Pure

Bangor University

DOCTOR OF PHILOSOPHY

Investigation of real-time spectral analysis techniques for use with pulsed ultrasonic doppler blood flow detectors.

Ruano, Maria da Graca Cristo dos Santos Lopes

Award date:
1992

Awarding institution:
Bangor University

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 10. Jul. 2024

INVESTIGATION
OF REAL-TIME SPECTRAL ANALYSIS TECHNIQUES
FOR USE WITH PULSED ULTRASONIC DOPPLER BLOOD-FLOW DETECTORS

by

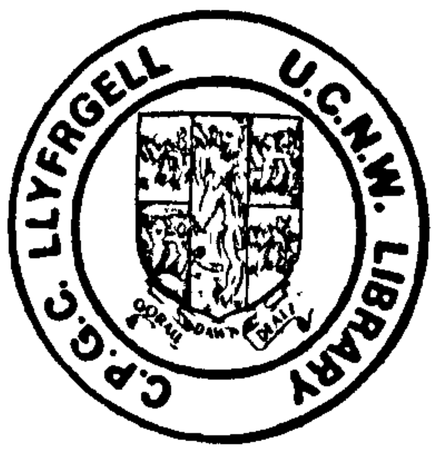
Maria da Graça Cristo dos Santos Lopes Ruano

Thesis submitted in candidature to the
UNIVERSITY COLLEGE OF NORTH WALES

for the degree of
DOCTOR OF PHILOSOPHY

School of Electronic Engineering Science
University College of North Wales, Bangor

February 1992



À Carina Alexandra e António

com muito amor

ACKNOWLEDGMENTS

I would like to express my sincere thanks and gratitude to Mr. Peter Fish for his invaluable advice, motivation, friendship, and supervision throughout the course of this study. I am also indebted to Prof. Peter Fleming for his guidance, assistance and support.

Appreciation and acknowledgments are due to all the staff and students of the S.E.E.S. To Prof. John O'Reilly for his overall support and encouragement. To Sian Hope, Keith France, and David Whitehead for their expert advice on software. To my colleagues from the Bangor Medical Ultrasound Group, in particular to Carlos Bastos for his support and productive discussions, and, to Fabian Garcia Nocetti for his helpful advice and joint work in continuing the parallel implementation of the modified covariance algorithm. To all portuguese students for their friendship.

I am also grateful to Prof. Anibal Duarte from the University of Aveiro, Portugal, whose advice and guidance was decisive to my coming to Bangor.

Special thanks and love to my parents, Arlete and Américo, and my sister, Maria da Conceição for their words of encouragement. To my parents-in-law for their continuous support.

I also owe a great debt of gratitude to my husband, António, for the many fruitful discussions, and above all to my daughter Carina for her cheerful forbearance in putting up with hours of neglect.

And last but not least, I would like to acknowledge the University of Aveiro, the Medical Research Council, the Science and Engineering Research Council, and, the 'Junta de Investigação Científica' (Portugal), for their financial support.

ABSTRACT

The goals of the work described here were the development of a method of selection of spectral estimation for use with pulsed Doppler ultrasonic blood flow instruments, and the use of this method to select an estimator and its implementation in a form suitable for real-time applications.

A study of estimation accuracy of the mean frequency and bandwidth using a number of spectral estimators was carried out. Fourier based, parametric, and, minimum variance estimators were considered. A Doppler signal simulator was developed to allow the accuracy tests required.

A method of selection of spectral estimators based on the accuracy of estimation of decisive signal parameters, under the constraint of low computational complexity has been proposed. This novel cost/benefit criterion, allows the possibility of weighting appropriate to estimator (mean frequency and bandwidth) and signal frequency importance (across the range of signal characteristics). For parametric spectral estimators, this criterion may also be used to select model order, leading to lower orders than FPE, AIC and CAT criteria. Its use led to the selection of a 4^{th} order modified covariance parametric method.

A new version of the modified covariance method for spectral estimation of real signals was developed. This was created with a view to the parallel partitioning of the algorithm for parallel implementation on a transputer-based system, using OCCAM. A number of parallel topologies were implemented. Their performance was evaluated considering estimation of a single, and a sequence of Doppler signal segments, revealing the feasibility of these parallel implementations to be achieved in real-time.

CONTENT

CHAPTER 1

CHAPTER 1

Introduction

1.1. GENERAL OVERVIEW

1.1.1. Motivation

Doppler ultrasound techniques have been increasingly used in noninvasive assessment of circulatory disease providing means of diagnosing and monitoring stenosed arteries [1].

Doppler ultrasound devices detect movement of an object in the path of the ultrasonic beam from the Doppler shift in the frequency of the signal scattered or reflected from the object. In this case, the objects are blood particles, essentially the red blood cells, which under pathological conditions have altered velocity patterns [2]. The blood velocity within an artery is periodic with the cardiac cycle, and the variation of blood velocity in each cardiac cycle (velocity waveform) changes with vessel disease and therefore contains diagnostically useful information. In addition, local to an intrusive vessel lesion there is disturbed flow which results in an increase in the range of Doppler shift frequency (spectrum broadening) [3]. Any Doppler signal analysis scheme must deal with both of these aspects.

The mathematical treatment of the interaction between the received ultrasonic signal from blood velocity measurements and the blood scattering particles has been developed in different ways. Angelsen [4] is one of the authors who developed a comprehensive analysis of this

relationship, assuming that the scattering of ultrasound from blood arises from random fluctuations of bulk elastic modulus intensity and density. This and the other proposed models, result in the conclusion that the Doppler signal from undisturbed flow is random, with a Gaussian probability density function, and a spectrum determined by the range of magnitudes of the velocities of the streamlines crossing the sample volume, and the sample volume characteristics.

Thus the Doppler signal from ultrasonic Doppler instruments is a random signal with a Gaussian probability density function in undisturbed flow and with a time-varying frequency spectrum determined by the time-varying-blood velocity field and the size and position, within the velocity field, of the instrument resolution cell [5]. If the heart rate is steady and there are no changes in the physiological state the signal is cyclo-stationary [3].

Typically two types of Doppler ultrasound devices are referred to in the literature. The continuous-wave and the pulse-wave instruments. The former has major disadvantage, the inability to distinguish signals from vessels at different levels, which is overcome by the use of pulsed Doppler ultrasound devices [6].

Analysis of the Doppler signals involves a sequence of procedures starting from the collection of clinical data by the technician till the application and consequent interpretation of the estimated Doppler spectrum. Throughout all these stages measurement errors are introduced [7]. Some of them are recognized and may be overcome by application of suitable spectral analysis techniques, namely windowing and spectral averaging.

Conventional spectral analysis of the Doppler signal uses a Fast Fourier Transform on sequential or overlapping windowed signal segments. The limitations of these techniques to estimate Doppler spectrum are extensively published. Some examples of comparative analysis of the results obtained using Fourier transformed windowed Doppler signals and other spectral estimation techniques are presented in [8 - 12].

Another class of spectral estimator methods is comprised of the modern parametric methods. Their potential advantage is the frequency resolution, while their major drawbacks are the computational burden and also the selection of a model order or model orders which can accurately represent the modelled signal. The commonly referred model order selector criteria perform the selection on the basis of full spectrum estimation [12 - 15], and, often lead to poor estimates of the correct model order when used with signals whose model order is known [16] [17].

These limitations induced the development of other spectral estimation methods than the parametric ones, such as the application of time-domain techniques to estimate spectral parameters [18].

Nevertheless there is still much work carried out on accurate estimation of the spectrum as a whole or of specific spectral parameters using parametric models, as may be observed on some recent publications on mean frequency and bandwidth estimation of narrow band signals [19] [20].

The spectral estimation methods above referred are essentially digital methods and are often software based. Another important strategy to estimate the Doppler spectrum is obtained by a mixture of software and hardware implementation. In this field, the application of Digital Signal Processors (DSPs) is commonly found (Eg. work reported by Schlindwein and Evans [21] on the feasibility of real-time autoregressive spectral estimation).

The demand for increasing processing power, together with the availability of hardware technology enabling elaborate calculations to be computed rapidly, led signal processing towards the use of multi-processor systems. For instance, the computation of Fourier Transforms using transputer arrays [22], and in parametric estimation, the development of fast algorithms for solution of Toeplitz systems of equations [23] and the implementation of a least squares algorithm in systolic array [24].

The above referred environment of Doppler signal processing aimed at cardiovascular disease detection motivated the development of the work described in this thesis.

1.1.2. Goals

- Development of a flexible but objective method of selection of spectral estimators based on measured estimation accuracy and the requirements of the user,taking also into account the cost of the implementation.
- Use of this method to select a spectral estimator for the measurement of mean frequency and bandwidth. The development of a Doppler signal simulator to allow the accuracy tests required by the above.
- Development of the selected estimator into a form suitable for real-time application using parallel processing techniques. Its implementation on a transputer-based system.

1.2. THESIS OUTLINE

Chapter 1 has given a general overview of some important features that motivated the usage of spectral analysis of Doppler shifted ultrasound scattered from blood to increase the sensitivity to blood vessel disease detection, describing the goals of the work developed. It also includes a thesis outline and a description of the major contributions achieved during this work.

Chapter 2 provides necessary background material for the three major disciplinary areas involved in this project, this is, Doppler ultrasound, spectral analysis, and, parallel processing. A brief description of Doppler ultrasound instruments, the relationship of the Doppler signal with the blood velocity, and the Doppler spectrum analysis techniques are included. A general overview of the most commonly used relationships between time and frequency representations of random signals, the Fourier based spectral estimation methods and typically employed windows, and the general concepts common to parametric spectral estimation methods are also presented. The parallel processing section gives a general idea about the steps need for the development and implementation of a numerical process on a multi-processor system, enables familiarization with the terminology employed in this field, and describes the transputer and its related parallel programming language - OCCAM.

Chapter 3 is concerned with the implementation and inspection of the Doppler signal simulations required to measure the performance of spectral estimators. Three algorithms were implemented, two of them based on reports of earlier work done in this field, and the other, developed from one of the previous algorithms to enable faster computations. Analysis of the behaviour of the amplitude distribution of the Doppler signal is presented and the criterion for selection of one of the simulation algorithms is indicated. Evaluation of the frequency content of the Doppler signal simulation obtained from the selected simulation algorithm is also presented.

Chapter 4 describes the measurements to be applied to assess spectral estimation accuracy and the selection of a suitable ensemble size for expected value estimation.

Chapter 5 starts with a more advanced description of the spectral estimation methods employed on this research work, giving special emphasis to the parametric methods (autoregressive, moving average, and autoregressive moving average methods) and the minimum variance method. The autoregressive methods comprise the Yule-Walker, Burg, and modified covariance spectral estimators. The Durbin method (a moving average method), although not

implemented alone as a Doppler spectral estimator, is also described. The least squares modified Yule-Walker equations and the Mayne Firoozan methods are the autoregressive moving average methods employed. This chapter deals also with the problem of parametric model order selection and spectral estimator selection. Conventional model order selection criteria are described. A new criterion enabling the selection of both a spectral estimation method (from parametric or non-parametric methods), and a model order(s) when parametric spectral estimators are considered, is proposed. The selection is constrained by the requirement of accurate spectral mean frequency and bandwidth estimation at low computational complexity. The results obtained when this new criterion is applied to the estimators described are reported.

Chapter 6 addresses the parallel implementation of the spectral estimation method selected in the previous stage of the work, the modified covariance method. Two approaches are presented. The first one, considering the estimation of the spectrum of a single Doppler data segment, is fully studied and described in this thesis. In this context, a description of the development of a parallel version of the modified covariance algorithm and comparison of its computational complexity behaviour with those attained by published versions of the same algorithm is presented; two approaches of parallel partitioning the modified covariance algorithm are introduced; different schemes of achieving the parallel implementation of the algorithm constituent blocks together with their performance results are described; results of parallel performance evaluation of the modified covariance spectral estimator as a single parallel program are given. The second approach, comprising the concurrent spectral estimation of a number of Doppler data segments, is being currently developed in collaboration with a research work colleague. It is included in this thesis for completeness of the subject of the parallel implementation of the modified covariance algorithm.

In Chapter 7 general conclusions from this work are given. Suggestions for future work are also described.

Throughout this thesis statistical concepts are used as important tools to judge and make decisions in comparative studies. Although important, their theoretical basis has not been included in this thesis, and, whenever necessary, a suitable reference is given.

Notational consistency has not been always easy to achieve throughout the thesis, since different disciplinary areas are involved. However, within each chapter, an attempt is made to maintain one symbol to represent one quantity. Typically the *italic* form is used to introduce a

specific term for the first time and to identify software program names. The round brackets are usually used with continuous signals, while the squared brackets are used with discrete signals.

1.3. CONTRIBUTIONS OF THIS RESEARCH WORK

In addressing the problem of increasing the sensitivity to blood vessel disease detection using spectral analysis of Doppler shifted ultrasound scattered from blood, this thesis makes the following specific contributions:

- The experimental study of the statistical behaviour of the mean frequency and bandwidth spectral estimators, obtained from different spectral estimator methods, and in the case of parametric spectral estimators, different model orders. This study required selection and implementation of a Doppler signal simulation algorithm.

- The development of a novel criterion of parametric model order selection, and, selection of spectral estimation methods (parametric or non-parametric), the cost/benefit criterion. The proposed criterion takes into account the cost (or computational complexity) of the spectral estimation algorithm employed, and allows different weights to be given to mean frequency and bandwidth estimation errors across a range of different signals. These signals are characterized by Gaussian probability density function, Gaussian spectral shape, mean frequencies varying from 1KHz to 8KHz, and half bandwidths of 5% (except for the 1KHz mean frequency signal), 10% and 20% of the mean frequency.

- The development of a new version of the modified covariance method for spectral estimation of real signals. This was created with a view to enable an efficient parallel partitioning for parallel implementation on a multi-processor system.

- The implementation of the new version of the modified covariance method on a transputer-based system using the parallel programming language, OCCAM. Each main computational block of the algorithm is implemented using different parallel architecture topologies, for which, performance measurements are taken. An overall OCCAM program is presented to enable estimation of narrow-band spectra of stationary Doppler ultrasound signal segments.

Considerations about feasibility of this parallel implementation are presented on basis of the performance measurements achieved.

CHAPTER 2

CHAPTER 2

Background

2.1. INTRODUCTION

The purpose of this chapter is to provide the reader the essential background of the major areas of this multidisciplinary project and in particular to enable a clear understanding of the motivations and goals described in chapter 1.

The first section of this chapter, concerned with the biomedical component of this research program, describes the Doppler ultrasound instruments, some definitions and physics theory related to the Doppler ultrasound signal model, the origin of the Doppler spectrum and some details of Doppler signal processing and analysis.

The second section, introduces the basic theory related to the analysis of the frequency spectrum of a signal and describes common terminology in this field. It also distinguishes two major spectral estimation techniques, those based on Fourier transformed signals, i.e. the *traditional or conventional spectral estimation methods*, from those sometimes called the *modern methods*, which are the parametric or model based estimators.

The third and last section of this chapter, introduces the area of parallel processing. The most common parallel processing architectures are described. The transputer and the OCCAM programming language are presented. Usual performance metrics employed in parallel processing are described.

2.2. DOPPLER ULTRASOUND

Doppler ultrasound techniques provide an important means for the noninvasive detection and measurement of the velocity of moving structures, and particularly blood, within the body. They are frequently used to assess circulatory diseases by detection of abnormalities in arteries, veins, and heart. An example of these diseases is plaque on the vessel wall, which reduces flow and pressure downstream in the vessel. If a severe reduction on the vessel lumen is observed, a change in the blood velocity waveform occurs.

The understanding of the mechanism by which the Doppler signal is produced, the relationship of the Doppler signal with the pulsatile flow patterns, and, the means of analysis and interpretation of the Doppler spectrum, are necessary in order to develop and improve analysis techniques to enable extraction of diagnostic information.

2.2.1. Doppler Ultrasound Instruments

Doppler ultrasound instruments are based on the idea of detecting the Doppler shift in the frequency of the ultrasound that is reflected or backscattered from moving structures. In the case under examination the scattering structures are the blood particles.

A signal with the *Doppler shift frequency* is obtained from a Doppler demodulator which mixes the received signal and a reference signal at a transmitted frequency in a non-linear device (usually a multiplier) and low pass filters its output to remove signal components at the transmitted frequency and above.

The Doppler shift (or difference) frequency (f_d) is defined as the difference between the received (f_r) and the transmitted frequency (f_t), the received frequency being the sum of the reference frequency with a frequency component proportional to the velocity of the blood particle (v). If the ultrasound is received in a direction 180° from the direction of transmission, then:

$$f_d = f_t - f_r = (2.\frac{f_t}{c}).v.\cos\theta \qquad (2.1)$$

where c the velocity of sound in tissue and θ the angle between the ultrasound beam and the direction of motion of the target [25].

In the above expression the speed of sound and the transmitted frequency are known in a particular situation, allowing the substitution of the parameters inside the brackets by a constant. Thus the relationship between the Doppler frequency and the velocity of the blood target is evident, making clear the importance of the analysis of the Doppler ultrasound spectrum to study disturbances of the blood flow in vessels.

The method of processing the information contained in the Doppler ultrasound signal developed through several stages. Initially performed by audible interpretation of the detected velocity changes, it moved to an analog display of the velocity patterns by means of a zero-crossing detector, and pushed by the disadvantages of this technique, moved towards the application of digital frequency methods using more sophisticated instruments [2].

The commonly encountered Doppler ultrasound systems may be classified as *Continuous-wave* or *pulsed-wave* instruments, dependent on the way the transmitted signal is emitted, continuously or in bursts.

Continuous-wave instruments use two transducers, one for transmitting ultrasound continuously and one for receiving the backscattered ultrasound. In this case the direction of transmission and reception are not exactly 180° as a result of the offset of the two transducers. For small offset, the change in (2.1) is negligible. These instruments are sensitive to flow within the region of overlap of the beams of the two transducers. Their weak range resolution leading to confusion of direction of signals from close vessels, the difficulty in quantification of the blood flow using this systems, the exaggerated sensitivity to vessel wall motions, are some of the disadvantages that lead to the usage of other instruments [26].

With a pulsed Doppler system (see Fig. 2.1) short bursts of ultrasound from a single transducer are generated, activating the receiver to obtain a short sample of the return signal. Only small sample volumes are considered, and their depth in tissue determines the distance of the transducer for which the reception of the echo from each transmission is still assured to be received. This way, the rate at which the pulses can be transmitted is limited to the *pulse repetition frequency (PRF)*, and the highest Doppler shift frequency is limited to the *Nyquist frequency* (PRF/2) [2].

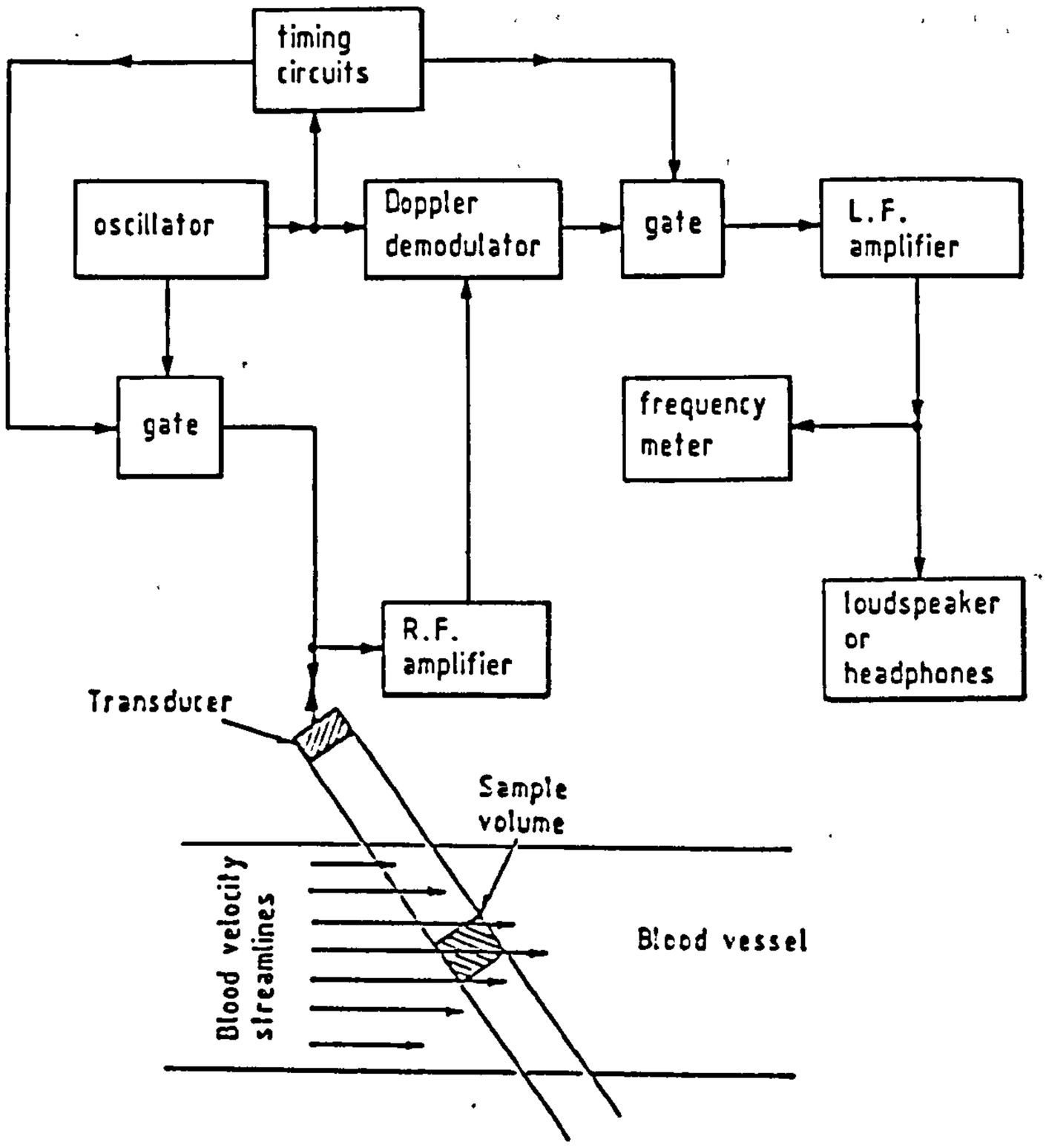


Fig. 2.1: Pulsed Doppler instrument block diagram.

