Estimation Of Partial Group Delay With Applications To Small Samples

by

Milan Mangeshkar

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

in

Statistics

APPROVED :

Robert V. Foutz, Chairman

Klaus H. Hinkelmann

Golde I. Holtzman

Marvin Lentner

Marion R. Reynolds, Jr.

February, 1995
Blacksburg, VA 24061

Estimation Of Partial Group Delay With Applications To Small Samples

by

Milan Mangeshkar

Robert V. Foutz, Chairman

Statistics

(ABSTRACT)

Partial group delay has an interpretation as a parameter that measures the time-lag relationship between two channels of a multiple time series after adjustments have been made for the influence of the remaining channels. The time-lagged relationship is typically studied frequency by frequency. In this dissertation a procedure for estimating the partial group delay parameter is proposed which is intended to work well even for small sample sizes. The only published procedure for estimating the partial group delay parameter is by Zhang and Foutz [1989]. The procedure by them is an asymptotic one and requires a fairly large sample size.

The proposed procedure for estimating the partial group delay parameter uses the frequency domain approach of time series analysis. The frequency domain approach is also known as spectral analysis and models a time series using sine-cosine functions. The two most important spectral tools used in the dissertation are the discrete Fourier transform and the periodogram ordinates.

The procedure consists of finding preliminary values for the partial group delay parameter. The mean of the preliminary values is then estimated using transforming and modeling techniques on the preliminary values. A key requirement for the procedure is that the periodogram and cross periodogram ordinates at each Fourier frequency are independent of the periodogram and cross periodogram ordinates at all other Fourier frequencies. Under this requirement, the estimate is uniformly minimum variance unbiased. The key requirement is satisfied as the sample size increases or if the channels of the multiple time series are Guassian white noise processes and are not cross correlated. The performance of the procedure is demonstrated using a simulation study and is compared to the only published procedure by Zhang and Foutz [1989].

Acknowledgments

I would like to thank my thesis chair Dr. Foutz for his valuable guidance and support throughout this dissertation. I would also like to express my sincere gratitude to him for spending his time and effort proof-reading this document several times.

I would like to thank Dr. Hinkelmann, Dr. Holtzman, Dr. Lentner and Dr. Reynolds for the valuable education they have provided me during my years at Virginia Tech and also for accepting to serve on my doctoral committee.

I would like to dedicate this thesis to my parents. I am indebted to them for the immense encouragement and support they have given me throughout my academic career. I would like to thank my husband Nandan for his encouragement, support, and patience throughout this work. I would also like to thank him for sharing this ride on the roller-coaster called 'Research' with me and being there for me all the time. I would also like to thank my brothers Vivek and Vishal, my aunt Satyavati Shinde and my parents-in-laws Sudha Joshi and Prabhakar Joshi for their love and encouragement.

I would like to thank my colleague Shanthi Sethuraman and my consulting partner Michele Marini for all the wonderful times I had with them discussing various statistical problems. Finally I would like to thank Linda Breeding for all the secretarial help.

Table Of Contents

List Of Illustrations

Chapter 1
Introduction

1.1 Motivation

Time series analysis involves analysis of observations that are correlated and taken over time. The techniques of time series analysis can be broadly classified into two major categories, namely, techniques in the time domain, and techniques in the frequency domain. In the present work we extensively make use of the latter techniques. Analysis in the frequency domain is typically referred as 'Spectral Analysis' and these techniques make use of the fact that the wave like patterns in a series can be modeled using a sine-cosine function.

In the present work we propose a new procedure for estimating the partial spectral parameter called the partial group delay. This parameter is also known as the partial time delay or the adjusted group delay. Throughout this work these terms will be used interchangeably. Note that estimation of partial group delay involves three or more series. The aim of the present chapter is to familiarize the reader with the term partial group delay and to put forth our motivation for introducing a procedure for estimating the partial group delay parameter. In the following paragraphs, using a few examples we try to accomplish this aim.

Consider monthly data for the following series : production of lumber, prices of wood, and prices of furniture. It is evident that changes in the production of lumber will have its effect on the prices of wood and prices of furniture. For instance, a drop in the lumber production will lead to a shortage of wood, thus causing an increase in the wood prices and furniture prices. In addition the increase in the wood prices will cause an additional increase in the furniture prices. This increase will not be reflected immediately but may happen over a period of time. It would be of interest to quantify the extent of this time lag, that is, one would like to know, how soon are the changes in wood prices reflected in the prices of the furniture after adjusting for the effect of the lumber production.

One can observe several such groups of economic series, for instance, consider monthly data for production of crude oil, production of gasoline, and gasoline prices. A natural entity would be to calculate the time lag for the changes in gasoline production to be reflected in the gasoline prices after adjusting for the effects of production of crude oil. Another example would be that of finding the time lag between say iron and steel industry, and housing after adjusting for the prime interest rates.

Groups of similar such series can also be observed in fields of study, such as, oceanography, seismology, meteorology, geo-physics, and signal processing. For instance, in oceanography one might be interested in knowing the time lag between the changes in air pressure that are reflected on the height of the tidal wave after adjusting the two series for the effect of wind velocity. Another important use of the adjusted time lag is that it can be used to estimate other parameters. For instance, in sonar and radar systems, the estimation of time delay between the transmitted and the received signal from a target can be used to find the location and the velocity of the target (see Quazi [1981] for more details).

The entity which determines the time lag of one series over another series after adjusting for the influences of the common series is called as the partial group delay. In the following paragraph a formal definition for partial group delay is given using the result by Zhang and Foutz [1989]. The problem of estimating the partial group delay is an important one but is rarely addressed in the literature. There is only one method by Zhang and Foutz [1989] for estimating the partial group delay. This procedure requires a fairly large sample size. Our basic motivation for the present research has been the unavailability of an alternate procedure for estimating the partial group delay that works well even for a small sample size.

Let Λ be a band of frequencies and let X_Λ, Y_Λ, and $\left(Z_{1,\Lambda}, Z_{2,\Lambda}, \ldots., Z_{p,\Lambda}\right)$ be the components of the continuous, weakly stationary, and stochastic processes X, Y, and $\left(Z_1, Z_2, \ldots., Z_p\right)$ respectively in the band Λ. As Λ shrinks to a single frequency say λ_0, the relationship between X_Λ, and $\left(Y_\Lambda, Z_{1,\Lambda}, Z_{2,\Lambda}, \ldots., Z_{p,\Lambda}\right)$ reduces to a simple linear time lagged relationship as shown below,

$$X_\Lambda(t) = \alpha Y_\Lambda(t-\tau) + \alpha_1 Z_{1,\Lambda}(t-\tau_1) + \ldots\ldots + \alpha_p Z_{p,\Lambda}(t-\tau_p) + \varepsilon_\Lambda$$

where $-\infty < t < \infty$, ε_Λ is uncorrelated with $\left(Y_\Lambda, Z_{1,\Lambda}, Z_{2,\Lambda}, \ldots., Z_{p,\Lambda}\right)$, and τ is called the partial group delay of X behind Y after adjusting for the delays $\tau_1, \tau_2, \ldots., \tau_p$ due to the series $\left(Z_{1,\Lambda}, Z_{2,\Lambda}, \ldots., Z_{p,\Lambda}\right)$ respectively.

In many instances the lead or the lag will be a function of the frequency, for example, a wave propagating through a dispersive medium will have its speed of propagation dependent on the frequency. Hence, in the literature the phrase 'time delay' is used to suggest that the lead or lag is constant and the phrase 'group delay' to convey that the lead or the lag is dependent on frequency.

Other important partial spectral parameters are the partial spectral density, partial cross spectral density, partial phase, and partial coherence. Partial coherence is a very important parameter and it quantifies the strength of the relationship between the two series after adjusting for the effect of the common influencing series. We refer the reader to chapter 2 for a detailed discussion on these parameters as a thorough understanding of these parameters will facilitate understanding the partial group delay parameter in greater depths.

1.2 Organization Of The Thesis

The dissertation is divided into six chapters. In chapter 2 we present the basic concepts in time series analysis with emphasis on spectral analysis. The thrust of the discussion is not on technical terms but rather on the intuitive understanding of the concepts especially in spectral analysis. This chapter is strongly recommended for the reader who is unfamiliar with the notions in spectral analysis. Having acquired the requisite background knowledge we present in chapter 3 brief discussions of the papers referred by us during the course of this dissertation. The literature discussed falls into five main categories, namely, papers concerned with the estimation of unadjusted time delay and other spectral parameters; paper discussing the existing procedure for estimating partial group delay; papers on the merits/demerits of transforming data; and papers describing spline models and their application. Again we do not give the technical details but rather point out some of the key issues which were of help to us. In chapter 4 we present step-by-step details of the procedure for estimating the partial group delay. The use of the proposed procedure is then demonstrated using simulation studies in chapter 5. Finally we end this dissertation with conclusions, ideas for further research and an example in chapter 6. In the appendix we have details of the Box-Cox transformation technique and present the C language code for the various programs that were written by us to demonstrate the procedure.

Chapter 2
Review Of Spectral Analysis

2.1 Introduction

A ray of light is composed of many different colors occurring in varying proportions. Using an optical device such as a prism, a ray of white light can be split into its constituent colors. In physics this is referred to as a **Spectrum**. Alternately, if the different colors and their respective proportions were known, a ray of light could be reconstructed by mixing the colors in their right proportions. So, spectral analysis in physics refers to studying the composition and properties of light. Spectral analysis in the context of time series refers to expressing a process as a sum of sinusoids (sine-cosine wave), thus displaying the patterns and the variability in the process.

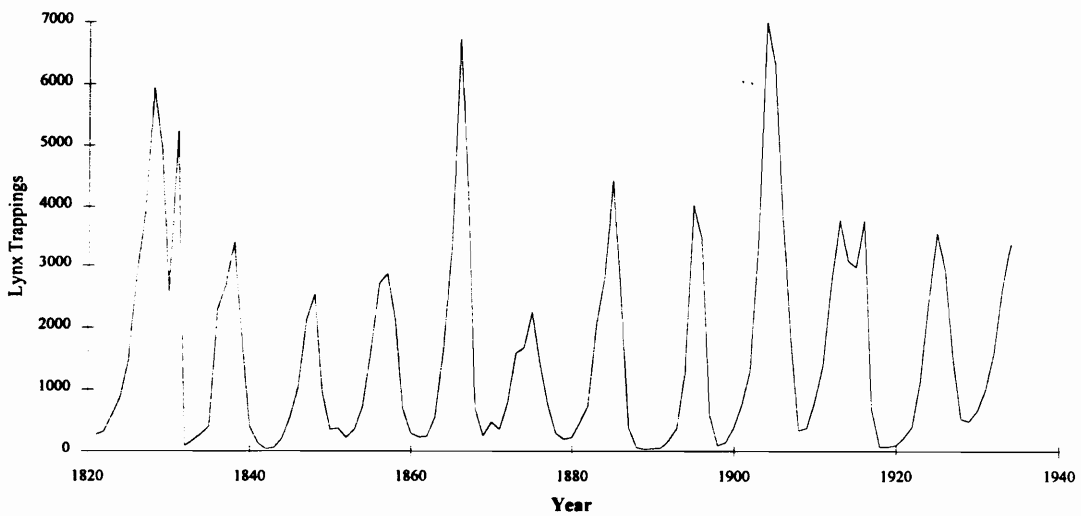
A **time series** is a sequence of observations made at regular intervals of time. By regular interval of time we mean that the observations are equispaced and the unit of time could be seconds, hours, days, months or years. Examples of such series are the daily temperatures, monthly unemployment figures, and annual rainfall figures.

The primary objectives for analyzing a time series are descriptive, inferential, prediction, and control. Using tools such as the **correlogram** or **periodogram**, data can be represented graphically. This will enable us to observe and study the patterns in the series, thus aiding us to understand and appreciate the underlying process which generated the series. Having depicted the series graphically, one would like to model the series, thereby quantifying the observed patterns. The fitted model in turn, would help to control the series and forecast values for the future. For instance, one could build a model for the sales data of a store based on previous months sales records. With the aid of the fitted model, the store can make predictions for the future months. The forecasted sales will aid the store in having just the right level of inventory, thereby increasing profits.

Some of the frequently used tools for analyzing data are the techniques of regression and analysis of variance. These techniques try to establish a relationship between a dependent variable and one or many independent variables. In addition, it is also assumed that the errors are independently distributed. In the analyses of time series we make use of the knowledge that the subsequent observations and hence subsequent errors are correlated. Often we try to seek a relationship between the past and the current observations. This relationship aids in understanding the nuances of the process and helps in predicting (forecasting) values for the future. Unlike other forms of analyses, in time series analysis the observations are first detrendend and the errors around the trend are modeled.

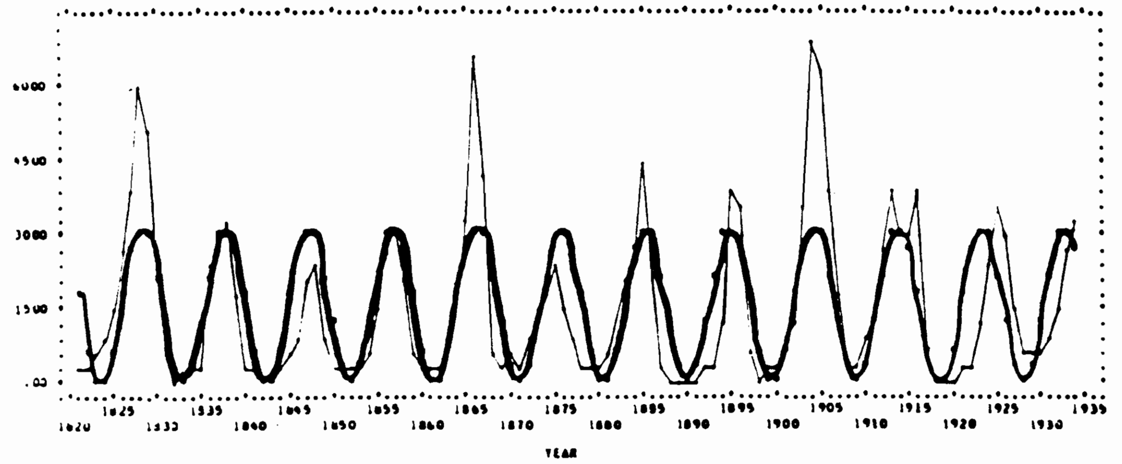
The two primary approaches employed in analyzing a time series are the time domain approach (Box Jenkins models) and frequency domain approach (Spectral Analysis). For the time domain approach, depending on the behavior of the series, the series is fitted with either an AR (autoregressive) model or a MA (moving average) model or an ARIMA (autoregressive integrated moving average) model. Spectral Analysis makes use of the fact that many phenomena in nature exhibit cyclical patterns. Using a Fourier transform the observed data are transformed to a sum of sinusoids (sine-cosine wave) and the variability in the data are studied frequency by frequency.

In this work data are analyzed using the frequency domain approach. The techniques used in spectral analysis are illustrated in the following example obtained from BMDP Statistical Software Manual. Figure 2.1 shows the annual number of Canadian lynx trappings for the years 1821-1934. The series exhibits an oscillatory pattern with approximately a 10 year period. To model this data set, a sine wave of the same period is fitted, this is shown in figure 2.2. The sine wave seems to fit the data well but it fails to take into account the heights of the peaks. To account for the peaks different sine waves with varying periods can be fitted. Figure 2.3 shows a sum of seven sine waves with periods ranging from 7½ to 12½ years fitted to the data. Further refinement can be obtained by taking the sum over several frequency bands as shown in figure 2.4.



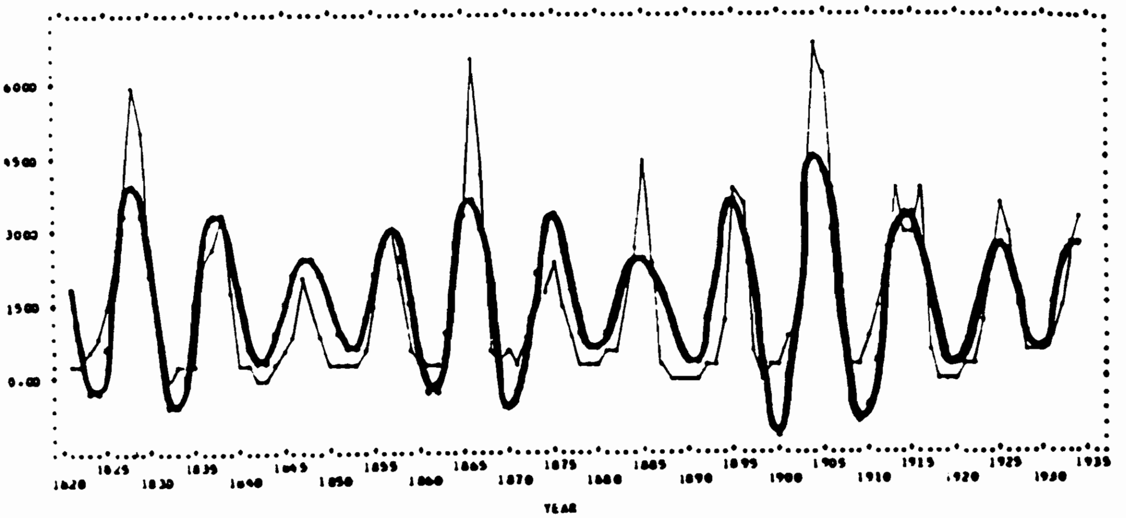
Plot Of Canadian Lynx Trappings [1821 - 1934]

Figure 2.2



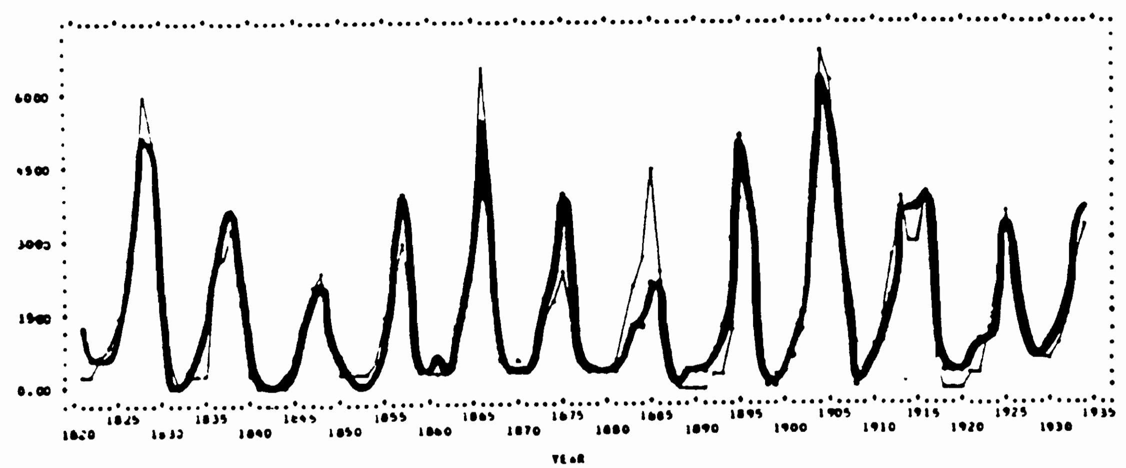
Plot Of Lynx Data Fitted With A Sine Wave Of Period 10 Years

Figure 2.2



Plot Of Lynx Data Fitted With A Sum Of Seven Sine Waves

Figure 2.3



Plot Of Lynx Data Fitted With Sum Of Several Sine Waves Over Several Frequencies

Figure 2.4

In the following sections, we describe some of the basic elements of spectral analysis. Emphasis is laid more on concepts than on technical details. For greater details the reader is referred to Koopmans [1974] as it has been extensively referred to in this work. The assumptions necessary for the analysis of a time series are listed and two forms of representation of a time series are given. The need to select an appropriate sampling interval is discussed in the sampling and aliasing section. Two crucial concepts namely transforming the data using Fourier transforms and constructing periodograms/cross-periodograms are also discussed. These two steps form the basis for analyzing any time series in the spectral domain. This is followed by sections which discuss univariate, bivariate, and multivariate spectral parameters and their interpretations. Finally, we end this chapter with a brief discussion on the notion of linear filters.

2.2 Assumptions

For the process $\{X(t)\}, -\infty < t < \infty$, under consideration, the following assumptions are made,

1. As the observations are made over time, we assume that subsequent observations are correlated.
2. The process under consideration is said to be stochastic (random), that is at time point t (say), the process assumes not a single value but rather a set of values. The series under study is then just one possible realization from the collection of all possible realizations.
3. The process is weakly stationary, that is, the mean of the process remains constant and the covariance depends upon τ, the displacement in time but not on time t. Also the variance of the process is finite. The property of weak stationarity can be represented mathematically as,

   $$E(X(t)) = \mu \qquad -\infty < t < \infty$$

   $$E(X(t)X(t+\tau)) = c(\tau) \qquad -\infty < t, \tau < \infty$$

   $$V(X(t)) < \infty \qquad -\infty < t < \infty$$

   We will assume throughout that $\mu = 0$.
4. The underlying process which generated the series is continuous and the series is just sampled over discrete time intervals. Typically, we assume that the series is sampled at time $t = 1, 2, \ldots\ldots, N$.

A point worth noting here is about an important class of processes called the nonstationary processes. These processes, unlike stationary processes do not wander about a constant mean. Instances of nonstationary processes are typically encountered in fields of study such as economics and business. An example of the same would be stock prices observed over several months. In general, although the process fluctuates at different levels at different times, the series shows a similar behavior when the differences in the levels are accounted for. Nonstationary processes can be modeled by assuming that the d-th difference of the process is stationary. Models which make this assumption are called the autoregressive integrated moving average models (ARIMA). In this work we do not consider such processes.

2.3 Representation Of A Time Series

Let $\{X(t)\}, -\infty < t < \infty$ be the time series of interest and let $x(t)$ represent the value taken by the series $\{X(t)\}$ at time point t. One way of modeling a series which exhibits a wave like pattern is by using the sine and cosine functions. For instance, the series could be represented exclusively by a sine function or by a cosine function or by a mixture of sine-cosine functions. Two forms of representing a time series, namely, the Cartesian representation, and the complex representation are discussed below. In this work the latter representation is of main concern to us. The Cartesian representation of a series $\{X(t)\}$ is given as,

$$X(t) = \sum_{\lambda} A_{\lambda} \mathrm{Sin}(\lambda t + \phi_{\lambda}) \qquad -\infty < t < \infty \qquad [2.1]$$

where λ is the angular frequency, A_{λ} is the amplitude, ϕ_{λ} is the phase, and the summation represents sums over different amplitudes, phases and frequencies.

Using the trigonometric relation,

$$\mathrm{Sin}(a+b) = \mathrm{SinaCosb} + \mathrm{CosaSinb}$$

and de Moivre relation,

$$e^{i\lambda t} = \mathrm{Cos}(\lambda t) + i\mathrm{Sin}(\lambda t)$$

the complex representation for the series $\{X(t)\}, -\infty < t < \infty$ can be constructed by expanding and rearranging expression [2.1] as given below,

$$X(t) = \sum_{\pm\lambda} C_\lambda e^{i\lambda t} \qquad -\infty < t < \infty \qquad [2.2]$$

where, $C_\lambda = \dfrac{A_\lambda e^{i\phi_\lambda}}{2i}$ and $C_{-\lambda} = \overline{C_\lambda}$

2.4 Sampling And Aliasing Effect

It has been stated in the assumptions that the process $\{X(t)\}$ is continuous and stochastic by nature. To gain better understanding of the underlying continuous process, the process needs to be sampled at discrete time interval say $t = 0, \pm1, \pm2, \ldots..$, in particular for the purpose of analysis we consider $t = 1, 2, .., N$. This time interval denoted by the symbol Δt is typically equispaced and is referred to as the sampling interval.

In spectral analysis we are concerned with functions of the type $e^{i\lambda t}$ (see expression [2.2]) where λ is the angular frequency and t is the time. When time t assumes discrete values say $t = 0, \pm1, \pm2, \ldots..$ the functions $e^{i\lambda t}$ and $e^{i(\lambda + 2\pi s)t}$ (s is some integer) become almost indistinguishable. That is, the components in $\{X(t)\}$ at frequency $\lambda \pm 2\pi, \lambda \pm 4\pi, \ldots...$ seem to have the same frequency λ. This phenomenon is called the **aliasing effect** and $\lambda \pm 2\pi s$ (s=1,2,3....) are said to be aliases of λ. As a result of this effect even the best estimates of the sampled series may prove to be poor estimates for the original continuous series.

The problem of aliasing can be minimized by choosing a proper sampling interval Δt. Using the sampling theorem an appropriate choice of Δt can be made. The theorem states that if for some frequency A, the power (variance) for the time series $\{X(t)\}$ outside the range $-A \le \lambda \le A$ is zero then the underlying continuous time series can be reconstructed from the sampled series, such that the sampling interval is $\Delta t = \pi k / A, \quad k = 0, \pm 1, \ldots$

2.5 Discrete Fourier Transforms

Let $\{X(t)\}$, $t = 1,2,3,\ldots,N$ be the sampled version of the process $\{X(t)\}, -\infty < t < \infty$. The discrete Fourier transform is defined as,

$$W_x(\lambda_k) = \frac{1}{N}\sum_{t=1}^{N} x(t)e^{-it\lambda_k} \qquad -\left[\tfrac{N-1}{2}\right] \le k \le \left[\tfrac{N}{2}\right] \qquad [2.3]$$

where [c] is an integer not greater than c and $\lambda_k = \dfrac{2\pi k}{N}$. The frequencies λ_k are called the Fourier frequencies. Note that, only discrete values of k will be used. The inequalities in expression [2.3] around k are used for compactness of the expression . In practice Fourier coefficients are found at Fourier frequencies λ_k, $k = 0,1,2,\ldots,\left[\tfrac{N}{2}\right]$.

In expression [2.3] each observation of the time series $\{X(t)\}$ at time t is multiplied by its corresponding observation on the sine and cosine waves. Each cross product is summed over all N observations and the average is found. This is equivalent to finding two covariances, namely, (i) between the series and the sine wave, and (ii) between the series and the cosine wave at the Fourier frequency λ_k , $-\left[\frac{N-1}{2}\right] \leq k \leq \left[\frac{N}{2}\right]$. Thus, finding the Fourier transforms of the data at Fourier frequency λ_k , $-\left[\frac{N-1}{2}\right] \leq k \leq \left[\frac{N}{2}\right]$ is equivalent to finding how well the data are modeled by the sine and cosine wave at Fourier frequency λ_k , $-\left[\frac{N-1}{2}\right] \leq k \leq \left[\frac{N}{2}\right]$.

If the time series $\{X(t)\}$ is Gaussian then the Fourier coefficients $W_x(\lambda_k)$ follow a multivariate complex normal distribution with mean 0 and variance $f_x(\lambda_k)$, called the spectral density function. The Fourier coefficients are uncorrelated provided the sample size is large and the spectral density function is smooth.

It can be shown that for the time series $\{X(t)\}$ sampled at discrete time points $t = 1,2,\ldots,N$, the variability (power) of the series $\{X(t)\}$ is given by,

$$\frac{1}{N}\sum_{t=1}^{N} x^2(t) = \sum_{k=-\left[\frac{N-1}{2}\right]}^{\left[\frac{N}{2}\right]} \left|W_x(\lambda_k)\right|^2 \qquad [2.4]$$

This implies that the variance accounted for by each Fourier frequency λ_k , $-\left[\frac{N-1}{2}\right] \leq k \leq \left[\frac{N}{2}\right]$ or rather by each wave pattern can be added to give the total variability for the series $\{X(t)\}$. If we were to consider the different waves as our various treatment groups, then this process of accounting for the variability as explained by each wave can be thought of as the technique of ANOVA.

For reasons of convenience we shall use the denominator $\sqrt{2\pi N}$ instead of N. Also for simplicity of an expression, the subscript k will be dropped from λ_k and the Fourier transform at a Fourier frequency λ will be represented as,

$$W_x(\lambda) = \frac{1}{\sqrt{2\pi N}} \sum_{t=1}^{N} x(t) e^{-i\lambda t} \qquad [2.5]$$

2.6 Periodograms And Cross Periodograms

Using the Fourier transforms $W_x(\lambda)$, periodogram ordinates are constructed as,

$$\begin{aligned} I_x(\lambda) &= W_x(\lambda)\overline{W_x(\lambda)} = \left|W_x(\lambda)\right|^2 \\ &= \left|\frac{1}{\sqrt{2\pi N}} \sum_{t=1}^{N} x(t) e^{-i\lambda t}\right|^2 \end{aligned} \qquad [2.6]$$

where $\overline{W_x(\lambda)}$ is the complex conjugate of $W_x(\lambda)$. Comparing expression [2.4] to expression [2.6], the periodogram ordinate can be interpreted as a measure of the amount of variability at frequency λ. Another way of describing a periodogram ordinate is that it is a sample statistic for measuring the variability in a series at a frequency λ.

The periodogram ordinates are asymptotically independent for $k \geq 0$ and follow a Chi-squared distribution with mean $f_x(\lambda_k)$ and the respective variances for $k \neq 0, \frac{N}{2}$; $k = 0$; and $k = \frac{\pi}{2}$ are $f_x^2(\lambda_k)$, $2f_x^2(0)$, and $2f_x^2(\pi)$.

For the bivariate case, let $\{X_j(t)\}$, and $\{X_k(t)\}, -\infty < t < \infty$ be the two series under consideration, then the cross periodogram ordinates are constructed at the Fourier frequencies λ as,

$$I_{jk}(\lambda) = W_j(\lambda)\overline{W_k(\lambda)} \qquad [2.7]$$

The cross periodogram ordinate is a measure of covariance between the series $\{X_j(t)\}$, and $\{X_k(t)\}$ at frequency λ.

2.7 Univariate Spectral Parameter

In probability theory, one way of summarizing the behavior of the random variable is by means of probability density function. Likewise in spectral analysis it would be of interest to measure the variability of not just a single instance but of the entire stochastic process $\{X(t)\}$, $-\infty < t < \infty$. Such a spectral parameter is called the autospectrum, commonly known as the **spectral density function**. It is denoted by $f_x(\lambda)$, where λ is a frequency of interest. Typically λ is taken to be the Fourier frequency.

The periodogram ordinates display the variability for one realization of the process and can be thought of as S^2, the sample variance in basic statistics. If repeated samples of size N were to be drawn from the stochastic process $\{X(t)\}$, $-\infty < t < \infty$ we would get a collection of periodogram ordinates at the various frequencies. The expected value of the ordinates at the respective frequencies can be found to give us the distribution of power for the stochastic process. Thus the spectral density function can be thought of as the quantity σ^2, the population variance in basic statistics. Note that, in this work the term power and variability will be used interchangeably.

The estimates of the univariate spectral parameters are easily derived from the periodogram ordinates. For example, it can be shown that the periodogram ordinate at frequency λ is an asymptotically unbiased estimate of $f_x(\lambda)$, the spectral density of the series $\{X(t)\}$. This is expressed in notations as,

$$\hat{f}_x(\lambda) = I_x(\lambda) = \left| \frac{1}{\sqrt{2\pi N}} \sum_{t=1}^{N} x(t)e^{-i\lambda t} \right|^2 \qquad [2.8]$$

However, for smaller sample sizes the periodogram ordinates are biased estimates of the spectral density function.

A major drawback of the periodogram ordinates is, that it is not a consistent estimator. That is, the variance of the periodogram ordinate does not tend to zero as the sample size N increases. This can be explained by observing the fact that the periodogram ordinates are Fourier transforms of $\hat{C}(\tau)$, the sample autocovariance function (see Koopmans [1974], pp. 74, expression 3.23, and pp. 266, expression 8.19),

$$I_x(\lambda) = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} e^{-i\lambda\tau} \hat{C}(\tau) \qquad [2.9]$$

where

$$\begin{aligned} \hat{C}(\tau) &= \frac{1}{N} \sum_{t=1}^{N-|\tau|} X(t+|\tau|)X(t) \qquad && |\tau| \le N-1 \\ &= 0 && |\tau| > N-1 \end{aligned} \qquad [2.10]$$

For lags near N-1, $\hat{C}(\tau)$ is an average of fewer pairs of observations. This leads to an unstable estimate of $\hat{C}(\tau)$ and hence an unstable $I_x(\lambda)$ irrespective of the sample size N.

The above drawback can be remedied by averaging the periodogram ordinates over a window. This is also known as smoothing a periodogram. A smoothed periodogram estimator is represented as,

$$\hat{f(\lambda)} = \sum_{\upsilon=-\left[\frac{N-1}{2}\right]}^{\left[\frac{N}{2}\right]} K(\lambda - \lambda_\upsilon) I_x(\lambda_\upsilon) \qquad [2.11]$$

where $K(\lambda)$ the window function is symmetric and real valued. It is also referred to as the periodic weight function. An example of such an estimator is the Danielle estimator. For the Danielle estimator a window is considered such that the frequency of interest is at the center of the window. The periodogram ordinates in this window are then averaged to give us an estimate of the spectral density for the frequency of interest. It can be shown (see Koopmans [1974], pp. 269, expressions 8.24-8.25) that the Danielle estimate is asymptotically unbiased and consistent.