2.2.2. Doppler Ultrasound Signal Model

A range of models of the process of generation of the Doppler signal and the interaction of ultrasound and blood have been published [27] [4] [5]. A brief description of a possible modelling of the Doppler signal is now presented.

The Doppler frequency signal received from a single scatterer on the resolution cell can be modelled by

$$z(t) = A.B.V.\cos(2.\pi.(f_t+f_d).t+\phi) \qquad (2.2)$$

where A is the amplitude of the transmitted sinusoidal continuous wave signal with frequency f_t, B is a factor depending on the compressibility and density of the scatterers and the

surrounding medium, V is the effective volume of the scatterer, f_d is the Doppler frequency and ϕ is the phase shift [27].

Within the resolution cell numerous scatterers rather than only one can be found, and their joint contribution (assuming no multiple scattering) may be regarded as the sum of the I individual scatterer contributions

$$y(t) = \sum_{i=1}^{I} z_i(t) \tag{2.3}$$

This signal, y(t), after demodulating (to downshift the signal spectrum by frequency f_t) leads to the general form of the Doppler signal

$$x(t) = A.B.\sum_{i=1}^{I} V_i.\cos\left(2.\frac{2\pi}{\lambda}.\upsilon(r_i).\cos(\theta).t + \phi_i\right) \tag{2.4}$$

where the scatterer velocity $\upsilon(r_i)$ is a function of the radial position assuming a stationary blood velocity profile during the period of measurement, and the phase shift ϕ_i is a random variable uniformly distributed over $[0,2\pi]$ when the scatterers are assumed to be uniformly distributed in the vessel and with random size, shape and orientation [27] [28].

The Doppler signal is therefore a rather complicated signal, function of three random variables V_i, r_i and ϕ_i, fact that enables the definition of x(t) as a stochastic process. Since it may also be regarded as a summation of a large number of uncorrelated contributions, the *Central Limit Theorem* [29] enables the classification of x(t) as a Gaussian random process. Therefore x(t) is perfectly characterized by its second moment, i.e., its *autocorrelation function*, since it is a zero mean Gaussian process [4].

Typically the Doppler signal is analyzed in a discrete form, this is, it is sampled at the pulse repetition frequency, determined by the maximum Doppler shift frequency that can be unambiguously detected (PRF/2). Thus the continuous Doppler signal x(t) is transformed into the time sequence $x(nt_s)$ with $n=0,\ldots,N-1$, when N is the data sequence length, and $f_s=1/T_s$ is the sampling frequency. For simplicity the discrete Doppler signal will be represented by x[n].

2.2.3. Doppler Signal Spectrum

The equation (2.1) representing the Doppler frequency is only valid under the ideal conditions of an infinitely wide ultrasound beam insonating uniformly the entire cross-section of the vessel,

and, considering one single target in motion. However in practice the ultrasound beam has finite width interrogating the vessel just on a sample area (sample volume) and each blood target by itself produces a spectrum of Doppler shift frequencies rather than a single frequency [30].

The Doppler spectrum is determined by the range of the velocities of the streamlines passing through the resolution cell (assuming undisturbed flow) and by the characteristics of the sample volume.

In order to extract quantitative diagnostic information from the backscattered Doppler signal, variations in the Doppler spectrum mean frequency (caused by change in the blood velocity waveform) and in the Doppler spectral width (induced by flow disturbance) should be accurately estimated by an appropriate signal processing scheme.

The most typical method of conversion of the Doppler time function to the frequency domain is obtained by application of a *Fast Fourier Transform (FFT)* to sequences of Doppler signal time segments. The magnitude of the resulting frequency function is the spectrum.

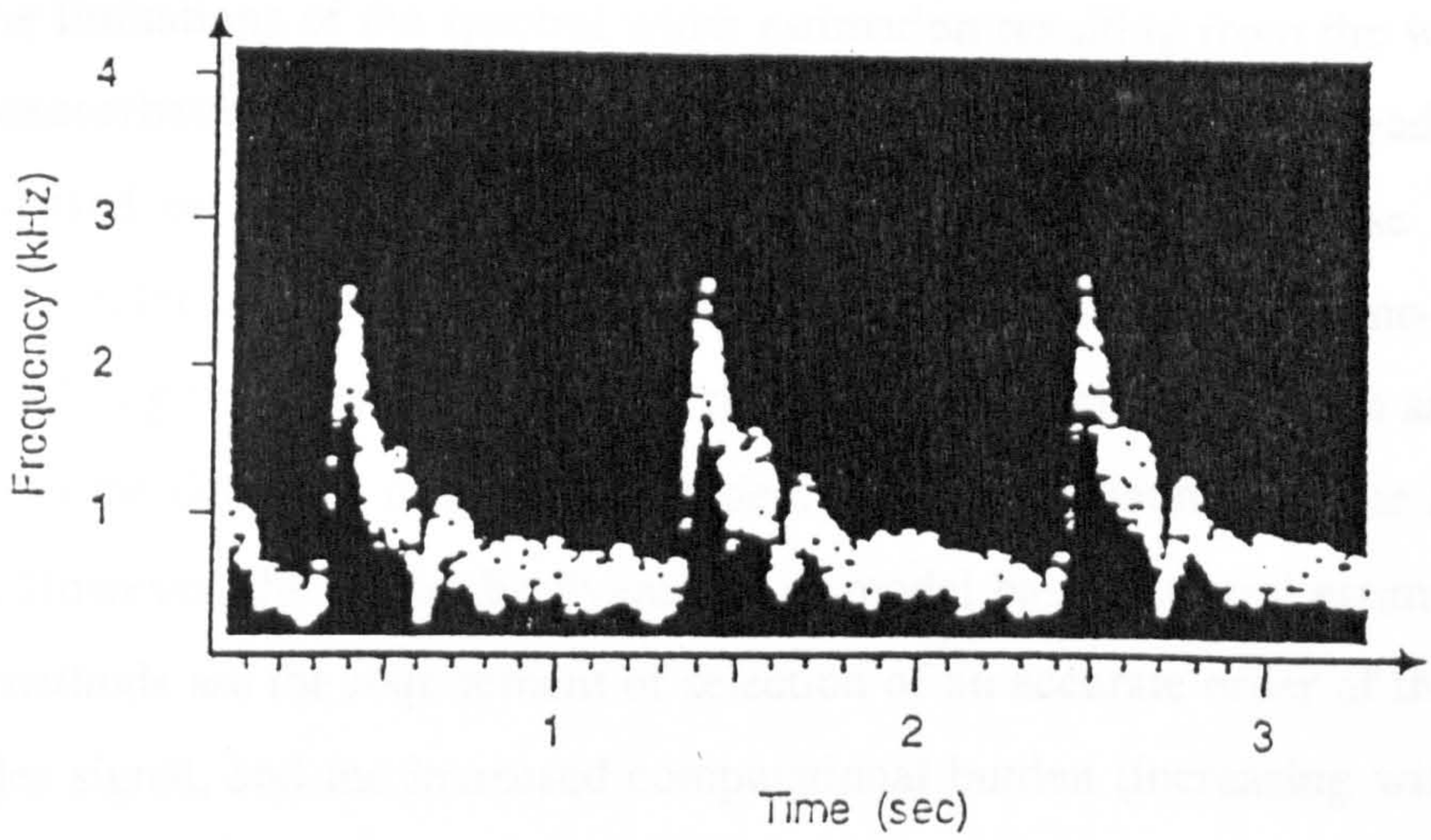


Fig. 2.2: Real-time Doppler spectrum showing 3 cardiac cycles from the centre of a normal common carotid artery using a 5MHz pulsed Doppler ultrasound instrument.

Usually the display of the spectral waveforms relates the time and frequency coordinates with the magnitude of the spectrum displayed as a grey-scale intensity. A typical display is shown in Fig. 2.2. Image brightness at a particular frequency and time indicates that frequency component power at that time. For each cardiac cycle, the relationships between velocity and time are displayed, and, at the same time, flow disturbances are denoted by the spectral broadening displayed. Due to the spectral variability of adjacent spectra, quantitative information

should only be obtained by ensemble averaging over several cardiac cycles, particularly if analysis about the mean frequency and spectral width is to be performed on a single point of the cardiac cycle, rather than on the entire spectrum. In this situation, ensemble averaging of the spectra from similar points in a number of consecutive cycles should be done [30].

The velocity of blood flow within arteries is periodic, resulting from the pumping action of the heart and the velocity profile alters in shape during each period as a result of the pressure pulse from the heart, the non-linear elastic properties of the arterial wall and pulse reflections from discontinuities in vessel-transmission-line-impedance at vessel branch points. As a consequence, the Doppler signal spectrum not only alters in mean frequency but also in shape throughout each pulse period, leading to a *non-stationary*, or *cyclo-stationary* signal. Current practice is to assume approximate stationarity over time segments of 2-20ms duration [27] [30] to which the Fourier transform are applied. Sequential or overlapping time segments may be considered, tapered by one of the available windows in an attempt to suppress the side-lobe effects on the spectrum [31].

The limitations of the spectral width estimation resulting from the window duration [17] [32] and exacerbated by the non-stationarity of the Doppler signal [3] lead to the investigation of model-based or parametric estimators [9] [11 - 13] [15] [33]. These methods, applied to stationary Doppler signal segments, offer potential improvements in the time-frequency resolution product, enabling the conservation of the shape of the velocity waveform and leading to greater sensitivity in the detection of spectral broadening, therefore increasing the sensitivity in disease detection. However, the major disadvantages of model based spectral estimation compared with classical methods are the requirement of selection of an accurate order of the model to represent the Doppler signal, and the increased computational burden (increasing with model order).

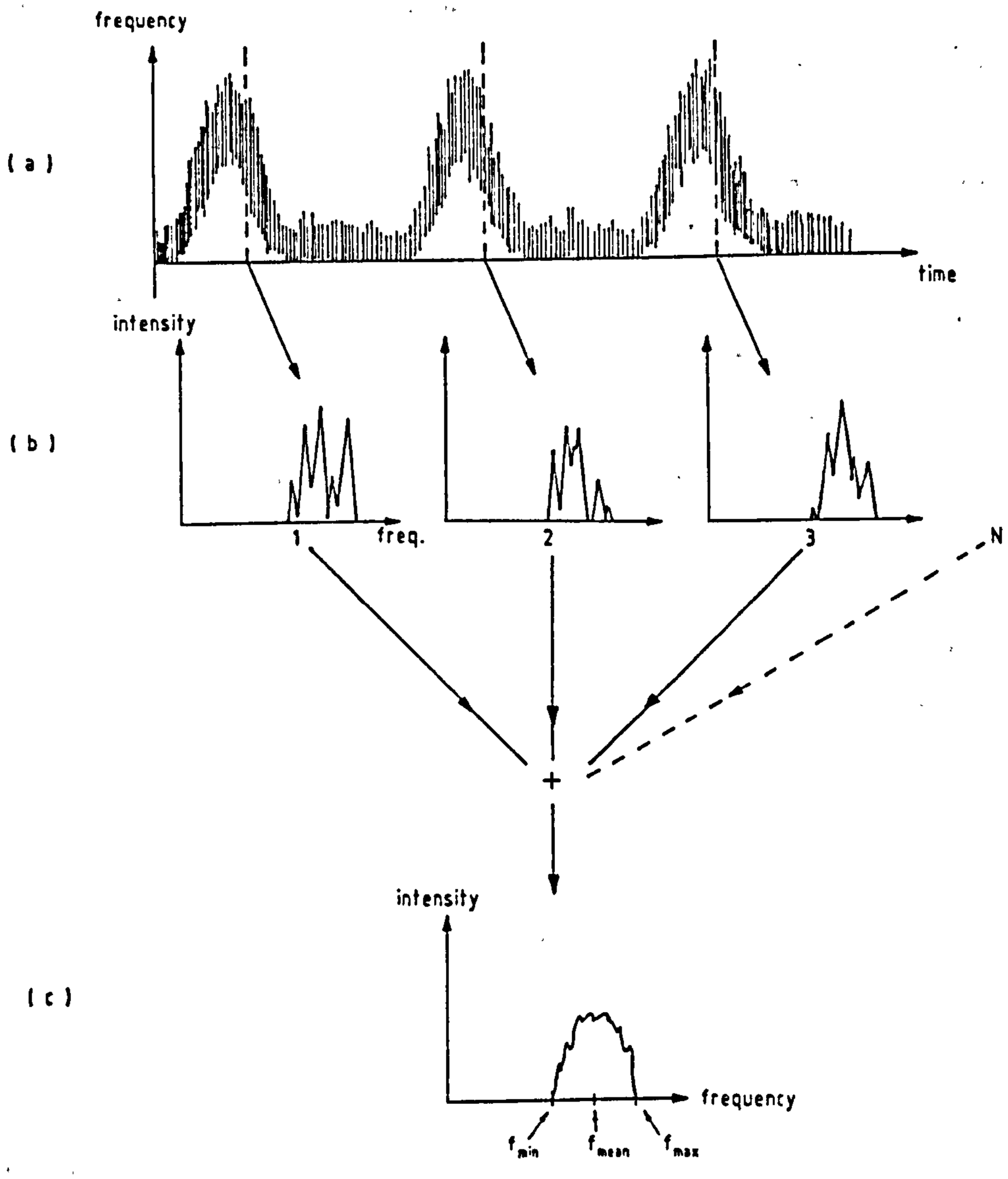


Fig. 2.3: Averaging frequency spectra of Doppler signals taken at similar points in N consecutive cardiac cycles. (a) Real-time spectrum, (b) individual spectra, and (c) averaged spectrum [34].

2.3. SPECTRUM ANALYSIS

Most of the signals encountered in practice are available for analysis over only a short period. Sonar and radar signals are examples of this class of signals. Signals with such characteristics usually assume the discrete form of a data sequence where valuable information is corrupted by unwanted noise and/or random effects. In these situations statistics play an important role in conjunction with signal estimation techniques to enable the extraction of pertinent information from the signal under analysis.

Spectral analysis is a very powerful tool used to characterize the frequency content of a signal, commonly performed via Fourier transforms, which are the mathematical foundation used to relate time or space signals or even a model of its signal to its frequency domain representation [32].

The spectral estimation methods based on the computation of the Fourier transform of either the signal or the autocorrelation function within a measurement interval are usually referred to in the literature as *conventional methods.*

Traditionally employed to obtain spectral estimates, the Fourier based methods have revealed weak ability to distinguish spectra of two or more closely allocated (in frequency) signals (the so called *frequency resolution*), and to suffer from leakage of the energy of the main lobe of a spectral response into the sidelobes due to windowing effects [35] [31].

Alternative techniques of spectrum estimation have been developed since early seventies in an attempt to alleviate the inherent limitations of the Fourier Transform approach.

A common philosophy adopted to classify the spectral estimation methods is to designate the conventional methods and the minimum variance method (a filter bank method) as *nonparametric methods*, while the other methods that consider more knowledge about the signal than that enclosed in the measurement interval are termed *parametric methods* [17].

Parametric spectrum estimation techniques are not so computationally efficient as the classical methods. However, the improvement that might be obtained in terms of frequency resolution performance justifies the study of the applicability of these modern methods particularly in the case of signals composed of short data records.

A general perspective of the theory used in the field of spectral analysis will now be developed. Some typical terms and mathematical expressions are pointed out to facilitate the foregoing study of spectral estimates obtained from the pulsed Doppler ultrasound signal under analysis. Some commonly used methods belonging to the class of conventional methods will be briefly described, and overall features of parametric model spectral estimation are pointed out. The minimum variance method will be described later in Chapter V.

2.3.1. Signals and their Frequency Transformation

Signals may be classified into several classes according to their characteristics. The general signal categories [36], *deterministic* (those whose future course may be determined by some linear analysis of the signal), *random* (those whose individual data may have occurred in any order),

continuous, and *discrete*, alone or combined, are frequency transformed with different approaches, and details of the procedure involved may be found in many textbooks and papers [37 - 39].

Concentrating for the purpose of this thesis on random signals only, a summary of the most used terminology and their respective meaning related to those signals will follow. This, however, will be done with the help of expressions developed specifically for other types of signal. For example, the definition of the *energy*, E, of a deterministic analog waveform (represented by a complex-valued variable x(t), a continuous function of time t), is

$$E = \int_{-\infty}^{\infty} |x(t)|^2 dt \tag{2.5}$$

where the x(t) satisfies the *Dirichlet conditions*, which require that the function is absolutely integrable, has a finite number of discontinuities, and a finite number of maxima and minima [37].

The equivalent energy relation in terms of frequency is obtained employing the *Parseval's theorem*

$$E = \int_{-\infty}^{\infty} x(t).x^*(t)\,dt = \int_{-\infty}^{\infty} X(f).X^*(f)\,df \tag{2.6}$$

where * stands for complex conjugate and X(f) is the Fourier transform of x(t). The function $S(f)=|X(f)|^2$ representing the distribution of energy as function of the frequency, is called the *Energy Spectral Density (ESD).*

Considering now the random process of interest rather than a deterministic signal, the estimation of its spectra characteristics will oblige the adoption of a statistical viewpoint.

Recalling (section 2.2.2) that the *Pulsed Doppler Ultrasound (PDU)* signal may be shortly modeled as a complex-valued signal x(t)

$$x(t) = x_c(t) + j x_s(t) \tag{2.7}$$

where $x_c(t)$ and $x_s(t)$ are the in-phase and quadrature baseband components respectively [11]. The blood velocity variations with time, causing the nonstationarity, may be overcome by considering periods of time (usually around 10ms) over which the velocity distribution remains approximately constant. In this case, the classification of the Doppler signal as a *Wide Sense Stationary (WSS)* process is assured [11].

Under these steady flow conditions, x(t) becomes a stationary complex Gaussian process [4], and the use of the covariance function enables full characterisation of the process in the time domain.

The covariance function $c_{xx}(t_1,t_2)$ is defined as the difference between the *Autocorrelation Function (ACF)* - $\gamma_{xx}(t_1,t_2)$ - of the process x(t) at times t_1 and t_2, and the product of the mean values of x(t) at times t_1 and t_2.

When a WSS process is considered, the mean value (μ) of the process is constant and the ACF does not depend on the specific times t_1 and t_2, but on the time-difference (or *lag*) $\tau=t_2-t_1$. In this case, the covariance function is

$$c_{xx}(\tau) = \gamma_{xx}(\tau) - |\mu|^2 \tag{2.8}$$

The covariance function evaluated for lag zero is equal to another commonly used statistic, the *variance* of the process, σ_x^2, given by

$$\sigma_x^2 = c_{xx}(0) = \gamma_{xx}(0) - \mu^2 \tag{2.9}$$

WSS Gaussian processes which are zero-mean, just like the Doppler signal [27], are completely statistically determined by the autocorrelation function.

In contrast with the deterministic signals, stationary random processes do not have finite energy and therefore do not possess a Fourier transform. The equivalent of the ESD in this case, that is, the distribution in frequency of the power of x(t), is called the *Power Spectral Density (PSD).*

This function presents always a real non-negative value, as for complex processes $\gamma_{xx}(\tau)=\gamma_{xx}^*(\tau)$ produces a real positive PSD, and for real processes, the ACF is real and even, producing again real positive values of PSD [17].

Usually in practice we deal with a single realization of the random process, for which we do not know the true ACF $\gamma_{xx}(\tau)$, but with which we can compute a time-averaged ACF.