Likewise for the time domain representation we can define a weighted covariance estimator for the spectral density function as,

$$\hat{f(\lambda)} = \frac{1}{2\pi} \sum_{\tau=-\infty}^{\infty} e^{-i\lambda\tau} W_m(\tau) \hat{C}(\tau) \qquad [2.12]$$

where $W_m(\tau)$, also called the lag window is the weight function. By applying smaller weights to unstable $\hat{C}(\tau)$, we can eliminate their effect and make the estimator consistent.

2.8 Multivariate Time Series

In practice it would be of interest to study two or more series simultaneously, thus leading to a multivariate system of study. Examples of such systems in time are encountered in fields of study such as agriculture, engineering, social sciences, biological sciences, and economics. The primary objectives for analyzing a multivariate time series would be to study the interrelationship among the univariate series that form the system or to know whether a series or a group of series influence another group of series. Brillinger [1975, pp. 1] points to a multivariate system of series generated from a set of signals recorded by an array of seismometers in the aftermath of an earthquake or nuclear explosion. Stock prices for various blue chip companies would form an interesting economic multivariate series. Monthly sales data of clothing items split up in categories such as skirts, blouses, shirts, trousers, and coats would be another example of multivariate time series.

Let $\mathbf{X}'(t) = \left(\{X_1(t)\}, \{X_2(t)\}, \ldots\ldots\ldots, \{X_p(t)\}\right)$, $-\infty < t < \infty$ be a multivariate system of interest, such that each $\{X_j(t)\}$, $j = 1,2,\ldots.,p$ is a univariate series. For $\mathbf{X(t)}$ to be a stationary stochastic process, each $\{X_j(t)\}$ should be weakly stationary. The covariance between the series $\{X_j(t)\}$, and $\{X_k(t)\}$, $j \neq k$ should be stationary, that is, it should depend on lag τ, and not on time t. It is expressed as,

$$C_{jk}(\tau) = E\left[X_j(t)X_k(t+\tau)\right] \quad -\infty < t,\tau < \infty, \quad 1 \leq j,k < p, \quad j \neq k \qquad [2.13]$$

When j=k , this becomes the condition of covariance stationarity.

The multivariate system can be adequately described by the parameter spectral density matrix (or covariance matrix). Other spectral parameters like phase, coherence, group delay can be derived from the spectral density matrix. We shall discuss below the parameters, how to estimate them and their interpretation.

The spectral density matrix $\mathbf{f}(\lambda)$, where λ is usually a Fourier frequency of interest has autospectrums, $f_{jj}(\lambda)$, $j = 1,2\ldots.p$, on the diagonals and the cross spectral densities $f_{jk}(\lambda)$, $j,k = 1,2\ldots.p$, $j \neq k$ on the off-diagonals. It is represented as,

$$\mathbf{f}(\lambda)=\begin{bmatrix} f_{11}(\lambda) & f_{12}(\lambda) & \ldots & f_{1p}(\lambda) \\ \vdots & \ddots & \cdots & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ f_{p1}(\lambda) & f_{p2}(\lambda) & \ldots & f_{pp}(\lambda) \end{bmatrix} \qquad [2.14]$$

Similarly the covariance matrix, $\mathbf{C}(\tau)$, with autocovariances $C_{jj}(\tau)$ $j=1,2....p$ and cross covariances $C_{jk}(\tau)$ $j,k=1,2....p$, $j \neq k$ is given by,

$$\mathbf{C}(\tau)=\begin{bmatrix} C_{11}(\tau) & C_{12}(\tau) & \ldots & C_{1p}(\tau) \\ \vdots & \ddots & \cdots & \cdots \\ \vdots & \cdots & \ddots & \cdots \\ C_{p1}(\tau) & C_{p2}(\tau) & \ldots & C_{pp}(\tau) \end{bmatrix} \qquad [2.15]$$

where $C_{jk}(\tau)$ is given by expression [2.13].

Estimates of the elements of the spectral density matrix are obtained by sampling each of the univariate series $\{X_j(t)\}$, $j=1,2,....,p$, at discrete time intervals, say $t=1,2,....,N$. To each univariate series Fourier transformation is applied at the Fourier frequencies $\lambda=\dfrac{2\pi k}{N}$, $k=0,1,2.....\left[\frac{N}{2}\right]$. Let the Fourier transforms at Fourier frequency λ be represented by $W_j(\lambda)$, $j=1,2,....,p$. The periodogram, and cross periodogram ordinates are constructed as,

$$\begin{aligned} I_{jj}(\lambda) &= W_j(\lambda)\overline{W_j(\lambda)} \qquad j=1,2,...,p \\ I_{jk}(\lambda) &= W_j(\lambda)\overline{W_k(\lambda)} \qquad j,k = 1,2....,p \quad j\neq k \end{aligned} \qquad [2.16]$$

As discussed in the previous section, $I_{jj}(\lambda)$ is an unbiased estimate of the autospectrum for the j-th univariate series. Similarly, it can be shown that the cross periodogram ordinate $I_{jk}(\lambda)$ estimates the cross spectral density $f_{jk}(\lambda)$, a complex valued function. The cross periodogram ordinates like the periodogram ordinates are asymptotically unbiased and are not consistent estimators. It can also be shown (see Koopmans [1974], pp. 124, expression 5.15) that the cross spectral density is a Fourier transform of the cross covariances, that is,

$$f_{jk}(\lambda)=\frac{1}{2\pi}\sum_{\tau=-\infty}^{\infty}e^{-i\lambda\tau}C_{jk}(\tau) \tag{2.17}$$

To make the cross periodograms consistent estimators, the real and the imaginary parts are smoothed. The cross spectral density is a measure of covariance between the series $\{X_j(t)\}$, and $\{X_k(t)\}$.

Koopmans [1974, pp. 137] defines the parameter phase at frequency λ as the angular lead or lag of the series $\{X_j(t)\}$ over the series $\{X_k(t)\}$. The parameter phase denoted by $\theta_{jk}(\lambda)$ for an observed series cannot be measured directly but can be calculated from the cross spectral density function $f_{jk}(\lambda)$ as

$$\theta_{jk}(\lambda)=-\mathrm{Arc}\tan\left(\frac{\mathrm{Im}\,f_{jk}(\lambda)}{\mathrm{Re}\,f_{jk}(\lambda)}\right) \tag{2.18}$$

In the paper by Hannan and Thomson [1973], it is mentioned by them that, if $\theta_{jk}(\lambda)$ is differentiable and has a smooth derivative then the estimate of $\theta_{jk}(\lambda)$ can be replaced by an estimate of $\theta'_{jk}(\lambda)$, often called the group or time delay. Group delay is a much more meaningful parameter because it has a direct interpretation as the time-lead or time-lag of the series $\{X_j(t)\}$ over the series $\{X_k(t)\}$ at frequency λ. The group delay at frequency λ is usually represented as follows,

$$\tau(\lambda)=\frac{\partial\theta_{jk}}{\partial\lambda} \tag{2.19}$$

The coefficient of coherence $\rho_{jk}(\lambda)$ at frequency λ is given by,

$$\rho_{jk}(\lambda) = \frac{|f_{jk}(\lambda)|}{\sqrt{f_{jj}(\lambda) f_{kk}(\lambda)}} \qquad [2.20]$$

It is interpreted as a measure of correlation between the series $\{X_j(t)\}$, and $\{X_k(t)\}$ or as a measure of linear association between the series $\{X_j(t)\}$, and $\{X_k(t)\}$. By linear association we mean the degree to which one series can be expressed as the output of a linear filter with the other series as the input. Using the Schwartz inequality, it can be shown that $\rho_{jk}(\lambda)$ satisfies $0 \le \rho_{jk}(\lambda) \le 1$. If the coefficient $\rho_{jk}(\lambda)$ is zero, it would imply lack of correlation and a coefficient of one signifies maximum association. One of the important properties of coherence is that if the univariate series are passed through respective linear filters with the same phase shift, then the coherence between the output of the linear filters is same as that of the input series. A nicer interpretation can be given using $\rho^2_{jk}(\lambda)$, the squared coefficient of coherence. The coefficient $\rho^2_{jk}(\lambda)$ measures the amount of power explained by expressing either one of the series as a linear combination of the other series.

2.9 Partial (Adjusted) Spectral Parameters

The univariate and multivariate parameters discussed above aided in explaining the relationship between two time series. In this work the focus is on slightly different parameters and is motivated by the following example. Electron-encephalogram studies, commonly known as **EEG** studies are conducted to understand how specific regions of the brain function. For example, one might want to know the centers where epilepsy is manifested. The brain is a very complicated surface and is made of millions of neurons.

These neurons are responsible for the electrical activities in the brain. To study these electrical activities a set of electrodes are placed at different sites in the brain. When an impulse is passed, data are recorded at the various sites. Data collected at each site can be thought of as an univariate series, thus generating a multivariate system of study. Suppose we study the electrical activities at Q chosen sites. In particular we are interested in the interrelationship between two of the Q sites and suppose we observe that the electrical activities at these two sites are almost the same. We are then led to conclude that either the neurons at the two sites are really connected by a physiological pathway or the neurons at the other $(Q-2)$ sites are driving the neurons at the sites in question. In this work the latter case is of concern to us. It is of interest to study the association between two sites after isolating the effects of some common influencing sites. To study these relationships, we will consider the parameters partial spectral density, partial phase, partial group delay and partial coherence.

Let $\mathbf{X}(t) = (\{X_1(t)\}, \{X_2(t)\}, \ldots\ldots, \{X_p(t)\})$, $-\infty < t < \infty$ be a multivariate system of study and let, $(X_{m_1}(t), X_{m_2}(t), \ldots\ldots, X_{m_q}(t))$ be the q series influencing the series $\{X_j(t)\}$, and $\{X_k(t)\}$. It is of interest to find whether the series $\{X_j(t)\}$, and $\{X_k(t)\}$ are genuinely related or do they appear to be related because of their association with the q series. To determine the exact relationship between the series $\{X_j(t)\}$, and $\{X_k(t)\}$, the effect of the q influencing series should be subtracted from the series $\{X_j(t)\}$, and $\{X_k(t)\}$. This is achieved by constructing linear functions of the q series that best approximates the series $\{X_j(t)\}$, and $\{X_k(t)\}$ respectively. In time series terminology the linear function is called a 'Linear Filter' and the process as 'filtering' the data.

Let, $\{\hat{X}_{j.\mathbf{m}}(t)\}$, $\mathbf{m}=(m_1,m_2,\ldots\ldots,m_q)$, be the best approximation of the series $\{X_j(t)\}$ obtained by constructing a linear filter of the q series, $(\{X_{m_1}(t)\},\{X_{m_2}(t)\},\ldots\ldots,\{X_{m_q}(t)\})$. The residual process for the series $\{X_j(t)\}$ can then be represented as,

$$V_{j.\mathbf{m}}(t)=X_j(t)-\hat{X}_{j.\mathbf{m}}(t) \qquad [2.21]$$

Similarly, let $\{\hat{X}_{k.\mathbf{m}}(t)\}$, $\mathbf{m}=(m_1,m_2,\ldots\ldots,m_q)$, be the best approximation of the series $\{X_k(t)\}$ obtained by constructing a linear filter of the q series, $(\{X_{m_1}(t)\},\{X_{m_2}(t)\},\ldots\ldots,\{X_{m_q}(t)\})$. The residual process for the series $\{X_k(t)\}$ is then given by,

$$V_{k.\mathbf{m}}(t)=X_k(t)-\hat{X}_{k.\mathbf{m}}(t) \qquad [2.22]$$

$\{V_{j.\mathbf{m}}(t)\}$, and $\{V_{k.\mathbf{m}}(t)\}$ are residual processes, free of the influence of the q common series and hence will be used to determine the true relationship between the series $\{X_j(t)\}$ and $\{X_k(t)\}$. Note that the notations in expressions [2.21], and [2.22] represent the entire time series observed at time t=1,2,...,N and not just the value at a specific time point t.

The cross spectral density between the series $\{X_j(t)\}$ and $\{X_k(t)\}$ after removing the influences of the q series at frequency λ is called the partial cross spectral density. It is equivalent to finding the cross spectral density between the residual series $\{V_{j.\mathbf{m}}(t)\}$ and $\{V_{k.\mathbf{m}}(t)\}$ and is given by,

$$f^{V}_{jk.\mathbf{m}}(\lambda)=f^{X}_{jk}(\lambda)-\mathbf{f}^{X}_{j\mathbf{m}}(\lambda)\mathbf{f}^{X}_{\mathbf{m}}(\lambda)^{-1}\overline{\mathbf{f}^{X}_{k\mathbf{m}}(\lambda)} \qquad [2.23]$$

where $\mathbf{f}^{X}_{j\mathbf{m}}(\lambda)$ is a 1xq vector of cross spectral densities between $\{X_j(t)\}$, and $\{X_{m_i}(t)\}$, $i=1,2,\ldots,q$. The qx1 vector, $\overline{\mathbf{f}^{X}_{k\mathbf{m}}(\lambda)}$ is the complex conjugate of the vector $\mathbf{f}^{X}_{k\mathbf{m}}(\lambda)$ of cross spectral densities between $\{X_k(t)\}$ and $\{X_{m_i}(t)\}$, $i=1,2,\ldots,q$ and $\mathbf{f}_{\mathbf{m}}^{X}(\lambda)$, is a qxq matrix of spectral, and cross spectral densities of $\{X_{m_i}(t)\}$, $i=1,2,\ldots,q$. The indices for j and k vary from 1,2,......p $(j \neq k)$, excluding the indices $m_1, m_2, \ldots\ldots m_q$. When $j=k$, we get the partial auto spectral density.

The coefficient of partial coherence between the series $\{X_j(t)\}$ and $\{X_k(t)\}$, is equivalent to calculating the coefficient of coherence between the residual processes $\{V_{j.\mathbf{m}}(t)\}$ and $\{V_{k.\mathbf{m}}(t)\}$, and is given by,

$$\rho_{jk.\mathbf{m}}(\lambda)=\frac{\left|f_{jk.\mathbf{m}}^{V}(\lambda)\right|}{\sqrt{f_{jj.\mathbf{m}}^{V}(\lambda)f_{kk.\mathbf{m}}^{V}(\lambda)}} \qquad [2.24]$$

$\rho_{jk.\mathbf{m}}(\lambda)$ is a measure of linear association between $\{X_j(t)\}$ and $\{X_k(t)\}$ at frequency λ, after removing the influence of the q series.

Partial phase, denoted by $\theta_{jk.\mathbf{m}}(\lambda)$, measures the angular displacement of $\{X_j(t)\}$ and $\{X_k(t)\}$ at frequency λ after removing the influence of the q series. The parameter $\theta_{jk.\mathbf{m}}(\lambda)$ is given by,

$$\theta_{jk.\mathbf{m}}^{V}(\lambda)=\arg\left(1-\frac{\mathbf{f}_{j\mathbf{m}}^{X}(\lambda)\mathbf{f}_{\mathbf{m}}^{X}(\lambda)^{-1}\overline{\mathbf{f}_{k\mathbf{m}}^{X}(\lambda)}}{f_{jk.\mathbf{m}}^{X}(\lambda)}\right)+\theta_{jk}^{X}(\lambda) \qquad [2.25]$$

where the term in the bracket, is the proportion of variability not explained by the regression of $\{X_j(t)\}$, and $\{X_k(t)\}$, individually, on the q series, $\left(\{X_{m_1}(t)\},\{X_{m_2}(t)\},\ldots\ldots,\{X_{m_q}(t)\}\right)$.

Partial group delay, $\tau_{jk.\mathbf{m}}(\lambda)$ expresses the true lead or lag $\{X_j(t)\}$ has on $\{X_k(t)\}$ at frequency λ, after removing the influence of the q series. This is equivalent to finding the group delay between the residual processes $\{V_{j.\mathbf{m}}(t)\}$ and $\{V_{k.\mathbf{m}}(t)\}$. It is given by,

$$\tau_{jk.m}(\lambda) = \frac{\partial \theta^{V}_{jk.m}}{\partial \lambda} \quad [2.26]$$

2.10 Linear Filters

The term '**filter**' originated in Electrical Engineering. The need to design systems which could attenuate/accentuate the input at certain frequencies gave rise to the theory of linear filters. A practical application of filters is found in the AM or FM radio tuners. This device suppresses the transmission of a signal in unwanted frequencies and transmits the signals only in its specified frequency range.

There are different types of filters. Filters found in audio amplifiers for example, pass all components in the frequency band $(-\lambda_0, \lambda_0)$. Such filters are called '**Low Pass**' filters. These filters reduce the effect of high frequency distortion. Others such as the one used in the base control of an audio amplifier passes all components outside the frequency band $(-\lambda_0, \lambda_0)$. These filters are called as '**High-Pass**' filters.

An important application of spectral analysis is in the study of systems which are linear and invariant in time. By invariant in time we mean, if the two inputs to the filter are the same except that they are displaced in time then the outputs will also be the same with the same displacement in time. A linear filter transforms an input series $\{X(t)\}$ to an output series $\{Y(t)\}$. Using the linear operator L the most generalized form of a linear filter can be represented as,

$$Y(t) = L(X(t)) \quad [2.27]$$

In particular, the output series $\{Y(t)\}$ of a linear filter with input $\{X(t)\}$, for a continuous process, can be represented as

$$Y(t)=\int_{0}^{\infty} g(u)X(t-u)du \qquad [2.28]$$

and that for a discrete process is,

$$Y_t=\sum_{u=0}^{\infty} g_u X_{t-u} \qquad [2.29]$$

where $g(u)$ and g_u are some deterministic function and are independent of the form of the input. The functions $g(u)$ and g_u measures the effect of the input in the time domain and are called as the **impulse response** function. One can also conceive of systems where the output is based on future input values. But such systems are unrealizable and hence will not be considered in our discussion.

Just as the impulse response function measures the effect of the input in the time domain, the complex valued **transfer function**, denoted by $B(\lambda)$, measures the effect of the input in the frequency domain. It can be shown that $B(\lambda)$ is a Fourier transform of the impulse response function $g(u)$ and can be represented as,

$$B(\lambda)=\int_{-\infty}^{\infty} g(u)e^{-i\lambda u}du \qquad [2.30]$$

For the frequency domain approach, the relationship between the spectral density of the output of a linear filter and its input is as follows,

$$\begin{pmatrix}\text{spectral density of the}\\ \text{output at frequency } \lambda\end{pmatrix}=\begin{pmatrix}\text{squared norm of}\\ \text{the transfer function}\end{pmatrix}*\begin{pmatrix}\text{spectral density of the}\\ \text{input at frequency } \lambda\end{pmatrix}$$

mathematically this can be represented as,

$$f_y(\lambda) = |B(\lambda)|^2 f_x(\lambda) \qquad [2.31]$$

Expression [2.31] shows the simplicity of working in the frequency domain. It shows that the value of the output spectral density depends on $|B(\lambda)|^2$ and the input spectral density at frequency λ and is not contaminated by any other frequency say λ'. In contrast the time domain representation given by expression [2.29] shows its contamination due to the inclusion of the input values at other time points as well. Analyzing the data in the frequency domain makes it possible to study the properties of the system separately at each frequency.

Chapter 3
Literature Review

3.1 Introduction

The objectives of this research are to introduce a procedure for estimating the parameter partial group delay and the procedure should be such that it works well even for a small sample size. The parameter partial group delay is defined as the lead or lag between two series of interest after eliminating the spurious effects of one or many common influencing series. In this chapter we give a brief overview of procedures available in the literature for estimating partial group delay and partial coherence in conjunction with the procedures available for estimating group delay and coherence. We use the terms group delay and coherence to refer to the lead (or lag) and degree of relationship respectively between two series of interest.

Literature has many papers addressing the problem of estimating group delay and coherence. To mention a few these are by Carter [1981,1987], Hannan and Thomson [1971, 1973, 1981, 1988], Hinich and Wilson [1992], Nikias and Pan [1988], and Ramsey and Foutz [1992]. The problem of estimating the partial group delay parameter hardly appears in the literature. In fact there is only one procedure by Zhang and Foutz [1989] for estimating the partial group delay. It is in the light of this situation that we think our research in this direction is important and it is our hope that the procedure we propose will prove to be of much use.

The material discussed in this chapter is divided into four sections. In the first section we discuss procedures that produce valid estimates under the standard assumption of normality and incoherence of the signal and noise series. We also discuss procedures that are to be used when the signal and or noise series are not Gaussian and the signal and the noise series are correlated. We thus hope to give the reader a broader picture of the procedures available for estimating group delay and coherence. These references have been of immense help in understanding the behavior of the various spectral parameters and their interrelationship with each other. In the second section we discuss the literature available for estimating the partial group delay. In the third section we discuss briefly the Box Cox transformation technique and in section four brief information regarding spline models is given. Finally, in section five we mention other miscellaneous references used in the development of this work.

3.2 Review Of Literature For Estimating Group Delay

The problem of estimating group delay is motivated well by Carter [1981] in the context of estimation techniques for passive sonar signal processing for naval systems. The estimation of the group delay is not just to find the lead or lag of one series over another but to go a step further and use this estimate of group delay to estimate the position and velocity of a moving acoustic source. The group delay problem has been considered by Foutz [1980] in the examination of the tree rings.

The literature for estimating group delay can be broadly divided into two categories, namely those procedures that fall under the class of generalized cross-correlators and those procedures that make use of higher ordered spectrum. In the following two paragraphs we discuss briefly the procedures that fall in the above two classes. We assume that each of the series to be examined is made up of two components, namely, a signal plus a noise component.

The procedures that fall in the class of generalized cross-correlators can be used provided for each of the series the signal and the noise series are incoherent (uncorrelated) and the noise series for the different time series under consideration are incoherent with each other. It is also assumed that the signal and the noise series are stationary Gaussian series. Procedures belonging to this class consists of two steps, namely, (i) compute a standard correlation function as given below,

$$R(\tau) = E[X_1(t)X_2(t+\tau)]$$

and (ii) maximize the function, such that τ that maximizes $R(\tau)$ provides an estimate of the group delay. Intuitively one can think of these procedures as those that find the peak of the sample cross-correlation function of the outputs of the two sensors. A detailed review of work on group delay estimation and coherence in the class of generalized cross-correlators is given by Carter [1987].

Procedures based on higher ordered spectra are used when the signal is non-Gaussian and noise sources are spatially correlated or when the signal and noise sources are correlated. Note that the extent of dependency (correlation) is not known. Estimation in the class of generalized cross-correlators is not considered as these procedures do not have the ability to suppress the effect of correlated noise sources. As a result the cross correlation function is now composed of the joint effect of the signal plus the noise and not just the signal. This is highly undesirable, and hence to circumvent this problem estimation using higher order spectrum is considered. Higher ordered spectra are given in terms of higher ordered cumulants which have the ability to preserve information on the non-Guassian stationary random processes. Higher ordered spectra have also been used to detect non-linearities in the mechanisms that generated the series. Also higher ordered spectra are independent of the correlated noises as polyspectra of order greater than two are zero. Estimates obtained by these methods are unbiased unlike the estimates obtained by the cross-correlator methods which are typically biased. For further details on higher ordered spectra the reader is referred to Akaike [1966]. For procedures based on higher ordered spectra to estimate the group delay and coherence the reader is referred to Hinich and Gary [1990], Nikias and Pan [1988], and Carter [1986].

We have extensively referred to the papers by Hannan and Thomson [1971, 1973, 1981, 1988] in this dissertation. The procedures suggested by them fall in the class of generalized cross-correlators. We discuss in the following paragraphs some of the key points illustrated in each of their papers.

It was observed by Akaike and Yamanouchi [1963] that when the time delay is large the conventional procedures used for estimating the coherence produces a biased estimate. This is referred to in the time series literature as 'coherency bias'. Hannan and Thomson [1971] introduced a procedure for estimating the coherence when the lag between the two series was big. The procedure is based on the likelihood function of complex valued Fourier transforms which asymptotically follow a complex multivariate normal distribution.

Hannan and Thomson [1973] suggest a procedure for estimating group delay. This procedure closely follows the Hannan and Thomson [1971] procedure. The procedure consists of Fourier transforming the data and obtaining cross-periodogram ordinates. A band B consisting of m Fourier frequencies is constructed around a frequency of interest say λ_0 and the estimated group delay is found as the value of τ that maximizes the function given below,

$$\hat{q}(\tau) = |\hat{p}(\tau)|^2 = \frac{1}{m}\sum_{B} I_{12}(\lambda)e^{-i\tau\lambda} \qquad [3.1]$$

An intuitive explanation for the procedure is as follows :- A narrow band is constructed around the frequency λ. Each series $\{X_j(t)\}$ is filtered in this narrow band to produce the output $\{X_j(\lambda,t)\}$. Let τ maximize the function $|E\{X_1(\lambda,t)X_2(\lambda,t+\tau)\}|$, then as the band narrows indefinitely the optimizing τ approaches the group delay. In this paper Hannan and Thomson suggest that phase is not defined for zero coherence. Also the reliability of the estimate of the group delay depends on coherence and for small coherence the rate of approach of the distribution of the estimate to its asymptotic form becomes slow.

A procedure for estimating group delay when it is not constant over all frequencies is suggested by Hannan and Thomson [1981]. This procedure uses modeling techniques for spectra and cross-spectra in the time domain unlike the procedure proposed by us in this work which uses modeling techniques in the frequency domain. Hannan and Thomson [1988] suggest a procedure for estimating the group delay when the signal to noise ratio is low for all frequencies. In Hannan and Thomson [1981] it was suggested that frequencies with low coherence can be omitted but in this situation it would not be an acceptable procedure as the signal-to-noise ratio is low for all frequencies. Hence they propose a procedure which uses a weighing scheme for the different frequencies. The procedure is an iterative one where an initial estimate of group delay is obtained. The data are realigned using this initial estimate. Estimate for the cross spectra is obtained using modeling techniques in the time domain. A new estimate for the group delay is obtained using a modified formula which closely resembles expression [3.1]. In the modified formula each of the frequencies are weighed by the quantity $\frac{\hat{\sigma}^2(\lambda_k)}{1-\hat{\sigma}^2(\lambda_k)}$ where $\sigma(\lambda_k)$ is the coherence at frequency λ_k. Thus, cross periodogram ordinates at frequencies with high coherence get weighed more and those with low coherence get weighed less. The procedure is repeated using the new estimate of group delay.

The method of Ramsey and Foutz [1992] is a two stage procedure. In stage I preliminary estimates for the spectral parameters are obtained and in stage II these preliminary estimates are modeled using polynomial regression techniques to obtain point estimators for the various spectral parameters. This method is different from the other methods in the sense that it uses modeling techniques in the frequency domain rather than in the time domain. Under a certain condition these estimates are uniformly minimum variance unbiased and the method provides confidence estimators that have exact confidence coefficients. The condition is assumed to hold asymptotically as the sample size increases. This procedure is also applicable for small sample sizes and requires that the series be preprocessed so that the condition holds approximately.

3.3 Review Of Literature For Estimating Partial Group Delay

The only procedure for estimating partial group delay is by Zhang and Foutz [1989]. This procedure closely follows the Hannan and Thomson [1973] procedure for estimating group delay. The authors derive the conditions for consistency and asymptotic normality of the estimating sequence. This procedure also falls in the class of generalized cross-correlators except that the peak is located in the sample cross spectrum of the residual processes. The residual processes are the ones that are obtained by adjusting for the effect of the common influencing series from the series of interest. For more details about this procedure the reader is referred to chapter 4.

3.4 Review Of Literature On Transformation Techniques

For issues related to the choice of an appropriate transformation and for merits/demerits of using transformed data an abundant amount of information is found in the literature. Especially the papers by Andrew [1971], Box and Cox [1964], Carroll and Ruppert [1981], Hinkley and Runger [1984], and Tukey [1955] draw interesting points regarding the pros and cons of transforming the data and working in the transformed metric.

In the analysis of data using linear models the following assumptions must hold :-

1. Simplicity of the structure of $E(y)$ (example : model should be additive)
2. Errors should be normally distributed
3. Variance of the error should be constant and
4. Observations should be independently distributed

Tukey [1955] suggests that we have two choices for data that does not satisfy these assumptions, namely, to transform the data so that the assumptions hold or to invent new methodology which will use data in its original form. One can readily see that the former alternative seems much simpler than the latter one, though the precise nature of the transformation will depend on what ills we are trying to rectify. For instance, Bartlett [1947] suggests that a transformation to stabilize variance can be obtained by determining the relationship between variance and mean. This could be done simply by plotting a graph of the variance versus the mean. To detect departures from normality Anscombe [1961], and Anscombe and Tukey [1963] suggest analysis of the residuals. Box and Tidwell [1962] suggests transformation of the dependent as well as independent variables.

Box and Cox [1964] suggests transformations of the dependent variable in the power family. An important point drawn by them is that the objective of transformation should not only be that the assumptions hold but it should also lend to easy interpretation of the results in the transformed metric. For instance, a formal analysis may show that $\sqrt{y}$ is the best transformation to achieve normality and constant variance for errors but one may have compelling arguments of ease of interpretation for working with say $\log(y)$. In this dissertation we have used the Box-Cox transformation technique to find a transformation so that the assumptions hold. The details of the procedure are given in the appendix.

Having obtained the right transformation, the traditional estimates in the original units are obtained by applying an inverse transformation on the transformed estimates. These estimates are typically biased. Neyman and Scott [1960] introduce a procedure which will obtain uniformly minimum variance unbiased estimates in original units. Carroll and Ruppert [1981] focus their attention on the issues related to prediction of observations in original units when data is transformed so as to follow a linear model.

3.5 Review Of Literature On Spline Models

Wold [1974] quotes from Rice [1969] that functions which express physical relationships are frequently of a disjointed or disassociated nature. In other words their behavior in one region maybe totally unrelated to their behavior in another region. It is precisely for this reason that approximating such functions by ordinary polynomials will prove to be inadequate. Instead the use of spline functions is recommended. Several references can be found in Buse and Lim [1977] for application of splines to real world problems. To mention a few, splines can be used to test for structural changes, wage determination, and in analysis of chemical data.

Spline functions are defined as piecewise polynomials of degree n. The abscissa where the pieces join are called the knots and fulfill continuity conditions for the function itself and its $(n-1)$ derivatives. Polynomials with no knots and piecewise polynomials with more than one discontinuous derivatives may also be considered as splines. Thus, in general splines are smooth continuous curves with one or more discontinuities. The degree of smoothness depends on the number of knots. Spline functions with fewer knots will be much smoother than functions with many knots. Though, the fit of the data will be much better with more knots.

For a spline function the parameters at the users disposal are (1) the degree n of the spline function, (2) the number of knots, and (3) position of the knots. Wold [1974] and Smith [1979] both agree that the latter two parameters of spline functions are important aspects but difficult to determine. Wold [1974] suggests that instead of treating knots as free parameters, one should choose knots so as to correspond to the overall behavior of the data. In the same context Simth [1979] comments that if knots are considered as parameters to be estimated then they enter into the regression analysis in a non-linear fashion and then one is faced with all the ills associated with non-linear regression. Thus, the choice of fixed knots seems to be the most simplified approach as one can then use ordinary least squares for estimation purposes.

There are different types of spline functions, for instance, Wold [1974] discusses B-splines, Smith [1979] analyzes data using '+' functions. Fuller [1969] describes linear and quadratic splines. Buse and Lim [1977] show how to fit a piecewise polynomial of varying degree with varying continuity restriction. The reader is also referred the SAS/STAT User's Guide version 6.0 (pages 1567-1575) for a number of numerical examples on how to fit a spline function. Results for spline functions using fixed knots can easily be obtained by using the 'Proc Transreg' procedure of the software SAS. To use Proc transreg the user specifies one or all of the following :- the degree of the spline, the number and position of the knots and the discontinuities. The model for a spline of degree three with discontinuities at knots $x = 5, 10, 15$ can be represented as follows,

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-5)^3 + \beta_5 (x-10)^3 + \beta_6 (x-15)^3 + \varepsilon$$

Thus, the spline in the above expression is a weighted sum of a constant, a straight line, a quadratic curve, a cubic curve for the portion of $x < 5$, a cubic curve for the portion of x between 5 and 10, a different cubic curve for the portion of x between 10 and 15, and a cubic curve for the portion of $x > 15$.

3.6 Review Of Literature On Other Relevant Material

The text books by Bloomfield [1976], Box and Jenkins [1976], Brillinger [1981], Brockwell and Davis [1991], Koopmans [1974], Priestley [1981], and Wei [1990] have aided in understanding the theory and application of time series. The papers by Akaike [1962], Akaike and Yamanouchi [1962], Deaton and Foutz [1980], and Hannan and Robinson [1973], Koopmans [1964a, 1964b] were useful in the context of spectral estimation. The papers by Granger [1969, 1980, 1981] investigate causality and are a good source of real world examples on which times series techniques can be applied. The text books by Cooper et al. [1974], Kellaway and Peterson [1976] introduced the EEG (electroencephalogram) technology to us. The authors extensively make use of the spectral tools to investigate and collect information about the centers in the brain where epilepsy is manifested. The text book by Pankratz [1991] provided data for the example in chapter 6. The text-books by Graybill [1976], and Myers [1990] were referred for queries regarding linear models and regression analysis respectively.