If an additional assumption is made, i.e., that the process is *ergodic* in the first and second moments, the substitution of time averages for ensemble averages is possible, allowing the replacement of the ensemble average ACF $\gamma_{xx}(\tau)$ by the time-average ACF $\gamma'_{xx}(\tau)$ within the time window (T) in consideration, as expressed in (2.10).

$$\gamma_{xx}(\tau) = \lim_{T\to\infty} \gamma'_{xx}(\tau)$$
$$= \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t+\tau)\,.\,x^*(t)\,dt \qquad (2.10)$$

Furthermore the Fourier transform of $\gamma'_{xx}(\tau)$ (not possible before the ergodicity assumption was made [17]) is now computed as indicated in (2.11) [40] [37]. When averaged and computed in the limit for T tending to infinity, equals the true PSD $\wp(f)$.

$$\hat{P}(f) = \int_{-\frac{T}{2}}^{\frac{T}{2}} \gamma_{xx}(\tau).e^{-j2\pi ft}\,dt$$
$$= \frac{1}{T} \left| \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t).e^{-j2\pi ft}\,dt \right|^2 \qquad (2.11)$$

$$\wp(f) = \lim_{T\to\infty} \mathcal{E}\,\hat{P}[(f)] \qquad (2.12)$$

If we now consider a sampled version of the random signal, x[n], obtained by sampling x(t) at f_s Hz, and assuming the random process bandlimited to $\pm 1/2T_s$ Hz, the *autocorrelation sequence (ACS)* of x[n] for lag k is described as

$$r_{xx}[k] = \mathcal{E}\left[\, x[n+k].x^*[n] \,\right] \qquad (2.13)$$

and its Fourier transform yields the PSD

$$P(f) = \sum_{k=-\infty}^{\infty} r_{xx}[k]\,.\,e^{-j2\pi kf/f_s} \qquad (2.14)$$

We may notice the existence of two alternatives to compute the PSD of a random process. One called the *direct method*, computes the PSD directly by Fourier transforming the signal x[n], and another, the *indirect method*, which first computes the autocorrelation sequence, and then Fourier transforms it to obtain the PSD [17].

2.3.2. Conventional Methods

The so called conventional or traditional methods are based on the direct and indirect alternatives of computing the PSD of a signal, now applied to finite data records. From now on discrete random processes will be assumed.

The PSD estimate based on the indirect approach may be given as [17],

$$\hat{P}_{BT}(f) = \sum_{k=-L}^{L} \hat{r}_{xx}[k] \cdot e^{-j2\pi k \frac{f}{f_s}} \qquad -\frac{1}{2T_s} \leq f \leq \frac{1}{2T_s} \tag{2.15}$$

where the estimates of the ACS have been forced to zero outside the range [-L,L].

Assuming N data samples available, one possible autocorrelation estimate is

$$\hat{r}_{xx}[k] = \frac{1}{N-k} \sum_{n=0}^{N-k-1} x[n+k] \cdot x^*[n], \qquad 0 \leq k \leq N-1 \tag{2.16}$$

This autocorrelation estimate is conjugate symmetric, and is a *consistent* estimate, i.e., is unbiased and its variance tends to zero as N increases [37].

An alternative autocorrelation estimate, as proposed by Jenkins and Watts, is

$$\bar{r}_{xx}[k] = \frac{1}{N} \sum_{n=0}^{N-k-1} x[n+k] \cdot x^*[n] \qquad 0 \leq k \leq N-1 \tag{2.17}$$

This estimate is *biased* for finite N but is asymptotically unbiased as N tends to infinity. For typical applications the mean square error of the former alternative tends to be higher than for the one obtained for the latter ACS estimate. Also the unbiased estimate can yield autocorrelation estimates that are not a valid autocorrelation sequence, since it can lead to autocorrelation matrices that are not positive semidefinite [32]. For these reasons, $\bar{r}_{xx}[k]$ is often the preferred estimator.

The direct approach of spectrum analysis is the *periodogram* PSD estimate

$$\hat{P}_{PER}[k] = \frac{1}{N} \left| \sum_{n=0}^{N-1} x[n] \cdot e^{-jkn\left(\frac{2\pi}{N}\right)} \right|^2 = \frac{1}{N} |X(k)|^2 \tag{2.18}$$

where X[k] is the Fourier transform of the discrete time signal of length N, x[n] [39].

However the periodogram presents a variance which does not decrease with increasing values of data sequence length considered (it is a inconsistent estimator of the PSD), a problem

which may be overcome by application of averaging techniques [39]. Bartlett suggested splitting the data sequence into nonoverlapping segments, for which a periodogram estimate was computed and then the estimated spectrum would be the average of those periodogram estimates. A slightly different approach was made in Welch method, where the segments may overlap, and each segment is windowed before their periodogram estimates are computed, the final power spectral estimate being obtained as the average of the modified periodograms.

Many of the problems of the periodogram PSD estimation techniques can be traced to the assumptions made about the data outside the measurement interval. The finite data sequence may be viewed as *windowing* an infinite sequence with a *boxcar* function. In terms of frequency, this corresponds to a convolution of the desired transform with a sinc function. If the power of the signal is concentrated in a narrow bandwidth, this convolution will have the effect of spreading this power over adjacent frequency regions, phenomena known as *leakage*. In addition to the distorting effects of leakage in the power estimation, it may also mask the presence of weak signals and prevent their detection. Other window functions (Bartlett, Hanning and Hamming for example [37] [39]) may be used to achieve better sidelobe levels than those of the boxcar function, therefore obtaining a better estimate. However, this can only be achieved by broadening the window's main lobe, therefore resulting in a reduction of spectral resolution.

2.3.3. Parametric Methods

Classical methods of estimation use Fourier transform operations on either windowed data or windowed ACS estimates. The unavailable data or unestimated ACS values outside the window are implicitly assumed zero.

Parametric methods of estimation drop this unrealistic assumption, and therefore avoid the need of windows, together with their distorting effects.

This type of spectral estimation is described in detail in chapter 5 and only in outline here.

Based on some a priori knowledge or assumptions about the process that generated the data, it is often possible to select a model which is a good approximation to the actual underlying process. Usually a better spectral estimate, both in terms of spectral fidelity and of frequency resolution is obtained, by basing the estimate on the model, whose parameters are estimated from the available data.

Thus the parametric approach to spectral estimation involves three steps.

The first one is the selection of an appropriate model that fits the measured data.

Many discrete-time random processes encountered in practice are well characterized by a time-series or rational transfer function filter model, driven by a white noise sequence. The transfer function H(z) between the z transform -U(z)- of the input sequence u[n] and the z transform -X(z)- of the output sequence x[n] may generally be expressed as

$$H(z) = \frac{X(z)}{U(z)} = \frac{B(z)}{A(z)} \tag{2.19}$$

where A(z) and B(z) represent respectively the Z-transforms of the *Autoregressive* (AR) and the *Moving Average* (MA) branches of the general *Autoregressive Moving Average* (ARMA) model.

The PSD of the output process of an ARMA filter driven by a white noise process with variance σ^2 is

$$P_{ARMA}(f) = \sigma^2 \left| \frac{B(z)}{A(z)} \right|^2_{z=e^{j2\pi\frac{f}{f_s}}} = \sigma^2 \left| \frac{\sum_{k=0}^{q} b[k].e^{-j2\pi k\frac{f}{f_s}}}{\sum_{k=0}^{p} a[k].e^{-j2\pi k\frac{f}{f_s}}} \right|^2 \tag{2.20}$$

where p poles and q zeros are considered. If the filter presents only p poles, the AR(p) model is obtained, while if the singularities presented are only q zeros, the MA(q) model is achieved.

Knowledge of the spectral shape of the PSD may help in the selection of one of these methods to a certain application. If the spectra has sharp peaks but no deep nulls an AR(p) model is recommended; if it has deep nulls but no sharp peaks the MA(q) model is preferred; if it has both extremes the choice is an ARMA model [32].

Thus the specification of the model includes also the model order, which is generally not known a priori.

Typically, practical experiments show that if a small model order is chosen the spectral estimate yields insufficient resolution while if a large model order is used the estimates will be statistically unstable presenting spurious peaks [17].

Some criteria have been developed to select an optimal model order, where the optimal represent the best compromise between bias and variance of the spectral estimate obtained. Published work on this field has proved that the model orders selected by a variety of methods do not differ too much from the selection made by means of the *Final Prediction Error* (FPE) criterion [41].

In principle, if two or more models can reasonably represent the spectrum, the model with fewer parameters should, intuitively, be the chosen one. However we must also weight the computational burden of determining the parameters. For this reason the AR model is often the selected model [32].

The second step in the parametric approach to spectrum estimation is the estimation of the parameters of the assumed model. This may be done using either the available data samples or the autocorrelation lags.

There are several possible approaches to produce model parameter estimates leading to different spectral estimation methods within each major class of parametric models.

Some are based on the same theory but giving different mathematical interpretation to equations, like the AR methods of *Yule-Walker*, *Covariance* and *Modified Covariance* methods, and some are based on a maximum entropy concept like the AR *Burg* method for which many different algorithm approaches have been developed.

However all methods present a common goal, the minimization of a mean squared error ρ computed as

$$\rho = \mathcal{E}[\,|x[n] - \hat{x}[n]|^2\,] = \|x[n] - \hat{x}[n]\|^2 \qquad (2.21)$$

where $\hat{x}[n]$ represent the estimated data sequence when a specific parametric model is assumed, and $\|\bullet\|$ stands for the Euclidean norm of vector • [17].

The third and last step is the computation of the spectral estimate. The estimated model parameters are inserted into the theoretical PSD expression appropriate to that model.

Other (nonparametric) methods exists that do not fit into the designation of classical estimation methods. One of them is the *Minimum Variance* method, also known as the *Capon* method. This method provides an estimate of the PSD by measuring the power at the output of a series of narrowband filters, which adapt according to the frequencies under consideration [17].

This method and some parametric methods will be further developed in Chapter 5.

2.4. PARALLEL PROCESSING

In the last few years the demand for faster, more accurate, and more flexible computers has been evident in areas as diverse as weather prediction, real-time control systems, and, signal processing.

Most of the currently available computer systems are based on the von Neumann architecture, where a single processing unit is connected to memory. In these systems, the typical processor operation is reading an instruction followed by data from memory and storing the results in memory again. Thus, the rate at which data can be read from and written to memory is a limitation of this architecture; this is known as the *von Neumann bottleneck* [42] [43].

Factors such as the achievement of very compact integrated circuit technology at low cost, were not matched by such good improvements in increasing clock rate. This led to the development of new computer architectures based on distributed processing units, that is, following a *parallel processing* approach [44].

The concept of parallel processing, this is, the completion of a certain job by a number of processors cooperating together for the same aim, is nowadays widely spread.

Parallel processing makes it possible to speed up the execution times of algorithms which are split into different tasks to be executed simultaneously. The speedup achieved by the use of parallel processing techniques increases (up to a limit) with the number of *processors* employed, however this increase is not linear with the number of processors used. Factors such as contention for shared resources, communication overheads and inherent serialism of the algorithm being implemented impose a practical limit on the effective speed-up achievable [42] [45].

Therefore the implementation of an algorithm in a parallel form is by no means a trivial task. Some algorithms present a set of operations that cannot be efficiently implemented in parallel, and in these cases, the performance of the system and the maximum number of processors that can be effectively employed is determined by the amount of sequential processing involved, as stated in *Amdahl's law* [46].

The procedure for converting a problem into a program for parallel execution on a multiprocessor is basically composed of three steps:

1- *identification* of the parallelism of the problem,

2- *partitioning* of the problem into tasks that can be executed concurrently, and

3- *allocation* (or scheduling) of those tasks on the available processors.

The extraction of the portions of a problem which can be parallelized is dependent on the ability of the programmer to identify those portions tacking into account the implementation facilities provided by the target programming language.

Partitioning and scheduling must be performed in order to minimise the parallel execution time on the target multiprocessor system, and depend on parameters such as number of

processors available, sequential execution times, communication overheads and scheduling overheads [47].

As far as the allocation of tasks to the processors is concerned, there are two general approaches, the *static allocation*, and the *dynamic allocation*. In the former the association of a group of tasks to a particular processor is resolved before running time and remains fixed during the execution of the program. In the latter approach, tasks are allocated to processors at running time according to a criterion previously established, such as processor availability, inter-task dependencies and task priorities.

A general parallel processing system is composed of several *Processing Elements* (Pes), acting on parts of a problem, and being able to operate concurrently and communicate with each other whenever necessary.

Typically the flexibility of each PE and the degree of connectivity between Pes determines different *parallel architectures*. An architecture whose Pes are characterized by performing simple tasks, establishing fast communications on a highly connected network, is termed a *fine-grain* architecture. On the other hand, the *medium-grain* architectures are more general purpose in nature; the degree of connectivity between such PEs is more complex and the communications between them tend to be slower. In general, the concept of *granularity* measures the size of an individual task that can be effectively executed on a parallel machine [42].

Another possible and popular classification of the parallel architectures is due to Flynn [48]. It categorizes the computer architectures according to whether the instruction stream and the data stream exist as single or multiple entities. Under this classification, the von Neumann architecture (the sequential or serial computer) is considered a *Single Instruction stream Single Data stream (SISD)*; the *SIMD* class, represents those computers where a *S*ingle *I*nstruction operates on *M*ultiple *D*ata simultaneously; the *MIMD* (*M*ultiple *I*nstruction stream *M*ultiple *D*ata stream) category is the more flexible one, since each processor can be performing totally different instructions at a given time on different data (see figure 2.4 for graphical representation of these architectures).

A typical example of a MIMD system is the *transputer*.

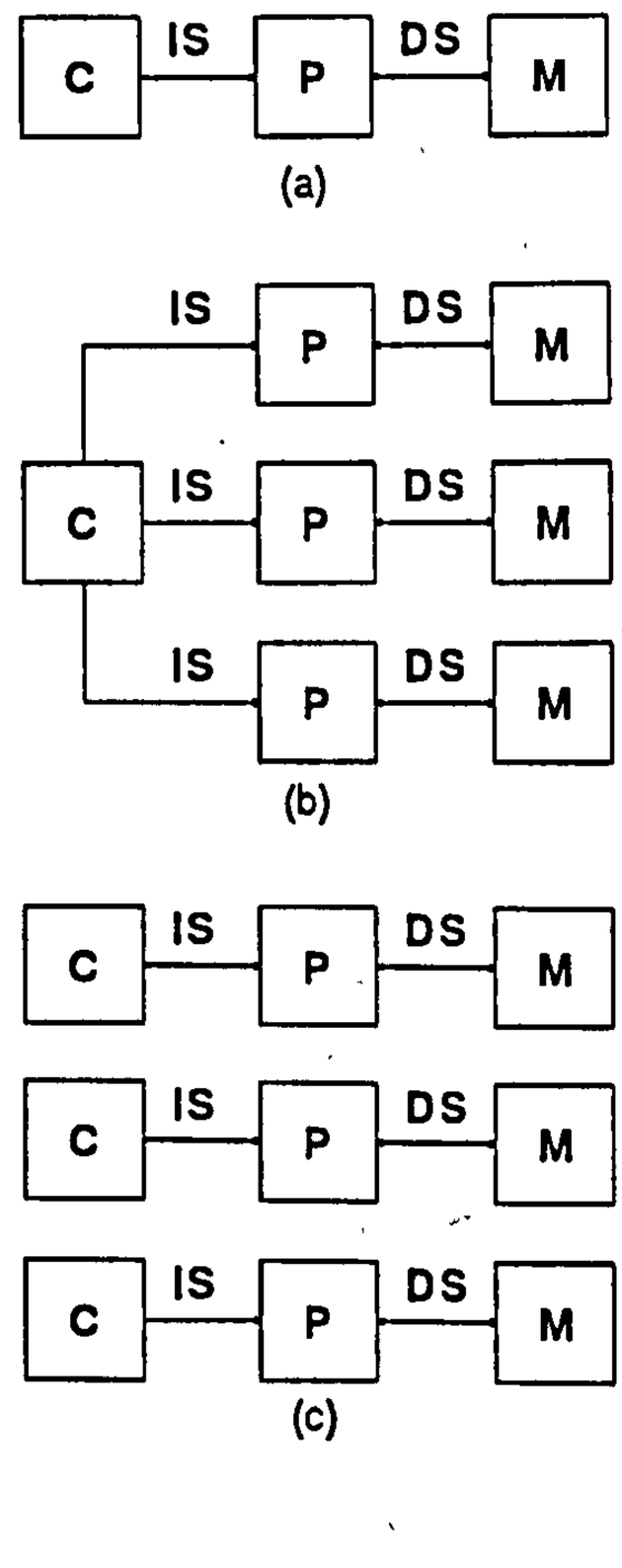


Fig. 2.4: Flynn's taxonomy of computer architectures: (a) SISM, (b) SIMD and (c) MIMD, (C=control unit, P=processor, M=memory, IS=instruction stream, DS=data stream).

The transputer is a programmable *Very Large Scale Integration (VLSI)* device containing a processor, local memory, and communication links for point-to-point inter-connection with other transputers.

Although the transputer might be used alone, it is designed specifically to implement parallel processing systems, and to achieve so, a number of transputers can be connected together in a *network*.

Programming languages dedicated to a multi-processor environment differ from sequential programming languages in several aspects. In a sequential programming language, action occurs strictly in sequence. However, with a parallel programming language, a means of performing a number of tasks (or processes), which are themselves sequential, in a concurrent way and with short execution times, must be provided.

The specific needs of parallel processing systems led to the development of the parallel processing language, *OCCAM*, dedicated to the transputer. Over the past few years, some

compilers of Modula-2, Fortran, ADA and C have also been developed, in an attempt to ease the portability of programs.

OCCAM is an abstract language whose basic unit, the *process*, presents inputs and outputs communicating with other processes via point-to-point links, called *channels*. Complex systems may be created by connecting together processes, where, since there are no shared variables, the communications are made solely through the channels.

OCCAM's clear and simple representation of a program as a hierarchical structure, regarded from outside as a single process (see Fig. 2.5), enables the operation of individual processes independently. On the other hand, the concurrent processes may be executed in different processors.

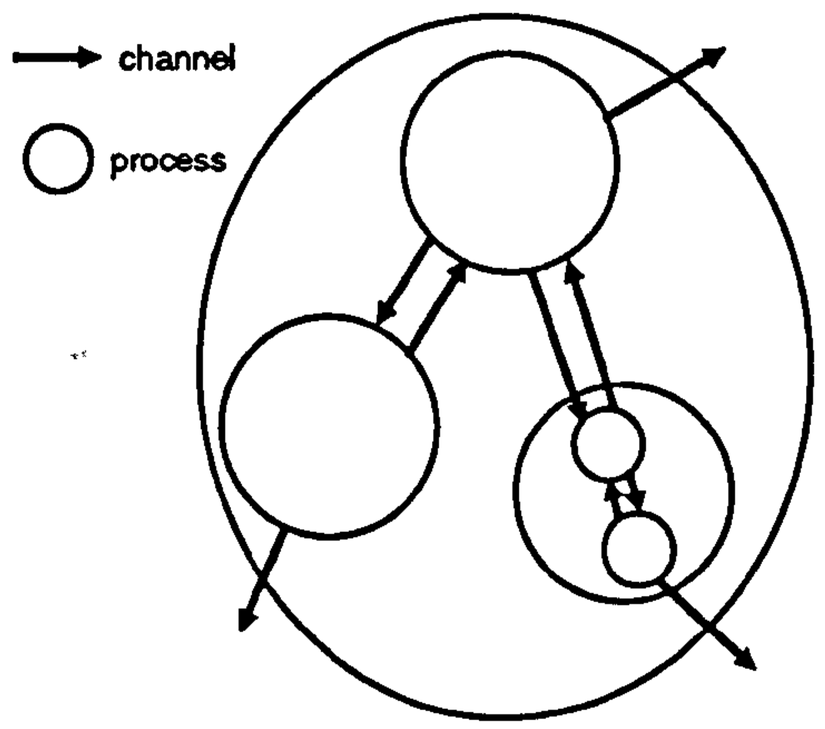


Fig. 2.5: Hierarchical structure of processes allowed by OCCAM.

A complete collection of processes, constituting an OCCAM program, may either be processed on a single transputer or on a network of transputers (Fig. 2.6).

The processes may be executed in parallel or in sequence. The latter procedure is usually taken as a first case study of the parallel implementation of an algorithm, enabling also results for comparative studies on the *parallel processing performance* of the algorithm in consideration.

As stated before, algorithms containing a significant number of sequential operations may not be suitable for parallel partitioning, or even, the number of processors used to achieve desired processing times may not produce an efficient usage of the same processors. For these reasons some suitable metrics (see section 2.3.3) have been developed to aid analysis of parallel systems.