Chapter 4
Estimating Partial Group Delay

4.1 Introduction

Consider the data obtained from an EEG (electroencephalogram) study. In particular, let X, Y, and Z be the three sites which are of interest to us. Suppose we have the following scenario: we observe that the electrical activities recorded at sites X and Y are almost similar to that at site Z except that the electrical patterns are delayed in time. This suggests that the neurons at site Z are driving the neurons at sites X and Y. Further let us suppose that the electrical activity observed at site Y appears to have a bigger delay than the electrical activity pattern at site X. This leads us to believe that the neurons at site X are connected by physiological pathway to the neurons at site Y. The question of interest is to find the exact delay that is observed at site Y due to the neurons at site X after removing the influence of the neurons at site Z from both sites X and Y. Often the term 'delay' is referred to as the lead of site X over site Y or the lag of site Y with respect to site X. A spectral parameter which measures the lead or lag of one series over another after removing the influences of some common series is called the partial group delay.

In the literature there is only one procedure by Zhang and Foutz [1989] to estimate the parameter partial group delay and this procedure requires a fairly large sample size. In the present work an effort has been made to put forth a procedure that will serve as another method for estimating the partial group delay parameter and yield better results under certain situations. The proposed technique is especially intended for use when sample sizes are not too big. In the following paragraph we briefly describe the two important steps of the proposed technique.

The proposed technique involves two main stages. In stage-I a preliminary estimate for partial group delay is found using the procedure by Zhang and Foutz [1989]. Their procedure was an extension of the work done on estimating the unadjusted group delay by Hannan and Thomson [1973]. The technique falls in the class of generalized cross-correlators. As explained in chapter 3, section 3.1 these techniques try to find the peak of the sample correlation function. So in the context of this work it would be the peak of the sample partial correlation. Traditionally group delay was estimated as the derivative of the phase and thus if the phase was badly estimated this would in turn lead to a bad estimate of group delay. The work by Hannan and Thomson [1973] is therefore significant in the sense that it is no longer necessary to estimate the phase in order to estimate the group delay. Stage-II of the procedure uses the preliminary estimates from stage-I in the procedure by Ramsey and Foutz [1992]. This stage treats the preliminary estimates as observations. Using various transforming and modeling techniques for these observations an estimate of the mean is obtained. Under an ideal condition this estimate is uniformly minimum variance unbiased.

We shall discuss, in the following paragraphs, details of the proposed technique when there is only one influencing series, the ideal condition, modification to the technique when there are two or more influencing series, how to compute the confidence intervals, an alternative method for finding the preliminary estimates, and changes required for the procedure when sample sizes are small.

4.2 Procedure - Stage I

Let $\{X(t)\}$, and $\{Y(t)\}$ represent the two series influenced by the common series $(\{Z_1(t)\},\{Z_2(t)\},......,\{Z_q(t)\})$, $t=1,2,...,N$. The series under consideration are assumed to be sampled at discrete time intervals from underlying processes that are stochastic, weakly stationary, and continuous. For each series construct the Fourier transforms at the Fourier frequencies as,

$$W_x(\lambda)=\frac{1}{\sqrt{2\pi N}}\sum_{t=1}^{N}x(t)e^{i\lambda t} \qquad [4.1]$$

$$W_y(\lambda)=\frac{1}{\sqrt{2\pi N}}\sum_{t=1}^{N}y(t)e^{i\lambda t} \qquad [4.2]$$

$$W_j(\lambda)=\frac{1}{\sqrt{2\pi N}}\sum_{t=1}^{N}z_j(t)e^{i\lambda t} \quad j=1,2.....,q \qquad [4.3]$$

where the Fourier frequencies are given by $\lambda=\frac{2\pi k}{N}$, $k=0,1,2,...,[\frac{N}{2}]$, where $[x]$ is an integer not greater than x.

From the Finite Fourier transforms the periodogram ordinates for the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z_j(t)\}$, $j=1,2,...q$ are constructed as,

$$I_x(\lambda)=W_x(\lambda)\overline{W_x(\lambda)} \qquad [4.4]$$

$$I_y(\lambda)=W_y(\lambda)\overline{W_y(\lambda)} \qquad [4.5]$$

$$I_j(\lambda)=W_j(\lambda)\overline{W_j(\lambda)}, \quad j=1,2,.....,q \qquad [4.6]$$

and the corresponding cross periodogram ordinates for distinct pairs of series are constructed as,

$$I_{xy}(\lambda)=W_x(\lambda)\overline{W_y(\lambda)} \qquad [4.7]$$

$$I_{xj}(\lambda)=W_x(\lambda)\overline{W_j(\lambda)} \quad \text{and} \qquad [4.8]$$

$$I_{yj}(\lambda)=W_y(\lambda)\overline{W_j(\lambda)} \quad j=1,2,...,q \qquad [4.9]$$

The procedure for obtaining preliminary estimates will be explained for the simplest case when we have only one influencing series, that is $q = 1$. The procedure with slight modifications can then be extended for the case $q > 1$. Let $\{Z(t)\}$ be the series influencing the series $\{X(t)\}$, and $\{Y(t)\}$. Let B_0 be a band centered at frequency ω_1 and containing m Fourier frequencies. Let $\{\lambda_{0,1}, \lambda_{0,2}, \ldots\ldots, \lambda_{0,m}\}$ be the m frequencies in the band B_0. Let B_L be another band to the left of band B_0, containing m Fourier frequencies $\{\lambda_{L,1}, \lambda_{L,2}, \ldots\ldots, \lambda_{L,m}\}$. Similarly, let B_R be a band to the right of band B_0, also containing m Fourier frequencies $\{\lambda_{R,1}, \lambda_{R,2}, \ldots\ldots, \lambda_{R,m}\}$. Note that the Fourier frequencies in all the three bands are distinct. The purpose of defining these three bands is to find a preliminary estimate at the center of the band B_0. The following three expressions summarize the notation used in this paragraph,

$$B_0 = \{\lambda_{0,1}, \lambda_{0,2}, \ldots\ldots, \lambda_{0,m}\} \quad [4.10]$$

$$B_L = \{\lambda_{L,1}, \lambda_{L,2}, \ldots\ldots, \lambda_{L,m}\} \quad [4.11]$$

$$B_R = \{\lambda_{R,1}, \lambda_{R,2}, \ldots\ldots, \lambda_{R,m}\} \quad [4.12]$$

Using the Fourier transforms in the bands B_0, B_L, and B_R we then find the smoothed periodogram and cross periodogram ordinates at each of the Fourier frequencies $\lambda_{0,j}$ $j = 1,2,\ldots..,m$ as,

$$S_{aa}(\lambda_{0,j}) = \frac{I_{aa}(\lambda_{L,j}) + I_{aa}(\lambda_{0,j}) + I_{aa}(\lambda_{R,j})}{3} \quad a = X, Y, Z \quad [4.13]$$

$$S_{ab}(\lambda_{0,j}) = \frac{I_{ab}(\lambda_{L,j}) + I_{ab}(\lambda_{0,j}) + I_{ab}(\lambda_{R,j})}{3} \quad a, b = X, Y, Z \quad a \neq b \quad [4.14]$$

We then find the partial cross spectral density of the series $\{X(t)\}$, and $\{Y(t)\}$ adjusted for the influences of the common series $\{Z(t)\}$ as given below,

$$\hat{g}_{xy.z}(\lambda_{0,j}) = S_{xy}(\lambda_{0,j}) - S_{xz}(\lambda_{0,j}) S_{zz}(\lambda_{0,j})^{-1} S_{zy}(\lambda_{0,j}) \quad [4.15]$$

The conventional way of adjusting the series $\{X(t)\}$, and $\{Y(t)\}$ for the delay due to the common influencing series $\{Z(t)\}$ is by using the method of linear filtering (see chapter 2, section 2.9 for details). The method of linear filtering involves expressing one series as a linear approximation of another series. Let L_1, and L_2 be two linear filters of the series $\{Z(t)\}$ that approximate the series $\{X(t)\}$ and $\{Y(t)\}$ respectively. Let $\{\hat{X}(t)\}$ and $\{\hat{Y}(t)\}$ be the best approximations of the series $\{X(t)\}$ and $\{Y(t)\}$ respectively. From these approximations construct the residual processes as follows,

$$\{V_x(t)\} = \{X(t)\} - \{\hat{X}(t)\} \qquad [4.16]$$

$$\{V_y(t)\} = \{Y(t)\} - \{\hat{Y}(t)\} \qquad [4.17]$$

From the residual processes $\{V_x(t)\}$ and $\{V_y(t)\}$ the cross spectral density is constructed. This is also termed the partial cross spectral density of $\{X(t)\}$ and $\{Y(t)\}$ adjusted for $\{Z(t)\}$. Note that the filtering is done for the entire series not just the individual data points. For more details on filtering the reader is referred to chapter 2, section 2.10. One easily can see that computing the partial cross spectral density as given in expression [4.15] is a much simpler way than computing it using the linear filtering method.

A preliminary estimate of τ, the partial group delay, is the value of τ that maximizes the function

$$\hat{q}(\tau) = \left| \frac{1}{m} \sum_{B_0} \hat{g}_{xy.z}(\lambda) e^{-i\tau\lambda} \right|^2 \qquad [4.18]$$

where Σ represents the sum over all frequencies in the band B_0. Represent this preliminary estimate of the group delay by $A(\omega_1)$ where ω_1 is the average of the Fourier frequencies in band B_0, that is, $\omega_1 = \frac{1}{m}\sum_{j=1}^{m} \lambda_{0,j}$. The estimate for τ in expression [4.18] can be explained intuitively, as the highest peak in the partial cross spectral density function. Thus, if the series were realigned by shifting $\{Y(t)\}$ by τ units, the coherence between $\{X(t)\}$ and $\{Y(t)\}$ would be maximum.

The process of constructing bands B_0, B_L, and B_R each containing m distinct Fourier frequencies, forming smoothed periodogram and cross periodgram ordinates, forming the partial cross spectral density, and maximizing expression [4.18] is repeated till we exhaust all the data points. Thus, at the end of stage-I we have n distinct preliminary estimates $A(\omega_1), A(\omega_2), \ldots\ldots, A(\omega_n)$ at distinct frequencies $\omega_1, \omega_2, \omega_3, \ldots, \omega_n$ respectively. Note that each of these estimates is formed from three sets of m distinct Fourier frequencies.

4.3 Procedure - Stage II

The preliminary estimates $A(\omega_1), A(\omega_2), \ldots\ldots, A(\omega_n)$ will be used in the method by Ramsey and Foutz [1992]. Their method uses transforming and modeling technique to obtain uniformly minimum variance unbiased estimates for the mean of the preliminary values found in Stage I. Since the preliminary estimates are highly variable and the assumption of normality may be violated, we need to find a transformation f^{-1}, such that each of the transformed variables, $B(\omega_j) = f^{-1}(A(\omega_j)), j = 1,2\ldots., n$ follows a linear model of the form,

$$\begin{pmatrix} B(\omega_1) \\ \vdots \\ B(\omega_n) \end{pmatrix} = W\underline{\beta} + \underline{\varepsilon} \qquad [4.19]$$

where W is a known matrix, β is a vector of unknown parameters and ε is a vector of independent normal random errors with mean 0 and variance σ^2.

In the simulation studies of chapter 5, we have used the Box-Cox transformation to find f^{-1}. This method simultaneously uses the modeling and transforming technique to identify the transforming parameter λ and the unknown parameters β. The Box-Cox transformation is given by,

$$B(\omega_j) = \frac{A(\omega_j)^{\lambda} - 1}{\lambda} \qquad [4.20]$$

and one possible model for the transformed variable $B(\omega_j)$, $j = 1,2....,n$, could be the polynomial model,

$$B(\omega_j) = \beta_0 + \beta_1\omega_j + \beta_2\omega_j^2 + ...+\beta_s\omega_j^s + e_j \quad j = 1,2,.....,n \qquad [4.21]$$

where the e_j's are random and normally distributed. Note that the transformation obtained should be invertible, so that the $A(\omega_j)$'s can be written as a Taylor series expansion in $B(\omega_j)$ $j = 1,2....,n$. That is,

$$A(\omega_j) = f(B(\omega_j)) \qquad j = 1,2....,n \qquad [4.22]$$

$$A(\omega_j) = f(0) + \sum_{h=1}^{\infty} \frac{1}{h!} f^{(h)}(0) B(\omega_j)^h \quad j = 1,2,...,n \qquad [4.23]$$

where $f^{(h)}(0)$ is the h^{th} derivative evaluated at zero. Finally, if expectation can be taken inside the summation then the mean $\theta(\omega_j)$, for the preliminary estimate $A(\omega_j)$ $j = 1,2....,n$ is a function of Gaussian moments,

$$\theta(\omega_j) = E[A(\omega_j)] = f(0) + \sum_{h=1}^{\infty} \frac{1}{h!} f^{(h)}(0) E\left[B(\omega_j)^h\right] \quad j = 1,2,...,n \qquad [4.24]$$

and its estimate is given by

$$\hat{\theta}(\omega_j) = E[A(\omega_j)] = f(0) + \sum_{h=1}^{\infty} \frac{1}{h!} f^{(h)}(0) T_h \quad j = 1,2,\ldots,n \qquad [4.25]$$

where T_h is an UMVU estimate of $E\left[B(\omega_j)^h\right]$ and is based on **b** the least square estimator of β and S^2, the error sum of squares. Thus T_h for odd moments, that is $E\left[B(\omega_j)^{2h+1}\right]$, is given by,

$$T_{2h+1} = \sum_{k=0}^{h} \frac{(2h+1)!}{(2k+1)!(h-k)!} \left(\mathbf{W}_j \mathbf{b}\right)^{2k+1} \left[\frac{S^2(1-\xi)}{4}\right]^{h-k} \frac{\Gamma(\frac{\upsilon}{2})}{\Gamma(\frac{\upsilon}{2}+h-k)} \qquad [4.26]$$

And T_h for even moments, that is $E\left[B(\omega_j)^{2h}\right]$, is given by,

$$T_{2h} = \sum_{k=0}^{h} \frac{(2h)!}{(2k)!(h-k)!} \left(\mathbf{W}_j \mathbf{b}\right)^{2k} \left[\frac{S^2(1-\xi)}{4}\right]^{h-k} \frac{\Gamma(\frac{\upsilon}{2})}{\Gamma(\frac{\upsilon}{2}+h-k)} \qquad [4.27]$$

where $\mathbf{W}_j$ is the jth row of matrix W, $\xi = \mathbf{W}_j (W'W)^{-1} \mathbf{W}_j'$, $\Gamma(x) = (x-1)!$ is the Gamma function, and $\nu = (\text{number of rows} - \text{number of columns})$ of matrix W. With this we complete stage two of the procedure and in the following paragraph we discuss the ideal condition necessary to get UMVU estimate.

4.4 Ideal Condition

The condition necessary for $\hat{\theta}(\omega_j)$ in expression [4.25] to be an UMVU estimate for the mean of $A(\omega_j)$, is that the periodogram and cross periodogram ordinates at each Fourier frequency should be independent of periodogram and cross periodogram ordinates at every other Fourier frequency. This condition is achieved if $\{X(t)\}$, $\{Y(t)\}$, and $\{Z_j(t)\}$, $j = 1,2,\ldots.,q$ are Gaussian white noise processes and the channels of $\mathbf{K}(t) = (X(t), Y(t), Z_1(t), \ldots. Z_q(t))$, $t = 1.2,..,N$ are not cross-correlated. Note that a white noise process is a weakly stationary stochastic process with continuous spectra and constant spectral density function (see Koopmans[1974], pp. 50). This condition is also satisfied asymptotically as the sample size $N \to \infty$ for processes that are linear (see Hannan [1970], pp. 248-249).

4.5 Procedure For $q > 1$

When we have more than one series influencing $\{X(t)\}$, and $\{Y(t)\}$, the above procedure needs to be modified slightly. Let $\{Z_j(t)\}$, $j = 1,2,....,q$, be the q influencing series. When $q = 1$, that is, when there was only one influencing series we formed one band to the left and right of the center band, such that, in all we had three bands. When we have more than one influencing series we construct $p = \left[\frac{q}{2}\right] + 1$ bands where [x] is an integer not greater than x. Thus, to the left of band B_0, we construct bands $B_{L_1}, B_{L_2} \ldots\ldots, B_{L_p}$ each containing m Fourier frequencies and to the right of band B_0, we have bands $B_{R_1}, B_{R_2} \ldots\ldots, B_{R_p}$ each containing m Fourier frequencies.

Each of the series involved is Fourier transformed at Fourier frequencies. Periodogram ordinates are formed for each of the series and cross periodogram ordinates are formed for each pair of series. The smoothed periodogram and cross periodogram ordinates at each of the frequency $\lambda_{0,j}$, $j = 1,2,\ldots,m$ are then constructed as follows,

$$S_{aa}(\lambda_{0,j}) = \frac{1}{2p+1} \sum_d I_{aa}(\lambda_{d,j}) \qquad a = X, Y, Z_1 \ldots Z_q \qquad [4.28]$$

$$d = L_1, \ldots, L_p, 0, R_1, \ldots, R_p$$

$$S_{ab}\left(\lambda_{0,j}\right)=\frac{1}{2p+1}\sum_{d} I_{ab}\left(\lambda_{d,j}\right) \qquad a,b=X,Y,Z_1,\ldots Z_q \qquad a\neq b$$

$$d \;=L_1,\ldots,L_p,0,R_1,\ldots,R_p \qquad [4.29]$$

Construct at frequency $\lambda_{0,j}$, $j=1,2,\ldots,m$ $\mathbf{M}_{zz}(\lambda)$, the pxp matrix containing the smoothed periodogram ordinates for $\{Z_j(t)\}$, $j=1,2,\ldots.,q$, and cross periodogram ordinates for $\{Z_j(t)\}$, and $\{Z_k(t)\}$, $j=1,2,\ldots.,q$ $j\neq k$; 1xp row vector $\mathbf{R}_{XZ}(\lambda)$ of the smoothed cross periodograms for the $\{X(t)\}$, and $\{Z_j(t)\}$, $j=1,2,\ldots.,q$ and px1 column vector $\mathbf{C}_{ZY}(\lambda)$ of the smoothed cross periodograms for the $\{Y(t)\}$, and $\{Z_j(t)\}$, $j=1,2,\ldots.,q$. Construct the estimates of the partial cross spectral density for the various frequencies as given below,

$$\hat{g}_{xy.z}\left(\lambda_{0,j}\right)=S_{xy}\left(\lambda_{0,j}\right)-\mathbf{R}_{xz}\left(\lambda_{0,j}\right)\mathbf{M}_{zz}\left(\lambda_{0,j}\right)^{-1}\mathbf{C}_{zy}\left(\lambda_{0,j}\right) \quad j=1,2\ldots,m \qquad [4.30]$$

A preliminary estimate of τ, the partial group delay is the value of τ that maximizes the function given in expression [4.18]

Represent this preliminary estimate of the partial group delay by $A(\omega_1)$ where $\omega_1=\frac{1}{m}\sum_{j=1}^{m}\lambda_{0,j}$ and repeat the process at different frequencies $\omega_2,\omega_3,\ldots\ldots\omega_n$ to give us the preliminary estimates $A(\omega_2),\ldots\ldots,A(\omega_n)$, respectively. Note that each of these estimates is formed from $(2\times p+1)$ sets of m distinct Fourier frequencies. These preliminary estimates are then used in the stage-II of the procedure. Note that Stage II of the procedure remains identical even when we have more than one influencing series.

4.6 Confidence Interval Estimation

Under the ideal condition, a 100(1-α)% confidence interval for $\theta(\omega_j)$ is constructed as follows: Stage-II of the procedure uses transforming and modeling technique to transform the preliminary estimates $A(\omega_j)$ to $B(\omega_j)$, $j = 1,2,\ldots,n$ Let ,

$$\mu = E\left[B(\omega_j)\right] \qquad [4.31]$$

and $$\sigma^2 = V\left[B(\omega_j)\right] \qquad [4.32]$$

Since the h^{th} moment of $B(\omega_j)$ is a function of μ and σ^2, we can write

$$E\left[B(\omega_j)^h\right] = M_h\left(\mu,\sigma^2\right) \qquad [4.33]$$

The sequence $\left\{M_h\left(\mu,\sigma^2\right)\right\}$ can be generated recursively as,

$$M_1\left(\mu,\sigma^2\right) = \mu \qquad [4.34]$$

$$M_2\left(\mu,\sigma^2\right) = \mu^2 + \sigma^2 \qquad [4.35]$$

$$M_h\left(\mu,\sigma^2\right) = \mu\, M_{h-1}\left(\mu,\sigma^2\right) + \sigma^2 (h-1) M_{h-2}\left(\mu,\sigma^2\right) \quad \text{if } h > 2 \qquad [4.36]$$

Thus, expression [4.24] can be written in terms of the moments as,

$$\theta(\omega_j) = E\left[A(\omega_j)\right] = f(0) + \sum_{h=1}^{\infty} \frac{1}{h!} f^{(h)} M_h\left(\mu,\sigma^2\right) \quad j = 1,2,\ldots,n \qquad [4.37]$$

Let g_1 and g_2 be such that, g_1 follows a chi-squared distribution with 1 degree of freedom and g_2 follows a chi-squared with υ degrees of freedom and are defined as follows,

$$g_1 = \frac{\left(\mathbf{W}_j\mathbf{b} - \mu\right)^2}{\xi\sigma^2} \qquad [4.38]$$

$$g_2 = \frac{S^2}{\sigma^2} \qquad [4.39]$$

Next, pick two constants c_1, and c_2 such that,

$$P(g_1 < c_1)P(g_2 > c_2) = 1 - \alpha \qquad [4.40]$$

In Ramsey and Foutz [1992] it is shown that a 100(1-α)% joint confidence set for (μ, σ^2) is given by,

$$E = \left\{ (\mu, \sigma^2):\ \sigma^2 < \frac{S^2}{c_2},\ \ \mathbf{W}_j\mathbf{b} - \sqrt{\xi c_1 \sigma^2} < \mu < \mathbf{W}_j\mathbf{b} + \sqrt{\xi c_1 \sigma^2} \right\} \qquad [4.41]$$

Finally, the set E can be mapped into the 100(1-α)% confidence interval of values for $\theta(\omega_j)$ as given by

$$\left\{ \theta(\omega_j) : (\mu, \sigma^2) \in E \right\}$$

where $\theta(\omega_j)$ is given by expression [4.37]

4.7 Finding Preliminary Estimates Using The Slope Method

In section 4.2 a method for finding preliminary estimates for partial group delay was described. In the present section we describe one more method for finding the preliminary estimates and we will refer to this method as the slope method. In the present work this method has not been extensively investigated by simulation studies and hence we refrain from making concrete comments regarding the merits of this procedure. However this is one idea for future research and we demonstrate the use of this procedure using some naturally occurring time series in chapter 6.

For the three series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ we obtain the Fourier transforms at Fourier frequencies, the periodogram ordinates and the cross periodogram ordinates as given by expressions 4.1 - 4.9 in section 4.2. Next we construct a band say B_l containing m Fourier frequencies $\{\lambda_{1,1}, \lambda_{1,2}, \ldots, \lambda_{1,m}\}$. Thus, a total of $M = \left[\frac{N}{2m}\right]$ such bands can be formed. Note that each of the bands contains m distinct Fourier frequencies and [x] is an integer not greater than x. Let $\{B_1, B_2, \ldots, B_M\}$ represent the M bands each containing m distinct Fourier frequencies.

For each of the bands B_j, $j = 1,2,..M$, smoothed periodogram and cross periodogram ordinates are constructed as follows :

$$S_{aa}(\lambda_L) = \frac{1}{m}\sum_{j=1}^{m} I_{aa}(\lambda_{L,j}) \quad a = X,Y,Z;\ L = 1,2,..,M \qquad [4.42]$$

$$S_{ab}(\lambda_L) = \frac{1}{m}\sum_{j=1}^{m} I_{ab}(\lambda_{L,j}) \quad a,b = X,Y,Z;\ a \neq b;\ L = 1,2,\ldots,M \qquad [4.43]$$

From the smoothed periodogram, and cross periodogram ordinates the estimated partial cross spectral density for the series $\{X(t)\}$ and $\{Y(t)\}$ adjusted for the effect of the series $\{Z(t)\}$ is obtained for each of the bands B_j, $j = 1,2,..M$ as follows,

$$\hat{g}_{xy.z}(\lambda_L) = S_{xy}(\lambda_L) - S_{xz}(\lambda_L) S_{zz}(\lambda_L)^{-1} S_{zy}(\lambda_L) \qquad L = 1,2,..M \qquad [4.44]$$

and for each of the bands B_j, $j = 1,2,..M$ the estimated partial phase is obtained as the argument of the estimated partial spectral density and is represented as follows,

$$\hat{\theta}_{xy.z}(\lambda_L) = \arg\left(\hat{g}_{xy.z}(\lambda_L)\right) = -\arctan\left[\frac{\operatorname{Im}\hat{g}_{xy.z}(\lambda_L)}{\operatorname{Re}\hat{g}_{xy.z}(\lambda_L)}\right] \qquad [4.45]$$

Finally, preliminary estimates for the partial group delay at frequency say λ_0 are obtained as follows,

$$\hat{\tau}(\lambda_0) = \frac{\hat{\theta}_{xy.z}(\lambda_{L+1}) - \hat{\theta}_{xy.z}(\lambda_L)}{\lambda_{L+1} - \lambda_L} \qquad L = 1,2,\ldots(M-1) \qquad [4.46]$$

where $\lambda_0 = \dfrac{\lambda_L + \lambda_{L+1}}{2}$.

This method thus results in $n = \left[\frac{M}{2}\right]$ preliminary estimates for the partial group delay. These n estimates will be used in Stage II of the procedure as described in section 4.3 to obtain uniformly minimum variance estimates for the mean of the preliminary values. Naively comparing the two procedures it appears that the present method is computationally simpler to use. Also for a bandwidth of say m the present method yields more preliminary estimates than the method of section 4.2. But as stated at the beginning of this section we refrain from making any conclusions about the merit of this procedure as no simulation studies have been conducted to test this procedure .

4.8 Procedure For Small Samples

When sample sizes are small, the procedure of Ramsey and Foutz [1992] needs slight modification because the condition of independence of periodograms and cross periodograms at each of the Fourier frequencies is violated. It is known that for a white noise process the periodogram ordinates and cross periodogram ordinates at the Fourier frequencies are independent. Hence, one way to satisfy the ideal condition is to prewhiten the series. The purpose of 'prewhitening' is to make the series nearly white noise, thereby satisfying the ideal condition. In the context of regression analysis the process of prewhitening can be compared to the process of transforming the data, such that the assumption of normality holds.

To prewhiten the multivariate process $\mathbf{F}(t)=\left(\{X(t)\},\{Y(t)\},\{Z_1(t)\},\ldots\ldots,\{Z_q(t)\}\right)$ construct a linear filter such that the $(q+2)$ channels of $\mathbf{F}(t)$ are transformed to a $(q+2)$ dimensional prewhitened series $\mathbf{G}(t)=\left(\{O(t)\},\{P(t)\},\{G_1(t)\},\ldots\ldots,\{G_q(t)\}\right)$. Let the matrix of the transfer function of the linear filter be represented by $\mathbf{B}(\lambda)$. At each Fourier frequency $\lambda=\dfrac{2\pi k}{N}, k=0,1,\ldots\left[\frac{N}{2}\right]$ construct $(q+2)$ dimensional matrices $I_G(\lambda)$ of periodogram and cross periodogram ordinates for the prewhitened multivariate series $\mathbf{G}(t)$. Since $\mathbf{G}(t)$ is white noise, the matrices $I_G(\lambda)$ at each Fourier frequency will be independent of the matrices at every other Fourier frequency. The matrices $I_G(\lambda)$ are re-transformed as,

$$I_T(\lambda) = \mathbf{B}(\lambda)^{-1} I_G(\lambda) \overline{\mathbf{B}(\lambda)^{-1}} \qquad [4.47]$$

where $\mathbf{B}(\lambda)^{-1}$ is the inverse of the complex transfer function matrix $\mathbf{B}(\lambda)$ and $\overline{\mathbf{B}(\lambda)^{-1}}$ is the transpose of its complex conjugate. Moreover the transformed matrices $I_T(\lambda)$ are also independent matrices at the Fourier frequencies. The procedure for estimating the partial group delay for small sample is carried out exactly as described in sections [4.2] and [4.3] except that the transformed matrices $I_T(\lambda)$ constructed for the multivariate series $\mathbf{G}(t)$ are used instead of the matrices $I(\lambda)$ for the multivariate series $\mathbf{F}(t) = (\{X(t)\},\{Y(t)\},\{Z_1(t)\},......,\{Z_q(t)\})$.

One way of prewhitening the series could be to fit a vector autoregressive model of order p. For example, for the case $q = 1$, that is, when there is only one influencing series $\{Z(t)\}$, we could prewhiten the series using a vector AR(1) process as follows,

$$\begin{pmatrix} X(t) \\ Y(t) \\ Z(t) \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} X(t-1) \\ Y(t-1) \\ Z(t-1) \end{pmatrix} + \begin{pmatrix} g_1(t) \\ g_2(t) \\ g_3(t) \end{pmatrix} \qquad [4.48]$$

The above expression can be rewritten as,

$$\mathbf{F}(t) = \mathbf{A}\mathbf{F}(t-1) + \mathbf{G}(t) \qquad [4.49]$$

rearranging terms we get,

$$\mathbf{G}(t) = \mathbf{F}(t) - \mathbf{A}\mathbf{F}(t-1) \qquad [4.50]$$

$$\mathbf{G}(t) = (1 - \mathbf{A}\mathrm{B})\mathbf{F}(t) \qquad [4.51]$$

The transfer function for the AR(1) process is given by ,

$$\mathbf{B}(\lambda) = 1 - \mathbf{A}e^{i\lambda} \qquad [4.52]$$

and the elements of the matrix **A** can be found by the usual technique of least squares.

Chapter 5
Application Of The Proposed Technique

5.1 Introduction

In chapter 4 the theoretical concepts of the proposed procedure for estimating partial group delay were discussed. Our objective in this chapter is fourfold : (i) With the use of simulated data we will demonstrate how this procedure could be used to estimate the partial group delay. (ii) On the same data sets we will apply the procedure by Zhang and Foutz [1989] to estimate the partial group delay. As mentioned in the previous chapters the procedure by Zhang and Foutz [1989] is the only other procedure available for estimating partial group delay. (iii) Next we compare the results of our procedure to the results of the procedure by Zhang and Foutz [1989]. (iv) Lastly we proceed to justify the need for the proposed procedure by establishing the fact that for certain situations our procedure is much better than that of Zhang and Foutz [1989].

The above objectives are demonstrated by means of two simulation studies. We consider the simplest case when q=1, that is, there is only one series $\{Z(t)\}$ which influences the series $\{X(t)\}$, and $\{Y(t)\}$. In these studies the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ were simulated using theoretical models in the time domain. The intent of simulating the series in the time domain was to demonstrate how this methodology could work in practice for real data sets. Using Fourier transforms the series were then converted to the frequency domain. The series $\{X(t)\}$, and $\{Y(t)\}$ were constructed as an additive effect of three components, namely, a signal, a noise and the influencing series $\{Z(t)\}$. The signal component of the series $\{Y(t)\}$ was the same as that of $\{X(t)\}$ except that it was delayed in time. Further it was assumed that the signal and the noise components for each of the series were uncorrelated and the noise components for the series $\{X(t)\}$, and $\{Y(t)\}$ were also uncorrelated.

There are numerous time domain models which one can employ to construct the signal. For instance, a simple choice could be an autoregressive model of order p [AR(p)], a moving average model of order q [MA(q)], a mixed autoregressive and moving average model of order p, q [ARMA(p, q)] or simply a white noise process. The same choice of models would also be available for the influencing series $\{Z(t)\}$. Models for different simulation studies can thus be constructed by using various combinations of the models for the signal and the influencing series. It would be practically impossible to try the procedure on each and every combination of the models and as such will be beyond the scope of this dissertation. We thus have selected only two interesting models for the simulation studies.

The following material of this chapter is divided into four sections. In the first two sections we discuss at length the two simulation studies that were conducted to demonstrate the procedure. In the next section we compare the results of the proposed procedure to the results of the Zhang and Foutz [1989] procedure and demonstrate for the simulated cases that the use of the proposed procedure will yield better results than the Zhang and Foutz [1989] method. Finally in the last section we give justification for the choice of the bandwidth 'm'.

5.2 Simulation Study I

In this study data were simulated for a theoretical partial group delay of 2, that is, after adjusting the two series for the effect of the series $\{Z(t)\}$ the series $\{X(t)\}$ led the series $\{Y(t)\}$ by 2 units. In this study the signal was an MA(1) process with parameter $\theta = 0.8$. The series $\{Z(t)\}$ was constructed as an AR(1) process with parameter $\phi = 0.5$. The material in this section is divided into three sub-sections. In the first sub-section details of the model that was used to simulate the data are given. In the second sub-section details of how the procedure was applied and estimates of partial group delay are presented and finally in sub-section three we compare the results of the proposed procedure to the results of the Zhang and Foutz [1989] method.