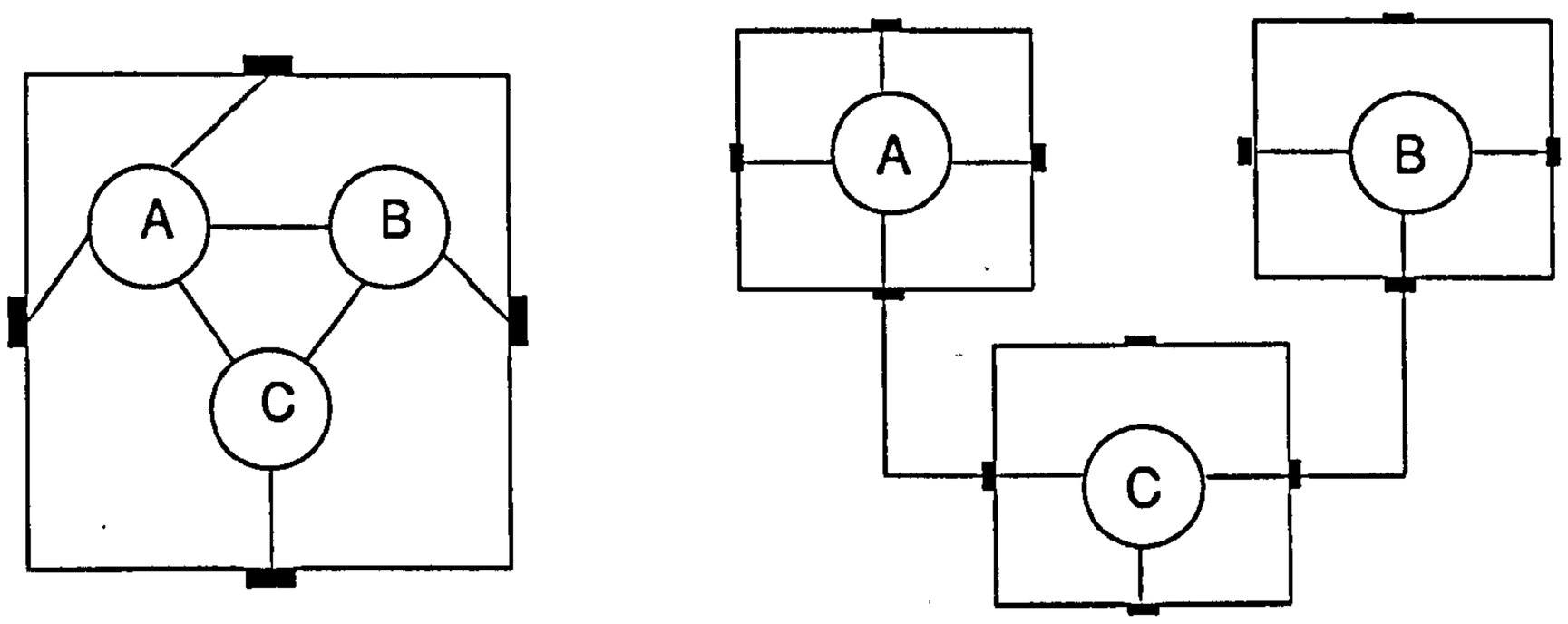


Fig. 2.6: An OCCAM program implemented on a single transputer or a network of transputers.

2.4.1. The Transputer

The transputer is a single-chip computer with communication links for direct connection to other transputers [49]. It is usually employed in a network configuration, presenting an architecture which supports concurrency of the collection of transputers of the network, and provides synchronisation of the links' communications [50].

A network of transputers incorporates features to support parallelism through its associated programming language, OCCAM [49].

Different transputer architectures, presenting different capabilities are currently available. The most common is the IMS T800 version, consisting of a 32-bit microprocessor, using a 64-bit floating-point unit, four high-speed serial communication links, with 4Kbytes of on-chip memory, and an external memory interface [51].

The transputer communication links implementing two OCCAM channels in each direction, enables the point-to-point connection between transputers at a data transfer rate of 1 Mbyte/second (for the T800 version of the transputer). Each transputer, possesses four of these links, allowing connection up to four other transputers.

There are several possible ways of connecting the transputers in a network. According to the aim of the parallel implementation, topologies such as pipeline, ring, array, binary tree, or, processor farm, are easily configured (see Fig. 2.7).

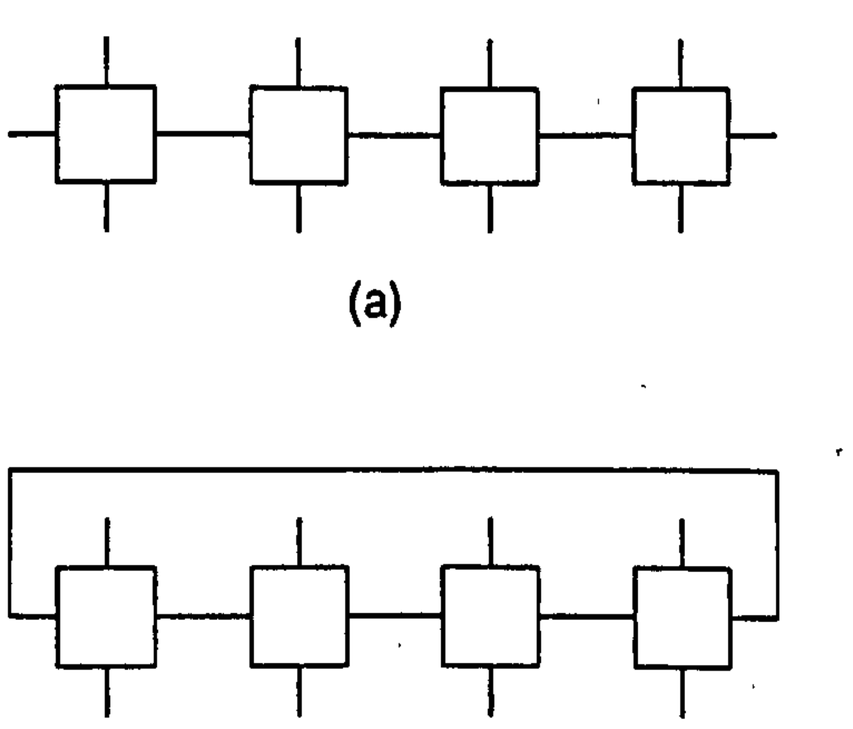


Fig. 2.7: Transputer-based topologies: **(a)** linear, **(b)** ring, **(c)** processor farm, **(d)** tree, and, **(e)** matrix.

The *processor farm* consists of a sequence of processors called *workers* and a *master* processor. The master processor schedules the tasks to the workers connected typically in a line. Each worker is used to monitor input data and either to process the data and return the result, or to transfer the data on to another worker [52].

A common development environment for OCCAM is the *Transputer Development System (TDS)*. The TDS is a hardware board that plugs into a PC, together with a software package that allows the user to develop and debug the OCCAM programs. The most commonly used card is the IMS B004 [53].

The TDS provides an editing environment, the *folding editor*, based on the concept of hiding from view (as if by folding a piece of paper) a sequence of text [54]. The sequence of text is enclosed in the *fold* and marked by a single fold line commenting the text enclosed. The fold lines are denoted by ... (three fullstops). This way, a program may be viewed in different levels of complexity. Thus, the TDS allows the edition of a program as a single fold line, which, when

opened, enables the view of other hidden folders. Consecutive opening of folders will therefore reveal a deeper analysis of the program.

Data are stored in separate files, as *filed* files of the *filing* system, and enclosed in a compilation fold (*foldset*) before compilation of the program. The foldsets are generally classified according to where the set of program contained in the fold is going to be executed, this is, as *.EXE* if it is going to be executed within the TDS, or *.PROGRAM* if it is to be executed on a network of transputers.

The OCCAM programs edited and compiled using the TDS may be executed on the TDS board transputer, or a network of transputers connected to the TDS, or even, on a transputer system acting independently from the TDS. Fig. 2.8 illustrates a typical transputer environment.

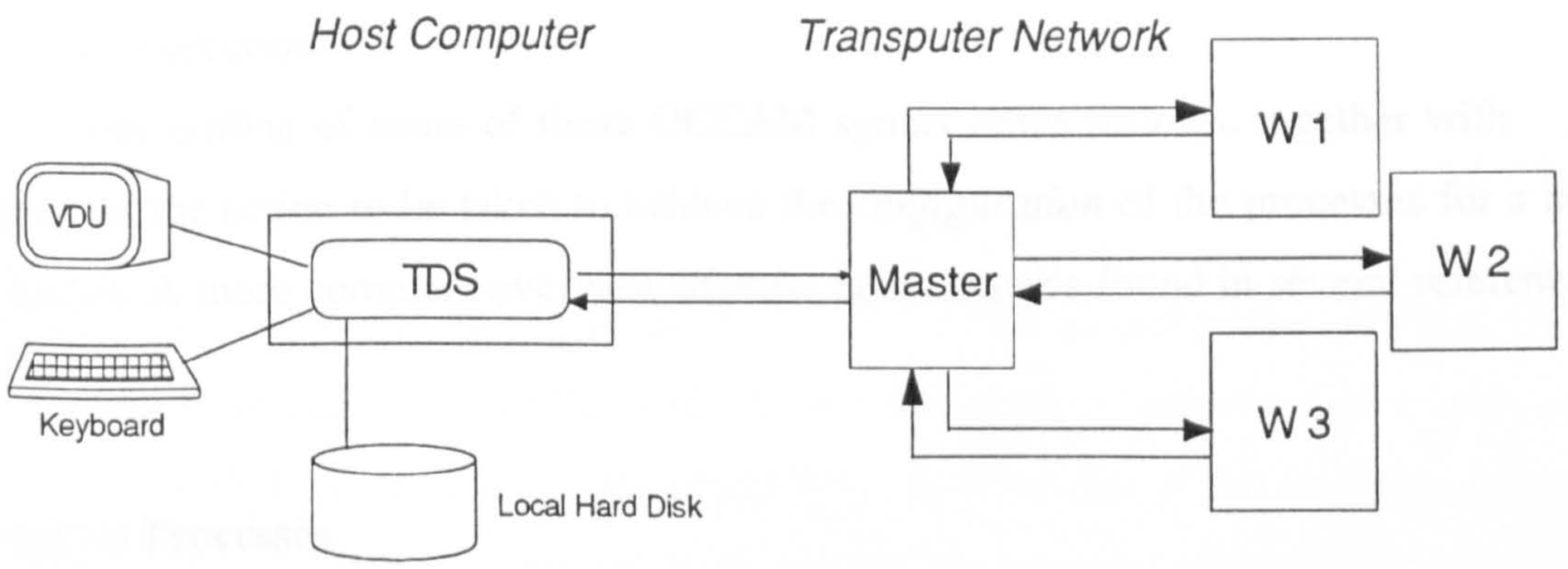


Fig. 2.8: Typical transputer environment.

2.4.2. OCCAM - the Parallel Programming Language

The OCCAM language was created from the concepts developed by David May in Experimental Programming Language (EPL), and influenced by the theoretical model of Communicating Sequential Processes (CSP) of Tony Hoare [55]. Since then it has continued development at INMOS Ltd [56].

The principles beyond the minimalist approach of the OCCAM language were derived from the 14th century philosopher William of Occam, whose name was given to the language,

and for whom duplication of language entities should be avoid and observable phenomena should be explained by the simplest approach [57].

OCCAM programs are viewed as a collection of sequential processes, each one executing concurrently and interacting or communicating with the other processes via input/output operations through channels. Each process represents by itself a certain aspect of the overall implementation, and each channel describes a specific connection between each of the processes. In this way, an OCCAM program assumes a clear and simple structure, and allows its application to one or more transputers.

An OCCAM program is built up from a number of *primitive processes*, which are combined to form *constructions*. Some of these constructions can be replicated a stated number of times, by means of *replicators*. The variables and data used in an OCCAM program must agree with the *data basics* defined by the language. Processes may be assigned with different *priorities* for execution.

A brief outline of some of these OCCAM syntax items follows, together with description of the action to be taken to achieve the *configuration* of the processes for a network of processors. A more complete overview of these items may be found in several references [56] [58 - 60].

A. Primitive Processes

The simplest process in an OCCAM program (the primitive process) is an action, assuming three possible forms:

- assignment process **v = e** -assign a expression value **e** to a variable **v**,
- input process **c ? v** -input a value **v** from a channel and assign to variable, and,
- output process **c ! e** -output a expression value **e** along a channel **c.**

Another two processes, called **SKIP** and **STOP**, are used for special purposes. SKIP is a process that starts, performs no action and terminates. STOP is a process that also does nothing, but unlike SKIP, it does not terminate. SKIP is used in situations such as to indicate the existence of a partially completed code, or when we require nothing to happen. When STOP is executed in a process, that process is stopped as well as any other interacting process.

B. Constructions

A construction is build from a number of processes, *process 1 ... process n*, combined in sequence, parallel or alternation, represented by the keywords:

- **SEQ** -sequential construction,
- **PAR** -parallel construction, and
- **ALT** -alternation construction,

and assuming a syntax format for the first two constructions as below

SEQ (or **PAR**)
process 1
.
.
process n

The sequential construction, similarly to the traditional computers, enables the set of processes to be executed one after the other, till the last process terminates its action.

In the parallel construction each process (composed of one of the primitive processes or a combination of processes) is executed concurrently, residing on a single transputer (achieved by time-slicing or "pseudo-parallelism"), or, on a network of transputers (where true parallelism is obtained).

The alternation construction allows a particular process from a list of alternative processes (*process 1 ... process n*) to be selected for execution, at a time instant dependent on the input guard (where *input 1 ... input n* represent the input guards associated with the respective processes) that first presents available data. The process associated with that first input guard is the only one (from the list of alternative processes) to be chosen for execution. This construction is represented in the following format:-

```
ALT
  input 1
    process 1
  .
  .
  input n
    process n
```

Some other more conventional constructions are provided by OCCAM:

- **IF** -conditional construction,
- **CASE** -selection construction and
- **WHILE** -repetition construction.

The conditional construction allows the selection and execution of the first process whose conditional instruction is true. Its format is

```
IF
  Boolean expression 1
    process 1
  .
  .
  Boolean expression n
    process n
```

The selection construction combines a number processes which are *options*, one of which is going to be selected by matching the value of a *selector* with the value of a constant expression, the *case expression*, associated with the option. In the case of unsuccessful matching, the associated process behaves as a STOP. The format of this construction is as follows:-

CASE
case expression 1
process 1
.
.
case expression n
process n

A loop is created within an OCCAM program with a **WHILE** construction when a Boolean expression assumes the value TRUE. The evaluation of the process is performed repetitiously till the Boolean expression changes its value. This construction has the format:-

WHILE *boolean expression*
process

C. Replicators

Like in other programming languages, OCCAM also possesses the possibility of repeating a specified number of times a process. This replication property may be applied to the constructions **SEQ, PAR, IF,** and **ALT.** Any of these constructions may be appended with a *replicator* which specifies a replication count. An *index* is incremented in steps of one from the value *start* for *count* values. The data type of these parameters must be integer. The format of the replicator is

index = *start* **FOR** *count*

where

- *index* is the occam identifier of the replicator index,
- *start* is an expression, value of which is the initial value of index, and
- *count* is an expression, the value of which is the number of times the process is replicated.

D. Data Basics

OCCAM programs operate on constants and variables (with values assigned by an assignment instruction or by an input), channels and timers. The program data identifier and its type must be declared before use. The data types allowed in OCCAM are

- data types - integer, floating-point, Booleans, etc. types,
- channels - inter-process communication links which allow values to be communicated between concurrent processes,
- timers - special input-only channels which provide access to the transputer's real-time clock facility and are useful in real time programming applications.

E. Other issues

The OCCAM language provides a means of establishing *priorities* among processes. Situations such as the implementation of real-time applications, the implementation of a program in parallel on a single transputer, or, the implementation of a process where specific processing of data is required upon its input from a channel, benefit from the available priority mechanisms.

Two levels of priority are available to processes with the OCCAM/transputer combination. A process running at low priority in parallel with a high priority one is automatically de-scheduled in order for the high priority process to execute.

The keyword **PRI** may be associated with the constructions PAR and ALT, where the textual ordering determines their priority order.

The general format assumed in these situations is

```
PRI PAR
  process 1
  .
  .
  process n
```

OCCAM has a useful set of *operators* (arithmetic, relational, boolean, bitwise and shift) and it also supports the use of *abbreviations*, *functions* and *procedures*.

F. Configuration

The mapping of processes onto one or more transputers implies the independent existence of these processes and the establishment of their communications through the transputers' physical links. This strategy enables the execution of concurrent processes on separate transputers for concurrent execution.

The logical mapping of processes is independent of the hardware configuration on which they will execute. A common approach is to first test an implementation on a single transputer to eliminate logical errors before allocation of the processes over a network of transputers.

There are three main actions which must be taken to configure processes onto a network of processors:

- specify the type of processors in the system,
- declare which processes will execute on which processor, and
- map occam channels to transputer links for inter-transputer communications.

These actions are achieved by using three language extensions:

PLACE PAR
PROCESSOR *number type*
PLACE *channel* **AT** *address*

The **PLACE PAR** extension specifies that the following named processes will be allocated to different processors. **PROCESSOR** is used to identify the processor in the network on which a process or processes are to execute, and the type of processor used (Eg. T8 for the T800 transputer). The last keyword, **PLACE**, specifies which hard channel between two processes on different transputers is going to be used.

2.4.3. Performance of Parallel Systems

The performance of parallel algorithms running on parallel processing systems is strongly dependent on the computers used to implement the algorithms, on the size of the problem considered, and the algorithms themselves used to parallel implement a solution.

Although there is no single metric to define the absolute performance of an algorithm [61], the performance measures *R/C*, *execution time*, *speedup*, and *efficiency* are commonly used to achieve an estimate of the general performance of an algorithm.

The R/C ratio is a measure of the effectiveness of the parallel implementation. Here R is the length of a run-time quantum and C is the length of the communications overhead produced by that quantum [42]. On a course-grain architecture with slow interprocessor communications this requires that individual tasks should involve a considerable computing effort. However, a fine-grain architecture with fast interprocessor communications will execute efficiently tasks having low computational effort.

Speedup, efficiency and execution times [62] are related metrics, expressing the effective usage of a particular machine. The execution time (T) is the effective elapsed time to run a particular job on a given machine. Speedup (s) is defined as the ratio of the elapsed time when executing a program on a single processor (the single processor execution time) to the execution time when P processors are used,

$$s = \frac{T[1]}{T[P]} \tag{2.22}$$

Efficiency (e) is defined as the average utilization of the P allocated processors,

$$e = \frac{T[1]}{P.T[P]} = \frac{s}{P} \tag{2.23}$$

Speedup assumes higher values as the number of processors increase (a linear development with the number of processors is desired), revealing high parallelization of the problem. Its dependency on the number of processors is also established by the efficiency metric. Efficiency close to unity suggests that the processors are used effectively.

CHAPTER 3

CHAPTER 3

Pulsed Doppler Ultrasound Signal Simulation

3.1. INTRODUCTION

The aim of this research work is to increase the sensitivity to blood vessel disease detection using spectral analysis of pulsed Doppler ultrasound signals scattered from blood. To achieve this, a comparative study of spectral estimation methods has to be performed. In order to compare and judge the different spectral estimation methods it is desirable to apply them to a common input reference signal. This reference signal should be able to accurately represent a pulsed Doppler ultrasound signal, and the characteristics of its time and frequency domain representations should also be known. Therefore, it is advisable that a Doppler simulated signal be applied on this work.

The simulated signals contemplate two extreme cases of spectral shape. The Gaussian and the rectangular shapes. The Gaussian spectral shape represents the situation where large vessels with undisturbed flow and small sample volumes are considered. The rectangular spectral shape simulates the case of the uniform sensitivity over the vessel cross-section, parabolic velocity profile, and, spectral broadening due to transit time effects and deviations from plane-wave conditions is negligible.

Two algorithms of Doppler ultrasound signal's simulation were considered. One based on Mo and Cobbold's [63] approach and the other based on Leuewen, Hoeks and Reneman's [64] theory. The algorithms are conceptually different. The former considers that the Doppler signal

is resultant from the backscatterer from a range of cell aggregates together with correlation of close scatterer position. The latter assumes that the Doppler signal arises from scatter from random fluctuations of bulk elastic modulus intensity and density. For simplicity, these algorithms will be referred in text as algorithm 1 and algorithm 2 respectively.

The following section describes the simulation algorithms employed, as well as a modification operated on algorithm 1 to compute the simulated signals in a faster computational way. Details of the software implementation performed are also presented.

The theoretical basis of the tests applied to the simulated signals are stated in section 3.3. The amplitude distribution of the simulated signals was tested using a Chi-square test (section 3.3.1.). The amplitude distribution of the spectrum was studied considering ensembles (of variable size) of spectra. Development of the theoretical expressions for both the distribution of amplitude of the ensemble averaged spectrum, and, its expected percentage relative variance, are included in section 3.3.2. Description of the implementation of these tests is also presented in section 3.3.3.

The results of the tests applied to the simulated signals, and the criteria used to select one of the algorithms are presented in section 3.4.

Finally, some conclusions are drawn in section 3.5.

3.2. SIMULATION ALGORITHMS

3.2.1. Algorithm 1

This algorithm is based on the Mo and Cobbold simulation study of a continuous wave Doppler ultrasound spectra in [63].

The continuous signal is simulated as

$$x(t) = \sum_{m=1}^{M} a_m . \cos(2\pi f_m t + \phi_m) \qquad (3.1)$$

where a_m is the amplitude of the Doppler signal component at frequency f_m, f_m is the m^{th} frequency point in a frequency range $[0, f_{max}[$, in which f_{max} represents the maximum frequency of the PSD of the Doppler signal, and ϕ_m is the phase shift of the m^{th} component, considering m=1,...,M.

The frequency f_m is given by (3.2), where $\Delta f = f_{max}/M$ is the frequency resolution.

$$f_m = (m - \frac{1}{2})\Delta f \qquad for\ m = 1,...,M \tag{3.2}$$

The amplitude a_m is computed as

$$a_m = \sqrt{2.S_x[f_m].\Delta f.y_m} \tag{3.3}$$

where $S_x[f_m]$ represents the PSD of the Doppler signal, and y_m is the m^{th} random variable of a set $\{y_m\}$ of independent Chi-square random variables with two degrees-of-freedom.

The PSD is chosen to represent either Gaussian or rectangular spectral shapes. The former is expressed by (3.4) [65], where f_c and b are (respectively) the mean frequency and half bandwidth of the spectrum.

$$S_x[f_m] = \frac{1}{b\sqrt{2\pi}}.e^{-\frac{1}{2}\left(\frac{f_m - f_c}{b}\right)^2} \tag{3.4}$$

The mean frequency is given by

$$f_c = \frac{\int_{-\infty}^{+\infty} f.S_x(f)\,df}{\int_{-\infty}^{+\infty} S_x(f)\,df} \tag{3.5}$$

The half bandwidth of the spectrum, defined as

$$b = \left(\frac{\int_{-\infty}^{+\infty} (f - f_c)^2.S_x(f)\,df}{\int_{-\infty}^{+\infty} S_x(f)\,df} \right)^2 \tag{3.6}$$

[20] where the integration in the denominator is unity [66], represents a rms bandwidth and is coincident with the standard deviation of the Gaussian function.

The expression of the PSD in the case of rectangular spectral shape, is generally expressed by (3.7) [5], where f_{max} is the maximum frequency observed and n is the profile constant, assuming in this study the value 2, representing the parabolic velocity profile.

$$S_x[f_m] = \left(1 - \frac{f_m}{f_{max}}\right)^{\frac{2-n}{n}} \tag{3.7}$$

Each random variable y_m (in equation (3.3)) has probability density function

$$f(y_m) = \begin{cases} \frac{1}{2}.e^{-\frac{y_m}{2}} & y_m \geq 0 \quad m = 1,...,M \\ 0 & otherwise \end{cases} \tag{3.8}$$

[63] being generated from

$$y_m = -2.\ln(b_m) \qquad m = 1,...,M \tag{3.9}$$

where b_m are independent random variables, each one uniformly distributed over [0,1].

The phase shifts ϕ_m in (3.1) are obtained from independent random variables uniformly distributed over $[0,2\pi]$.

Each sinusoidal component in (3.1) represents the contribution of the scatterers whose velocities are coded into Doppler shifts within narrow frequency bands of size Δf. The sum of these contributions results in the Doppler signal.

A requirement of this algorithm (as proved by Mo and Cobbold [63]) is that the number of frequency points considered, M, should be much greater than the product $f_{max}T$, say $M=cf_{max}T$, for c representing a constant assuming different values. The larger the M values used in the simulation, the more closest to a continuous spectrum will the spectrum of the simulated signal be.

This algorithm originally applied by Mo and Cobbold [63] to simulate continuous Doppler ultrasound signals, was now sampled at a frequency f_s, giving rise to a discrete Doppler ultrasound signal.

The data sequence was obtained computing x(t) in expression (3.1), at times nT_s for n=0,..., N-1, where T_s is the sampling period.

3.2.2. Algorithm 2

This other algorithm is based on the simulation model that Leuewen, Hoeks and Reneman [64] used on their simulation of real-time frequency estimators for pulsed Doppler systems.

If $\bar{P}[k]$ is the Fourier transform of the desired Doppler signal, then the simulated Doppler signal x[n] can be obtained as the real part of the inverse Fourier transform ($\bar{x}[n]$) of $\bar{P}[k]$ [67].

$$\bar{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \bar{P}[k].e^{j2\pi\frac{kn}{N}} \qquad for \quad n=0,...,N-1 \tag{3.10}$$

That is

$$x[n] = Re(\bar{x}[n]) \qquad for \quad n=0,...,N-1 \tag{3.11}$$

and $\bar{P}[k]$ obtained by application of a FFT algorithm, contains information about the PSD of the Doppler signal, $S_x[f_k]$, weighted by a zero mean Gaussian function presenting unit variance. The real (X[k]) and imaginary (Y[k]) parts of this frequency domain signal are

$$\begin{aligned} X[k] &= Re(\bar{P}[k]) = \sqrt{S_x[f_k]}.C_{1k} \\ Y[k] &= Im(\bar{P}[k]) = \sqrt{S_x[f_k]}.C_{2k} \end{aligned} \tag{3.12}$$

where k=0,....,(N/2-1) and, C_{1k} and C_{2k} are independent random variables, representing the zero mean Gaussian weighting function.

The PSD $S_x[f_k]$ is computed as in algorithm 1 (see (3.4) and (3.7)), using f_k instead of f_m, where f_k=0,....,(N-1)f_s/(2N), and f_c, b and f_{max} have the same meaning as in algorithm 1.

In order to have a real signal x[n], $\bar{P}[k]$ must be created such that exactly half of its points are the complex conjugate of the other half. Thus both its real (X[k]) and imaginary (Y[k]) parts are computed for half the number of sample points N [64], and then

$$\begin{aligned} X(N-k+1) &= X(k+1) \\ Y(N-k+1) &= -Y(k+1) \\ X(\frac{N}{2}+1) &= Y(\frac{N}{2}+1) = 0 \end{aligned} \tag{3.13}$$

3.2.3. Modified Algorithm 1

In algorithm 1 the Doppler signal is simulated by adding M sinusoidal components as given by (3.1). Since M has to be made much greater than $f_{max}T$, the number of sample points contained in the Doppler signal simulation using this algorithm is much larger than the one using algorithm 2, leading to greater computational times. To overcome this problem some modifications have been introduced in algorithm 1, when the simulation of a Gaussian spectral shaped signal is considered.

In the case of the Gaussian spectral shaped signal simulation, the effective number of computational loops required for the evaluation of the summation expressed in (3.1) are reduced. This is due to the need of truncating the Gaussian function at a minimum (f_{min}) and maximum (f_{max}) frequencies. Therefore, $S_x[f_m]$ assumes non-zero values only over a finite range of frequencies $[f_{min}, f_{max}]$, outside which the a_m amplitude of the Doppler signal is forced to zero.

Imposing this constraint, the software loop that calculates the parameter a_m, needs to consider only the sampling points in the range $[cTf_{min}, cTf_{max}]$. Thus, the PSD $S_x[f_m]$ may be evaluated only in M' points given by

$$M' = \quad c.T.(f_{max} - f_{min}) \tag{3.14}$$

The frequency values f_{min} and f_{max} are established using statistical practical considerations about the Gaussian distribution. The area under the curve of the probability density function of a Gaussian random variable is concentrated (95%) between the limits $\mu \pm 2\sigma$ [68], where μ and σ are respectively the statistical mean and standard deviation of the random variable. Using this reasoning, the amplitude of the simulated signal correspondent to an α percentage of its maximum value, will determine the limiting frequencies, f_{min} and f_{max}.

Since the effective value of the PSD employed during the simulations is the square root of $S_x[f_m]$, the maximum amplitude is $(b(2\pi)^{1/2})^{-1/2}$, so, the maximum and minimum frequencies are given by

$$f_{max} = f_c + 2.b.\sqrt{\ln(100) - \ln(\alpha)}$$
$$f_{min} = f_c - 2.b.\sqrt{\ln(100) - \ln(\alpha)} \qquad \textbf{(3.15)}$$

The above modifications have been implemented as a new version of the simulation algorithm 1.

3.2.4. Implementation of the Simulation Algorithms

These algorithms, algorithm 1 (original and modified versions) and algorithm 2, were implemented and tested on a VAX 8600 with an operating system VMS V5.3. They were included in a software program named *DOPSYNA*, written in Fortran 77 [69], which makes use of routines of the NAG software package [70], and employs for graphical outputs the library Ginograf version 2.3 [71].

The Doppler signal was simulated considering values of mean frequencies from 500Hz up to 8kHz with intervals of one octave and with half bandwidth values of 5%, 10%, and 20% of the mean frequency. The data sequence duration used was of 10ms with a maximum possible variation of 2%.

The relationship between the mean frequencies and the respective sampling frequencies was conditioned by three factors:

a) the requirement of a number of sampling points which is a power of two, due to the algorithm of FFT employed,

b) the fact that algorithm 1 is more efficient if the sampling frequency is kept just above twice the maximum signal frequency [63],

c) the limitation on the range of admissible frequencies of the PSD (see (3.15)), imposed by an amplitude threshold of α% of the maximum amplitude of the PSD. The threshold value was α=10.

Under these conditions the sampling frequencies are as shown in Table 3.1.

mean freq. f_c (kHz)	samp. freq. f_s (kHz)	no. sample pts N
0.5	3.2	32
1	6.4	64
2	6.4	64
4	13	128
8	26	256

Table 3.1: Signal simulation mean frequencies and, respective sampling frequencies and number of sampling points.

3.3. TESTS APPLIED TO THE SIMULATED SIGNALS

3.3.1. Testing the Amplitude of the Simulated Signal

In order to simplify the application of tests to the amplitude distribution of the simulated Doppler ultrasound signal, the amplitudes were first normalized. This way the symmetric curve of the distribution of amplitudes, representing a Gaussian PDF centred on the mean amplitude a and with a standard deviation s_a, would be transformed into a standard normal distribution centred on zero.

Considering x the random variable representing the amplitude distribution of the simulated signal, the normalized variable z and its PDF are given [72] by

$$z = \frac{x-a}{s_a} \in]-\infty, +\infty[\qquad f(z) = \frac{1}{\sqrt{2\pi}} . e^{-\frac{z^2}{2}} \tag{3.16}$$

The amplitude of the Doppler signal x(t) expressed by (3.1) has been proved to have a Gaussian PDF with zero mean [63]. Thus, in order to validate the simulation, the Gaussianity of the amplitude of the simulated Doppler signal has to be tested.

A Chi-square was chosen to determine if the statistical hypothesis 'the amplitude of the signal is Gaussian' was acceptable or not.

The significance level, i.e., the probability to reject the hypothesis when it is true, was considered to be of 5%, a reasonable value also chosen by Mo and Cobbold [63].

The time-domain simulated signal, existing as a discrete data sequence of N sample points is primarily classified into a statistical frequency distribution of amplitudes. This frequency distribution of amplitudes is composed of K classes. Generally the classes have equal length, which is computed to be approximately the ratio of the difference between the largest and the lowest amplitude and the number of classes required [73],

$$length\ of\ class = \frac{amplitude_{max} - amplitude_{min}}{no.\ of\ classes\ (k)} \tag{3.17}$$

Each element of the discrete signal is therefore assigned to one class only, and the classes cover the whole range of amplitudes in consideration. To avoid ambiguous allocation of elements between adjacent classes, for each class the admissible amplitudes of the signal belong to a right opened interval.

However, the time-domain signal simulation following a Gaussian PDF presents very low amplitudes towards the tail of the Gaussian curve. Thus, the upper limit of the first class, and, the lower limit of the last class have to be imposed, as it will be shown below. Among these two amplitude limits, K-2 classes (with equal lengths) are created.

The frequency distribution of amplitudes is then compared with a Gaussian distribution. The goodness-of-fit is measured by the variable [74],

$$\chi^2 = \sum_{i=1}^{K} \frac{(n_i - N.p_i)^2}{N.p_i} \tag{3.18}$$

where i=1,...,K represent the number of classes in which the histogram has been divided, N is the total number of sample points, n_i the number of sample points in the i^{th} class, and, p_i represents the probability of the random variable being inside class i. If $P(\chi^2 > c) < 0.05$ where c is a constant, the hypothesis of the generated signal being Gaussian is accepted.

To apply this test two requirements must be satisfied:

a) the statistic χ^2 as defined above, can be said to have approximately a Chi-square distribution if the total number of samples considered is sufficiently large. In the case, χ^2 presents a number of degrees-of-freedom given by [68]

$$degress-of-freedom = K-1-(number\ of\ estimated\ parameters) \quad (3.19)$$

b) the other relevant requirement to validate the application of the Chi-square test is the need of having each class of the histogram with at least five sample points, this is, making

$$N.p_i > 5 \quad (3.20)$$

in equation (3.18) [68].

Since the number of sampling points is determined by the product of the sampling frequency and the data sequence duration, and these parameters are limited by signal's specifications, the following approach is employed: the establishment of upper and lower limits of the first and last classes (respectively) of the frequency distribution of the amplitudes, are symmetric, due to the use of a standardised Normal distribution. Considering n_c as the minimum number of elements per class, where $n_c>5$, the upper limit of the first class will be given by the deviate associated with the lower tail probability n_c/N of the standardised Normal distribution. This way, the K-2 inner classes created with equal lengths, and the shape of the Gaussian curve, assure that if the first of these inner classes satisfies

$$N.p_i > n_c \quad (3.21)$$

all the other inner classes will also satisfy the same condition. If not, either n_c or K have to be adjusted.

3.3.2. Testing the Ensemble Averaged Amplitude of the Spectrum

The analysis of the frequency content of the simulated Doppler signals, was carried out by means of Fourier Transforming the simulated signal.

Due to the random nature of the Doppler signal, individual estimates are random, and spectral averaging should be performed to reduce the variance of the spectrum [30]. The selection of a size of ensemble which guarantees low spectral variance values is our concern. To achieve so, theoretical developments are presented to establish an expression for the relative variance of the

averaged amplitude of the spectrum obtained by Fourier Transforming the signal simulated by algorithm 2.

Representing the complex spectrum obtained by Fourier Transforming the simulated signal by

$$\{\overline{X_j}\} = \{\overline{X_1}, \overline{X_2}, \ldots, \overline{X_{N_h}}\} \tag{3.22}$$

where the index j represents the order of the sampling point considered, and $N_h=N/2$ ($N=f_sT$) denotes the need of computation of just half the spectrum (due to its symmetry), we may consider without loss of generality only one sample of the ensemble, and analyze its statistical distribution.

Let one component j of the spectrum be represent by

$$\overline{X_j} = XR_j + \sqrt{-1}\,.XI_j \tag{3.23}$$

where XR_j and XI_j are the real and imaginary parts of $\overline{X}_j$. These components, given by (3.12), may now be rewritten as

$$\begin{aligned} XR_j &= \sqrt{S_x[f_j]}\,.Y_R \\ XI_j &= \sqrt{S_x[f_j]}\,.Y_I \end{aligned} \tag{3.24}$$

For both components the first factor is given by the same function, the square-root of the PSD that is expected to be followed by the simulated Doppler signal. Y_R and Y_I are independent random variables following a Gaussian distribution with zero mean and unity variance.

We shall deal with the modulus of the spectrum, so each component j of the vector expressed in (3.22) will generally be represented by X_j, presenting as well a constant factor,

$$\begin{aligned} X_j = |\overline{X_j}| &= \sqrt{XR_j^2 + XI_j^2} \\ &= \sqrt{S_x[f_j]}\,.\sqrt{Y_R^2 + Y_I^2} \end{aligned} \tag{3.25}$$

The last factor of the above equation is a random variable, whose PDF we want to characterize.

It is known that the square of a random variable Y_R (or Y_I) with Gaussian distribution characterized by zero mean and unit variance, is a random variable Z with PDF

$$g(z) = \begin{cases} \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{\sqrt{z}} \cdot e^{-\frac{z}{2}} & z>0 \\ 0 & z \leq 0 \end{cases} \tag{3.26}$$

which corresponds to the PDF of a Chi-square random variable with one degree of freedom [74].

The sum of two independent random variables Z_1 and Z_2 with distribution χ_1^2, is a random variable W whose moment generating function $M_W(t)$ is

$$M_W(t) = E[e^{tW}] = E[e^{tZ_1}].E[e^{tZ_2}] \tag{3.27}$$

since Z_1 and Z_2 are independent random variables [Meyer, 83]. Thus,

$$M_W(t) = (1-2t)^{-\frac{1}{2}}.(1-2t)^{-\frac{1}{2}} = (1-2t)^{-1} \tag{3.28}$$

which is the moment generating function of a random variable following a χ_2^2 (two degrees of freedom) [75].

The square root of a random variable W with distribution χ_2^2 is a random variable U with PDF given by (3.29), following a Rayleigh distribution [66].

$$g(u) = \begin{cases} u.e^{-\frac{u^2}{2}} & u \geq 0 \\ 0 & u<0 \end{cases} \tag{3.29}$$

Thus, the random variable of (3.25) is Rayleigh distributed characterized by the following mean and variance

$$mean = \sqrt{\frac{\pi}{2}} \tag{3.30}$$

$$variance = \frac{4-\pi}{2}$$

[Papoulis, 84], which allows the definition of the random variable X_j of (3.25) as Rayleigh distributed with mean μ_j and variance σ_j^2 given by

$$\mu_j = \sqrt{S_x[f_j]} \cdot \sqrt{\frac{\pi}{2}}$$
$$\sigma_j^2 = S_x[f_j] \cdot \frac{4-\pi}{2} \qquad (3.31)$$

according to the properties of these statistics [74].

As stated before, we are interested in analyzing the statistical behaviour of the averaged amplitude of the spectrum. To do so, let us consider the modulus of the elements of the vector expressed in (3.22), rewritten taking into account the concepts of samples of an ensemble,

$$\{X_{ij}\} = \{X_{i1}, X_{i2}, \ldots, X_{iN_h}\} \qquad for\ i = 1,\ldots,L \qquad (3.32)$$

where the index i represents the i^{th} sample of a total number of L samples.

Computing the average of the above L vectors, the averaged amplitude of the j^{th} frequency component of the spectrum, XM_j, where j may vary up to N_h

$$\{XM_j\} = \left\{\frac{1}{L}\sum_{i=1}^{L} X_{ij}\right\} = \left\{\frac{1}{L}\sum_{i=1}^{L} X_{i1}, \frac{1}{L}\sum_{i=1}^{L} X_{i2}, \ldots, \frac{1}{L}\sum_{i=1}^{L} X_{iN_h}\right\} = \qquad (3.33)$$
$$= \{XM_1, XM_2, \ldots, XM_{N_h}\}$$

Each XM_j of the vector $\{XM_j\}$ is the statistical mean of the L observations of the random variable X_j, Therefore its mathematical expectation (E[•]) is coincident with the theoretical mean of X_j, and, its variance (Var[•]), is the theoretical variance divided by the number of samples in the ensemble [74] [76]. Making use of (3.31),

$$E[XM_j] = \mu_j = \sqrt{S_x[f_j] \cdot \frac{\pi}{2}} \qquad (3.34)$$
$$Var[XM_j] = \frac{\sigma_j^2}{L} = S_x[f_j] \cdot \frac{4-\pi}{2L}$$

Since we are also interested in determining the variance of the averaged amplitude of the spectrum (expressed in (3.33)), each component j of $\{XM_j\}$, for j=1, ...,N_h may be regarded by itself as a sample of an ensemble of size N_h. This way, the sample variance is computed as

$$\frac{1}{N_h}\sum_{j=1}^{N_h}(XM_j - E[XM_j])^2 \tag{3.35}$$

To obtain a percentage relative variance of the averaged amplitude of the spectrum we may consider the computation of

$$\frac{1}{N_h}\sum_{j=1}^{N_h} ZM_j \qquad with \quad ZM_j = \frac{(XM_j - E[XM_j])^2}{(E[XM_j])^2}.100 \tag{3.36}$$

The value of the summation in (3.36) can be obtained making use of an auxiliary random variable YM_j, which is the relative averaged amplitude of the spectrum, defined as

$$YM_j = \frac{XM_j}{E[XM_j]} \tag{3.37}$$

This variable, presenting unit expectation, enables an equivalent expression of the summation in (3.36),

$$\frac{1}{N_h}\sum_{j=1}^{N_h}(YM_j - 1)^2 \tag{3.38}$$

simplifying the calculation of the percentage relative variance of the averaged amplitude of the spectrum, by computing only the variance of YM_j (see (3.39)), where the approximate value is obtained by assuming that a high number of sample points is considered.

$$Var[YM_j] = \frac{Var[XM_j]}{(E[XM_j])^2} = \frac{4-\pi}{\pi L} \approx \frac{0.2732}{L} \tag{3.39}$$

3.3.3. Implementation of the Tests

The *DOPSYNA* program (see section 3.2.4.) includes also an FFT algorithm, transcribed from Kay' software package [17] to enable the computation of the Doppler signal simulated spectrum. The FFT routine processes complex vectors with a number of points equal to a power of two, and enables the computation of either the direct or the inverse Fourier Transform.

DOPSYNA is a modular program which enables the user to select various processing options. All specifications of the Doppler signal are introduced by the user. The program enables the study of the amplitude distribution of the Doppler ultrasound signal simulation by means of a Chi-square test. The Chi-square value and the correspondent significance level is presented on screen to the user. Optionally, a plot of the histogram of amplitudes of the signal against the theoretical Gaussian curve may be obtained.