5.2.1 Model For Simulated Data

The series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ were constructed as,

$$Z(t) = .5Z(t-1) + \gamma(t) \quad [5.1]$$

$$X(t) = .8Z(t+1) + \varepsilon_x(t) \quad [5.2]$$

$$Y(t) = .6Z(t+2) + \varepsilon_y(t) \quad [5.3]$$

where $\gamma(t)$ was a white noise process with mean 0 and variance 0.06 and was not correlated with the signal $\{Z(t)\}$. The residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ were constructed as,

$$\varepsilon_x(t) = S(t) + \alpha(t) \quad [5.4]$$

$$\varepsilon_y(t) = S(t+2) + \beta(t) \quad [5.5]$$

where $\{\alpha(t)\}$, and $\{\beta(t)\}$ were white noise processes with mean 0 and variance 0.4 and were uncorrelated with the signal process $\{S(t)\}$. Also the white noise process $\{\alpha(t)\}$

was uncorrelated with the white noise process $\{\beta(t)\}$. The signal process $\{S(t)\}$ was constructed as,

$$S(t) = \eta(t) + 0.8\eta(t-1) \qquad [5.6]$$

where $\{\eta(t)\}$ was a white noise process with mean 0 and variance 2.85. Expression [5.6] can also be written as,

$$S(t) = (1 + 0.8B)\eta(t) \qquad [5.7]$$

where B is the backward shift operator that takes $\eta(t)$ to $\eta(t-1)$. Since $\{\eta(t)\}$ is a white noise process with mean 0 and variance 2.85, its spectral density is given by,

$$f_\eta(\lambda) = \frac{\sigma_\eta^2}{2\pi} = \frac{2.85}{2\pi} \qquad [5.8]$$

The transfer function of the output process $\{S(t)\}$ can be constructed (see Koopmans [1974], pp. 166) as follows,

$$B(\lambda) = \left(1 + 0.8 * e^{i\lambda}\right) \qquad [5.9]$$

It can also be shown (see Koopmans [1974], pp. 90-91) that the spectral density of the output process $\{S(t)\}$ of a linear filter with input as the white noise process $\{\eta(t)\}$ is given by,

$$f_s(\lambda) = \left|1 + 0.8e^{i\lambda}\right|^2 f_\eta(\lambda) \qquad [5.10]$$

$$f_s(\lambda) = \left|1 + 0.8e^{i\lambda}\right|^2 \frac{2.85}{2\pi} \qquad [5.11]$$

The residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ are in fact series $\{X(t)\}$, and $\{Y(t)\}$ respectively adjusted for the effect of the common series $\{Z(t)\}$ and can be written in terms of the process $\{\eta(t)\}$ as follows,

$$\varepsilon_x(t) = (1 + 0.8B)\eta(t) + \alpha(t) \qquad [5.12]$$

$$\varepsilon_y(t) = (1 + 0.8B)\eta(t+2) + \beta(t) \qquad [5.13]$$

The output $\{O(t)\}$ of a linear filter L with input $\{I(t)\}$ and contaminated with noise $\{N(t)\}$ is written as follows,

$$O(t) = L(I(t)) + N(t) \qquad [5.14]$$

and the corresponding spectral density for the output $\{O(t)\}$ is given by (see Koopmans [1974], pp. 145-146),

$$f_O(\lambda) = |B(\lambda)|^2 f_I(\lambda) + f_N(\lambda) \qquad [5.15]$$

where $B(\lambda)$ is the transfer function of the linear filter L. Expressions [5.12] and [5.13] are similar to expression [5.14] and using expression [5.15] the theoretical spectral densities for the residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ can be obtained. In fact, the spectral densities are identical as the white noise processes $\{\alpha(t)\}$, and $\{\beta(t)\}$ have the same variances, namely 4, the same transfer function and are given by,

$$f_{\varepsilon_x\varepsilon_x}(\lambda) = f_{\varepsilon_y\varepsilon_y}(\lambda) = \frac{1}{2\pi}\left[2.85\left|1+.8e^{i\lambda}\right|^2 + 0.4\right] \qquad [5.16]$$

To obtain the cross spectral density between the residual processes $\{\varepsilon_x(t)\}$ and $\{\varepsilon_y(t)\}$ we first obtain their cross-covariances at lag τ as follows,

$$C(\tau) = E\left[\varepsilon_x(t)\varepsilon_y(t+\tau)\right] \qquad [5.17]$$

using expanded versions of expressions [5.12] and [5.13] in expression [5.17] we get,

$$C(\tau) = \left\{\begin{array}{l} E[\eta(t)\eta(t+\tau+2)] + 0.8 * E[\eta(t)\eta(t+\tau+1)] + \\ 0.8 * E[\eta(t-1)\eta(t+\tau+2)] + 0.64 * E[\eta(t-1)\eta(t+\tau+1)] + \\ \text{cross product terms for signal and noise} \end{array}\right\} \qquad [5.18]$$

The cross product terms are zero as the signal and noise processes are uncorrelated and $C(\tau)$ is as given below,

$$C(\tau)=\begin{cases}\sigma_\eta^2 & \text{if } \tau=-2\\ 0.8*\sigma_\eta^2 & \text{if } \tau=-1\\ 0.8*\sigma_\eta^2 & \text{if } \tau=-3\\ 0.64*\sigma_\eta^2 & \text{if } \tau=-2\end{cases} \qquad [5.19]$$

The cross spectral density between the residual processes $\{\varepsilon_x(t)\}$ and $\{\varepsilon_y(t)\}$ is then given by,

$$f_{\varepsilon_x\varepsilon_y}(\lambda)=\frac{1}{2\pi}\sum_{\tau=-\infty}^{\infty}e^{-i\tau\lambda}C(\tau) \qquad -\pi<\lambda<\pi \qquad [5.20]$$

Using expression [5.19] and $\sigma_\eta^2=2.85$ in expression [5.20] we get,

$$f_{\varepsilon_x\varepsilon_y}(\lambda)=\frac{2.85}{2\pi}\left[(1+0.64)*e^{2i\lambda}+0.8*e^{i\lambda}+0.8*e^{3i\lambda}\right] \qquad [5.21]$$

$$f_{\varepsilon_x\varepsilon_y}(\lambda)=\frac{2.85*e^{2i\lambda}}{2\pi}\left[(1+0.64)+0.8*e^{-i\lambda}+0.8*e^{i\lambda}\right] \qquad [5.22]$$

$$f_{\varepsilon_x\varepsilon_y}(\lambda)=\frac{1}{2\pi}\left[2.85\left|1+.8e^{i\lambda}\right|^2\right]e^{i2\lambda} \qquad [5.23]$$

The phase for the residual processes or the partial phase for the series $\{X(t)\}$ and $\{Y(t)\}$ after adjusting for the series $\{Z(t)\}$ is obtained from the cross spectrum of the residuals as,

$$\theta_{\varepsilon_x\varepsilon_y}(\lambda)=\operatorname{Arg}\left(f_{\varepsilon_x\varepsilon_y}(\lambda)\right)=\operatorname{Arg}\left(e^{i2\lambda}\right) \qquad [5.24]$$

$$\theta_{\varepsilon_x\varepsilon_y}(\lambda)=2\lambda \qquad [5.25]$$

and the partial group delay is given by the derivative of the partial phase as follows,

$$\tau(\lambda)=\frac{d\theta_{\varepsilon_x\varepsilon_y}(\lambda)}{d\lambda}=2 \qquad [5.26]$$

As can be seen in the above expression the partial group delay is constant for all frequencies λ.

The corresponding coherence for the residual processes $\{\varepsilon_x(t)\}$ and $\{\varepsilon_y(t)\}$ is given as follows,

$$\sigma(\lambda)=\frac{\left|f_{\varepsilon_x\varepsilon_y}(\lambda)\right|}{\sqrt{f_{\varepsilon_x\varepsilon_x}(\lambda)f_{\varepsilon_y\varepsilon_y}(\lambda)}} \qquad [5.27]$$

$$\sigma(\lambda)=\frac{\left[2.85\left|1+.8e^{i\lambda}\right|^2\right]}{\left[2.85\left|1+.8e^{i\lambda}\right|^2+0.4\right]} \qquad [5.28]$$

and in the following table the partial coherences as given by expression [5.28] for some of the commonly used frequencies are presented ,

Table 5.2.1 Partial Coherences For Simulation Study I

Frequency	0	$\frac{\pi}{4}$	$\frac{\pi}{2}$	π
Par.Coherence	0.95848	0.95179	0.921167	0.22178

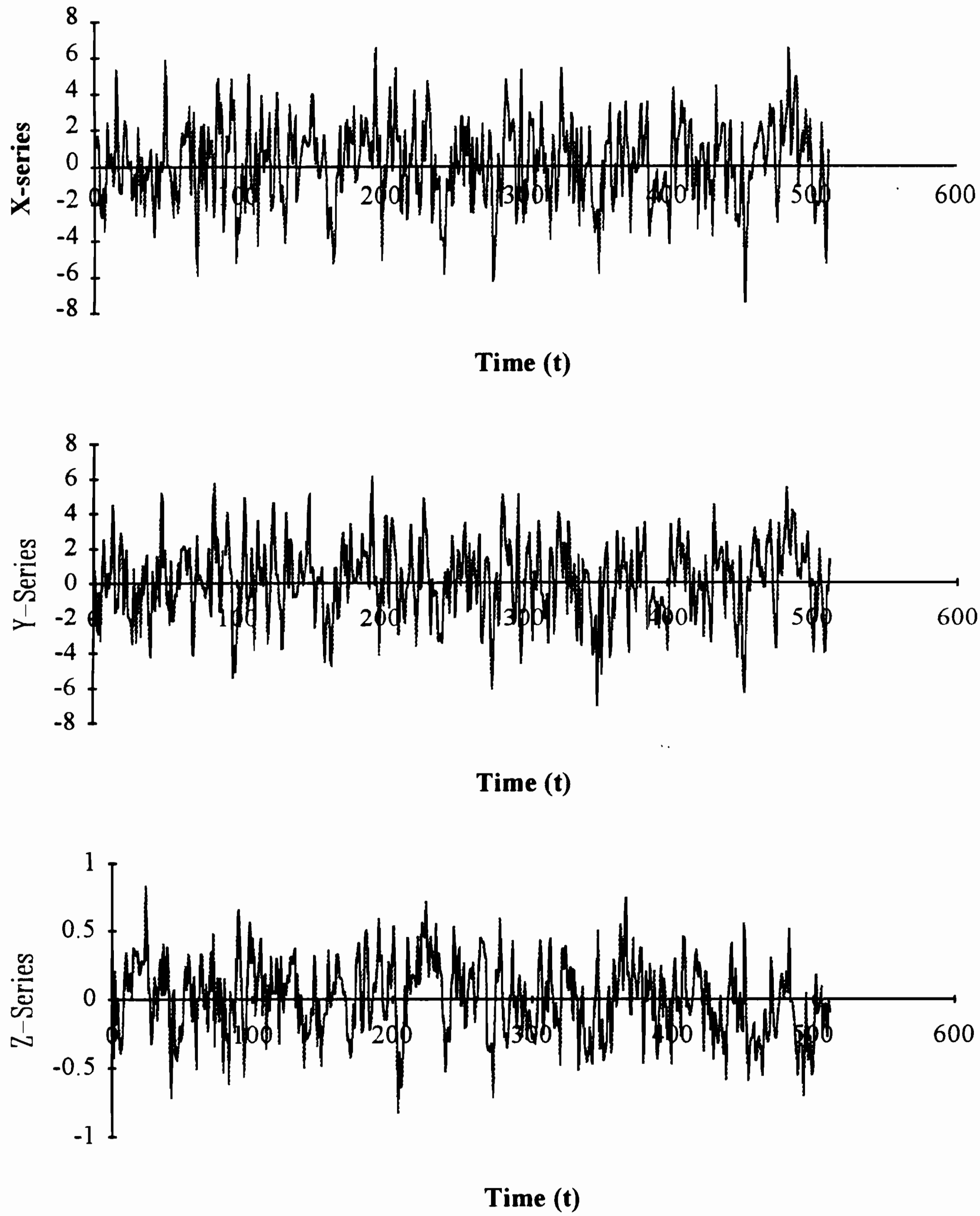
5.2.2 Results

Using the model of section [5.2.1] 100 data sets were simulated. The simulation was conducted on a personal computer using the statistical package SAS. It took approximately 20 minutes to generate 100 data sets. Each data set consisted of 512 observations. In the context of spectral analysis this can be considered as a small sample. In fact the literature does not specify what constitutes a large sample and what qualifies as a small sample. It seems to be an unwritten law that a data set of several 1000 observations would constitute a large sample and a data set of a few 100 observations would qualify as a small sample.

To give the reader a visual idea of the nature of the series, we have on the following two pages shown plots for two sets of simulated series. In the figures a separate graph for each of the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ are plotted versus time. The plots in the first figure are for data set number 1 and the one in the next figure are for data set number 25. Plots for the remaining 98 data sets can be generated easily using SAS. They are not included in this work so as to keep this work concise and brief.

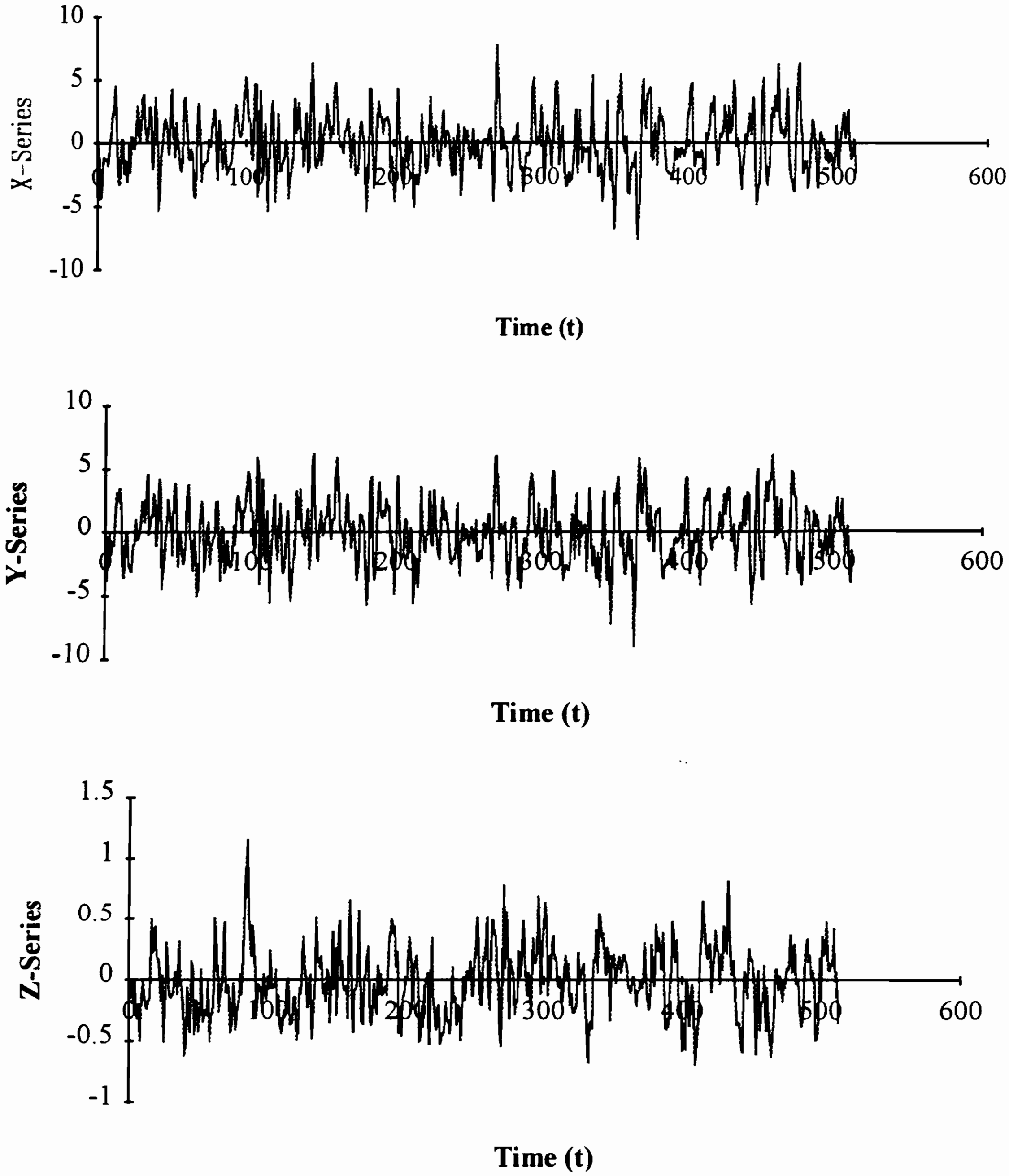
The series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ generated in the time domain were first tapered using a 5% taper and then converted to the frequency domain using a Fourier transform at the Fourier frequencies. We had a total of 256 distinct Fourier frequencies at which the periodogram and cross periodogram ordinates were constructed. The bands B_0, B_L, and B_R were formed. Each band contained eight Fourier frequencies, that is, $m = 8$. The choice of $m = 8$ will be justified in section [5.5]. Having constructed the bands B_0, B_L, and B_R smoothed periodogram and cross periodogram ordinates were formed. The partial cross spectral densities were obtained for the various frequencies and the function $\hat{q}(\tau)$ (see chapter 4, expression 4.18) was maximized to give the preliminary estimate for the partial group delay. For each data set ten preliminary estimates were obtained.

The preliminary estimates were used in the procedure by Ramsey and Foutz [1992] to obtain UMVU estimates for the mean of the preliminary estimate at the respective frequencies. Only the first eight preliminary estimates were used and the last two were discarded. The reasons for doing so are as follows : (i) Its relatively simple to model over a narrower band than over a wider band of frequencies. (ii) A low partial coherence implies that the magnitude of the relationship between $\{X(t)\}$ and $\{Y(t)\}$ is considerably low, thus leading to preliminary values for the partial group delay that are badly



Plot For Data Set # 1 - Study I

Figure 5.2.1



Plot For Data Set # 25 - Study I

Figure 5.2.2

estimated. Note that the partial coherences at the discarded frequencies $\lambda = 2.43596$ and 2.73049, were 0.75 and 0.55 respectively and the partial coherences at other frequencies were greater than 0.89. As a result of low partial coherence the preliminary values estimated at these frequencies were highly inflated and hence we were justified in eliminating them. This seems to be an acceptable practice by many statistician in the estimation of spectral parameters. For instance Hannan and Thomson [1981] make the following suggestion in the context of estimating phase :

> ***"Figs. 3 and 4 show $\hat{\phi}$ for two simulations and indicate how inaccurate is $\hat{\phi}$ for low σ. Undoubtedly better results would have been achieved had frequencies above about 1.25 radians been eliminated."***

Recall that the Ramsey and Foutz [1992] method involves modeling the preliminary estimates as functions of frequencies. The two modeling techniques employed were polynomial regression and spline models. In all there were six data sets that were modeled using regression models. These models were typically first, second, or third order models. Thirty-two data sets were modeled using spline regression. Here the knots varied from one through four. The remaining sixty-two data sets had to be transformed using the Box-Cox transformation technique. The Box-Cox parameter λ typically varied between zero and one.

In the following table, we present the partial group delay estimates using the proposed procedure and the Zhang and Foutz [1989] method at frequency λ=1.25786. The table also includes 90% confidence intervals for the two methods at the same frequency.

Table 5.2.2 : Partial Group Delay Estimates For Proposed Procedure And Zhang & Foutz [1989] Procedure - Study I

Data Set	Proposed Est	Proposed 90% C.I LBD	UBD	Zhang's Est.	Zhang's 90% C.I. LBD	UBD
1	2.173	-0.43	4.777	3.5	2.759387	4.240613
2	2.377	-0.878	20.274	0.6	-0.14061	1.340613
3	4.328	-1.007	36.253	-3.1	-3.84061	-2.35939
4	3.423	1.159	22.625	4.9	4.159387	5.640613
5	4.015	1.115	25.238	1.3	0.559387	2.040613
6	0.951	-0.775	2.678	1.1	0.359387	1.840613
7	1.282	-1.848	30.762	-3.2	-3.94061	-2.45939
8	1.783	0.227	3.338	1.9	1.159387	2.640613
9	3.433	0.554	6.312	3.4	2.659387	4.140613
10	2.328	0.077	22.147	-0.8	-1.54061	-0.05939
11	2.729	0.619	4.84	4.7	3.959387	5.440613
12	1.777	-0.754	24.19	-1.8	-2.54061	-1.05939
13	3.971	1.213	17.895	1	0.259387	1.740613
14	4.85	1.159	8.541	4.3	3.559387	5.040613
15	-0.625	-2.865	1.616	0.3	-0.44061	1.040613
16	0.561	-1.19	8.871	-1.3	-2.04061	-0.55939
17	2.351	-0.473	126.566	4.2	3.459387	4.940613
18	2.226	-2.744	53.755	4.1	3.359387	4.840613
19	5.875	2.198	9.553	6	5.259387	6.740613
20	1.432	-0.498	3.363	2.5	1.759387	3.240613
21	6.098	4.102	8.094	4.6	3.859387	5.340613
22	3.851	0.534	7.167	1.8	1.059387	2.540613
23	5.615	0.387	35.556	1.9	1.159387	2.640613
24	2.08	-1.389	22.172	-4.6	-5.34061	-3.85939
25	2.652	1.428	3.876	2.5	1.759387	3.240613
26	2.13	0.756	3.504	1.6	0.859387	2.340613
27	2.898	0.7	67.279	4.1	3.359387	4.840613
28	4.009	0.355	71.829	3.8	3.059387	4.540613
29	6.17	3.197	9.144	5.7	4.959387	6.440613
30	2.79	-0.919	24.898	-3.1	-3.84061	-2.35939
31	5.397	-0.182	34.251	2.6	1.859387	3.340613
32	-2.548	-4.961	-0.136	0.9	0.159387	1.640613
33	2.043	-0.101	27.782	-0.4	-1.14061	0.340613
34	1.617	-3.12	48.442	-2.9	-3.64061	-2.15939
35	2.147	-0.058	4.352	0.3	-0.44061	1.040613
36	-0.82	-3.2	1.56	-1	-1.74061	-0.25939
37	2.111	0.815	15.763	4.3	3.559387	5.040613
38	0.825	-0.937	15.356	-2.4	-3.14061	-1.65939
39	3.633	-0.094	32.04	1.4	0.659387	2.140613

Data Set	Proposed Est	Proposed 90% C.I LBD	UBD	Zhang's Est.	Zhang's 90% C.I. LBD	UBD
40	3.147	0.632	5.662	5.7	4.959387	6.440613
41	3.733	1.97	18.208	0.4	-0.34061	1.140613
42	1.867	-0.481	20	-1.5	-2.24061	-0.75939
43	2.125	0.717	3.533	2.6	1.859387	3.340613
44	2.929	1.03	4.828	4.8	4.059387	5.540613
45	-0.811	-1.348	-0.274	-0.8	-1.54061	-0.05939
46	2.294	-1.105	25.51	2.6	1.859387	3.340613
47	0.865	-0.704	17.065	1	0.259387	1.740613
48	1.44	-0.533	16.978	-1.3	-2.04061	-0.55939
49	2.114	-6.419	80.115	5.4	4.659387	6.140613
50	3.142	1.475	4.808	3.5	2.759387	4.240613
51	4.999	1.142	8.857	8	7.259387	8.740613
52	2.751	-0.056	28.229	1.5	0.759387	2.240613
53	-0.378	-6.398	5.642	-2.9	-3.64061	-2.15939
54	2.165	-1.162	28.293	-2.8	-3.54061	-2.05939
55	1.537	-0.051	3.125	1.6	0.859387	2.340613
56	2.055	-9.851	13.961	4.3	3.559387	5.040613
57	-1.05	-2.484	0.383	-0.1	-0.84061	0.640613
58	2.625	-0.739	35.908	1.4	0.659387	2.140613
59	1.934	-0.649	4.517	1.8	1.059387	2.540613
60	2.17	1.177	3.162	2.4	1.659387	3.140613
61	3.118	-2.984	57.533	-6.2	-6.94061	-5.45939
62	3.29	-1.696	19.629	3.8	3.059387	4.540613
63	1.467	-0.565	3.498	0.3	-0.44061	1.040613
64	1.192	-0.49	47.502	-0.1	-0.84061	0.640613
65	2.24	-0.879	46.947	1	0.259387	1.740613
66	3.353	2.434	6.835	2.2	1.459387	2.940613
67	2.921	1.106	19.098	4.7	3.959387	5.440613
68	7.091	1.228	38.764	7.5	6.759387	8.240613
69	1.756	-1.804	48.624	-2.6	-3.34061	-1.85939
70	2.961	1.052	4.87	2.6	1.859387	3.340613
71	1.616	-0.31	13.006	2.3	1.559387	3.040613
72	0.112	-2.543	2.767	-0.3	-1.04061	0.440613
73	4.009	1.642	17.08	4.4	3.659387	5.140613
74	3.291	-1.449	28.444	-1.7	-2.44061	-0.95939
75	0.277	-1.581	139.075	-2.4	-3.14061	-1.65939
76	0.795	-0.887	20.515	5.3	4.559387	6.040613
77	2.097	-0.88	31.548	2.8	2.059387	3.540613
78	-2.422	-6.381	1.536	-1.1	-1.84061	-0.35939
79	4.36	0.127	26.161	7.4	6.659387	8.140613
80	2.053	-0.918	32.456	-0.1	-0.84061	0.640613
81	3.952	0.577	22.341	1.6	0.859387	2.340613

Data Set	Proposed Est	Proposed 90% C.I LBD	UBD	Zhang's Est.	Zhang's 90% C.I. LBD	UBD
82	1.75	-3.992	7.492	2.8	2.059387	3.540613
83	2.142	1.146	10.066	2.1	1.359387	2.840613
84	2.653	-0.434	30.879	3.5	2.759387	4.240613
85	1.6	-0.794	37.022	-1.5	-2.24061	-0.75939
86	3.449	1.508	5.39	1.3	0.559387	2.040613
87	7.443	2.887	11.999	7.1	6.359387	7.840613
88	2.165	0.846	32.318	0.9	0.159387	1.640613
89	2.126	1.42	118.283	1.7	0.959387	2.440613
90	1.577	-0.194	9.485	1.5	0.759387	2.240613
91	2.325	-3.289	55.509	9.4	8.659387	10.14061
92	2.871	1.046	21.031	4.1	3.359387	4.840613
93	1.586	0.563	2.61	1.3	0.559387	2.040613
94	2.118	-0.086	297.733	-0.8	-1.54061	-0.05939
95	2.048	0.619	18.093	3.2	2.459387	3.940613
96	3.437	-0.048	33.464	3.4	2.659387	4.140613
97	1.952	-11.033	102.53	0.4	-0.34061	1.140613
98	1.533	-2.687	41.394	-2.4	-3.14061	-1.65939
99	-5.511	-10.79	-0.232	-6.4	-7.14061	-5.65939
100	1.965	-1.168	33.323	-0.1	-0.84061	0.640613

5.2.3 Comparison Of The Results

In this section we compare the results obtained using the proposed procedure to an existing procedure. This other procedure is by Zhang and Foutz [1989] and as mentioned previously is the only other procedure for estimating partial group delay. The summary statistics for the estimated partial group delay using the proposed method and the method by Zhang and Foutz [1989] at frequency λ=1.25786 are presented in the following table:

Table 5.2.3 **Simulation Study I - Statistics At Frequency λ=1.25786**

STATISTICS	PROPOSED METHOD	ZHANG'S METHOD
MEAN	2.34349	1.552
STD. DEVIATION	1.834841	3.01951
VARIANCE	3.552627	9.1174406
BIAS	0.34349	-0.448
MEAN SQUARE ERROR	3.670612	9.318175
NOS. OF 90% C.I CAPTURING τ	88	25
N	100	100

The true partial group delay for the simulated data sets was 2. Observing just the numbers it appears that the bias for partial group delay estimate using the proposed technique and the Zhang and Foutz [1989] procedure is the same. A one sample Z test conducted for the means of each of the methods indicated that the two means were not significantly different from 2, the true value of the partial group delay. The p-values for the proposed method and Zhang and Foutz [1989] method were 0.0614 and 0.267 respectively.

The variance and the mean square error for the partial group delay estimate using the proposed method is far less than their counterparts obtained using the procedure by Zhang and Foutz [1989]. A t-test was conducted to test the significance of the difference between the two dependent variances. We consider the variances dependent because the same data sets were used by the two procedures to estimate the partial group delay. The t-test indicated that the variances for the two methods in fact are significantly different from each other (p-value < .001). See Ferguson [1976] , pp. 180 for the t-statistic.

Recall that using the procedure by Zhang and Foutz [1989] we obtain asymptotic confidence interval and the intervals obtained using the proposed technique are exact confidence intervals provided the ideal condition (see chapter 4, section 4.4) is satisfied. For the proposed method 90% confidence intervals were computed. Out of the 100 intervals computed, 88 intervals captured the true value of the partial group delay, that is, the true value 2 was enclosed in 88% of the intervals. Only 25 of the 100, 90% confidence intervals computed for the Zhang and Foutz [1989] method could capture the true partial group delay of 2.

5.3 Simulation Study II

We conducted another study to estimate the partial group delay by the proposed procedure. The signal process and other series involved were constructed using a different model. In this study also we had only one influencing series, namely $\{Z(t)\}$. Recall that in study I the signal was a MA(1) process and the influencing series $\{Z(t)\}$ was an AR(1) process. In this study the signal process and the series $\{Z(t)\}$ were both white noise processes. The interesting feature of this model is that it has constant partial coherence at all frequencies and compared to the previous model it is relatively simple. Secondly, we introduced a higher lag to see if any difficulties were encountered while applying the procedure. Thus, for this study data were simulated for a a theoretical partial group delay of 5, thus the series $\{X(t)\}$ led the series $\{Y(t)\}$ by 5 units after adjusting for the effect of the series $\{Z(t)\}$. In the following sections we describe in detail the model used, the results obtained, and comparison of the proposed procedure to that of Zhang and Foutz [1989] procedure.

5.3.1 Model For Simulated Data

For this study the signal $S(t)$ was generated as a white noise process with mean 0 and variance 0.8. The influencing series was also a white noise process with mean 0 and variance 0.6. The series $\{X(t)\}$, and $\{Y(t)\}$ were generated as,

$$X(t) = 0.4 * Z(t+1) + \varepsilon_x(t) \quad [5.29]$$

$$Y(t) = 0.6 * Z(t+3) + \varepsilon_y(t) \quad [5.30]$$

The residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ were constructed as,

$$\varepsilon_x(t) = S(t) + \alpha(t) \quad [5.31]$$

$$\varepsilon_y(t) = S(t+5) + \beta(t) \quad [5.32]$$

where $\{\alpha(t)\}$, and $\{\beta(t)\}$ were white noise processes with mean 0 and variance 0.05 and were uncorrelated with the signal process $\{S(t)\}$. Also the white noise process $\{\alpha(t)\}$ was uncorrelated with the white noise process $\{\beta(t)\}$.

The spectral density for the white noise process $\{S(t)\}$ is given by,

$$f_s(\lambda) = \frac{\sigma_s^2}{2\pi} = \frac{0.8}{2\pi} \quad [5.33]$$

and since the residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ were sums of two uncorrelated white noise processes their spectral densities are given as follows,

$$f_{\varepsilon_x\varepsilon_x}(\lambda) = \frac{1}{2\pi}\left[\sigma_s^2 + \sigma_\alpha^2\right] = \frac{1}{2\pi}[0.8 + 0.05] = \frac{0.85}{2\pi} \quad [5.34]$$

$$f_{\varepsilon_y\varepsilon_y}(\lambda) = \frac{1}{2\pi}\left[\sigma_s^2 + \sigma_\beta^2\right] = \frac{1}{2\pi}[0.8 + 0.05] = \frac{0.85}{2\pi} \quad [5.35]$$

Thus the spectral densities for the residual processes $\{\varepsilon_x(t)\}$, and $\{\varepsilon_y(t)\}$ are identical. To obtain the cross spectral density for the residual processes we first obtain the cross covariance as follows,

$$C(\tau) = E\left[\varepsilon_x(t)\varepsilon_y(t+\tau)\right] \quad [5.36]$$

$$= E\left[\left\{S(t)+\alpha(t)\right\} * \left\{S(t+\tau+5)+\beta(t+\tau)\right\}\right] \quad [5.37]$$

$$= E\left[S(t) * S(t+\tau+5) + \text{cross product terms}\right] \quad [5.38]$$

$$= \frac{\sigma_s^2}{2\pi} \qquad \text{if } \tau = \text{-}5 \quad [5.39]$$

The expected value of the cross product terms are zero as the signal and noise processes are assumed to be uncorrelated. The cross spectral density of the residual processes $\{\varepsilon_x(t)\}$ and $\{\varepsilon_y(t)\}$ is then given by,

$$f_{\varepsilon_x\varepsilon_y}(\lambda) = \frac{1}{2\pi}\sum_{\tau=-\infty}^{\infty} e^{-i\tau\lambda}C(\tau) \qquad -\pi < \lambda < \pi \quad [5.40]$$

$$f_{\varepsilon_x\varepsilon_y}(\lambda) = \frac{e^{i5\lambda} * 0.85}{2\pi} \qquad -\pi < \lambda < \pi \quad [5.41]$$

The partial phase for the series $\{X(t)\}$ and $\{Y(t)\}$ after adjusting for the series $\{Z(t)\}$ is obtained from the cross spectrum of the residual processes $\{\varepsilon_x(t)\}$ and $\{\varepsilon_y(t)\}$ as,

$$\theta_{\varepsilon_x\varepsilon_y}(\lambda) = \operatorname{Arg}\left(f_{\varepsilon_x\varepsilon_y}(\lambda)\right) = \operatorname{Arg}\left(e^{i5\lambda}\right) \quad [5.42]$$

$$\theta_{\varepsilon_x\varepsilon_y}(\lambda) = 5\lambda \quad [5.43]$$

and the partial group delay is given by the derivative of the partial phase as follows,

$$\tau(\lambda) = \frac{d\theta_{\varepsilon_x\varepsilon_y}(\lambda)}{d\lambda} = 5 \quad [5.44]$$

Thus we see that for this simulation study also the partial group delay is constant at all frequencies λ.