An ensemble of simulated Doppler signals may be created, according to the user's specification of size of the ensemble. Graphical outputs of the ensemble averaged spectra of the simulated signals can be plotted against the expected amplitude of the spectrum, together with the percentage relative variance of the ensemble averaged amplitude spectrum.

3.4. RESULTS

3.4.1. Selection of one Simulation Algorithm

The amplitude distribution of the simulated signals created by simulation algorithms 1 and 2 was tested. Since algorithm 1 presents greater computational effort than algorithm 2, specially when a rectangular spectral shape is chosen, the first step of comparison of algorithm behaviour was concerned with the Gaussian spectral shape and the use of the modified algorithm 1, proposed in section 3.2.3.

The results of the Chi-square test applied to the simulated signals obtained from both algorithms are shown on the following tables. The constant M' is the number of sinusoidal components used in modified algorithm 1. M' is computed in accordance with equations (3.14) and (3.15) putting α=10. The tables indicate the limits of the significance levels obtained in each test (a total of 5), when the number of degrees of freedom are chosen considering the estimation of two parameters, the mean frequency and the bandwidth of the spectrum (see (3.19)).

algorithm	test 1	test 2	test 3	test 4	test 5
1	25-50	5-25	1-5	25-50	5-25
2	25-50	25-50	5-25	50-75	5-25

(a)

algorithm	test 1	test 2	test 3	test 4	test 5
1	5-25	25-50	5-25	5-25	5-25
2	25-50	50-75	50-75	5-25	5-25

(b)

algorithm	test 1	test 2	test 3	test 4	test 5
1	50-75	5-25	25-50	25-50	25-50
2	5-25	5-25	25-50	5-25	25-50

(c)

algorithm	test 1	test 2	test 3	test 4	test 5
1	25-50	5-25	5-25	50-75	25-50
2	5-25	5-25	5-25	5-25	5-25

(d)

algorithm	test 1	test 2	test 3	test 4	test 5
1	25-50	50-75	5-25	25-50	50-75
2	25-50	50-75	5-25	50-75	5-25

(e)

algorithm	test 1	test 2	test 3	test 4	test 5
1	5-25	5-25	50-75	50-75	50-75
2	75-95	5-25	50-75	25-50	50-75

(f)

Tables 3.2: Results of the Chi-square test applied to simulated signals with: f_c=500Hz, f_s=3.2kHz, 1 degree of freedom and **(a)** b=25Hz, M'=608, **(b)** b=50Hz, M'=1216, **(c)** b=100Hz, M'=1212; f_c=1kHz, f_s=6.4kHz, 5 degrees of freedom and **(d)** b=50Hz, M'=608, **(e)** b=100Hz, M'=606; and, **(f)** f_c=8kHz, f_s=26kHz, 7 degrees of freedom, b=1.6kHz, M'=971.

The above Chi-square test results show that the Doppler signals simulated by both algorithms presented similar properties within the significance values assumed.

Although algorithm 1 was improved in terms of computational time for the case of Gaussian spectral profile, its modified version was still much slower than algorithm 2. In fact, running both algorithms under similar specifications, rough measures of typical processing times for the modified algorithm 1 and algorithm 2 may be contrasted on the Table 3.3, where case 1 represents the simulated signal with f_c=500Hz, b=25Hz, f_s=3.2kHz and M'=576, and, case 2 corresponds to a signal with f_c=8kHz, b=1.6kHz, f_s=26kHz and M'=8369.

algorithm	case 1	case 2
modified alg. 1	1 min : 47/100 sec	1 min : 5123/100 sec
alg. 2	40/100 sec	61/100 sec

Table 3.3: Comparison of processing times using by algorithms 1 (modified) and 2 for two different simulation cases.

Therefore, as the underlying theories which gave rise to both algorithms are essentially the same [27], the selection of one of these algorithms relied on considerations of computational time efficiency, leading to the selection of algorithm 2.

3.4.2. Testing the Selected Algorithm

The simulated Doppler ultrasound signal obtained from algorithm 2 was fully tested in terms of both the Gaussianity of its amplitude distribution and the characteristics of its PSD.

The signal was deliberately oversampled to ensure that the sampling did not become a consideration in the statistical measurements. The simulated signal was tested in the situation of a sampling frequency of 2^{17}Hz (131072Hz) and a data sequence of 1 second.

Two types of signal were simulated, one with a rectangular spectral shape with a bandwidth of $f_s/2$, and the other with a Gaussian spectral shape with a mean frequency of $f_s/4$ and a half bandwidth of 5% of the mean frequency.

These simulated signals presented as a common feature very low amplitudes. As a consequence, the mean amplitude of the signal and its standard deviation assumed also very small values.

Common values of these statistics, mean amplitude (a) and standard deviation (s_a), are $3*10^{-6}$ and $3.7*10^{-3}$, and 0 and $1.5*10^{-5}$, respectively for rectangular and Gaussian spectral shapes.

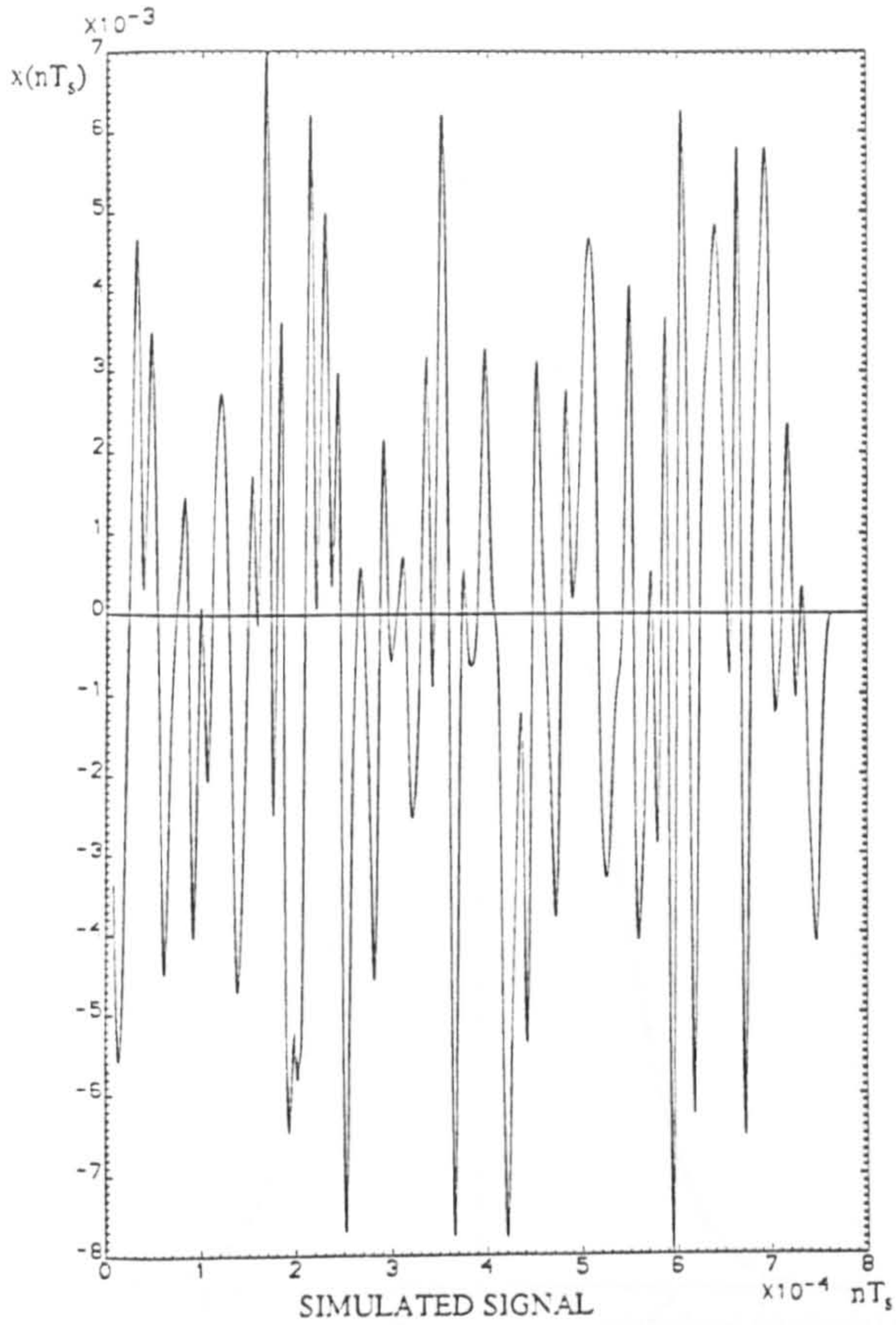
These very low amplitudes of the simulated Doppler signal are justified by the expression of the inverse Fourier Transform of a signal, generally given by

$$\bar{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \bar{X}[\,k].e^{j2\pi\frac{kn}{N}} \qquad n=0,\ldots,N-1 \tag{3.40}$$

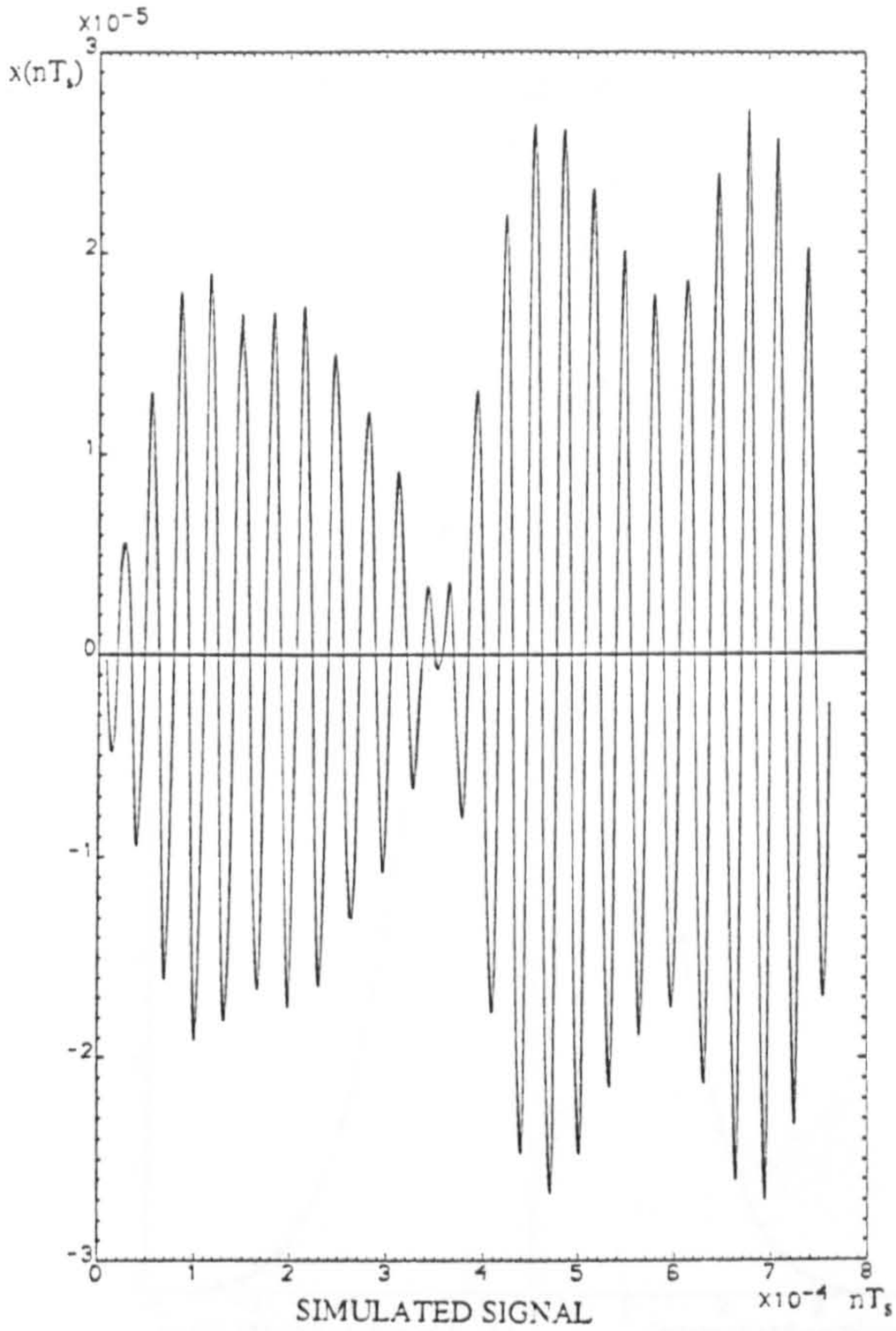
where $N=f_sT$ is very large and $\bar{X}[k]$, assumes very low values in the case of a Gaussian spectral profile.

Fig. 3.1 presents the simulated Doppler ultrasound signal for rectangular and Gaussian spectral shapes. Since the number of sample points considered is very large, only the first 100 points are shown.

The existence of a non-zero value of a, specially in the case of a rectangular spectral shape justifies the standardization mentioned in section 3.3.1.



(a)



(b)

Fig. 3.1: Simulated Doppler signal with (**a**) rectangular spectral shape with $f_s=2^{17}$Hz and $f_{max}=f_s/2$, (**b**) Gaussian spectral shape with $f_s=2^{17}$Hz and $f_c=f_s/4$ and $b=0.05f_c$.

Considering the simulated signal with the specifications above mentioned, the frequency distribution of amplitudes was graphically output in a form of an histogram with 40 classes. The Chi-square variable presented 37 degrees of freedom assuming, for the previously established 5% significance level, the value [70],

$$\chi^2_{37,0.95} = 52.19 \qquad \textbf{(3.41)}$$

Tests applied to several simulated signals proved that on 6% and 13% of the simulations of rectangular and Gaussian spectral shapes respectively, the simulated signals were outside the significance level.

The histograms of the amplitude distribution of the two simulated Doppler signals shown in Fig. 3.1 are presented in Fig. 3.2, whose statistical characteristics are shown in Table 3.4.

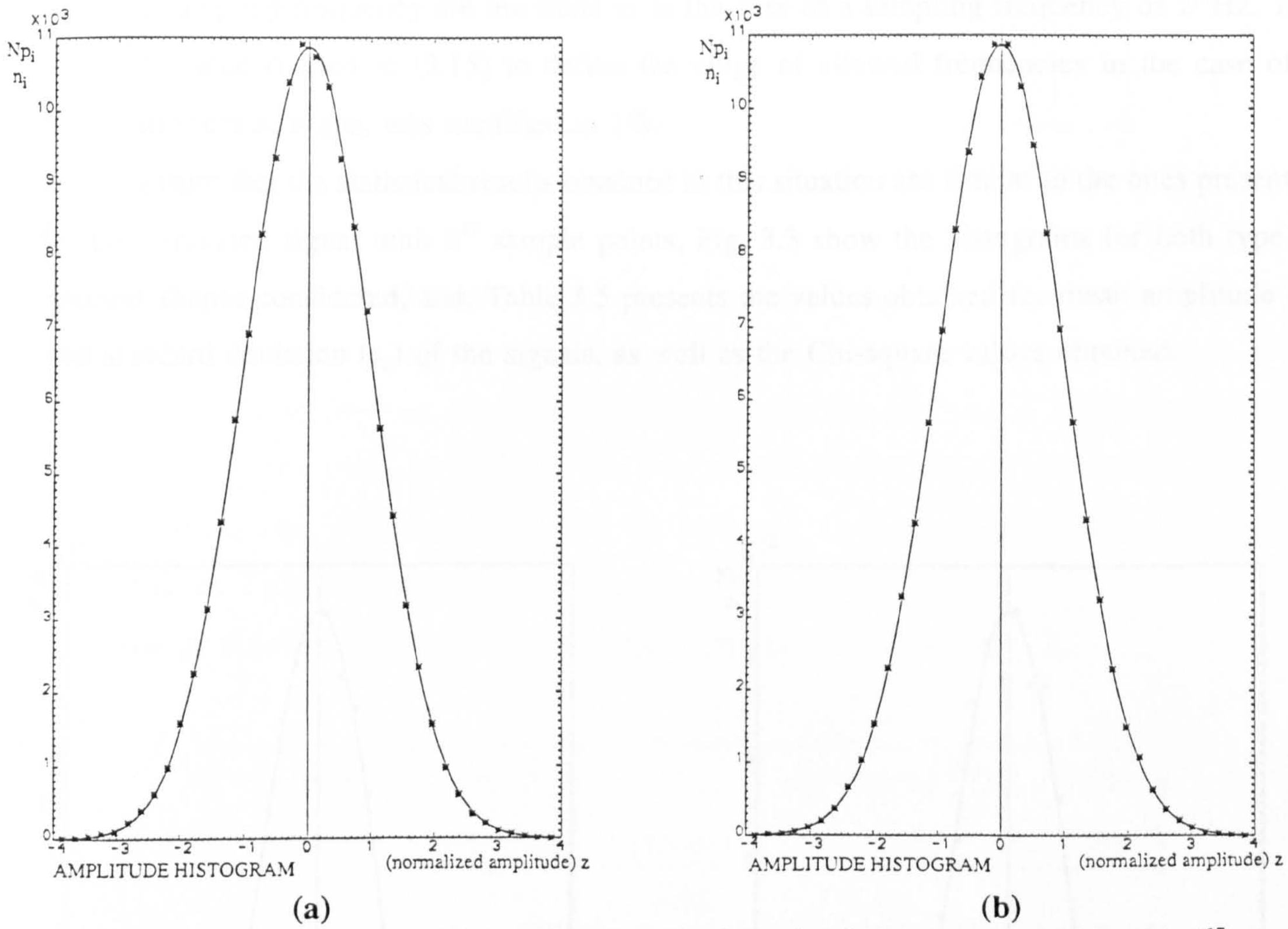


Fig. 3.2: Histogram of the amplitude distribution of the simulated Doppler signal with 2^{17} sample points with (**a**) rectangular spectral shape, (**b**) Gaussian spectral shape, against the theoretical amplitude distribution (solid line).

Spectral shape	a	s_a	$\chi^2_{observed}$	$\chi^2_{37,0.95}$
rectangular	$-0.3*10^{-5}$	$3.9*10^{-3}$	48.47	52.19
Gaussian	0	$1.5*10^{-5}$	36.94	52.19

Table 3.4: Characteristics of the amplitude distribution of the simulated signals of Fig. 3.2.

To study the spectrum of the simulated signal it is necessary to consider an ensemble which is in general constituted by a large number of samples. Since the simulation of one signal with 2^{17} points is by itself a slow process, the execution of a number of samples would lead to long computational times. For this reason, simulated signals with only 2^{13} (8192) points are considered. For simplicity this signal is sampled at 2^{13}Hz with a data sequence of 1 second.

The relations between the maximum frequency, the mean frequency and the half bandwidth with the sampling frequency are the same as in the case of a sampling frequency of 2^{17}Hz. The threshold value α used in (3.15) to define the range of allowed frequencies in the case of a Gaussian spectral shape, was modified to 1%.

To assure that the statistical results obtained in this situation are similar to the ones presented by the simulated signal with 2^{17} sample points, Fig. 3.3 show the histograms for both type of spectral shapes considered, and, Table 3.5 presents the values obtained for mean amplitude (a) and standard deviation (s_a) of the signals, as well as the Chi-square values obtained.

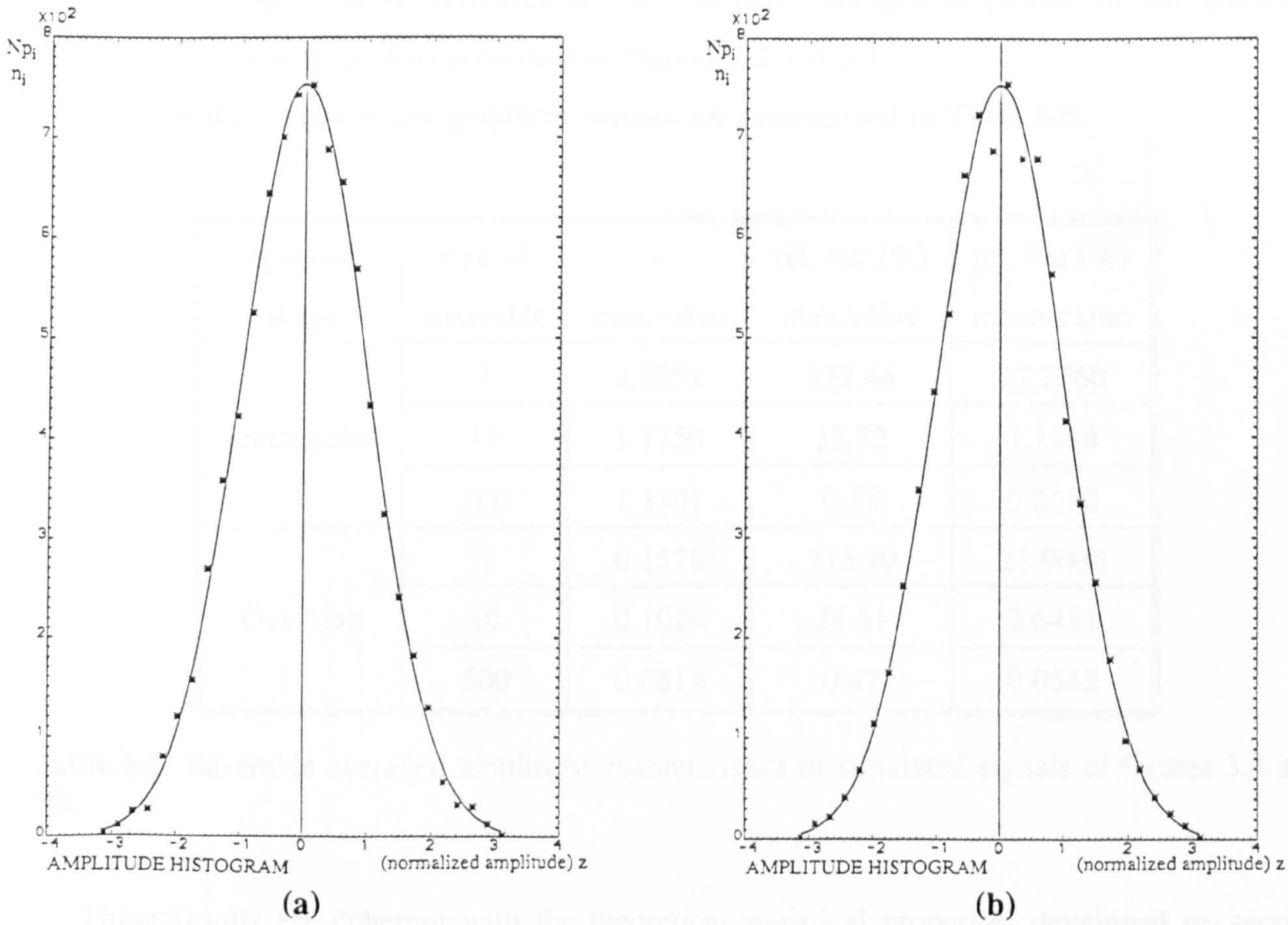


Fig. 3.3: Histogram of the amplitude distribution of the simulated Doppler signal with 2^{13} sample points with (**a**) rectangular spectral shape, (**b**) Gaussian spectral shape, against the theoretical amplitude distribution (solid line).

Spectral shape	a	s_a	$\chi_{observed}^2$	$\chi_{37,0.95}^2$
rectangular	$-0.4*10^{-5}$	$15.6*10^{-3}$	31.76	40.11
Gaussian	0	$24.8*10^{-5}$	25.57	40.11

Table 3.5: Characteristics of the amplitude distribution of the simulated signals of Fig. 3.3.

As stated earlier, *DOPSYNA* has graphical output facilities to help on the analysis of the spectrum of the simulated signal. The development of the ensemble averaged amplitude of the spectrum with different sizes of ensemble is illustrated in Figures 3.4 and 3.6 for ensembles of size 1 (i.e. a single signal), 10, and 500. The theoretical expected curve is included for reference.

The percentage relative variances of the ensemble averaged amplitude of the spectrum (computed by (3.36)) are also presented in Figures 3.5 and 3.7.

Some of the results of the graphical outputs are summarized in Table 3.6.

spectral shape	size of ensemble	a max.value	rel. var.(%) max.value	rel. var.(%) mean value
rectangular	1	2.6958	132.46	27.2760
	10	1.7750	17.32	2.7414
	500	1.3307	0.38	0.0552
Gaussian	1	0.1578	315.99	27.9006
	10	0.1029	24.81	2.6431
	500	0.0814	0.47	0.0542

Table 3.6: Ensemble averaged amplitude characteristics of simulated signals of figures 3.4 and 3.6.

These results are coherent with the theoretical statistical properties developed on section 3.3.2.. In fact, the practical values depicted on the last column of the table above closely agree with the expected values given by (3.39), this is, 0.273/L (*100%).

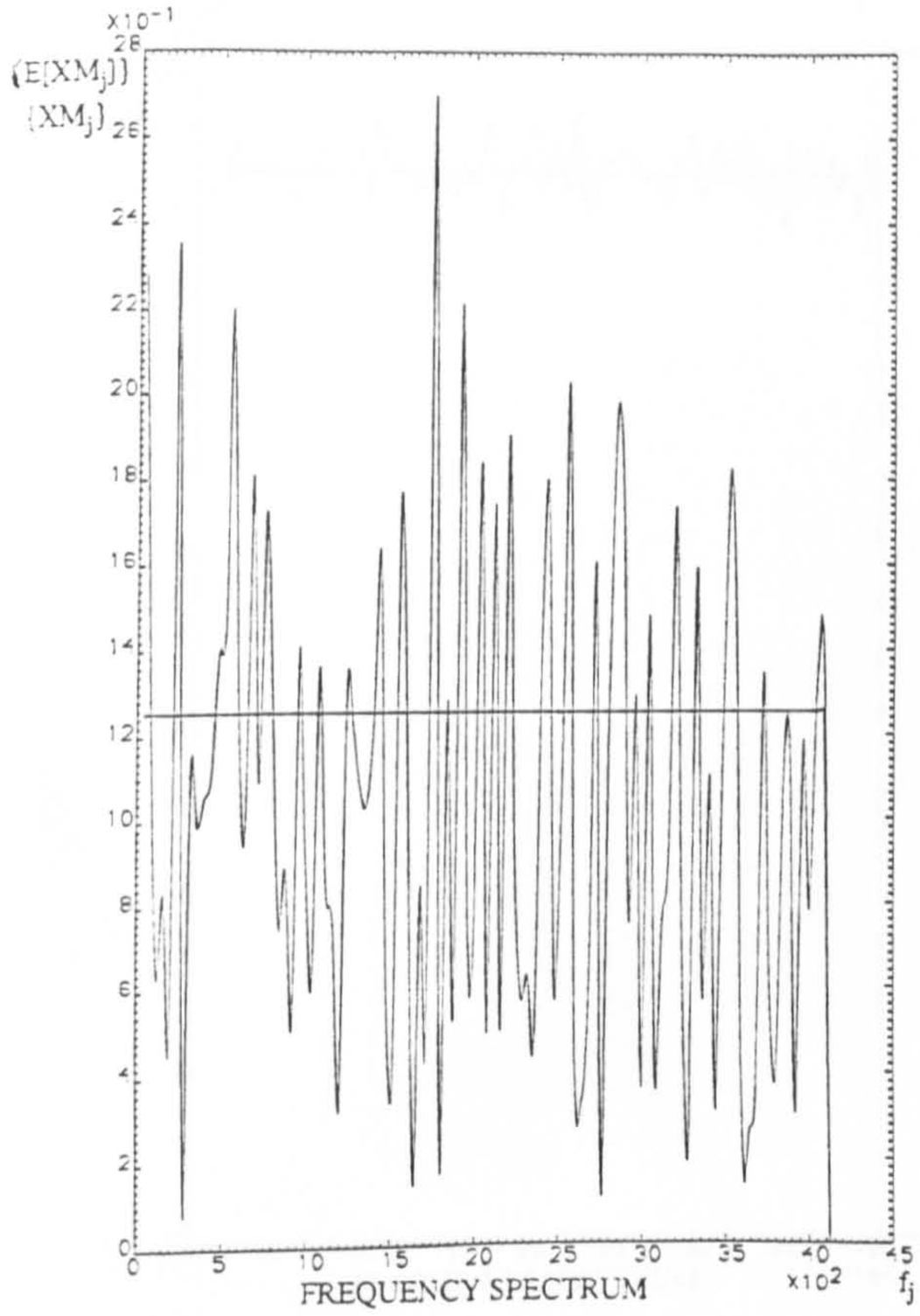


Fig. 3.4 (a)

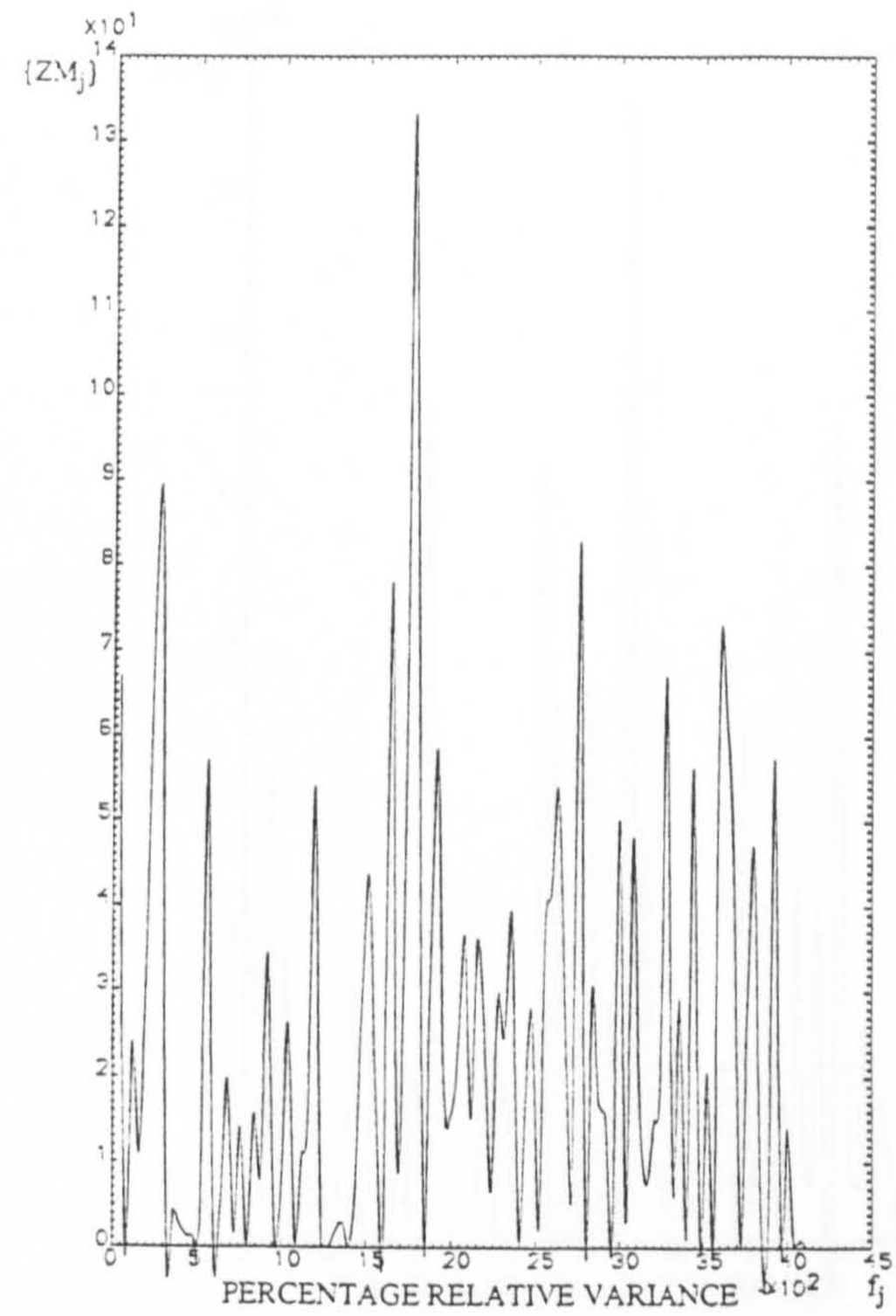


Fig. 3.5 (a)

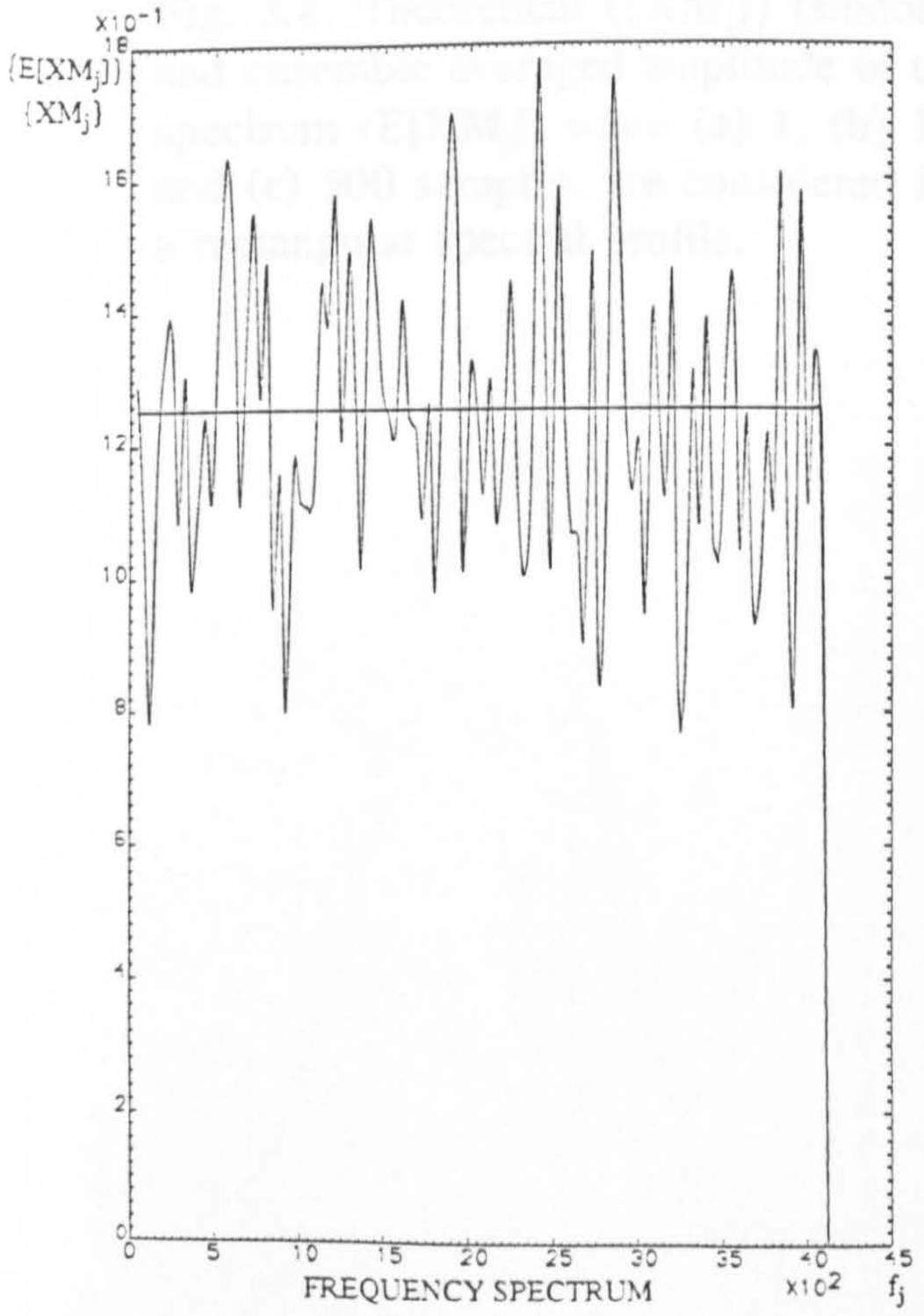


Fig. 3.4 (b)

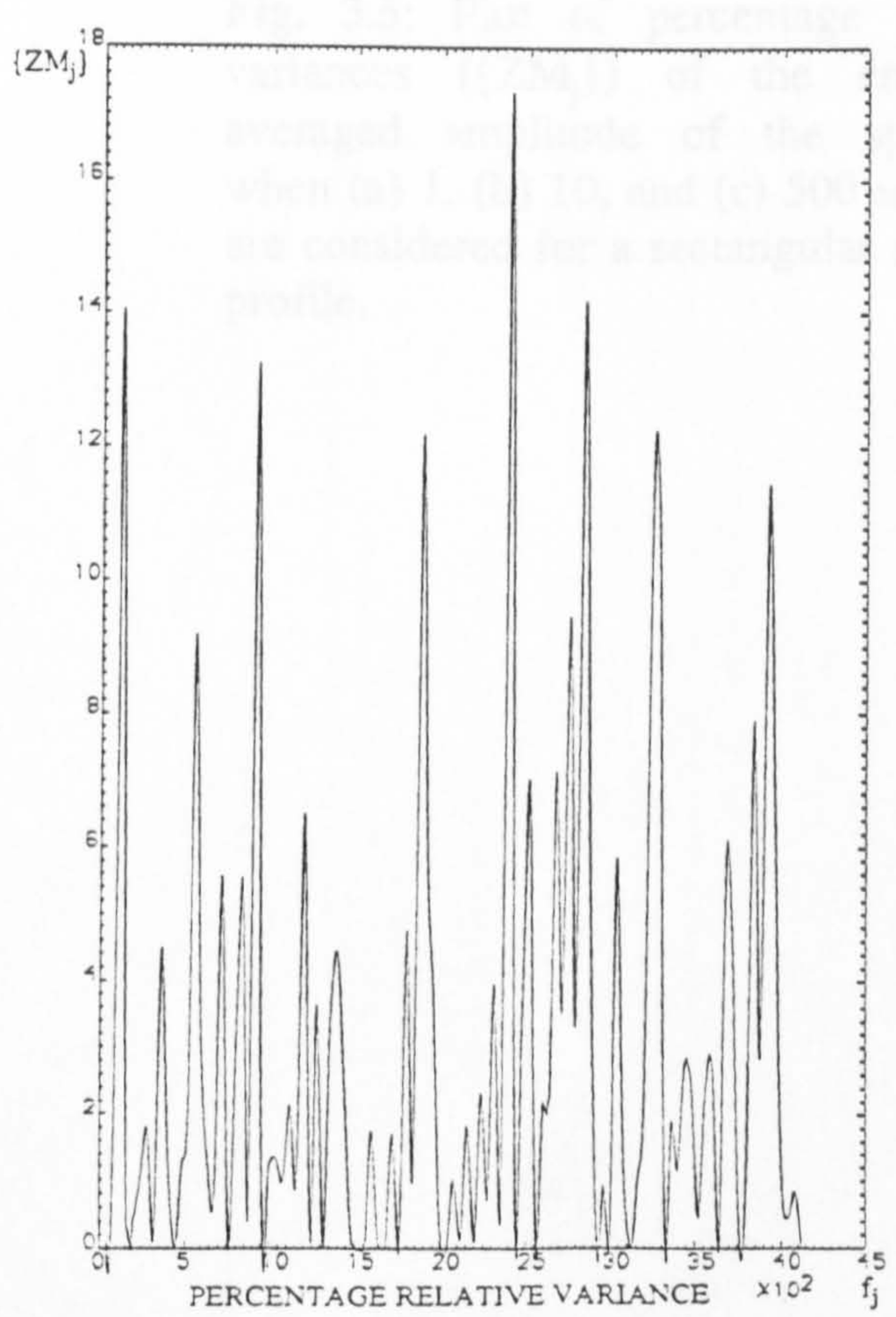


Fig. 3.5 (b)

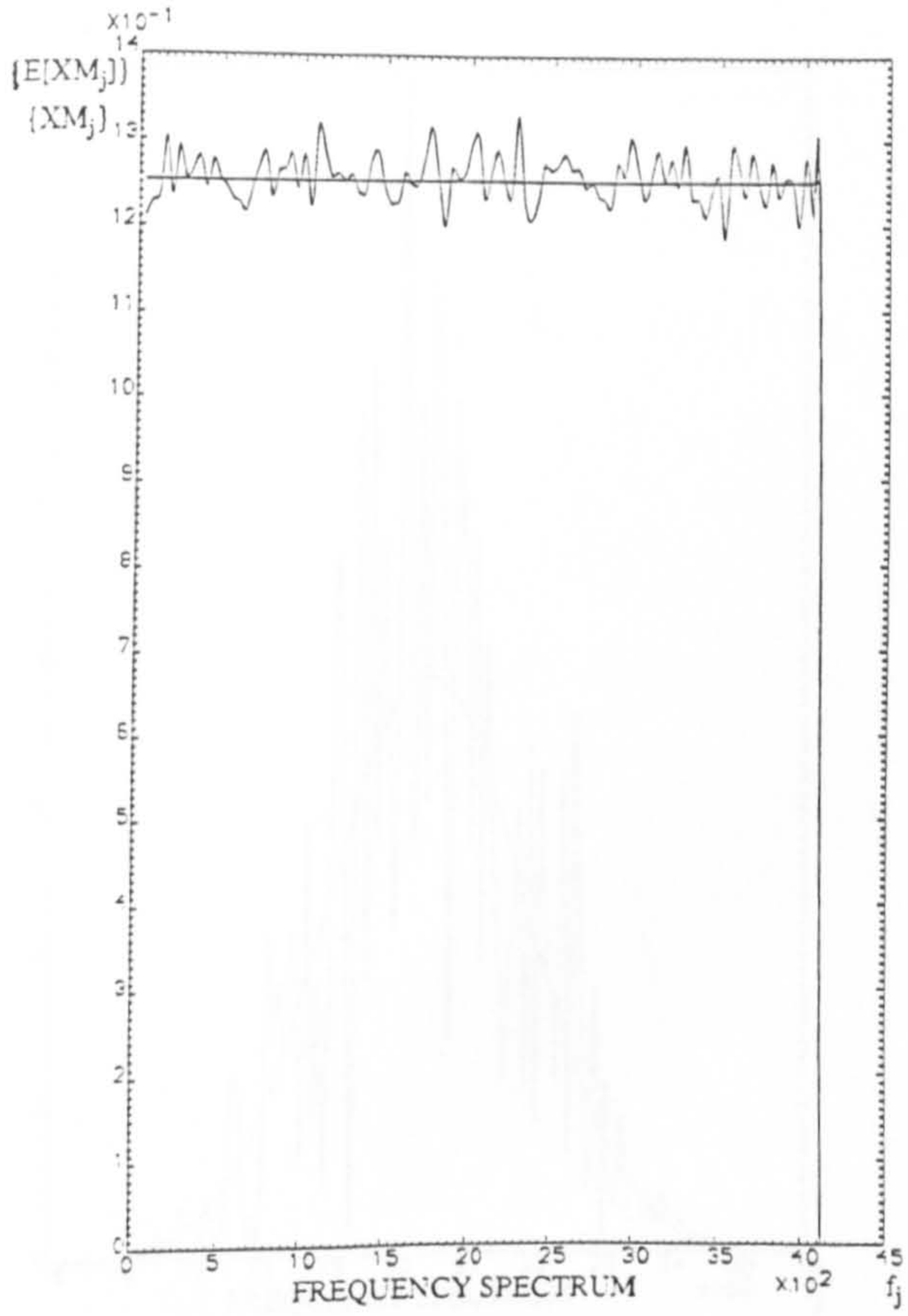


Fig. 3.4 (c)

Fig. 3.4: Theoretical ($\{XM_j\}$) (smooth) and ensemble averaged amplitude of the spectrum ($E[XM_j]$) when (**a**) 1, (**b**) 10, and (**c**) 500 samples, are considered for a rectangular spectral profile.