The interesting thing about this study was that the partial coherence was very high and most importantly constant at all frequencies λ. The partial coherence is given by,

$$\sigma(\lambda) = \frac{\left|f_{\varepsilon_x\varepsilon_y}(\lambda)\right|}{\sqrt{f_{\varepsilon_x\varepsilon_x}(\lambda)f_{\varepsilon_y\varepsilon_y}(\lambda)}} \quad [5.45]$$

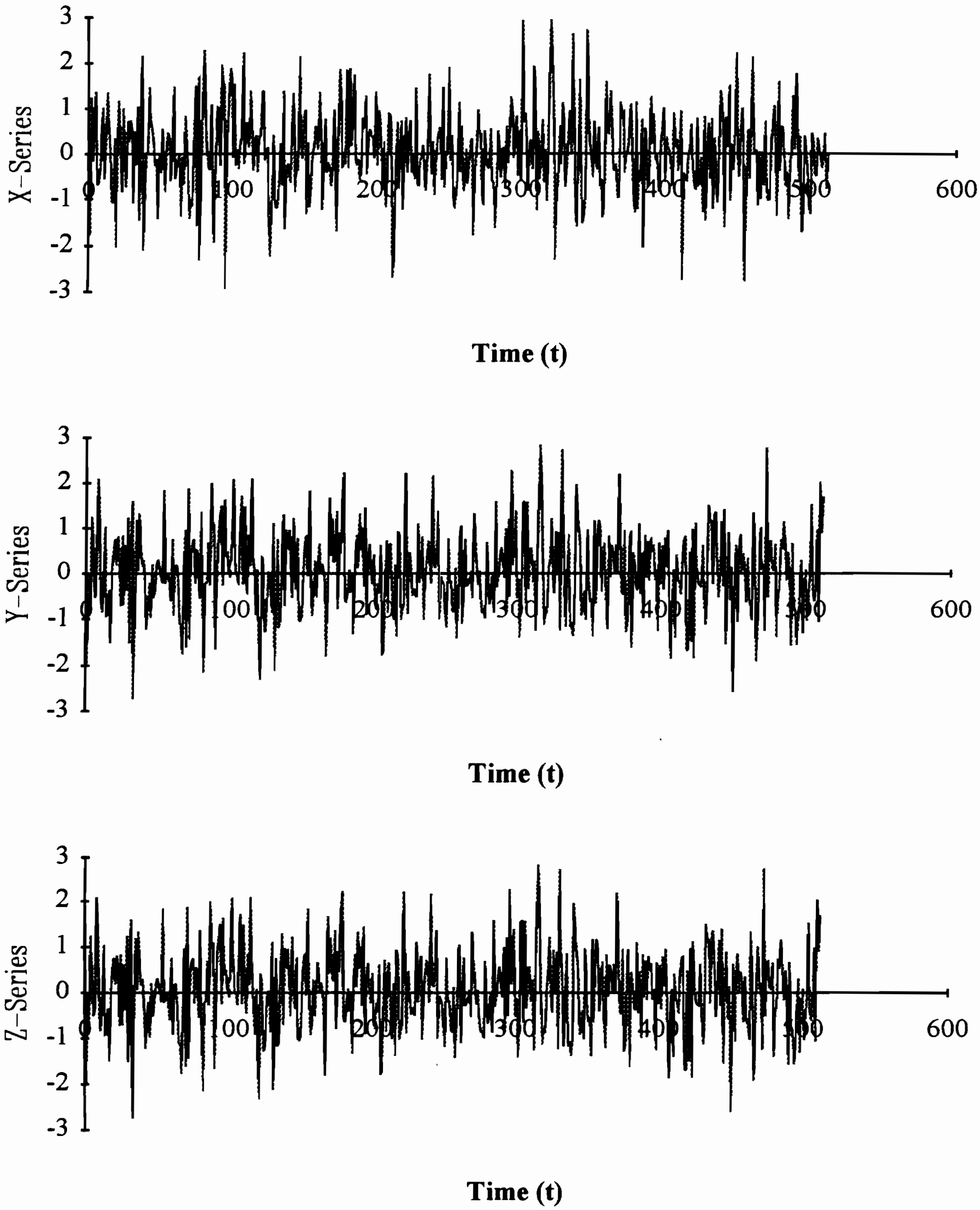
$$\sigma(\lambda) = \frac{\sigma_s^2}{\sigma_s^2 + \sigma_\alpha^2} = \frac{0.8}{0.8 + 0.05} = \frac{0.8}{0.85} \qquad [5.46]$$

$$\sigma(\lambda) = 0.9411764 \qquad \forall\, \lambda \qquad [5.47]$$

5.3.2 Results

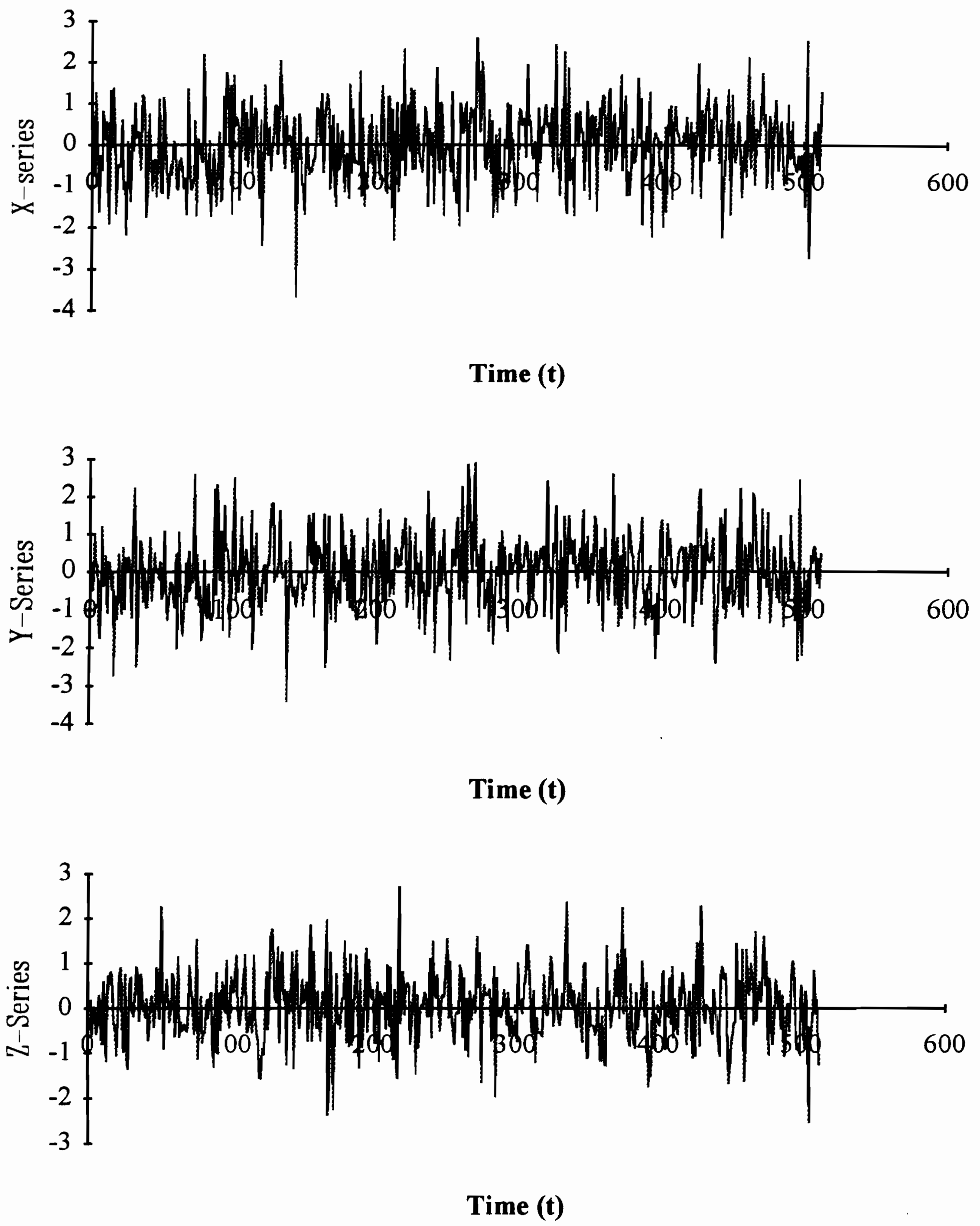
Using the simulation model given in the above section 100 data sets were generated on the IBM 3090 system. It took about 20 minutes to generate these data sets. For each of the data sets, the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ consisted of 512 observations. The plots on the following two pages will give the reader a visual idea of the series generated. The plots are for data set number 10 and 80 respectively and in the plots the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ are plotted versus time. For this simulation study the first 50 data sets were used for the proposed procedure and the remaining 50 data sets were used to estimate the partial group delay by the Zhang and Foutz [1989] procedure.

The steps for estimating the partial group delay except for a few changes are very identical to those of study I . First using a 5% taper the series $\{X(t)\}$, $\{Y(t)\}$, and $\{Z(t)\}$ were tapered. The series were then converted to frequency domain using Fourier transform. The periodogram and cross periodogram ordinates were formed. Each of the bands B_0, B_L, and B_R contained 8 Fourier frequencies, thus, $m = 8$. The partial cross spectral densities were obtained from the smoothed periodogram and cross periodogram ordinates. The function $\hat{q}(\tau)$ (see chapter 4, expression 4.18) was then maximized to yield the preliminary estimates for the partial group delay. Each data set generated 10 preliminary estimates.



Plot For Data Set # 10 - Study II

Figure 5.3.1



Plot For Data Set # 80- Study II

Figure 5.3.2

The preliminary estimates generated for this study were more stable than those obtained in study I. The primary reason for this is that the partial coherence between the series $\{X(t)\}$, and $\{Y(t)\}$ after adjusting for the effect of the series $\{Z(t)\}$ was constant and very high ($\sigma(\lambda) = 0.9411764$) for all frequencies. Hence all ten preliminary estimates were used in stage II of the procedure. Recall that for simulation study I we had used only eight preliminary estimates in stage II of the procedure. The preliminary estimates at frequencies $\lambda = 2.43596$ and 2.73049 were discarded because owing to a low partial coherence (0.75 and 0.55 respectively) the preliminary estimates at these two frequencies were highly unstable.

In stage II of the procedure which is application of the Ramsey and Foutz [1992] procedure we obtain UMVU estimates for the mean of the preliminary values. In this stage we express a relationship between the preliminary estimates and the frequencies using modeling techniques such as, polynomial regression, and spline models. When the errors were not normally distributed we resorted to transforming techniques such Box-Cox transformation technique. In all twenty-six data sets were transformed using the Box-Cox transformation, the transforming parameter λ varied between -0.2 and 1 . Seven data sets were modeled using spline regression. The number of knots varied from one to four. Seventeen data sets were modeled using polynomial regression, where the degree of the polynomial varied between one and three.

The partial group delay estimates for frequency λ=1.25786 obtained using the proposed procedure are presented in table [5.3.1] and those obtained using the Zhang and Foutz [1989] procedure are presented in the table [5.3.2]. For the two procedures we also present the 90% confidence intervals in their respective tables.

Table 5.3.1 Partial Group Delay Estimate For Proposed Procedure - Study II

Data Set	Proposed Est	Proposed 90% C.I LBD	UBD
1	2.305	1.036	56.339
2	4.481	2.163	6.799
3	8.926	4.883	12.97
4	5.574	2.805	17.484
5	9.407	4.771	14.044
6	6.153	1.542	46.981
7	6.338	3.099	9.577
8	4.714	1.123	8.305
9	3.029	0.246	5.812
10	3.889	-0.283	29.469
11	3.897	0.929	6.866
12	5.804	2.403	27.592
13	4.739	2.838	6.641
14	5	0.695	34.515
15	3.776	1.248	6.305
16	7.336	5.293	9.379
17	5.144	2.27	22.011
18	8.128	4.953	24.636
19	6.395	4.621	41.691
20	6.217	4.655	31.116
21	9.168	2.546	66.327
22	4.935	0.606	40.429
23	5.786	2.617	18.334
24	4.893	3.727	6.06
25	1.004	-2.138	4.147
26	8.352	4.988	11.717
27	9.877	3.217	27.794
28	10.424	4.738	38.825
29	3.772	-1.003	34.096
30	5.154	2.328	7.979
31	5.725	4.878	8.232
32	7.091	1.543	47.786
33	3.429	1.101	5.757
34	5.911	4.483	17.821
35	-0.767	-3.497	1.963
36	6.909	0.45	26.568
37	6.216	3.104	9.327
38	7.958	3.027	30.97
39	3.032	1.031	5.032
40	3.267	1.577	4.957
41	6.819	2.276	26.636

Data Set	Proposed Est	Proposed 90% C.I LBD	UBD
42	9.52	3.426	320.96
43	3.03	0.506	5.555
44	7.119	3.452	10.787
45	-1.144	-3.727	1.439
46	5.4	2.253	25.152
47	4.232	1.31	7.155
48	4.665	0.46	28.985
49	-0.123	-3.603	3.357
50	7.314	3.216	29.961

Table 5.3.2 Partial Group Delay Est. For Zhang & Foutz [1989] Method - Study II

Data Set	Zhang's Est.	Zhang's 90% C.I. LBD	UBD
51	4.1	3.597854	4.602146
52	0.8	0.297854	1.302146
53	3	2.497854	3.502146
54	5.1	4.597854	5.602146
55	4	3.497854	4.502146
56	-1.9	-2.40215	-1.39785
57	-0.9	-1.40215	-0.39785
58	4.8	4.297854	5.302146
59	8.6	8.097854	9.102146
60	-0.2	-0.70215	0.302146
61	-0.2	-0.70215	0.302146
62	14.9	14.39785	15.40215
63	3.8	3.297854	4.302146
64	-2.6	-3.10215	-2.09785
65	5.4	4.897854	5.902146
66	4.8	4.297854	5.302146
67	6.1	5.597854	6.602146
68	3.8	3.297854	4.302146
69	9.6	9.097854	10.10215
70	-3.1	-3.60215	-2.59785
71	9.1	8.597854	9.602146
72	7.5	6.997854	8.002146
73	-2.8	-3.30215	-2.29785
74	7.4	6.897854	7.902146
75	9.8	9.297854	10.30215
76	6.5	5.997854	7.002146
77	7.6	7.097854	8.102146
78	0.3	-0.20215	0.802146

Data Set	Zhang's Est.	Zhang's 90% C.I. LBD	UBD
79	4.6	4.097854	5.102146
80	12.9	12.39785	13.40215
81	9.3	8.797854	9.802146
82	5.1	4.597854	5.602146
83	10.9	10.39785	11.40215
84	5.8	5.297854	6.302146
85	-0.1	-0.60215	0.402146
86	-2.3	-2.80215	-1.79785
87	2.5	1.997854	3.002146
88	6.6	6.097854	7.102146
89	5	4.497854	5.502146
90	3.3	2.797854	3.802146
91	6.9	6.397854	7.402146
92	2.3	1.797854	2.802146
93	10.3	9.797854	10.80215
94	1	0.497854	1.502146
95	2.1	1.597854	2.602146
96	4.1	3.597854	4.602146
97	13.7	13.19785	14.20215
98	3.2	2.697854	3.702146
99	8.5	7.997854	9.002146
100	-0.5	-1.00215	0.002146

5.3.3 Comparison Of The Results

To compare the results of our procedure to that of Zhang and Foutz [1989] we consider the following table where the summary statistics for the estimated partial group delay using the proposed method and the method by Zhang and Foutz [1989] at frequency $\lambda = 1.25786$ are given:

Table 5.3.3 Simulation Study II - Statistics At Frequency λ=1.25786

STATISTICS	PROPOSED METHOD	ZHANG'S METHOD
MEAN	5.4044	4.61
STD. DEVIATION	2.59288353	4.401171
VARIANCE	6.72304498	19.37031
BIAS	0.4044	-0.39
MEAN SQUARE ERROR	6.88658434	19.55241
NOS. OF 90% C.I CAPTURING τ	44 (88%)	7 (14%)
N	50	50

Recall that the true partial group delay of the simulated data sets was 5. Since we used different data sets for each of the two methods the estimates generated will be considered as two independent groups. A one sample Z-test conducted on the respective means indicated that they did not differ significantly from the true value of 5. The p-values for the proposed procedure and for Zhang and Foutz [1989] procedure were 0.27 and 0.53 respectively. For this study also the variance and the mean square error for the proposed procedure are smaller than those for the procedure by Zhang and Foutz [1989]. It was indicated by a t-test that the two variances were significantly different (p-value = 0.001).

The 90% confidence intervals obtained for the proposed procedure showed that the true partial group delay of 5 was captured by 44 data sets, that is, the true value of the parameter was enclosed in 88% of the confidence intervals. The 90% confidence intervals computed for the Zhang and Foutz [1989] method showed that only 7 out of the 50 data sets captured the true value of the parameter.

5.4 Conclusions

In the above two sections with the help of simulation studies we demonstrated our procedure and compared our procedure to that of Zhang and Foutz [1989]. In this section we summarize the results as shown by the two simulation studies and try to convince the reader that the use of the proposed procedure for estimating partial group delay in some situations is better than the existing procedure by Zhang and Foutz [1989], thus justifying the need for the proposed procedure and fulfilling our fourth objective.

The two simulation studies indicate that the bias for the two method appears to be almost the same but are opposite in sign. However the variance for the proposed procedure is smaller than the variance for the procedure by Zhang and Foutz [1989]. In fact it has been shown in both the studies that the two variances are significantly different. Since there is no formal test to compare the mean square errors we have no other choice but to compare the numbers and this indicates in both the studies that the proposed procedure has a smaller mean square error than that for the Zhang and Foutz [1989] method.

The most important test of this procedure in our view is the ability of the confidence intervals to capture the true value of the parameter. It has always been stressed in elementary statistics that an interval estimate is more preferable to a point estimate. For the proposed procedure the true value of the partial group delay, that is $\tau = 2$ was captured by 88% of the intervals in simulation study I and in simulation study II 88% of the intervals captured the true value ($\tau = 5$) of the partial group delay. To test whether the sample estimate $\hat{p} = 0.88$ was significantly different from the true value $p = 0.90$ a Z-test was conducted. The p-values were 0.54 and 0.67 for simulation study I and II respectively. Hence $\hat{p} = 0.88$ did not differ significantly from the true value $p = 0.90$. Recall that the confidence intervals obtained using Zhang and Foutz [1989] method are asymptotic. As seen in both simulation studies they drastically fail in capturing the true value of the parameter. In study I only 25% of the intervals captured the true value of $\tau = 2$ and in study II only seven out of fifty (14%) intervals captured the true partial group delay of 5. In fact, Z-test for proportions indicates that the sample estimates $\hat{p} = 0.25$ and $\hat{p} = 0.14$ are significantly different from the true value $p = 0.90$. The p-values for the two studies were less than 0.0001. This suggests that for small samples the asymptotic intervals obtained by Zhang and Foutz [1989] method are meaningless. The proposed procedure on the other hand provides exact intervals and the use of such a procedure when sample sizes are small would lead to more meaningful interval estimates. Thus, we can conclude that the use of the proposed procedure when sample sizes are not big will yield better results than the procedure by Zhang and Foutz [1989].

5.5 Choice Of The Bandwidth (m)

To find the smoothed periodogram and cross periodogram estimates we formed three bands B_0, B_L, and B_R and each of these bands consisted of eight Fourier frequencies, that is $m = 8$. Before settling for $m = 8$, the proposed procedure was tried for bandwidths of 2, 3, 4, 5, 6, and 7. To save time this search for bandwidth did not involve all the 100 data sets, only the first twenty-five data sets were used. This investigation revealed that for smaller bandwidths, especially $m = 2$ to 5 the preliminary estimates were highly unstable, in fact some of the data sets gave meaningless estimate values like -400, 312, 500 and such other absurd numbers. And it was observed that as the bandwidth got closer to eight frequencies the preliminary estimates were stable.

For this procedure we do not recommend a specific bandwidth of $m = 8$. Rather the choice of the bandwidth will change from one set of series to another. A smart analyst will make the choice of the bandwidth depending on the amount of data available, nature of the series involved, the relationship between the series, and the nature of the partial coherence at each frequencies. In fact the reader familiar with time series literature will find that there are no formal and specific guidelines for choosing the bandwidth.

Chapter 6
Conclusions And Topics For Further Research

6.1 Concluding Remarks

In the previous five chapters we discussed in details the theoretical aspects of the procedure. We also demonstrated the use of the procedure by means of two simulation studies. In the present chapter we try to summarize the results of our findings and discuss topics for future research.

(i) In this dissertation we have introduced a procedure for estimating the partial group delay parameter. This parameter is defined as the time lag between a group of series after adjusting for the effects of the common influencing series. The procedure is a two step procedure. In step I we find preliminary values for the partial group delay using the Zhang and Foutz [1989] method. In Step II we use transforming and modeling techniques to obtain estimates of the mean for the preliminary values.

(ii) The estimates obtained using the procedure are uniformly minimum variance unbiased. Further we have also shown how to obtain $100(1-\alpha)\%$ confidence intervals for the estimated mean of the preliminary values.

(iii) The procedure was demonstrated on data simulated using specific models.

(iv) The results obtained using the proposed procedure were compared to the results obtained using the Zhang and Foutz [1989] method. The comparison showed that the proposed procedure gave more meaningful results than the Zhang and Foutz [1989] method especially in the context of confidence interval estimates.

6.2 Discussion Of Topics For Further Research

We discuss in the present section some of the topics for further research. These topics were encountered while working on the present dissertation. Some of the topics listed would qualify as topics for dissertations.

(i) More work needs to be done on the choice of m the bandwidth, and the sample size N. At present there does not seem to be a formal procedure for choosing an appropriate bandwidth m. Typically the bandwidth is chosen as a compromise between resolution and stability of an estimator. By resolution we mean the ability of an estimator to distinguish fine structures of a spectrum, and by stability we mean an estimator having a small variance.

(ii) A method for finding preliminary estimates for the partial group delay was discussed in the present dissertation. Alternate methods need to be investigated. These methods can be obtained by extending procedures which are available for the unadjusted group delay. One such idea is to find the preliminary estimates using the slope method as discussed in section 4.7 of chapter 4. In the following section we demonstrate the use of this method using some naturally occurring time series. To arrive at any concrete conclusions regarding the use of the slope method to find preliminary estimates simulation studies need to be conducted.

(iii) For the two simulation studies conducted we assumed a single influencing series $\{Z(t)\}$. In practice there could be more than one influencing series. For such a situation the present procedure with slight modifications (see section 4.5 of chapter 4) can be used. To explore this scenario simulation studies could be conducted

(iv) The models for the simulation studies of chapter 5 were such that the partial group delay was constant at each and every frequency. However, the partial group delay ($\tau(\lambda)$) could be different at different frequencies (λ). For instance the partial group delay could be a linear function of the frequencies ($\tau(\lambda) = \tau\lambda$). The use of the present procedure for such a situation could be demonstrated using simulation studies.

(v) The procedure proposed in the present work can be extended to other partial spectral parameters like partial spectral density, partial cross spectral density, partial phase, and partial coherence.

(vi) Extending the procedure to estimating partial coherence would be an iterative procedure. The estimates of partial coherence tend to be highly biased if the partial group delay between the residual series is large. This in time series literature is termed as the 'coherency bias'. One way of reducing this bias would be to realign the series before estimating the partial coherence. Thus, the procedure would involve two steps, namely initial estimates for the partial group delay would be obtained, the series are realigned and estimates for partial coherence obtained. These new estimates of partial coherence would be used as weights for the cross correlation function to give new estimates for the partial group delay and hence new estimates of partial coherence. The process would thus be repeated iteratively until convergence is attained.

6.3 Application Of Slope Method To Vendor Data

6.3.1 Introduction

The Dow Jones Industrial Average, Standard and Poor's Average, New York Stock Exchange Index, Nasdaq composite index, and the American Stock Exchange Index are some of the leading economic indicators that measure the level of the economic activity. These indexes or indicators determine where the economy has been, where it is now, and where it is going in the future. To compute these indexes the stock and/or bond prices of a few important companies are taken into consideration. These prices are then averaged to give the value of the index. For instance, the Dow Jones Industrial Average is a composite of stocks and bonds of a total of 30 companies including Chrysler, Dupont, General Electric, General Foods, Proctor and Gamble, Exxon, Texaco, United Aircraft, and Woolworth.

The aim of the present section is to briefly demonstrate the idea of using another method for finding the preliminary estimates for the partial group delay as suggested in point number (ii) of section 6.2. This method is referred as the slope method and for greater details the reader is referred to section 4.7 of chapter 4. Observing the results obtained we can only make a simple conclusion that the method holds some promise as being an additional method for finding preliminary estimates for the partial group delay. Since detailed simulation studies have not been conducted to test this procedure we refrain from drawing any concrete conclusions regarding the use of this procedure.

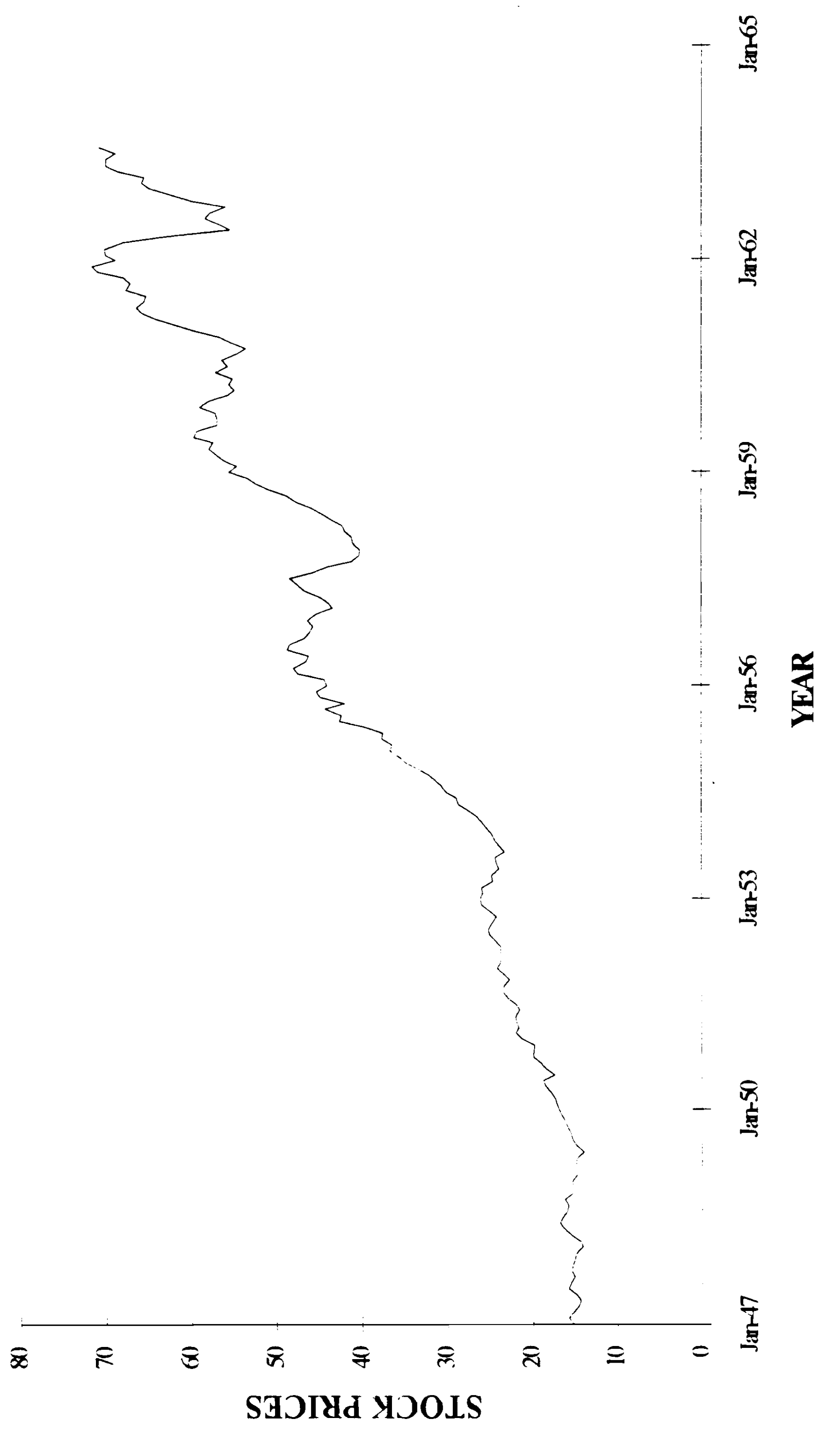
6.3.2 Spectral Analysis Of Vendor Data

The example considered in the present section consists of three economic series and these are as follows : Standard and Poor's 500 index of stock market prices, U.S Department of Commerce's index of industrial production, and index of vendor performance. The latter two indexes along with 20 other indexes are published monthly in the magazine 'Survey Of Current business'. Vendor performance is percent of businesses reporting slower deliveries. The index for stock market prices, as the name suggests is a composite of 500 stocks. The index of industrial production, and vendor performance are composites of 12 series. These three series were chosen because they seem to change directions before the general level of the economic activity changes direction. Data and information regarding the three series were obtained from the text by Pankratz [1991, pages 232-233].

For the present example we will assume that the index for vendor performance is the common influencing factor for the index of industrial production, and the index of stock market prices. The reason for this is as follows : As the series for vendor performance increases, more businesses experience slower deliveries. This leads to an artificial increase in the levels of order and services. This in turn reflects higher future levels of sales and production which would eventually lead to higher profits on the part of the investors thereby increasing the stock prices. Also, as the index for stock prices increases the index for industrial production also increases. Therefore the question of interest is how soon are the changes in the index for stock prices reflected in the index for industrial production after removing the influence of the index for vendor performance from both series.

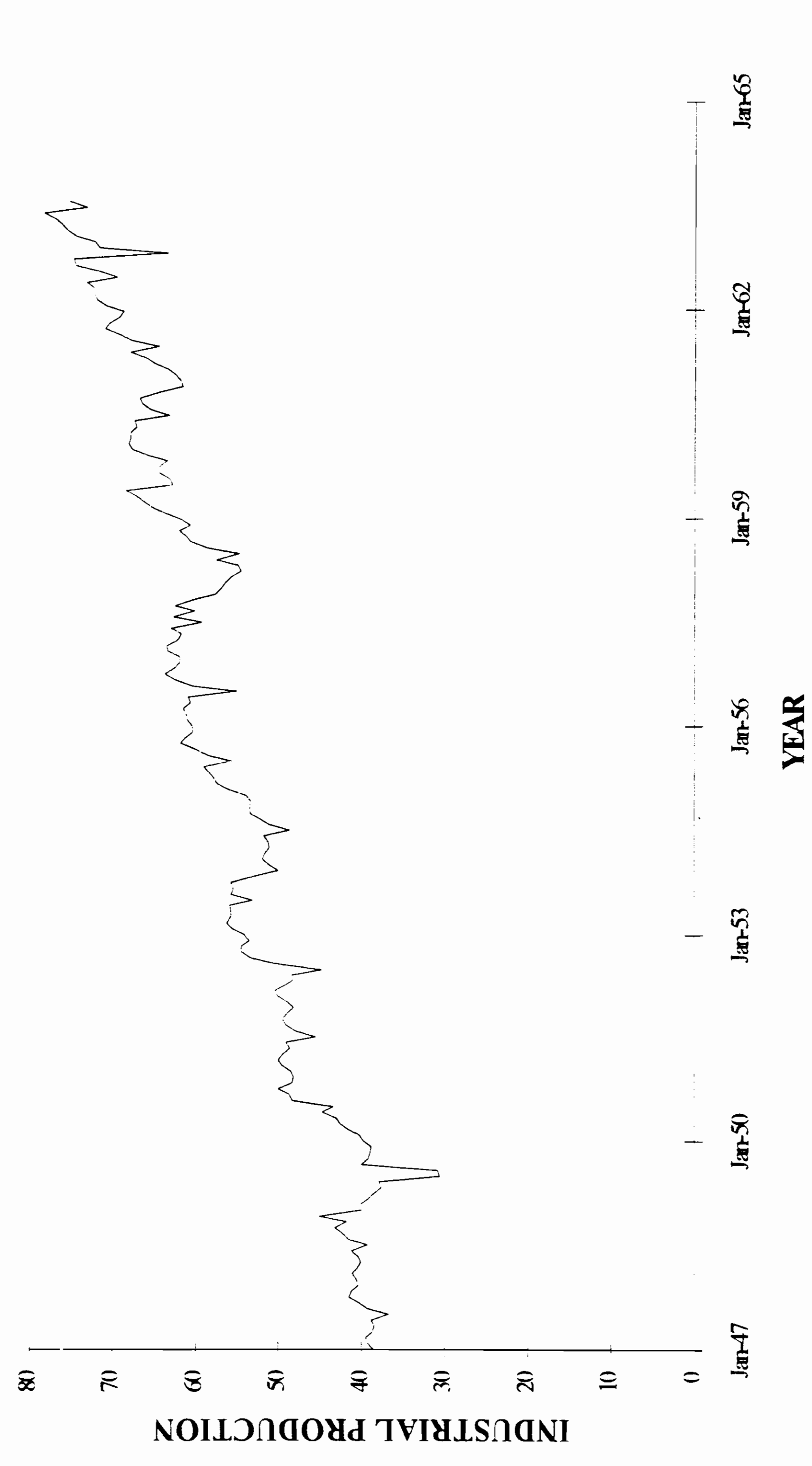
To estimate the partial group delay, data from January 1947 to August 1963 was considered for the above three series. Thus we had a total of 200 observations. The series for the index of industrial production, index of stock market prices, and vendor performance are plotted on the following pages respectively in figures [6.3.2.1], [6.3.2.**Error! Bookmark not defined.**], and [6.3.2.5]. The plots for the series of industrial production and stock market prices show an upward trend and the series for vendor performance shows some cyclical trends and is fairly stationary. To account for the upward trend and to make them stationary the series of industrial production and stock market prices were fitted with a straight line.

Using the vendor performance series and the residual series for stock prices and industrial production, basic spectral analysis was performed using the procedure 'Proc Spectra' from the software SAS. The intent of this was twofold, namely to investigate whether the data exhibited cycles of longer period or shorter period and to find the partial coherences at these dominant frequencies. In figures [6.3.2.7], [6.3.2.9], and [6.3.2.11] are plots of spectral densities for index of stock market prices, industrial production, and vendor performance. In figures [6.3.2.13], [6.3.2.15], and [6.3.2.17] are the plots of the squared coherency for each of the pairs of series, namely index of stock prices and industrial production; index of stock prices and vendor performance; and index of industrial production and vendor performance respectively. The plots of spectral density for each of the series indicates that the lower frequencies are the dominant frequencies. This implies that the cycles exhibited by the data has a longer period. The plots for the squared coherency also indicates that the three series exhibit a pairwise relationship.



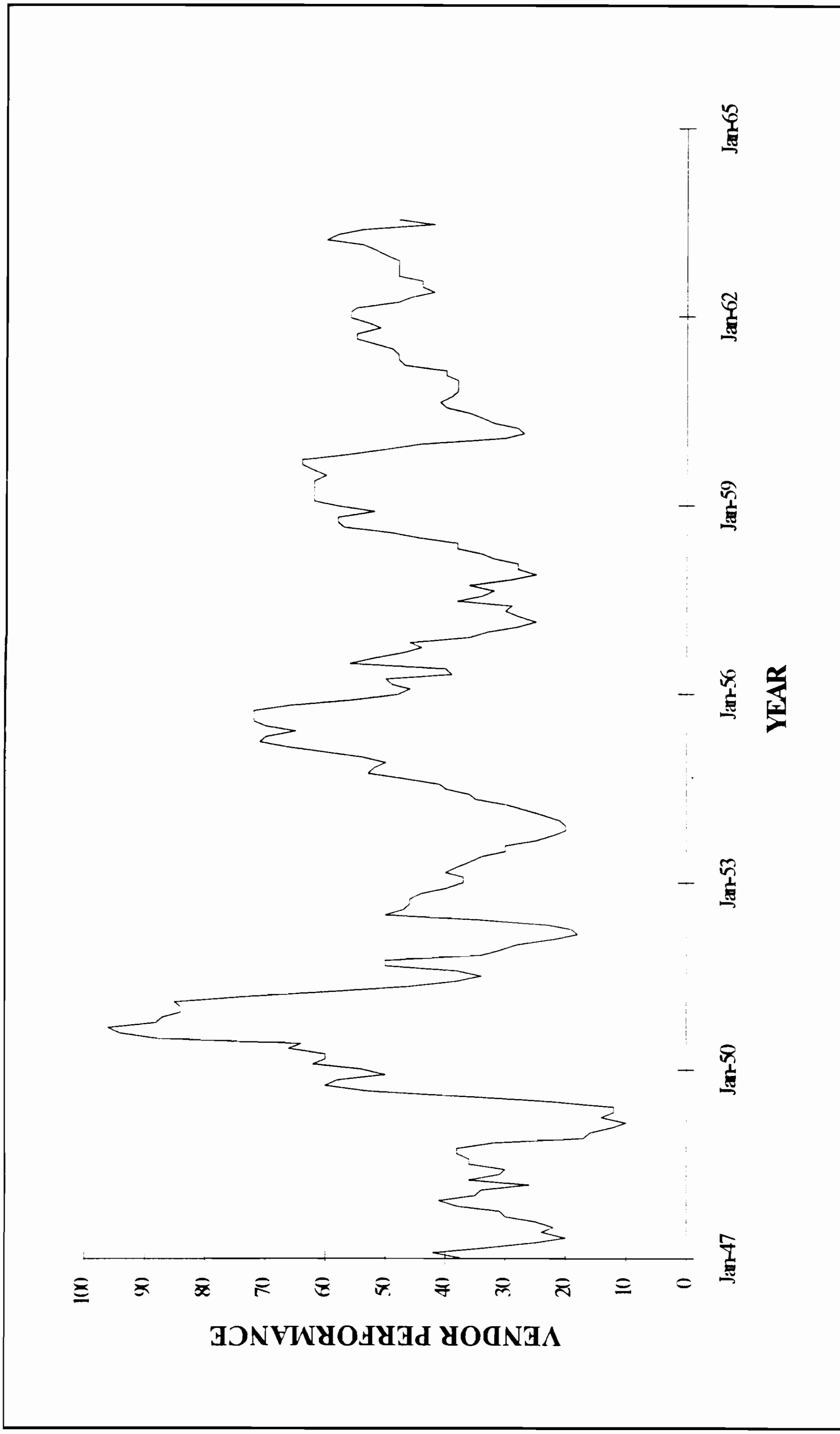
PLOT FOR INDEX OF STOCK PRICES (JAN 1947-AUG 1963)

Figure 6.3.2.1



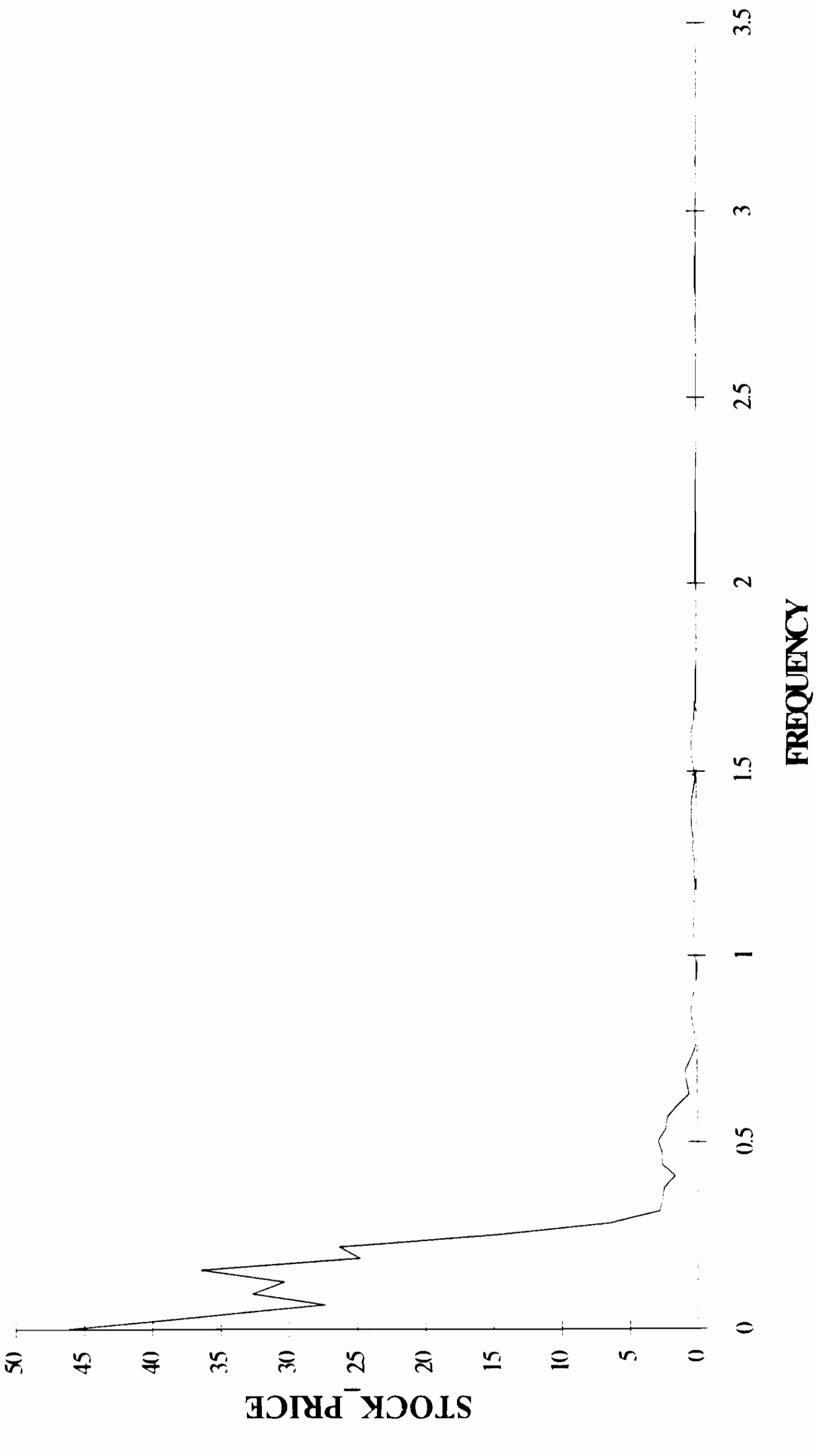
PLOT FOR INDEX OF INDUSTRIAL PRODUCTION (JAN 1947- AUG 1963)

Figure 6.3.2.2



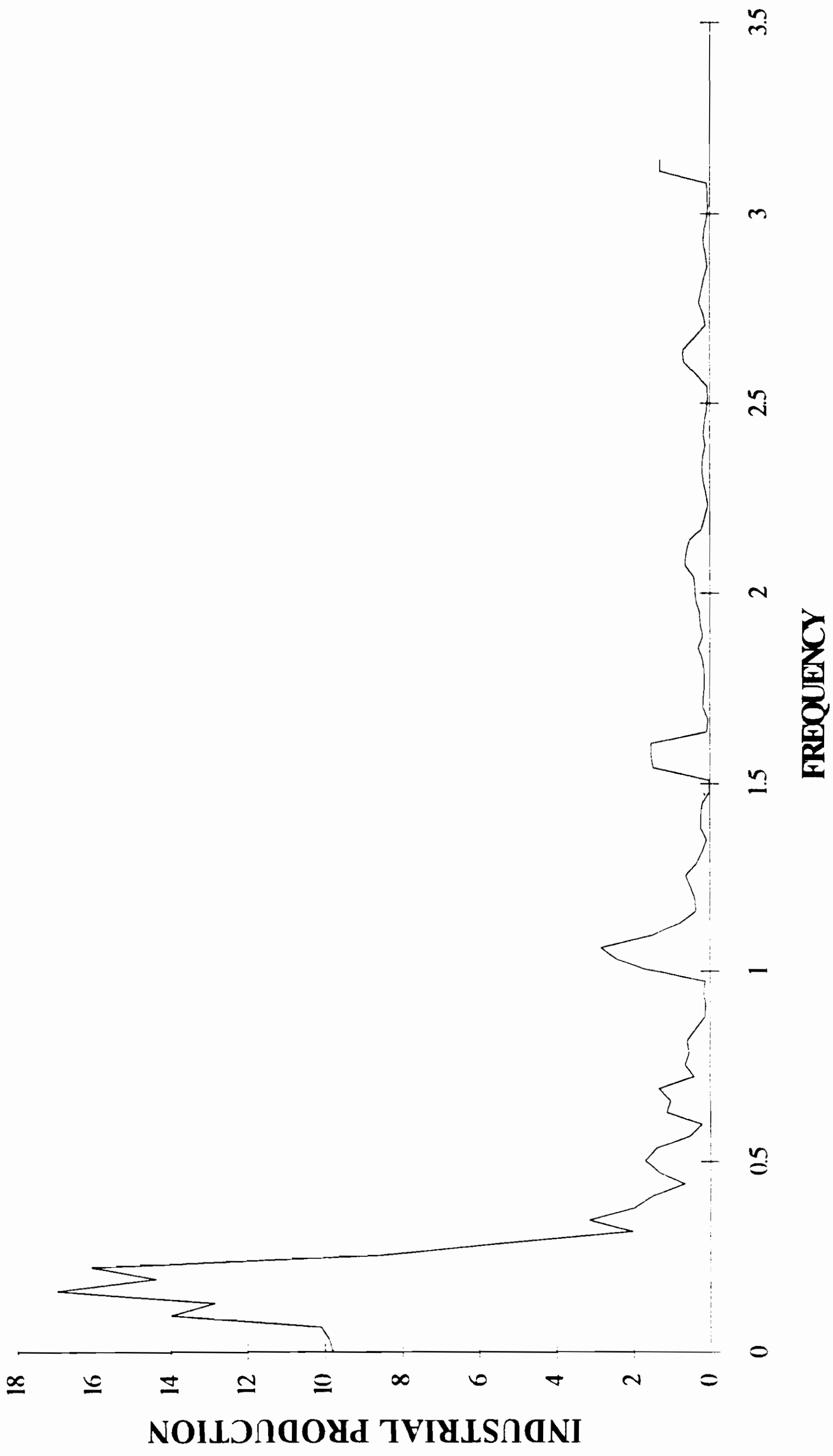
PLOT FOR INDEX OF VENDOR PERFORMANCE (JAN1947 - AUG1963)

Figure 6.3.2.3



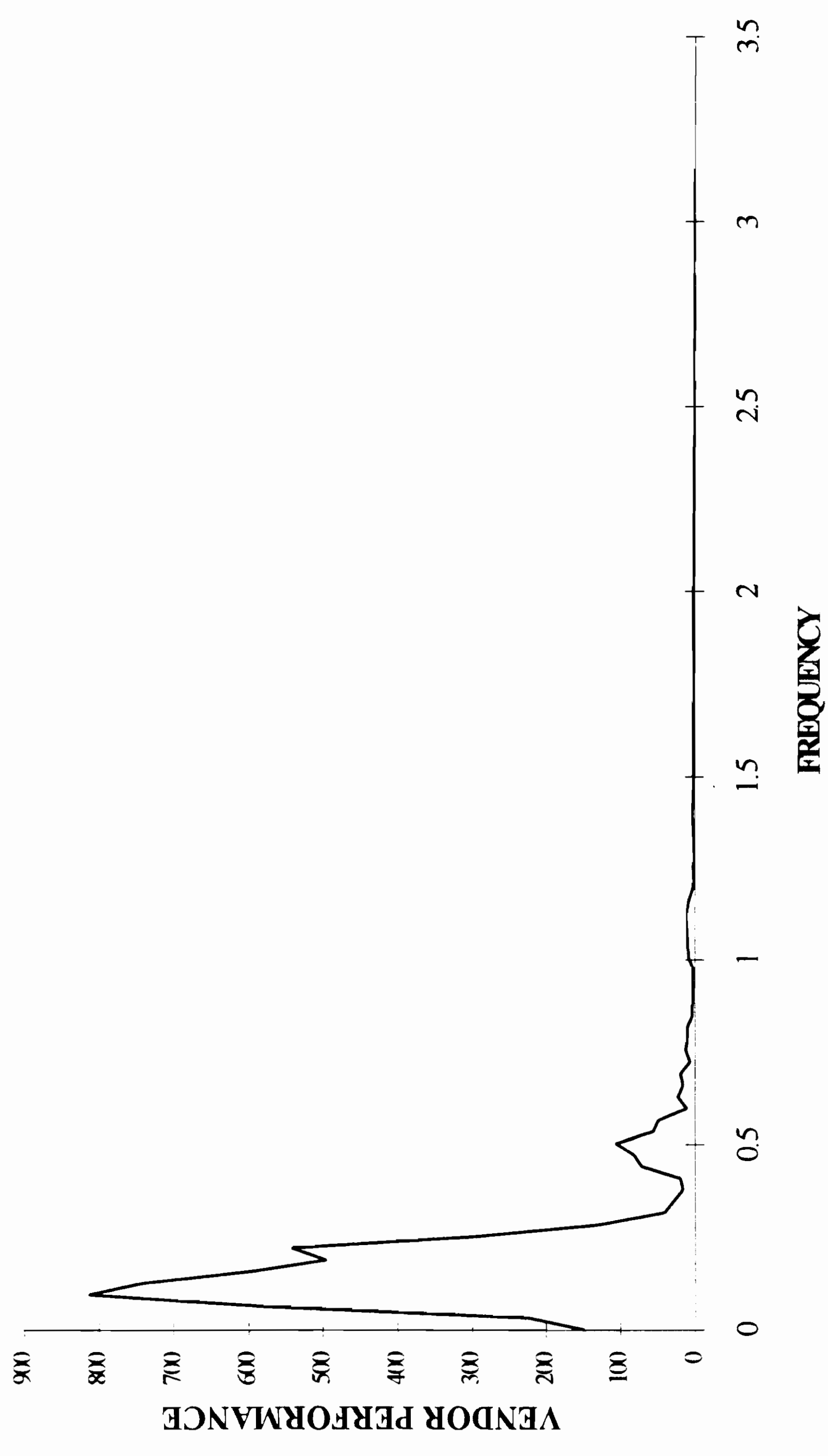
SPECTRAL DENSITY OF INDEX FOR STOCK PRICES

Figure 6.3.2.4



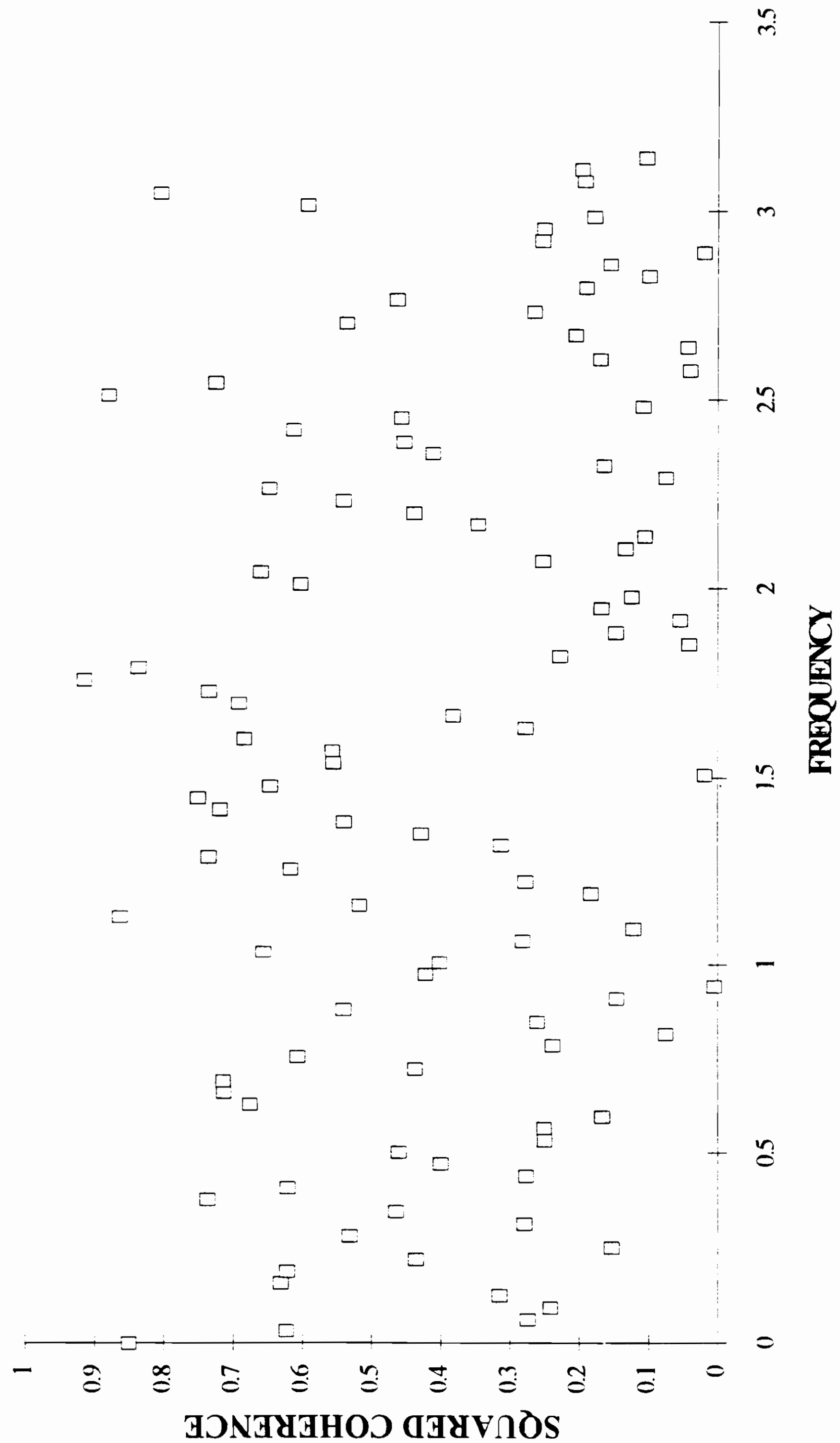
SPECTRAL DENSITY OF INDEX FOR INDUSTRIAL PRODUCTION

Figure 6.3.2.5



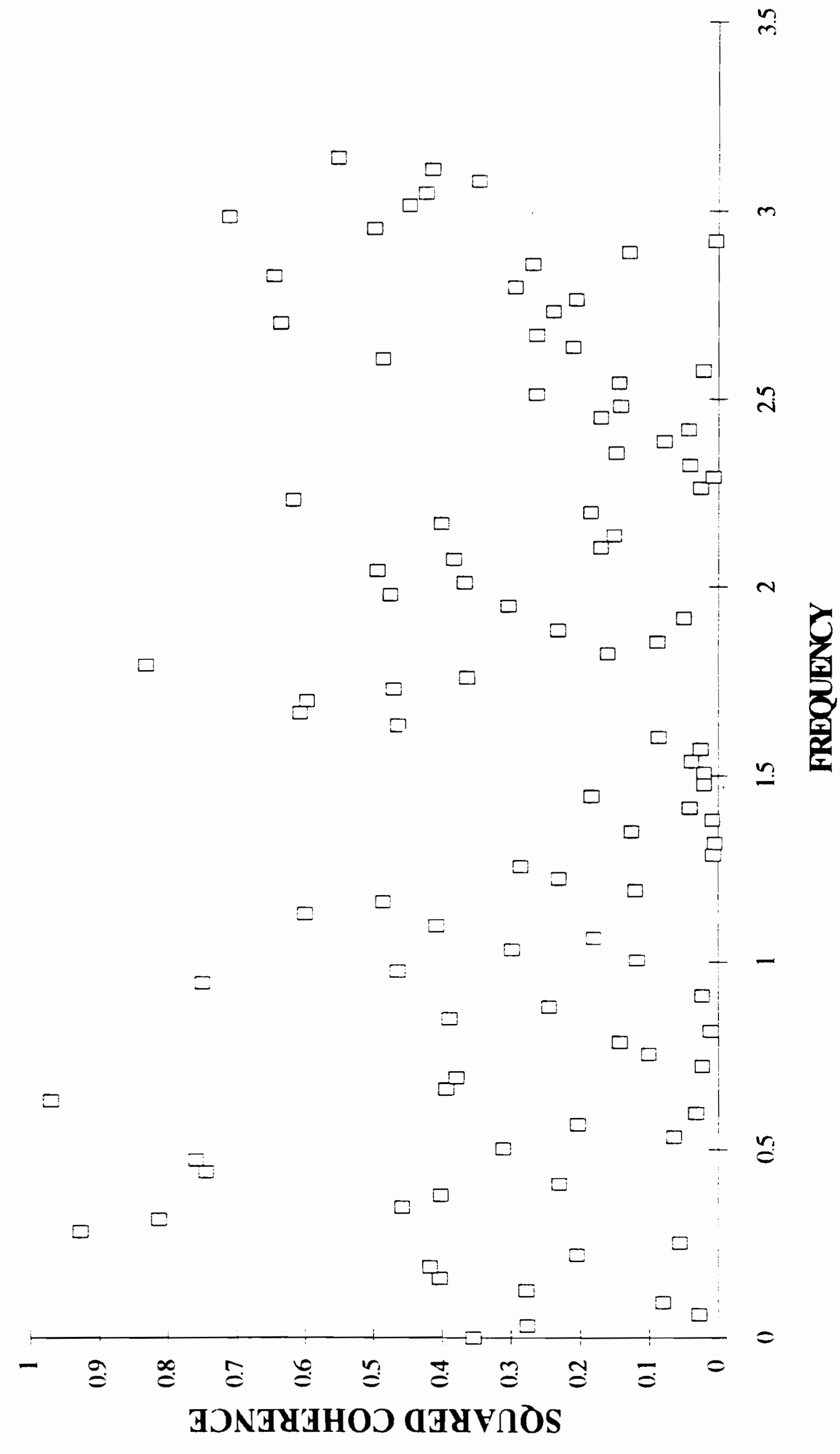
SPECTRAL DENSITY OF INDEX FOR VENDOR PERFORMANCE

Figure 6.3.2.6



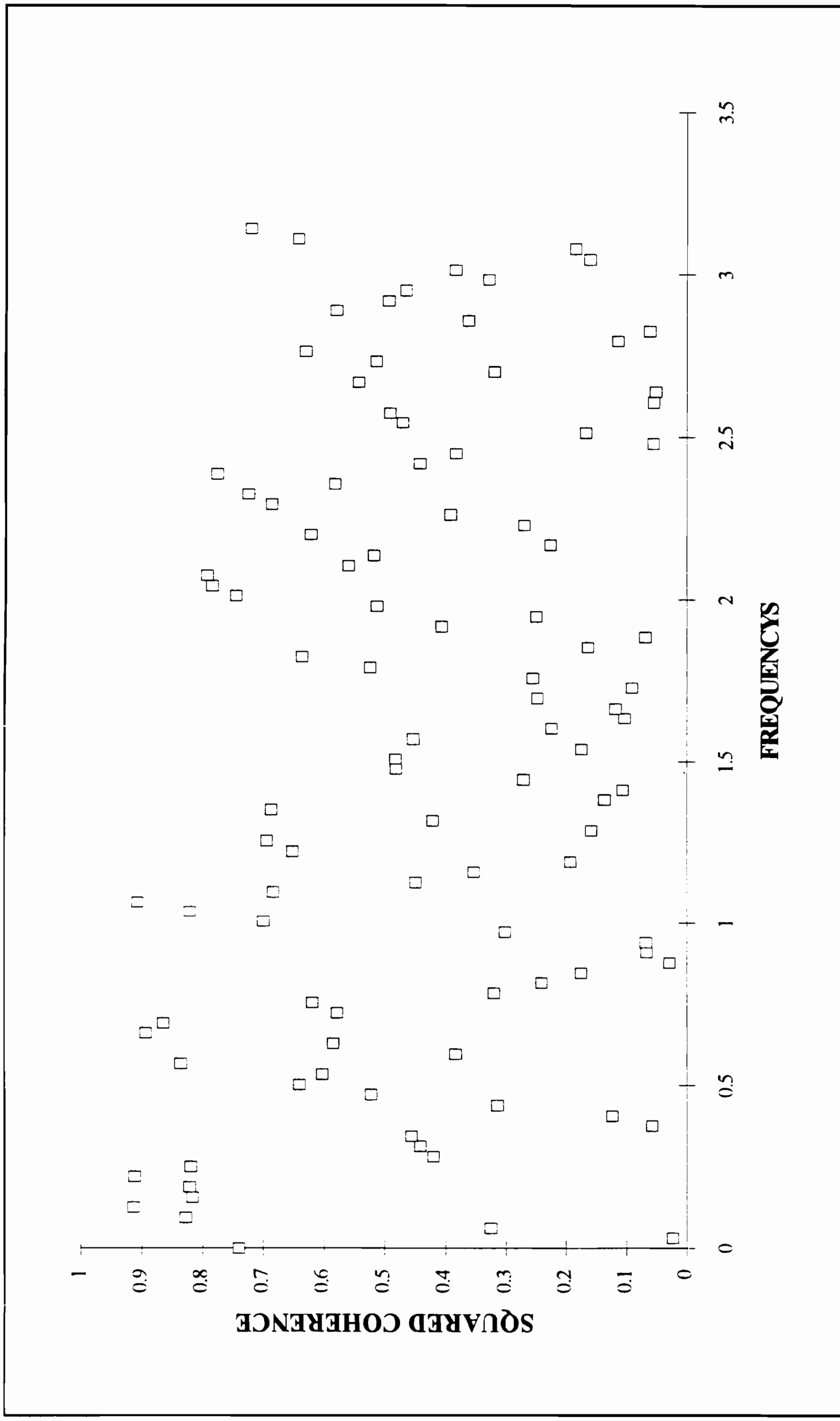
SQUARED COHERENCE OF INDEX FOR STOCK PRICES AND INDUSTRIAL PRODUCTION

Figure 6.3.2.7



SQUARED COHERENCE OF INDEX FOR STOCK PRICES AND VENDOR PERFORMANCE

Figure 6.3.2.8



SQUARED COHERENCE OF INDEX FOR INDUSTRIAL PRODUCTION AND VENDOR PERFORMANCE

Figure 6.3.2.9

6.3.3 Application Of The Slope Method

Having obtained the preliminary information regarding the behavior of the three series, we then proceeded to apply the proposed procedure. The series for index of vendor performance, the residual series for the index of stock prices, and industrial production were used to find the preliminary estimates for the partial group delay. Unlike the method of chapter 5 for this example the preliminary estimates were formed by the method described in section 4.7, chapter 4. Recall that in this method for each of the series we obtain the Fourier transforms at Fourier frequencies, the smoothed periodogram ordinates, and the smoothed cross periodogram ordinates. The series for the index of stock prices and industrial production are then adjusted for the influence of the series for the index of vendor performance. For the adjusted (residual) processes we then obtain the estimated cross spectral density referred to in this work as the estimated partial cross spectral density. Next the estimated partial phase is obtained from the estimated partial cross spectral density and the estimated partial group delay is then obtained as the slope between consecutive estimated phases.

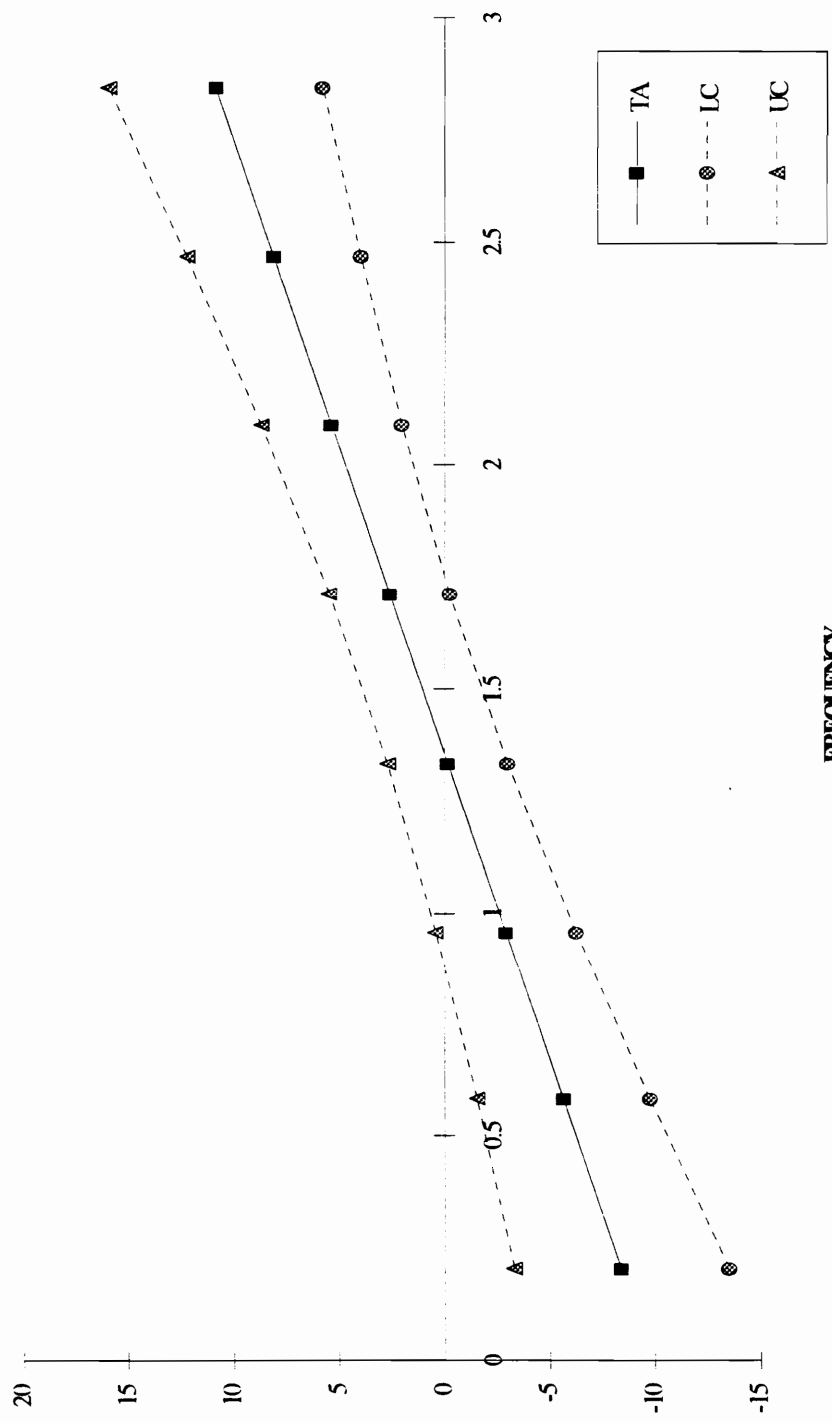
For the vendor data a bandwidth of $m = 6$ was selected, that is six Fourier transforms were used to find the smoothed periodogram and smoothed cross periodogram ordinates. Recall that the method of Zhang and Foutz [1989] to find preliminary estimates requires three bands, each containing m Fourier frequencies and the present method requires only one band containing m Fourier frequencies and thus is much more simplified than the method of Zhang and Foutz [1989]. Thus, at the end of Stage-I we had eight preliminary estimates for the partial group delay.

The preliminary estimates of stage-I were used in stage-II of the procedure, which is the modeling stage. No transformation was required for the preliminary estimates as a plot of the preliminary estimates versus their respective frequencies indicated a straight line relationship. In fact a linear regression of preliminary estimates on the respective frequencies indicated that a straight line fit was more than adequate ($R^2 = 0.95$). Also a test of normality on the residuals indicated that the assumptions of normality and homogeneity of variances were not violated. In table 6.3.3.1 we have presented the partial group delay estimates in the column titled $\hat{\tau}$ and the 90% lower and the upper confidence limits in the next two columns for the proposed procedure.

Table 6.3.3.1 : $\hat{\tau}$ Using Proposed Method

		PROP	90% C.I	
N	FREQ	$\hat{\tau}$	LCL	UCL
1	0.2042	-8.4	-13.491	-3.309
2	0.58119	-5.651	-9.778	-1.524
3	0.95819	-2.902	-6.235	0.431
4	1.33518	-0.153	-3.007	2.701
5	1.71217	2.595	-0.259	5.449
6	2.08916	5.344	2.011	8.677
7	2.46615	8.093	3.966	12.22
8	2.84314	10.841	5.751	15.932

On the following page is a plot of the partial group delay estimates and its 90% interval estimates at the respective frequencies obtained using the proposed procedure. We now



Plot Of Partial Group Delay Estimates And 90% Confidence Intervals

Figure 6.3.3.1

Figure 6.3.3.2

select a frequency of interest say 0.58119. At this frequency we can say that changes in the index of industrial production are reflected approximately five and a half months after changes in the index of stock market prices.

In table 6.3.3.2 are presented the partial group delay estimates and 90% confidence intervals obtained using the Zhang and Foutz [1989] method.

Table 6.3.3.2 : $\hat{\tau}$ Using Zhang & Foutz [1989] Method

		ZHANG	90% C.I	
N	FREQ	$\hat{\tau}$	LCL	UCL
1	0.14137	115.688	17.418	5924.614
2	0.51836	65.955	22.929	1733.517
3	0.89535	36.833	13.087	1090.083
4	1.27235	22.771	6.866	1005.945
5	1.64934	17.345	4.997	881.986
6	2.02633	16.773	5.235	732.847
7	2.40332	18.954	5.543	920.293
8	2.78031	20.946	2.891	2773.175

At frequency 0.51836 for the Zhang and Foutz [1989] procedure the stock market prices lead the index of industrial production by approximately 65 months. Observing the market trends it is elementary knowledge that changes in the index of stock market prices should have an immediate effect on the index of industrial production. In this context comparing the above two point estimates and their respective interval estimates the proposed method using the slope method to find preliminary estimates seems to give reasonable estimates than the Zhang and Foutz [1989] method. Note that though the frequencies are not exactly the same they approximately are.

At this point based on only this example we cannot conclude that the slope method is another alternative method for finding the preliminary estimates for partial group delay but merely remark that the method holds a great deal of promise.

Bibliography

Akaike, H. [1962]. *On The Design Of Lag Window For The Estimation Of Spectra.* Ann. Inst. Stat. Math., Vol. 14, pp. 1-21.

Akaike, H. [1966]. *Note On Higher Order Spectra.* Ann. Inst. Stat. Math., Vol. 18, pp. 123-126.

Akaike, H. [1966]. *On The Use Of Non-Gaussian Process In The Identification Of A Linear Dynamic System.* Ann. Inst. Stat. Math., Vol. 18, pp. 269-276.

Akaike, H. and Yamanouchi, Y [1962]. *On The Statistical Estimation Of Frequency Response Function.* Ann. Inst. Stat. Math., Vol. 14, pp. 23-56.

Andrews, D. F. [1971]. *A Note On The Selection Of Data Transformations.* Biometrika Vol. 58, No.2, pp. 249-254.

Anscombe, F. J. [1961]. *Examination Of Residuals.* Proc. Fourth Berkeley Symp. Math. Statist. and Prob., Vol 1, pp. 1-36.

Anscombe, F. J. and Tukey, J. W. [1963]. *The Examination And Analysis Of Residuals.* Technometrics, Vol. 5, pp. 141-160.

Bartlett, M. S. [1947]. *Properties Of Sufficiency And Statistical Test.* Proc. Roy. Soc. A, Vol. 160, pp. 268-282.

Bloomfield, P. [1976]. *Fourier Analysis Of Time Series : An Introduction.* New York: John Wiley & Sons.

Box, G. E. P. and Cox, D. R. [1964]. *An Analysis Of Transformation (With Discussion).* J. R. Statist. Soc. B, Vol. 26, pp. 211-252.

Box, G.E.P. and Jenkins, G.M. [1976]. *Time Series Analysis : Forecasting And Corntrol.* Holden-Day, Oakland, California.

Box, G. E. P. and Tidwell, P. W. [1962]. *Transformation Of The Independent Variables.* Technometrics, Vol. 4, pp. 531-550.

Brillinger, D. R. [1981]. *Time Series : Data Analysis And Theory.* McGraw-Hill Inc., New York.

Brillinger, D. R. and Krishnaiah, P. R. [1983]. *Time Series In The Frequency Domain.* Handbook Of Statistics, Vol. 3, New York : Elsevier Science Publishing Co., Inc.

Brockwell, P. J. and Davis, R. A. [1987]. *Time Series : Theory And Application.* New York : Springer Verlag.

Buse, A. and Lim, L. [1977]. *Cubic Splines As A Special Case Of Restricted Least Squares.* J. Am. Statist. Assoc., Vol. 72, pp. 64-68.

Carroll, R. J. and Ruppert, D. [1981]. *On Predicition And The Power Transformtion Family.* Biometrika, Vol. 68, No. 3, pp. 609-615

Carter, G. C. [1981]. *Time Delay Estimation For Passive Sonar Signal Processing.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-29, No.3, pp. 463-470.

Carter, G. C. [1987]. *Coherence And Time Delay Estimation.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-75, No.2, pp. 236-255.

Cooper, R., Shaw, J. C., and Osselton, J. W. [1974]. *EEG Technology.* Butterworth & Co. (Publishers) Ltd.

Deaton, M. L. and Foutz, R. V. [1980]. *Group Delay And The Time-Lag Relationship Between Stochastic Processes.* J. Time Series Anal., Vol. 1, pp. 111-118.

Dixon, W. J. and Brown, M. B. [1979]. *BMDP : Biomedical Computer Programs.* Berkley: University Of California Press.

Ferguson, G. A. [1976]. *Statistical Analyis In Psychology & Education.* McGraw-Hill, Inc., United States.

Foutz, R.V. [1980]. *Estimation Of A Common Group Delay Between Two Multiple Time Series.* J. Amer. Statist. Assoc., Vol. 75, pp. 779-788.

Gibson, M. M. [1981]. *Delay Estimation Of Disturbances On The Basilar Membrance.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-29, No.3, pp. 621-622.

Granger, C. W. J. [1969]. *Investigating Causal Relations By Econometric Models And Cross-Spectral Methods.* Econometrica, Vol. 37, No. 3, pp. 424-438.

Granger, C. W. J. [1980]. *Testing For Causality. J. Economic Dynamics and Control.* J. Economic Dynamics And Control, Vol. 2, pp. 329-352.

Granger, C. W. J. [1981]. *Some Properties Of Time Series Data And Their Use In Econometric Model Specification.* J. Econometrics, Vol. 16, pp. 121-130.Granger, C. W. J. [1969].

Graybill, F. A. [1976]. *Theory And Application Of The Linear Model.* Duxbury Press, Boston, Massachusetts.

Hamon, B.V. and Hannan, E. J. [1974]. *Spectral Estimation Of Time Delay For Dispersive And Non-Dispersive Systems.* J. R. Statist. Soc. C. (Appl. Statist.), Vol. 23, No. 2, pp 134-142.

Hannan, E. J. and Thomson, P. J. [1971]. *The Estimation Of Coherence And Group Delay.* Biometrika Vol. 58, No. 3, pp. 469-481.

Hannan, E. J. and Thomson, P. J. [1973]. *Estimating group delay.* Biometrika Vol. 60, No. 2, pp. 241-253.

Hannan, E. J. and Thomson, P. J. [1981]. *Delay Estimation And The Estimation of Coherence And Phase.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-29, No.3, pp. 485-490.

Hannan, E. J. and Thomson, P. J. [1988]. *Time Delay Estimation.* J.Time Series Anal., Vol 9, No. 1.

Hannan, E.J. and Robinson, P.M. [1973]. *Lagged Regression With Unknown Lags.* J. R. Statist. Soc. B., Vol 35.

Hinich, M. J. and Gary, R. W. [1992]. *Time Delay Estimation Using The Cross Bispectrum.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-40, No. 1, pp. 106-1139.

Hinkley, D. V. and Runger, G. [1984]. *The Analysis Of Transformation.* J. Am. Stat. Assoc. Vol. 79, No. 386, pp. 302-309.

Kellaway, P. and Peterson, I. [1976]. *Quantitative Analytic Studies In Epilepsy.* Raven Press, New York.

Kernighan, B. W. and Ritchie D. M. [1988]. *The C Programming Language.* Prentice Hall, New Jersey.

Koopmans, L. H. [1964a]. *On The Coefficient Of Coherence For Weakly Stationary Stochastic Processes.* Ann. Math. Statist., Vol. 35, pp. 532-549

Koopmans, L. H. [1964b]. *On The Multivariate Analysis Of Weakly Stationary Stochastic Processes.* Ann. Math. Statist., Vol. 35, pp. 1765-1780.

Koopmans, L. H. [1974]. *The Spectral Analysis Of Time Series.* Academic Press, New York.

Neyman, J. and Scott, E. L. [1960]. *Correction For Bias Introduced By A Transformation Of Variables.* Ann. Math. Stat. Vol.31, pp. 643-655.

Nikias, C. L. and Pan, R. [1988]. *Time Delay Estimation In Unknown Gaussian Spatially Correlated Noise.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-36, No.11, pp.1706-1714.

Myers, R. H. [1990]. *Classical And Modern Regression With Application.* PWS-Kent, Boston, Massachusetts.

Pankratz, A. [1991]. *Forecasting With Dynamic Regression Models.* John Wiley & Sons, New York.

Priestley, M. B. [1981]. *Spectral Analysis And Time Series.* Academic Press, San Diego.

Quazi, A. H. [1981]. *An Overview On The Time Delay Estimate In Active And Passive Systems For Target Localization.* IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-29, No.3, pp. 527-530.

Ramsey, P. J. [1989]. *UMVU Estimation Of Phase And Group Delay With Small Samples.* Ph. D dissertation, Virginia Polytechnic Institute And State Univ., Blacksburg, VA.

Ramsey, P. J. and Foutz, R.V. [1992]. *Asymptotically Optimal Spectal Estimation With Applications To Small Samples.* Technical Report No. 92-6, Dept. Of Statistics, Virginia Polytechnic Institute And State Univ., Blacksburg, VA.

Rice, J. R. [1969]. *The Approximation Of Functions, Vol II.* Addison -Wesley, Reading, Massachusetts. pp. 123.

SAS/STAT User's Guide : Statistics [version 6]. SAS Institute Inc. Cary NC

Smith, P. L. [1979]. *Splines As A Useful And Convenient Statistical Tool.* The American Statistician, Vol 33, No. 2, pp. 57-62.

Thomson, P. J. [1990]. *Quadratic-Inverse Spectrum Estimates : Applications To Palaeoclimatology.* Phil. Trans. R. Soc. Lond. A, Vol. 332, pp. 539-597.

Tukey, J. W. [1957]. *On The Comparative Anatomy Of Transformations*. Ann. Math. Stat., Vol 28, pp. 602-632.

Wei, W. W. S. [1990]. *Time Series Analysis : Univariate And Multivariate Methods.* Addison Wesley Publication, Redwood City, CA.

Wold, S. [1974]. *Spline Functions In Data Analysis.* Technometrics, Vol. 16, pp. 1-11.

Zhang, N. F. and Foutz, R. V. [1989]. *Estimating Partial Group Delay.* Biometrika, Vol. 76, No. 1, pp. 57-63.

Appendix

A.1 Box-Cox Transformation

The two most commonly used techniques for analysis of linear models are regression, and analysis of variance. The assumptions made by these techniques are as follows,

i. Simplicity of the structure for the mean response, that is, E(y) should be linear

ii. Error variance should be constant

iii. Errors should be normally distributed and

iv. Observations should be independent

When one or more of the above conditions are not satisfied we are left with two choices, namely, transformation of the data, or inventing new techniques so that the data could be used as is. The first choice definitely seems an easier one. The analyst can then either transform the response variable, the regressor variable, or both of them. One such technique available for transforming the response variable is the Box-Cox transformation technique and is discussed in the following paragraph.

The Box-Cox transformation technique simultaneously transforms the response variable as follows,

$$v = \begin{cases} \dfrac{y^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln y & \text{if } \lambda = 0 \end{cases} \qquad [A.1]$$

or ,

$$v = \begin{cases} \dfrac{(y+\lambda_2)^{\lambda_1}-1}{\lambda_1} & \text{if } \lambda_1 \neq 0,\ y > -\lambda_2 \\ \ln\ (y+\lambda_2) & \text{if } \lambda_1 = 0,\ y > -\lambda_2 \end{cases} \qquad [A.2]$$

and modeling the transformed variable v as given by the following expression,

$$v_i = \beta_0 + \beta_1\omega_{1i} + \ldots\ldots + \beta_p\omega_{pi} + \varepsilon_i \qquad i = 1,2,3..,n \qquad [A.3]$$

Estimation of the β's and the λ's is done simultaneously using maximum likelihood method. The ε_i's in expression [A.3] are independently, and normally distributed random variables with common variance σ^2. The likelihood function for expression [A.3] is given as follows,

$$L(\beta,\lambda,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}}\exp\left\{\frac{-1}{2\sigma^2}(v-\omega\beta)'(v-\omega\beta)\right\} * J(\lambda,y) \qquad [A.4]$$

where $J(\lambda,y)$ the Jacobian is given as follows,

$$J(\lambda,y) = \prod_{i=1}^{n}\frac{\partial v_i}{\partial y_i} = \prod_{i=1}^{n} y_i^{\lambda-1} = (\tilde{y})^{\lambda-1} \quad \forall\, y > 0 \qquad [A.5]$$

and,

$$J(\lambda,y) = \prod_{i=1}^{n}\frac{\partial v_i}{\partial y_i} = \prod_{i=1}^{n}(y_i+\lambda_2)^{\lambda-1} \quad \forall\, y > -\lambda_2 \qquad [A.6]$$

and it accounts for the change in scale. Hence to get a scaled likelihood function we standardize the v_i's and use them in expression [A.4]. The standardization of the v_i's is done as follows,

$$z_i = \begin{cases} \dfrac{y_i^{\lambda}-1}{\lambda\prod_{i=1}^{n} y_i^{\lambda-1}} & \text{if } \lambda \neq 0 \\ \dfrac{\ln\ y_i}{\prod_{i=1}^{n} y_i^{\lambda-1}} & \text{if } \lambda = 0 \end{cases} \qquad [A.7]$$

or,

$$z_i = \begin{cases} \dfrac{(y_i+\lambda_2)^{\lambda_1}-1}{\lambda_1 \prod\limits_{i=1}^{n}(y_i+\lambda_2)^{\lambda_1-1}} & \text{if } \lambda_1 \neq 0,\ \lambda_2 > -y \\ \dfrac{\ln\ (y_i+\lambda_2)}{\prod\limits_{i=1}^{n}(y_i+\lambda_2)^{\lambda_1-1}} & \text{if } \lambda_1 = 0,\ \lambda_2 > -y \end{cases} \qquad [A.8]$$

Note that the Jacobians are merely geometric means. Hence the maximum log likelihood function is given as follows

$$\ln\{L(\beta,\lambda)\} = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\hat{\sigma}^2) - \left\{\frac{-(z-\omega\beta)'(z-\omega\beta)}{2\hat{\sigma}^2}\right\} \qquad [A.9]$$

where $\hat{\sigma}^2$ is the maximum likelihood function of σ^2 and for a fixed λ, $\hat{\sigma}^2 = \dfrac{\text{ssresiduals}}{n}$.

Hence maximizing the above expression with respect to λ is equivalent to minimizing apart from constant terms the following expression,

$$\ln\{L(\beta,\lambda)\} = -\frac{n}{2}\ln(\hat{\sigma}^2) \qquad [A.10]$$

Thus, maximizing the log likelihood function with respect to λ is the same as minimizing the sum of squares residuals.

The above procedure for Box-Cox transformation can be summarized as follows :

1. Select λ on a grid say, [-n, n].
2. Transform the response variable y's to v's using either expression [A.1], or [A.2].
3. Divide each v by the Jacobian $J(\lambda, y)$ to obtain standardized variables z's as given by expression [A.7], or [A.8].
4. Fit a linear model to the z's and obtain the sum of squares residuals.
5. Compare the current sum of squares residuals with the previous value and if the current value is less than the previous one then stop else goto the next step.
6. Increment λ and goto step 2.

A.2 Programming Code

We include in this section programs written for demonstrating the procedure. Note that we have not included all the programs but the most important one so as to keep this section brief.

```
/* *************************************************************** */
/* Title : FFT.C                                                    */
/* Date Written  : September 28, 1993                               */
/* Date Modified : September 28, 1994                               */
/* Author : Milan Mangeshkar                                        */
/* This program does the following things :                         */
/*      (1) Finds Fourier Transforms for the X,Y, and Z series      */
/*      (2) Finds the periodograms ordinates for the Z series       */
/*      (3) Finds the cross periodogram ordinates for X-Y, X-Z, & Z-Y */
/* *************************************************************** */
/* This program needs the foll. arguments :  (1) # of Data points   */
/*                                           (2) # of simulations   */

#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include "C:\USERS\MILAN\SOURCE\FFT.H"

/* HASH DEFINES */

#define EPSILON           1.0e-10

#define READMODE      "r"
#define WRITEMODE     "w"

#define IPFL_PFX      "C:\\USERS\\MILAN\\PARGD\\SIMDATA\\GD512"
#define OPFL_PFX      "C:\\USERS\\MILAN\\PARGD\\FFT\\FFT"
#define PRIFILENM     "C:\\USERS\\MILAN\\PARGD\\FFT\\FFT.PRI"

/* FILEPOINTERS ARE */

FILE    *fpOutFile;
FILE    *fpPriFile;

void main (int argc, char **argv)
{
short   sNumItems, sCtr = 0;
long    lNumDataPts;
```

```
char      szIpFileName[255],szOpFileName[255];

          if(argc==3)
          {
                    lNumDataPts = atol(argv[1]);
                    sNumItems  = atoi(argv[2]);
                    if ((fpPriFile = fopen(PRIFILENM, WRITEMODE)) == NULL)
                    {
                              printf("ERROR : COULD NOT OPEN FILE **FFT.PRI**\n");
                              exit(-1);
                    }

                    while (sCtr++ < sNumItems)
                    {
                              sprintf(szIpFileName,"%s.%d", IPFL_PFX, sCtr);
                              sprintf(szOpFileName,"%s.%d", OPFL_PFX, sCtr);
                              printf("ITERATION #:%d, %s, %s\n",sCtr, szIpFileName, szOpFileName);
                              fprintf(fpPriFile, *********************************************\n");
                                   fprintf(fpPriFile,"ITERATION  #:%d,  %s,  %s\n",sCtr,  szIpFileName,
szOpFileName);
                              Find_FFT(lNumDataPts,szIpFileName, szOpFileName);
                    }
          }
          else
          printf("INVALID ARGUMENTS GIVEN TO THE PROGRAM\n");

          printf("**PROGRAM TO FIND FFT & PERIODOGRAMS COMPLETED SUCCESSFULLY
***\n");
          fprintf(fpPriFile, "No of data pts = %ld\t", lNumDataPts);
          fprintf(fpPriFile, "No of FFT's    = %lf\n", floor(lNumDataPts/2.0));
          fclose(fpPriFile);
}

void Find_FFT(long lNumDataPts,char *szIpFileName,char *szOpFileName)
{
int                 i, iLoopvar, nobs;
static    short     sCallCt = 0;
long                lNumItems, fft_Items;
static double       *XSeries, *YSeries, *ZSeries;
static double       *rWX, *rWY, *rWZ,       *iWX, *iWY, *iWZ;
static double       *pgrmZZ;
static double       *rpgrmXY, *rpgrmXZ, *rpgrmZY,  *ipgrmXY, *ipgrmXZ, *ipgrmZY;
double              taper;

FILE      *fpDatFile;

          /* OPEN THE DATA FILE */

   if ((fpDatFile = fopen(szIpFileName, READMODE)) == NULL)
                    {
```

```
                        printf("ERROR : COULD NOT OPEN FILE **FFT.IN**\n");
                        exit(-1);
                }

        if ((fpOutFile = fopen(szOpFileName, WRITEMODE)) == NULL)
                {
                        printf("ERROR : COULD NOT OPEN FILE **FFT.OUT**\n");
                        exit(-1);
                }

  lNumItems   = lNumDataPts;
        fft_Items          = floor(lNumItems/2.0);

/*      *****************************
        * MALLOC SPACE if 1st invocation  *
        *******************************    */

         if (!sCallCt)
         {
                sCallCt++;
                if ((XSeries = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                        Print_Err();
                if ((YSeries = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                        Print_Err();
                if ((ZSeries = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                        Print_Err();

                if ((rWX = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((rWY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((rWZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((iWX = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((iWY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((iWZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();

                if ((pgrmZZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((rpgrmXY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((rpgrmXZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((rpgrmZY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
                if ((ipgrmXY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
                        Print_Err();
```

```
		if ((ipgrmXZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		if ((ipgrmZY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();

	} /* End of mallocing space */

  /* READ THE  X, Y, Z SERIES */

	/*	printf("*******PROGRAM EXECUTION BEGINS*******\n");*/

	iLoopvar = 0;
	while((!(feof(fpDatFile))) && (iLoopvar++ < lNumItems))
			fscanf(fpDatFile, "%d %lf %lf %lf\n", &nobs, XSeries+iLoopvar-1,
						YSeries+iLoopvar-1, ZSeries+iLoopvar-1);

	if (iLoopvar != lNumItems)
	{
		printf("ERROR: INCORRECT DATA FILE OR COMMAND LINE ARGUMENT\n");
		exit(-1);
	}

/*	** *****************
	* TAPER THE SERIES *
	*******************	*/

	taper = 0.0;
	for(i=0;i<5;i++)
	{
		XSeries[i] = XSeries[i] * taper;
		YSeries[i] = YSeries[i] * taper;
		ZSeries[i] = ZSeries[i] * taper;

		XSeries[lNumItems-1-i] = XSeries[lNumItems-1-i] * taper;
		YSeries[lNumItems-1-i] = YSeries[lNumItems-1-i] * taper;
		ZSeries[lNumItems-1-i] = ZSeries[lNumItems-1-i] * taper;

		taper = taper + 0.2;
  }

/*	************************************************
	* GET THE FAST FOURIER TRANSFORMS FOR X Y AND Z*
	************************************************	*/

	fft(lNumItems, fft_Items, XSeries, rWX, iWX);
	fft(lNumItems, fft_Items, YSeries, rWY, iWY);
	fft(lNumItems, fft_Items, ZSeries, rWZ, iWZ);

/*	**************************************
	* GET THE PERIODOGRAMS  FOR  Z SERIES	*
```

```
         ***************************************          */

         prdgm(fft_Items, rWZ, iWZ, pgrmZZ);

/*       *********************************************
         * GET THE CROSS PERIODOGRAMS  FOR X, Y AND Z*
         *********************************************     */

         crspgrm(fft_Items,rWX,rWY,iWX,iWY,rpgrmXY,ipgrmXY);
         crspgrm(fft_Items,rWX,rWZ,iWX,iWZ,rpgrmXZ,ipgrmXZ);
         crspgrm(fft_Items,rWZ,rWY,iWZ,iWY,rpgrmZY,ipgrmZY);

/*       ************************************************************
         * WRITE THE PERIODOGRAM & CROSS PERIODOGRAMS ORDINATES TO *
         * AN OUTPUT FILE                                          *
         ************************************************************    */

         for (i=0; i<fft_Items; i++)
                  fprintf(fpOutFile,
                "%4d\t%+8.5f\t\t%+8.5f\t\t%+8.5f\t\t%+8.5f\t\t%+8.5f\t\t%+8.5f\t\t%+8.5f\n",
                  (i+1),pgrmZZ[i],rpgrmXY[i],ipgrmXY[i],rpgrmXZ[i],ipgrmXZ[i],
                                  rpgrmZY[i],ipgrmZY[i]);

   /*    *************************
         *CLOSE ALL THE FILES      *
         *************************   */
         fclose(fpDatFile);
         fclose(fpOutFile);

} /* END OF FIND_FFT ROUTINE */

void fft(long lNumItems,long fft_Items,double *p, double *rW,double *iW)
{
int      t,j;
double   frequency, const_term;

         for(j=0; j<fft_Items; j++)
         {
                  rW[j] = 0.0;
                  iW[j] = 0.0;

                  for(t=0;t<lNumItems; t++)
                  {
                           frequency = (2.0*M_PI*(j+1)*(t+1))/lNumItems;
                           rW[j] += p[t] * cos(frequency);
                           iW[j] += p[t] * sin(frequency);
                  }

                  const_term       = 2.0*M_PI*lNumItems;
                  rW[j]      /= sqrt(const_term);
```

```
                    iW[j]        /= sqrt(const_term);
                                  /*   iW[j]  = -iW[j];  */

   }

}/* END OF FFT ROUTINE */

void prdgm(long fft_Items,double *rW,double *iW,double *pgrmii)
{
int i;

          for(i=0; i<fft_Items; i++)
                    pgrmii[i] = (rW[i] * rW[i]) + (iW[i] * iW[i]);

} /* END OF PRDGM ROUTINE */

void crspgrm(long fft_Items,double *rW1,double *rW2,double *iW1,double *iW2,
                              double *rpgrm12,double *ipgrm12)
{
int i;

          for (i=0; i<fft_Items; i++)
          {
                    rpgrm12[i] = (rW1[i]*rW2[i]) + (iW1[i]*iW2[i]);
                    ipgrm12[i] = (iW1[i]*rW2[i]) - (rW1[i]*iW2[i]);
   }

} /*END OF CRSPGRM ROUTINE */

void Print_Err(void)
{
          printf("ERROR : CANNOT MALLOC SPACE **** \n");
  printf("*******TERMINATING THE PROGRAM******\n");
  exit(-1);
}

/*  ********END OF FFT.C **********/
```

```
/* ********************************************************************** */
/* Title : GDXS.C                                                          */
/* Date Written  : September 28, 1993                                      */
/* Date Modified : September 28, 1994                              */
/* Author : Milan Mangeshkar                                               */
/*                                                                 */
/* This program finds the Preliminary X's for the time delay              */
/* ********************************************************************** */
/* This program needs the foll. arguments : (1) # of Data points          */
/*                                          (2) Bandwidth                         */
/*                                          (3) # of simulations                  */

#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include "C:\USERS\MILAN\SOURCE\GDXS.H"

/* HASH DEFINES */

#define EPSILON             1.0e-10

#define READMODE      "r"
#define WRITEMODE     "w"

#define IPFL_PFX      "C:\\USERS\\MILAN\\PARGD\\FFT\\FFT"
#define OPFL_PFX      "C:\\USERS\\MILAN\\SIM2AV8\\XS\\GDXS"
#define PRIFILENM     "C:\\USERS\\MILAN\\SIM2AV8\\XS\\GDXS.PRI"

/* FILEPOINTERS ARE */

FILE    *fpOutFile;
FILE    *fpPriFile;

void main (int argc, char **argv)
{
short   sNumItems, sCtr = 0;
int     ibandwidth;
long    lNumDataPts;
char    szIpFileName[255],szOpFileName[255];

        if(argc==4)
        {
                lNumDataPts     = atol(argv[1]);  /* # of data points */
                ibandwidth  = atoi(argv[2]);  /* for bandwidth   */
                sNumItems       = atoi(argv[3]);  /* for # of simulations */

                if ((fpPriFile = fopen(PRIFILENM, WRITEMODE)) == NULL)
                {
                        printf("ERROR : COULD NOT OPEN FILE **GDXS.PRI**\n");
                        exit(-1);
```

```
                }

                while (sCtr++ < sNumItems)
                {
                        sprintf(szIpFileName,"%s.%d", IPFL_PFX, sCtr);
                        sprintf(szOpFileName,"%s.%d", OPFL_PFX, sCtr);
                        printf("ITERATION #:%d, %s, %s\n",sCtr, szIpFileName, szOpFileName);
                        fprintf(fpPriFile, "*******************************************\n");
                fprintf(fpPriFile, "ITERATION  #%d,  %s  %s\n", sCtr, szIpFileName, szOpFileName);
                        Find_Xs(lNumDataPts,ibandwidth,szIpFileName, szOpFileName);
                }
        }
        else
        printf("INVALID ARGUMENTS GIVEN TO THE PROGRAM\n");

        printf("*PROGRAM TO FIND PRE. ESTIMATES COMPLETED SUCCESSFULLY *****\n");
         fclose(fpPriFile);
}

void Find_Xs(long lNumDataPts,int midItems,char *szIpFileName,char *szOpFileName)
{
int             i, iLoopvar, nobs;
int             bw;
static  short   sCallCt = 0;
float           halfbw;
long            lNumItems, fft_Items,mItems,nItems;
static double   *pgrmZZ;
static double   *rpgrmXY, *rpgrmXZ, *rpgrmZY,  *ipgrmXY, *ipgrmXZ, *ipgrmZY;
static double   *fZZ;
static double   *rfXY, *rfXZ, *rfZY,      *ifXY, *ifXZ, *ifZY;
static double   *rgXY,  *igXY;
double          taper;

FILE            *fpDatFile;

        bw      = 3*midItems;
        halfbw  = bw/midItems;

        /* OPEN THE DATA FILE AND THE OUTPUT FILE*/

        if ((fpDatFile = fopen(szIpFileName, READMODE)) == NULL)
                {
                        printf("ERROR : COULD NOT OPEN FILE **GDXS.IN**\n");
                        exit(-1);
                }

        if ((fpOutFile = fopen(szOpFileName, WRITEMODE)) == NULL)
                {
                        printf("ERROR : COULD NOT OPEN FILE **GDXS.OUT**\n");
                        exit(-1);
```

```
		}

	lNumItems	= lNumDataPts;
	fft_Items	= floor(lNumItems/2.0);
	mItems		= floor(fft_Items/bw);
	nItems		= midItems*mItems;

/*	**********************************
	* MALLOC SPACE if 1st invocation	*
	**********************************	*/

	 if (!sCallCt)
	 {
		sCallCt++;

		 if ((pgrmZZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((rpgrmXY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((rpgrmXZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((rpgrmZY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((ipgrmXY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((ipgrmXZ = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();
		 if ((ipgrmZY = (double *)malloc(fft_Items*sizeof(double)))==NULL)
			Print_Err();

	  if ((fZZ = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		if ((rfXY = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		if ((rfXZ = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		if ((rfZY = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();

		if ((ifXY = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		if ((ifXZ = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		if ((ifZY = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();

		if ((rgXY = (double *)malloc(nItems*sizeof(double)))==NULL)
			Print_Err();
		 if ((igXY = (double *)malloc(nItems*sizeof(double)))==NULL)
```

```
                    Print_Err();

   } /* End of mallocing space */

 /*     ***************************************************************
        * READ THE PERIODOGRAM & CROSS PERIOGRAMS FROM THE INPUT FILE *
        ***************************************************************   */

        iLoopvar = 0;
        while((!(feof(fpDatFile))) && (iLoopvar++ < fft_Items))
                fscanf(fpDatFile, "%d %lf %lf %lf %lf %lf %lf\n", &nobs,
                        pgrmZZ+iLoopvar-1,rpgrmXY+iLoopvar-1,ipgrmXY+iLoopvar-1,
                        rpgrmXZ+iLoopvar-1,ipgrmXZ+iLoopvar-1,rpgrmZY+iLoopvar-1,
                        ipgrmZY+iLoopvar-1);

        if (iLoopvar != fft_Items)
        {
                printf("ERROR: INCORRECT DATA FILE OR COMMAND LINE ARGUMENT\n");
                exit(-1);
        }

/*      *********************************************
        * GET THE SMOOTHED SPECTRAL DENISTY FOR Z   *
        *********************************************   */

        spectral_density(mItems,bw,halfbw,midItems,pgrmZZ,fZZ);

 /*     *********************************************
        * GET THE SMOOTHED CROSS SPECTRAL DENISTIES *
        *********************************************   */

        cross_spect_density(mItems,bw,halfbw,midItems,rpgrmXY,ipgrmXY,rfXY,ifXY);
        cross_spect_density(mItems,bw,halfbw,midItems,rpgrmXZ,ipgrmXZ,rfXZ,ifXZ);
        cross_spect_density(mItems,bw,halfbw,midItems,rpgrmZY,ipgrmZY,rfZY,ifZY);

/*      *********************************************
        * GET THE RESIDUAL CROSS SPECTRAL DENISTIES *
        *********************************************   */

        residual_spec_density(nItems,rfXY,rfXZ,rfZY,fZZ,ifXY,ifXZ,ifZY,rgXY,igXY);

/*      ************************************************
        * FIND TAUHAT THAT MAXIMIZES THE COVARIANCE *
        ************************************************   */

        find_tauhat(lNumItems,mItems,bw,halfbw,midItems,rgXY, igXY);

  /*    **********************
        *CLOSE ALL THE FILES  *
        **********************    */
```

```
        fclose(fpDatFile);
         fclose(fpOutFile);

} /* END OF MAIN ROUTINE */

void spectral_density(long mItems,int bw,float halfbw,int midItems,double *pgrm,double *fZZ)
{
int i,j,k,l,element;

        for(i=0; i<mItems; i++)
                for (j=0; j<midItems; j++)
                {
                        l       = midItems * i;
                         element = l+j;
                        fZZ[l+j] = 0.0;
                        for(k=0; k<bw; k=k+midItems)
                                fZZ[element]   = fZZ[element] + pgrm[(bw*i)+j+k];
                                fZZ[element]   = fZZ[element] / halfbw;
    }

}/* END OF SPECTRAL_DENSITY ROUTINE */

void cross_spect_density(long mItems,int bw,float halfbw,int midItems, double *rpgrm, double *ipgrm,
                                                double *rf12,double *if12)
{
int i,j,k,l,bw_mult_i, l_plus_j, element;

        for(i=0; i<mItems; i++)
  {     bw_mult_i = bw * i;
                for (j=0; j<midItems; j++)
                {
                        l = midItems * i;
                         l_plus_j = l+j;
                        rf12[l_plus_j] = 0.0;
                        if12[l_plus_j] = 0.0;
                        for(k=0; k<bw; k=k+midItems)
                        {
                                element = bw_mult_i + j + k;
                                rf12[l_plus_j]   = rf12[l_plus_j] + rpgrm[element];
                                if12[l_plus_j]   = if12[l_plus_j] + ipgrm[element];
                         }
                        rf12[l_plus_j]       = rf12[l_plus_j] / halfbw;
                        if12[l_plus_j]       = if12[l_plus_j] / halfbw;
                }
  }

} /* END OF CROSS_SPECT_DENSITY */
```

```
void residual_spec_density(long nItems,double *rfXY, double *rfXZ, double *rfZY, double *fZZ,double
                              *ifXY, double *ifXZ, double *ifZY, double *rgXY, double *igXY)
{
int i;

        for(i=0; i<nItems; i++)
        {
                rgXY[i] = rfXY[i] - (((rfXZ[i]*rfZY[i])-(ifXZ[i]*ifZY[i]))/fZZ[i]);
                igXY[i] = ifXY[i] - (((rfXZ[i]*ifZY[i])+(ifXZ[i]*rfZY[i]))/fZZ[i]);
  }

}/* END OF RESIDUAL_SPEC_DENSITY */

void find_tauhat(long lNumItems,long mItems,int bw,float halfbw, int midItems ,double *rgXY,double
*igXY)
{
int         i,j, max_flag, element;
int         iendnumber=200;
double       tau,tau2,taumax,qhat,qhatmax,wk,sumofwk;
double       A, B;
static double     *taulambda, *rP, *iP;
static    short   sCallCt = 0;

  if (!sCallCt)
        {
         sCallCt++;
                if ((taulambda = (double *)malloc(midItems*sizeof(double)))==NULL)
                        Print_Err();
                if ((rP = (double *)malloc(midItems*sizeof(double)))==NULL)
                        Print_Err();
                if ((iP = (double *)malloc(midItems*sizeof(double)))==NULL)
                        Print_Err();
  }

        fprintf(fpPriFile, " N\t  WK\t\t  TAU\t\t\t  QHAT\n");

        for(i=0; i<mItems; i++)
        {
                tau = -lNumItems;      taumax= -lNumItems;
                qhatmax = -999999999.0;

                /* FIND THE MIDDLE FREQUENCY */

                sumofwk=0.0;
                for(j=0; j<midItems; j++)
                        sumofwk = sumofwk + (bw*i)+halfbw+j;

                sumofwk = sumofwk/midItems;
                wk                = (2.0 * M_PI * sumofwk)/lNumItems;
```

```
		/* FIND THE MAXIMUM QHAT */

		while(tau < lNumItems)
		{
			A=0.0; B=0.0;
			 for(j=0; j<midItems; j++)
			{
				taulambda[j] = 0.0;  iP[j] = 0.0;
				taulambda[j] = (tau * 2.0 * M_PI * ((bw*i)+halfbw+j))/lNumItems;
				rP[j]       = 0.0;
				element  = (midItems*i)+j;
				rP[j] = rgXY[element]*cos(taulambda[j]) +
						igXY[element]*sin(taulambda[j]);
				iP[j] = igXY[element]*cos(taulambda[j]) -
						rgXY[element]*sin(taulambda[j]);
				A          = A + rP[j];
				 B         = B + iP[j];
			}

			A = A/midItems;
			B = B/midItems;

			qhat = A*A + B*B;

			if(qhat >= qhatmax)
			{
				qhatmax = qhat;
				taumax  = tau;
			}

			tau = tau + .1;

		} /* END OF WHILE ROUTINE */

		if (taumax < 0)
			tau2 = taumax + lNumItems;
		else
			tau2 = lNumItems - taumax;

		if (abs(tau2) < abs(taumax))
			taumax = tau2;

  fprintf(fpPriFile, "%3d\t %+8.5f\t %+8.3f\t\t %+8.3f\n", (i+1),wk,taumax,qhatmax);
  fprintf(fpOutFile, "%4d  %+8.5f    %+8.3f\n", (i+1),wk,taumax);

	} /* END OF i LOOP */

} /* END OF ROUTINE FIND_TAUHAT */

void Print_Err(void)
```

```
{
        printf("ERROR : CANNOT MALLOC SPACE **** \n");
  printf("*******TERMINATING THE PROGRAM******\n");
  exit(-1);
}
/* END OF GDXS.C PROGRAM
```

```
/* ******************************************************************           */
/* Title            : BOXCOX.C                                                 */
/* Date Written : JANUARY 15, 1993                                             */
/* Date Modified: NOVEMBER 5, 1994                                             */
/* Author                   : Milan Mangeshkar                                 */
/* This program uses the preliminary X's and finds a transformation            */
/* using BOXCOX technique such that the errors are distributed normally        */
/* Note :If the # of data points exceed 100 then change the Xmat in       */
/*   POLYREGR.C  routine appropriately                                         */
/* *********************************************************************/
/* The arguments required by the program are: (1) order of the model          */
/*                                            (2) # of preliminary X's    */
/*                                            (3) # of simulations             */

#include    <stdio.h>
#include <stdlib.h>
#include <math.h>

#include "C:\USERS\MILAN\SOURCE\BOXCOX2.H"

#define READMODE      "r"
#define WRITEMODE     "w"

#define  EPSILON      1.0e-10 /* THIS IS FOR CHECKING WHETHER VALUE = 0 */
#define NMAX_NOS      50      /* THIS NOS IS FOR TAYLORS SERIES EXPANSION */

#define LAMBDA1START        -1.0
#define  LAMBDA1END          1.01

#define IPFL_PFX      "C:\\USERS\\MILAN\\SIM2AV8\\XS\\GDXS"
#define OPFL_PFX      "C:\\USERS\\MILAN\\SIM2AV8\\UNWTED\\ORDER2\\BCI"
#define OUTFILENM     "C:\\USERS\\MILAN\\SIM2AV8\\UNWTED\\BOXCOX.OUT"
#define ESTFILENM     "C:\\USERS\\MILAN\\SIM2AV8\\UNWTED\\EST.FIL"

FILE     *fpOutFile;
FILE     *fpEstFile;

void main(int argc, char ** argv)
{
short    sNumItems, iOrder, sCtr = 0;
long     lNumItems;
char     szIpFileName[255],szOpFileName[255];

         if (argc==4)
         {
                  iOrder    = atoi(argv[1]);
                  lNumItems = atol(argv[2]);
                  sNumItems = atoi(argv[3]);
                  if ((fpOutFile = fopen(OUTFILENM, WRITEMODE)) == NULL)
                  {
```

```
                printf("Error2 : Could not open BOXCOX.OUT file\n");
                exit(-1);
            }
            if ((fpEstFile = fopen(ESTFILENM, WRITEMODE)) == NULL)
            {
                printf("Error2 : Could not open ESTIMATE.OUT file\n");
                exit(-1);
            }
            while (sCtr++ < sNumItems)
            {
                sprintf(szIpFileName,"%s.%d", IPFL_PFX, sCtr);
                sprintf(szOpFileName,"%s.%d", OPFL_PFX, sCtr);
                printf("Iteration:%d, %s, %s\n",sCtr, szIpFileName, szOpFileName);
                 fprintf(fpOutFile,"****************************\n");
                fprintf(fpOutFile,"Iteration:%d, %s, %s\n",sCtr, szIpFileName, szOpFileName);
                Find_Trans(sCtr,iOrder, lNumItems, szIpFileName, szOpFileName);
            }
        }
        else
            printf("Invalid arguments\n");

        printf("******* End of Boxcox Iteration Successfully ********\n");

        fclose(fpOutFile);
        fclose(fpEstFile);
}

void Find_Trans(short ctr,int iOrder, long lNumItems, char *szIpFileName, char *szOpFileName)
{

FILE            *fpDatFile;
FILE            *fpOutFileCI;
char            sym,YesFlag;
int             i,v, iLoopvar,iNrows, iNcols,nobs,case_nos,NN, half_eles, lambda1inv;
static double   *Xdat,*Wk,*Ydat,*Xbetahat,*zeta;
static double   *Predat, *Resdat;
static short    sCallCt=0;
static double   c2[70];
double          SSRes,theta,lbd,ubd, lambda1,lambda2,zlbd,zubd;

        /* open the input and output data files */

        if ((fpDatFile = fopen(szIpFileName, READMODE)) == NULL)
        {
            printf("Error1 : Could not open INPUT file\n");
            exit(-1);
        }

        if ((fpOutFileCI = fopen(szOpFileName, WRITEMODE)) == NULL)
        {
```

```
            printf("Error2 : Could not open BCI file\n");
            exit(-1);
    }

     if(iOrder > 4)
    {
            printf("Error : Program cannot fit a poly. reg of order > 4\n");
            exit(-1);
    }

    /*  # of observations define the rows for Xmatrix and order of the polynomial
        to be fitted defines the columns for the Xmatrix */

    iNrows = lNumItems;
    iNcols = iOrder + 1 ;

    /* allocate space for the input data initially*/

    if (!sCallCt)
    {
            if((Xdat    = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();
            if((Wk      = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();

            /*      This space is allocated for the transformed x-data */

            if((Ydat    = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();

            /*      This space is allocated for use in polynomial regression */

            if((Xbetahat = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();
            if((zeta     = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();
            if((Predat   = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();
            if((Resdat   = (double *)malloc(lNumItems*sizeof(double)))==NULL)
                    Print_Err();

            /* FORM THE 90*CHI-SQUARE VALUE TABLE */

            Form_c2(c2);

  }  /* END OF OUTERMOST IF STATEMENT */

/* read the input data */

        iLoopvar = 0;
```

```
	while((iLoopvar++ < lNumItems))
		fscanf(fpDatFile, "%d %lf %lf\n", &nobs,Wk+iLoopvar-1,Xdat+iLoopvar-1);

  if (iLoopvar != (lNumItems+1))
	{
		printf("Error : Incorrect data file or command line arguments\n");
		exit(-1);
	}

	if((BoxCox(lNumItems,iNrows,iNcols,Xdat,Wk,Xbetahat,zeta,Predat,Resdat,Ydat,&lambda1,&la
mbda2))!=0)
	{
		printf("Error: In function BoxCox\n");
		exit(-1);
	}

	YesFlag = 'Y';
	if ((LAMBDA1END-lambda1) <= EPSILON)
		YesFlag = 'N';

	if (YesFlag == 'Y')
	{
		if((PolyReg(iNrows,iNcols,Ydat,Wk,Xbetahat,zeta,Predat,Resdat,&SSRes)) != 0)
		{
			printf("Error: In function PolyReg, TERMINATING THE PROGRAM\n");
			exit(-1);
		}

   /* printf("COMPUTING THETA AND THE CONFIDENCE INTERVAL ........\n\n");*/
   /* v = degrees of freedom */

		v = iNrows - iNcols;

		/* list the various case_nos */

		if((lambda1 > EPSILON) && (lambda1 <1))			/* lambda1 = .a */
			case_nos = 1;
		else if (fabs(lambda1) <= EPSILON)		/*  lambda1 = log */
			case_nos = 2;
		else if((lambda1 >= 1) || (lambda1 <= -1))	/*  lambda1 = a or -a */
			case_nos = 3;
		else if((lambda1 < EPSILON) && (lambda1 > -1))  /*  lambda1 = -.a */
			case_nos = 4;
		else
			case_nos = 5;

		if(case_nos == 1)
			NN = ceil(1/lambda1);
		else
```

```
                    NN = NMAX_NOS;

    if((case_nos == 1) || (case_nos == 4))
                    lambda1inv = ceil(1/fabs(lambda1));

            fprintf(fpOutFile,    "lambda2=%lf\tlambda1=%lf\tSSRES=%+lf\tD.F(v)=%d\tcase=%d
NN=%d\n",lambda2,lambda1,SSRes, v,case_nos,NN);
            printf("lambda2=%lf\t              lambda1=%lf\t              SSRes=%8.3f\t
NN=%d\n\n",lambda2,lambda1,SSRes,NN);

    /* computing theta */

            half_eles = ceil(iNrows/2.0);

    for(i=0; i<iNrows; i++)
            {

        if((Compute_theta(NN,case_nos,v,lambda1,lambda1inv,Xbetahat[i],zeta[i],SSRes, &theta)) != 0)
                    {
                            printf("Error: In function Compute_theta_%d,TERMINATING THE
PROGRAM\n", case_nos);
                            exit(-1);
                    }

                      if((Compute_CI(NN,case_nos,v,lambda1,lambda1inv,c2,Xbetahat[i], zeta[i],
SSRes, &lbd, &ubd))!=0)
                    {
                            printf("Error:   In   function   Compute_CI,TERMINATING   THE
PROGRAM\n");
                            exit(-1);
                    }

                    theta = (theta-lambda2);
                    lbd   = (lbd  -lambda2);
                    ubd   = (ubd  -lambda2);

                    fprintf(fpOutFileCI, "%3d %8.6f % 8.3f % 8.3f % 8.3f % 8.3f\n",
                                            (i+1),Wk[i],Xdat[i],theta,lbd,ubd);

                    if(i==half_eles)
                    {
                            fprintf(fpEstFile,  "%3d\t%  3.2f\t%8.6f\t%  8.3f\t%  8.3f\t%  8.3f\t%
8.3f\n",
                                            ctr,lambda1,Wk[i],Xdat[i],theta, lbd, ubd);
                    }

        } /* end of the for loop */
    }
    else
  {    fprintf(fpEstFile, "%3d\n",ctr);
```

```
                printf("lambda1 = %+lf discard data set #%3d\n",LAMBDA1END,ctr);
  }

        fclose(fpDatFile);
        fclose(fpOutFileCI);

} /* end of main */

int BoxCox(long lNumItems, int iNrows, int iNcols, double *Xdat, double *Wk,double *Xbetahat,
                        double    *zeta,double    *Predat,double    *Resdat,double    *Ydat,    double
*lambda1conv,double *lambda2conv)
{

int             i,j,k,convflag1,convflag2;
double          log_gmean,gmean,SSRes,SSResprev,SSResprev2,diff;
double          lambda1,lambda1prev,lambda2,lambda2prev;
double          small_nos,lambda2end;
static double   *Vdat;
static short    sCallCt = 0;

        if (!sCallCt)
        {
          sCallCt++;
                if((Vdat = (double *)malloc(lNumItems*sizeof(double))) == NULL)
                        Print_Err();
  }

  convflag1 = 1;
  SSResprev2 = 999999999999.0;

        /*FIND THE SMALLEST NEGATIVE NUMBER*/

        small_nos = 0.0;
        for(i=0; i<lNumItems; i++)
        {
                if((Xdat[i] < 0.0) && (Xdat[i] < small_nos))
                        small_nos = Xdat[i];
        }

        if (small_nos < 0.0)
        {
                lambda2    = -small_nos + 0.5;
                lambda2end =  lambda2   + 0.5;
  }
        else
  {
                lambda2 = 0.0;
                lambda2end = 0.01;
        }
```

```
while((convflag1 == 1)&&(lambda2 < lambda2end))
        {
                SSResprev = 999999999999.0;

                /* find the geometric mean of the data */

                log_gmean = 0.0;            gmean = 0.0;
                for(i=0; i<lNumItems; i++)
                        if (fabs(Xdat[i])> EPSILON)
                                log_gmean = log_gmean + log(Xdat[i]+lambda2);

                log_gmean = log_gmean/lNumItems;
        gmean   = exp(log_gmean);

                convflag2 = 1;
                lambda1  = LAMBDA1START;

                while((convflag2 == 1) && (lambda1 < LAMBDA1END))
                {
                        /* transform Xdat to Ydat = ((Xdat+lambda2)**lambda1-1)/lambda1 */

                        /* compute  Vdat = ydat / exp((lambda1-1.0)*log(gmean)) */

                        if (fabs(lambda1) <= EPSILON)   /* i.e when lambda1 is zero */
                        {
                                for(i=0; i<lNumItems; i++)
                                {
                                        if (fabs(Xdat[i]) <= EPSILON)
                                                Vdat[i] = 0.0;
                                        else
                                                Vdat[i] = log(Xdat[i]+lambda2) * gmean;
                                }
                        }
                        else                            /* i.e when lambda1 is other than 0 */
                        {
                                for(i=0; i<lNumItems; i++)
                                {
                                        if (fabs(Xdat[i]) <= EPSILON)
                                                Vdat[i] = 0.0;
                                        else
                                                Vdat[i] = (pow((Xdat[i]+lambda2), lambda1) - 1.0) /
(lambda1 * pow(gmean,(lambda1-1.0)));
                                }
                        }

                        if((PolyReg(iNrows,iNcols,Vdat,Wk,Xbetahat,zeta,Predat,Resdat,&SSRes))   !=
0)
                        {
                                printf("Error: In function PolyReg\n");
                                exit(-1);
```

```
                }

                if(SSRes <= SSResprev)
                {
                        SSResprev  = SSRes;
                        lambda1prev = lambda1;
                }
                else
                        convflag2 = 0;

                lambda1 = lambda1 + .1;


        } /* end of inner while loop */


        if (SSResprev <= SSResprev2)
        {
                SSResprev2   = SSResprev;
                *lambda1conv = lambda1prev;
                *lambda2conv = lambda2;
        }
        else
                convflag1 = 0;

        lambda2 = lambda2+.10;

} /* END OF THE OUTER WHILE LOOP */

if((-0.1 < *lambda1conv) && (*lambda1conv < 0.1))
        *lambda1conv = 0.0;

if (fabs(*lambda1conv) <= EPSILON)  /* i.e when lambda1 = 0 */
{
        for(i=0; i<lNumItems; i++)
        {
                if (fabs(Xdat[i] <= EPSILON))
                        Ydat[i] = Xdat[i];
                else
                        Ydat[i] = log(Xdat[i] + *lambda2conv);
        }
}
else
{
        for(i=0; i<lNumItems; i++)
        {
                if (fabs(Xdat[i] <= EPSILON))
                        Ydat[i] = Xdat[i];
                else
```

```
                                        Ydat[i] = (pow((Xdat[i]+ *lambda2conv), *lambda1conv) - 1.0) /
*lambda1conv;
                    }
        }

        return(0);

}/* end of BoxCox fuction */

int Compute_theta(int NN,int case_nos,int v,double lambda1,int lambda1inv,
            double Xbetahat,double zeta,double SSRes,double *theta)
{

int j;
double Exp_Value, constterm;

        *theta = 1.0;

        for(j=1; j<(NN+1); j++)
        {
                switch (case_nos)
                {
                        case 1 : Find_Const_1(j,lambda1inv,&constterm); /* for lambda1=.a */
                                        break;
                        case 2 : Find_Const_2(j,&constterm);        /* for lambda1=0  */
                             break;
                        case 3 : Find_Const_3(j,lambda1,&constterm);    /* for lambda1=a or -a */
                                        break;
                        case 4 : Find_Const_4(j,lambda1inv,&constterm); /* for lambda1=-.a */
                                        break;
                        default  : printf("Incorrect case_nos=%d\n",case_nos);
                               break;
                } /* end of switch */

                /* compute E[(Ydat)**j]  */

                Exp_Value = 0.0;
                if((Compute_Tn(j,v,Xbetahat,zeta,SSRes,&Exp_Value))!=0)
                {
                        printf("Error: In function Compute_Tn\n");
                        exit(-1);
                }

                *theta = *theta + constterm * Exp_Value;

        }/* end of for loop */

        return(0);

}/* end of compute_theta routine */
```

```
void Find_Const_1(int j,int lambda1inv,double *constterm)
{
int                diff;
double  factlambda1, factj, factdiff;

/*  this routine is for lambda = .a */

          factorial(lambda1inv, &factlambda1);

          diff  =  lambda1inv - j;
          factorial(j, &factj);
          factorial(diff, &factdiff);

          *constterm = factlambda1 / ( factj * factdiff * pow(lambda1inv,j) );

}/* end of Find_Const_1 routine */

void Find_Const_2(int j,double *constterm)
{

/*        this routine is for lambda = log */

double   factj;

                   factorial(j, &factj);

                   *constterm = 1/factj;

}/* end of Compute_theta_2 routine */

void Find_Const_3(int j,double lambda1, double *constterm)
{
int                k;
double   factj, prodct;

/* this routine is for lambda1 = -a or a */

          factorial(j, &factj);
          prodct = 1.0;
          for(k=1; k<(j+1); k++)
                   prodct = prodct * (1 - ((k-1)*lambda1));

          *constterm = prodct / factj;

}/* end of Compute_theta_3 routine */

void Find_Const_4(int j,int lambda1inv,double *constterm)
{
```

```
int                 k;
double   factj,num;

/* this routine is for lambda1 = -.a */

          factorial(j, &factj);
          num = 1.0;
          for(k=1; k<j+1; k++)
          num = num * (lambda1inv+k-1);

          *constterm = num / (factj * pow(lambda1inv,j));

}/* end of Compute_theta_4 routine */

int Compute_Tn(int nn,int v,double Xbetahat,double zeta,double SSRes,double *Exp_Value)
{
int       rem, k, diff, n;
double      vv;
long double num, deno;
double      fact2n , fact2k, factdiff, gmvv1, gmvv2;

          *Exp_Value = 0.0;
          n = floor(nn/2);

          vv = v/2.0;
          gamma_func(vv, &gmvv1);

  if(nn == (2*n+1))     /* implies n is odd */
          {
                    factorial((2*n+1), &fact2n);

                    for(k=0; k<n+1; k++)
                    {
                              diff = n - k;
                              factorial((2*k+1), &fact2k);
                              factorial(diff,  &factdiff);
                              vv = (v/2.0) + diff;
                              gamma_func(vv, &gmvv2);

                              num = fact2n * pow(Xbetahat,(2*k+1)) *
                                  pow((SSRes*(1-zeta)),diff) * gmvv1;

                              deno = fact2k * factdiff * pow(4,diff) * gmvv2;

                              *Exp_Value   = *Exp_Value + num / deno;

                    }/* end of for loop */

  } /* end of odd routine */
          else          /* n is even number */
```

```
        {
    factorial((2*n), &fact2n);
                for(k=0; k<n+1; k++)
                {
        diff = n - k;
                        factorial((2*k), &fact2k);
                        factorial(diff, &factdiff);

                        vv = (v/2.0) + diff;
                        gamma_func(vv, &gmvv2);

                        num = fact2n * pow(Xbetahat,(2*k)) *
                            pow((SSRes*(1-zeta)),diff) * gmvv1;

                        deno = fact2k * factdiff * pow(4,diff) * gmvv2;

       *Exp_Value   = *Exp_Value + (num / deno);

                }/* end of for loop */

        } /* end of else */

        return(0);

} /* end of Compute_Tn routine */


int Compute_CI(int NN, int case_nos,int v,double lambda1,int lambda1inv,double *c2,
                              double Xbetahat, double zeta,double SSRes,double *lbd, double *ubd)
{
int       i,j;
double  c1,sigsq, product, mu, thetahat;


        *ubd = 0.0;          *lbd = 0.0;

  /*      c1 = 2.71;      10% chi-square value for 1 d.f */

        c1 = 5.0239;      /* .025% chi-square value for 1 d.f */

        for(i=0; i<30; i++)
        {
        sigsq = 0.0;
                sigsq = (SSRes / c2[v-1]) * ((i+1) / 30.0);

                for(j= 0; j<30; j++)
                {
        product = 0.0;  mu = 0.0;
                        product = zeta * c1 * sigsq ;
                        mu      = Xbetahat - sqrt(product)+2.0*sqrt(product)*((j+1)/30.0);
```

```
        if((Compute_Thetahat(NN,case_nos,lambda1,lambda1inv,mu,sigsq,&thetahat))!=0)
                        {
                        printf("Error: In function Compute_Thetahat\n");
                        exit(-1);
                        }

                        if((i==0) && (j==0))
                        {
                                *ubd = thetahat;
                                *lbd = thetahat;
        }

                        if(thetahat > *ubd)
                                *ubd = thetahat;

                        if(thetahat < *lbd)
         *lbd = thetahat;
      }
         }
    return(0);

} /* end of compute_CI routine */

int Compute_Thetahat(int NN,int case_nos,double lambda1,int lambda1inv,
               double mu,double  sigsq, double *thetahat)
{
int          j;
double       constterm;
static double  *Exp_Value;
static short   sCallCt = 0;

        if (!sCallCt)
        {
        sCallCt++;
                if((Exp_Value = (double *)malloc(NN*sizeof(double)))==NULL)
                        Print_Err();
        }

        for(j=0; j<NN; j++)
                Exp_Value[j] = 0.0;

        if((Compute_Exp(NN, mu, sigsq,Exp_Value))!=0)
        {
                printf("Error: In function Compute_Exp\n");
                exit(-1);
        }

  *thetahat = 1.0;
```

```
        for(j=1; j<(NN+1); j++)
        {
                switch (case_nos)
                {
                        case 1 :  Find_Const_1(j,lambda1inv,&constterm);/*for lambda1=.a */
                break;
                        case 2 :  Find_Const_2(j,&constterm);                         /*for
lambda1=0  */
                        break;
                        case 3 :  Find_Const_3(j,lambda1,&constterm);  /*for lambda1=a or -a */
                                        break;
                        case 4 :  Find_Const_4(j,lambda1inv,&constterm);/*for lambda1=-.a */
                break;
                        default:   printf("Incorrect case=%d\n", case_nos);
                                    break;
                } /* end of switch */

    *thetahat = *thetahat + (Exp_Value[j-1] * constterm);
  }
  return(0);
}/* end of Compute_thetahat */

int Compute_Exp(int NN,double  mu, double sigsq,double *Exp_Value)
{
int i;

        Exp_Value[0] = mu;

        if(NN > 1)
                Exp_Value[1] = mu * mu + sigsq;

        if(NN > 2)
  {
                for(i=2; i<NN; i++)
                        Exp_Value[i] = sigsq * i * Exp_Value[i-2] + mu * Exp_Value[i-1];
  }
  return(0);
} /* end of Compute_Exp routine */

void factorial(int n, double *factn)
{
int i;

        *factn = 1.0;

        if( n > 1)
        {
        for(i=1; i<n+1; i++)
                        *factn = *factn * i;
```

```
        }

} /* end of factorial routine */

void gamma_func(double vv, double *gmvv)
{
int i, n;
double   sum;

        if(vv == 0.5)
                *gmvv = 2.0*0.8862269;  /* gamma of half is sqrt of pi */
        else
        {
                n = ceil(vv);
                if (vv == n)
                {
                        *gmvv = 1.0;
                        if(vv > 1.0)
                        {
                                for(i=1; i<n; i++)
                                        *gmvv = *gmvv * i;
                        }
                }
                else
                {
                        *gmvv = 2.0 * 0.8862269;
                        for(i=1; i<n; i++)
                                *gmvv = *gmvv * (i-0.5);
    }

        }/* end of 1st else */

}/* end of gamma_func */

void Form_c2(double *c2)
{

        /*      THESE ARE THE .975% CONFIDENCE COEFFICIENTS */

        c2[0]  = 0.0010;  c2[1]  = 0.0506;  c2[2]  = 0.2158;  c2[3]  = 0.4844;
        c2[4]  = 0.8312;  c2[5]  = 1.2373;  c2[6]  = 1.6899;  c2[7]  = 2.1797;
        c2[8]  = 2.7004;  c2[9]  = 3.2470;  c2[10] = 3.8157;  c2[11] = 4.4038;
        c2[12] = 5.0087;  c2[13] = 5.6287;  c2[14] = 6.2621;  c2[15] = 6.9077;
        c2[16] = 7.5642;  c2[17] = 8.2307;  c2[18] = 8.9065;  c2[19] = 9.5908;
        c2[20] = 10.2829; c2[21] = 10.9823; c2[22] = 11.6886; c2[23] = 12.4012;
        c2[24] = 13.1197; c2[25] = 13.8439; c2[26] = 14.5734; c2[27] = 15.3079;
        c2[28] = 16.0471; c2[29] = 16.7908; c2[30] = 17.5387; c2[31] = 18.2908;
        c2[32] = 19.0467; c2[33] = 19.8063; c2[34] = 20.5694; c2[35] = 21.3359;
        c2[36] = 22.1056; c2[37] = 22.8785; c2[38] = 23.6543; c2[39] = 24.4330;
        c2[40] = 25.2145; c2[41] = 25.9987; c2[42] = 26.7854; c2[43] = 27.5746;
```

```
        c2[44] = 28.3662; c2[45] = 29.1600; c2[46] = 29.9562; c2[47] = 30.7545;
    c2[48] = 31.5549; c2[49] = 32.3574;

} /*END OF FUNCTION FORM_C2 */

void Print_Err(void)
{
        printf("malloc returned error\n");
        exit(-1);
}


/* END OF BOXCOX.C PROGRAM */
```

Vita

of

Milan Mangeshkar

Personal : Born on April 16, 1963 in Bombay India

Education : **Ph.D. in Statistics,** February 1995
Virginia Tech, Blacksburg, VA
Dissertation : A Procedure For Estimation Of Partial Group Delay With Applications To Small Samples

M.S. in Statistics, May 1986
University Of Bombay, India

B.S. in Statistics, May 1984
University Of Bombay, India

Work Experience : **Graduate Teaching Assistant,** Department of Statistics, Virginia Tech, August 1990 - August 1994

Programmer Analyst, TELCO, India, July 1986-July 1989

Honors And Activities : Awarded the best student presenter at Virginia Academy of Sciences conference held in April 1994,
Member since 1991 of Mu Sigma Rho, a statistics honor society,
Member of American Statistical Association