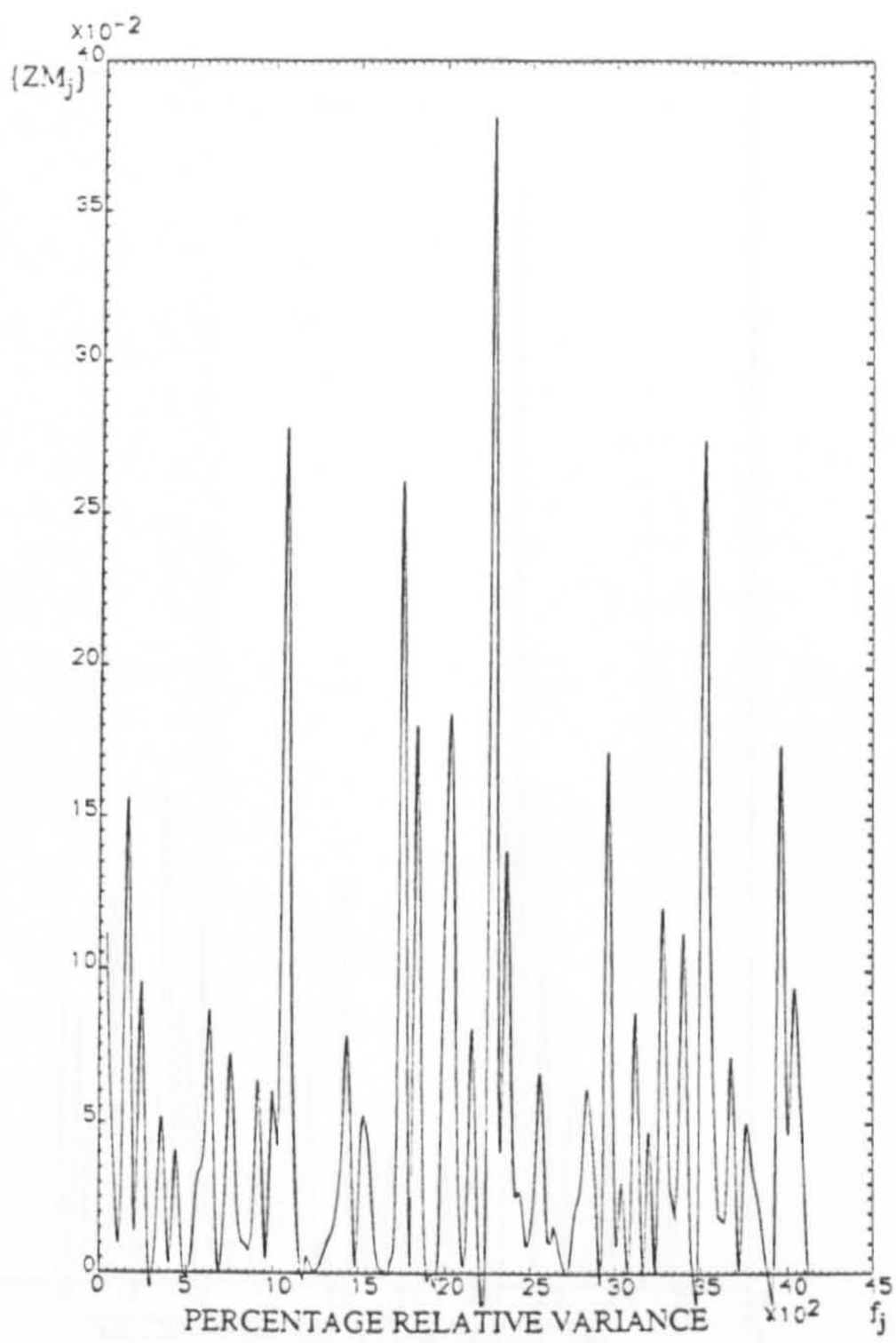


Fig. 3.5 (c)

Fig. 3.5: Plot of percentage relative variances ($\{ZM_j\}$) of the ensemble averaged amplitude of the spectrum when (**a**) 1, (**b**) 10, and (**c**) 500 samples, are considered for a rectangular spectral profile.

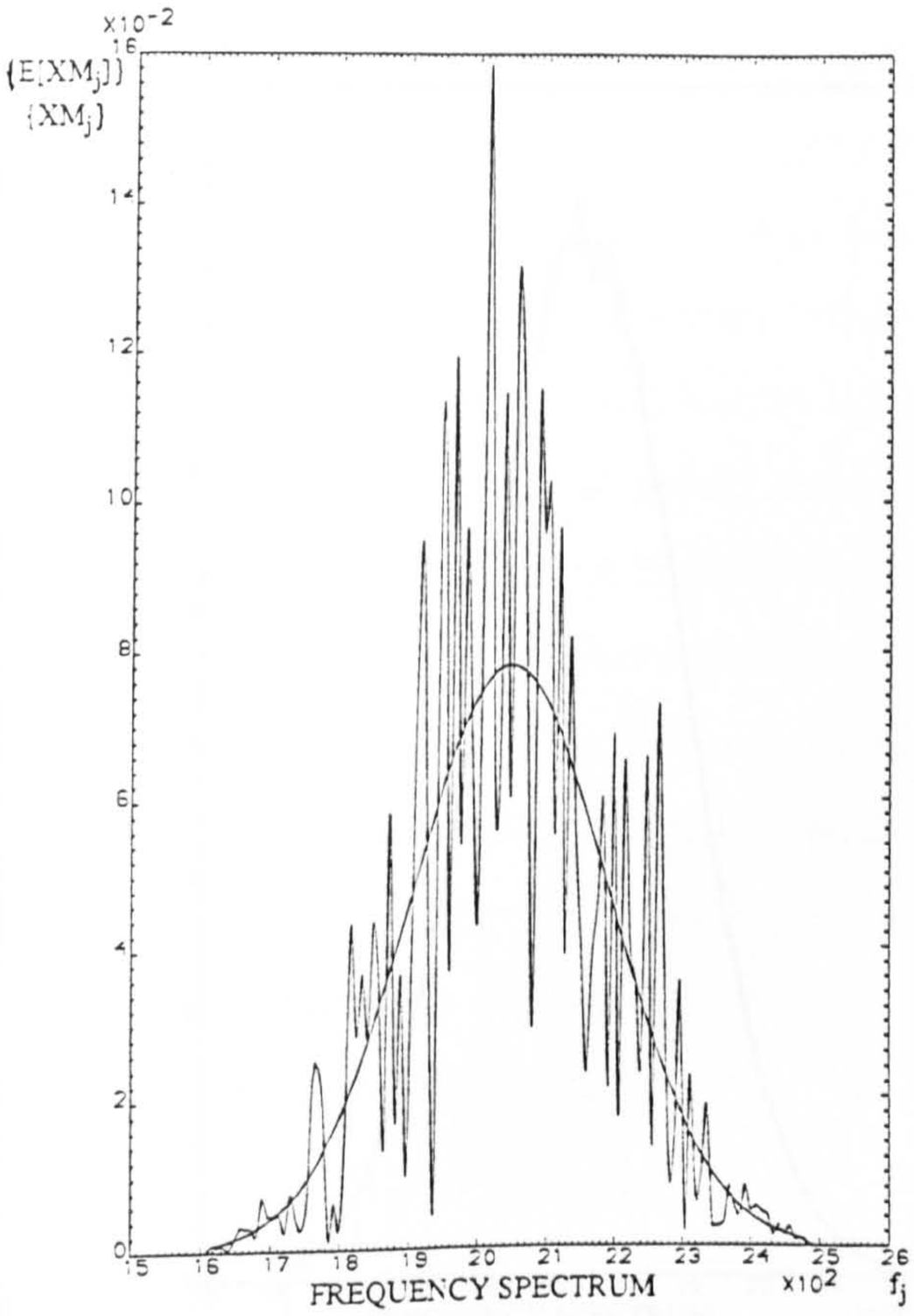


Fig. 3.6 (a)

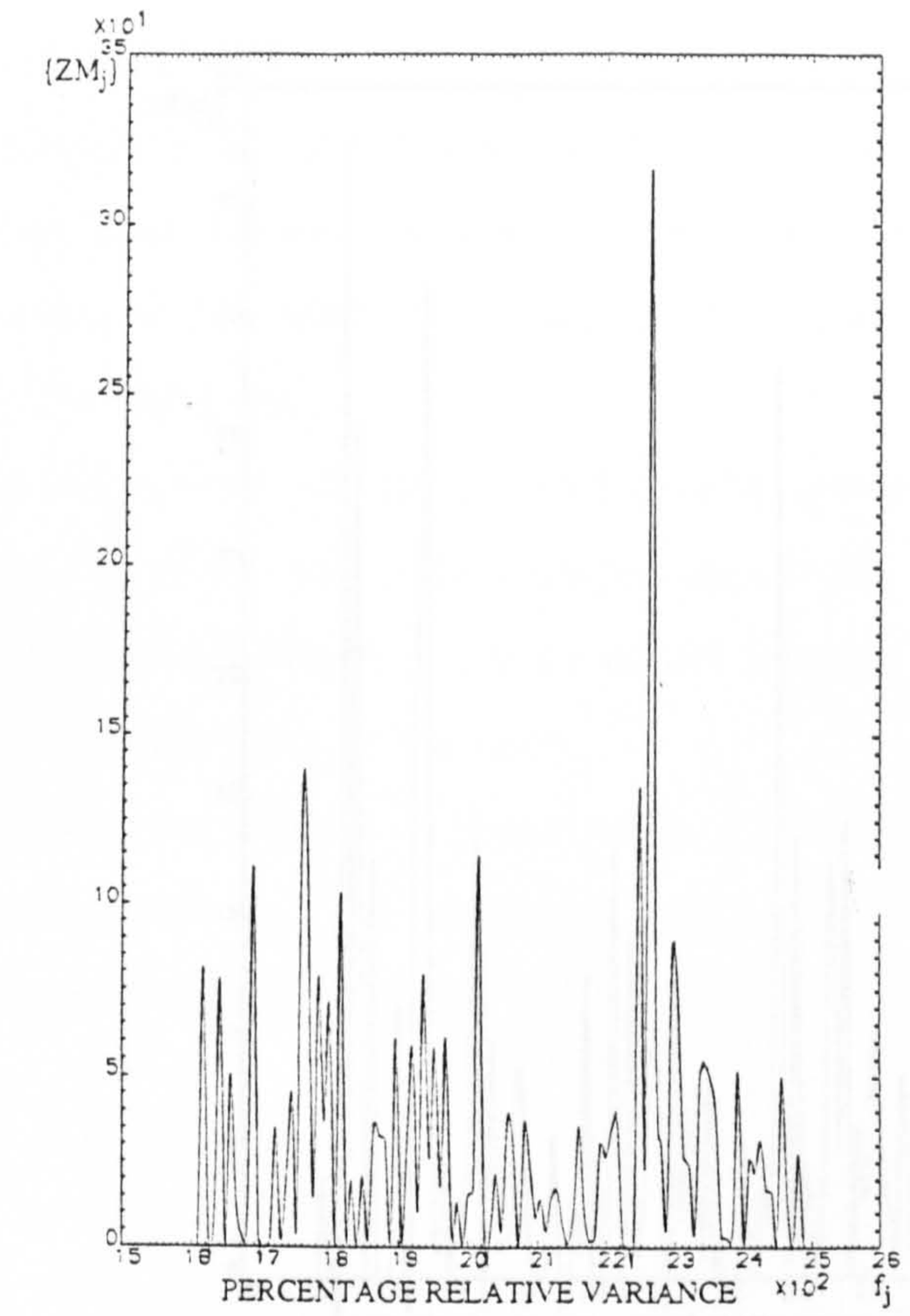


Fig. 3.7 (a)

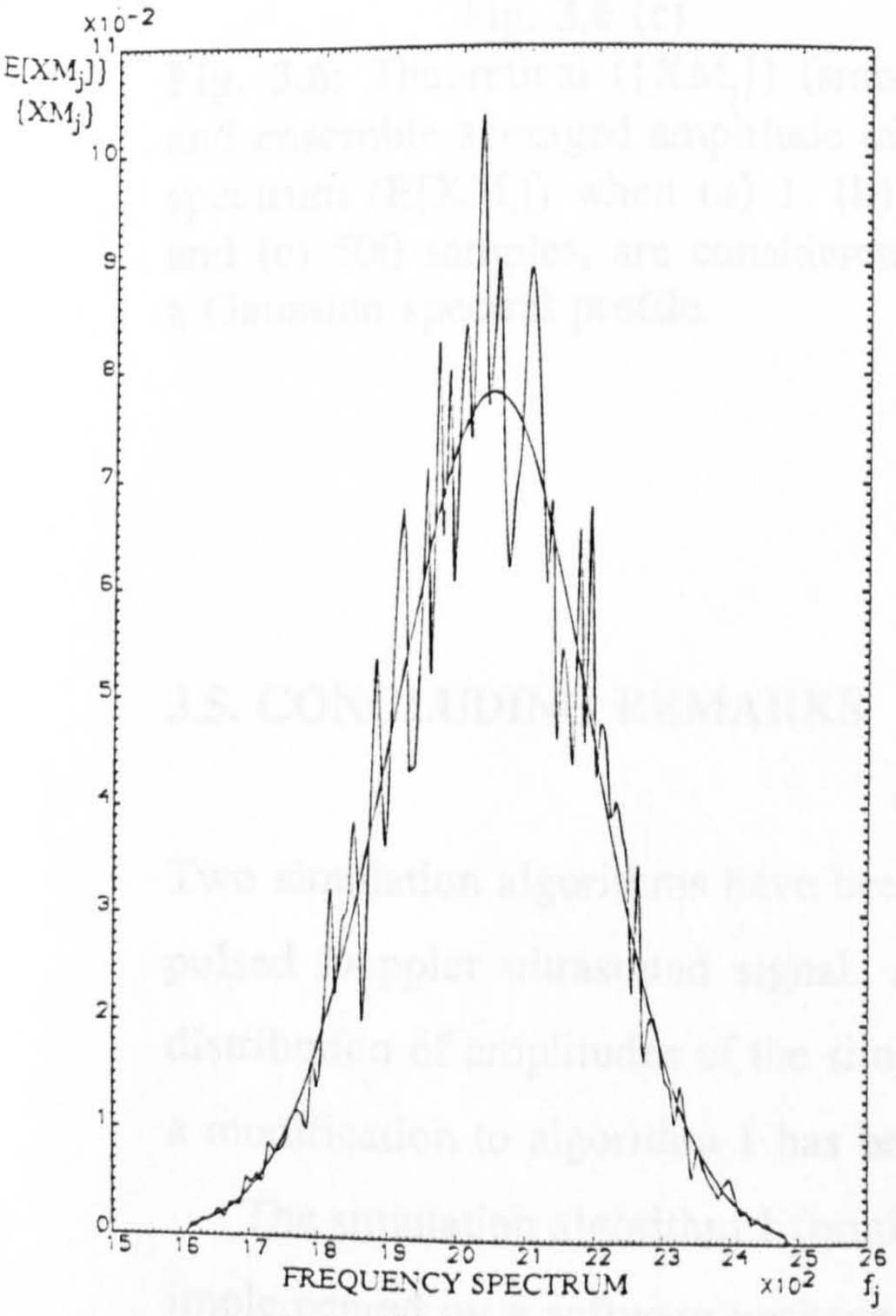


Fig. 3.6 (b)

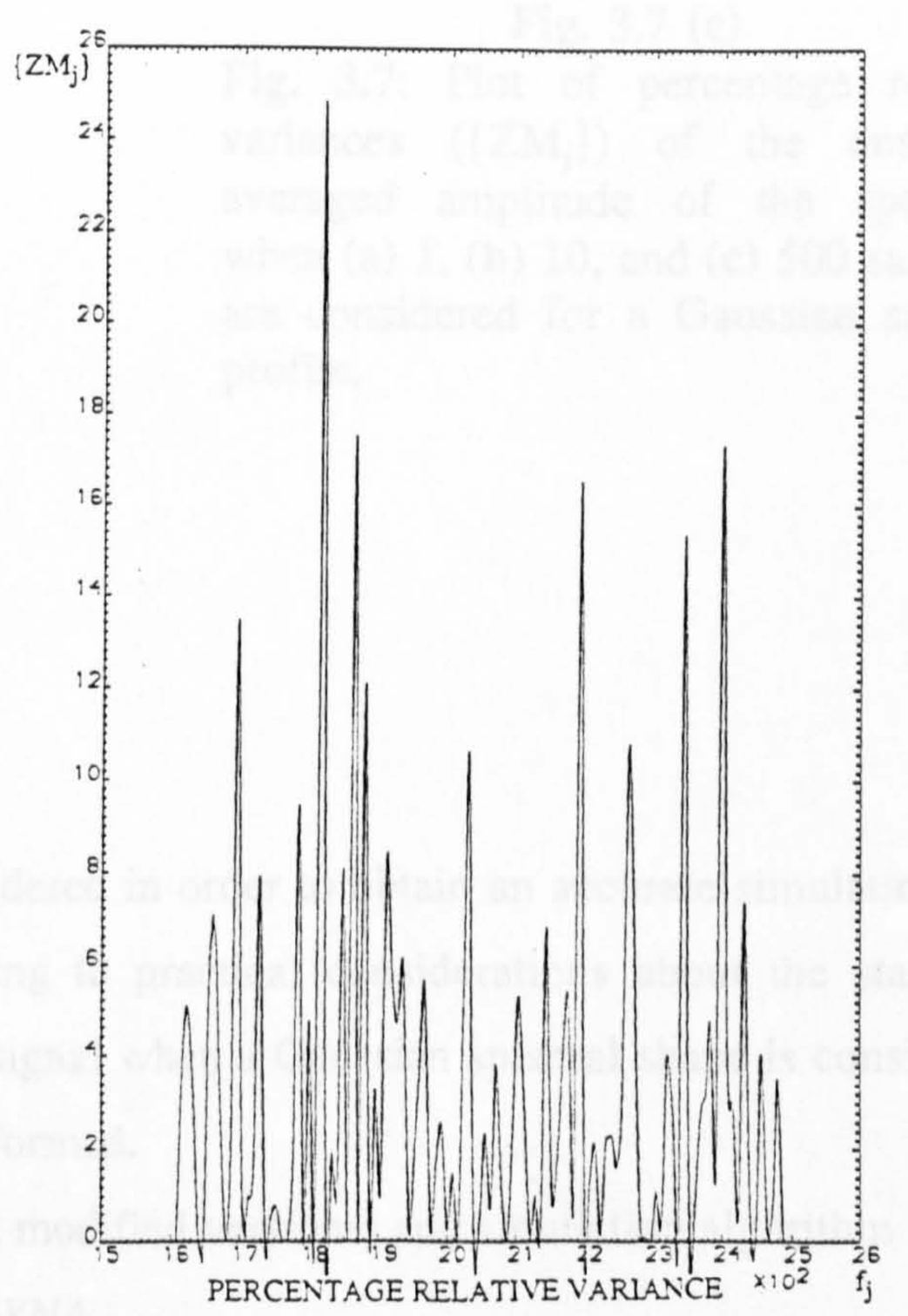


Fig. 3.7 (b)

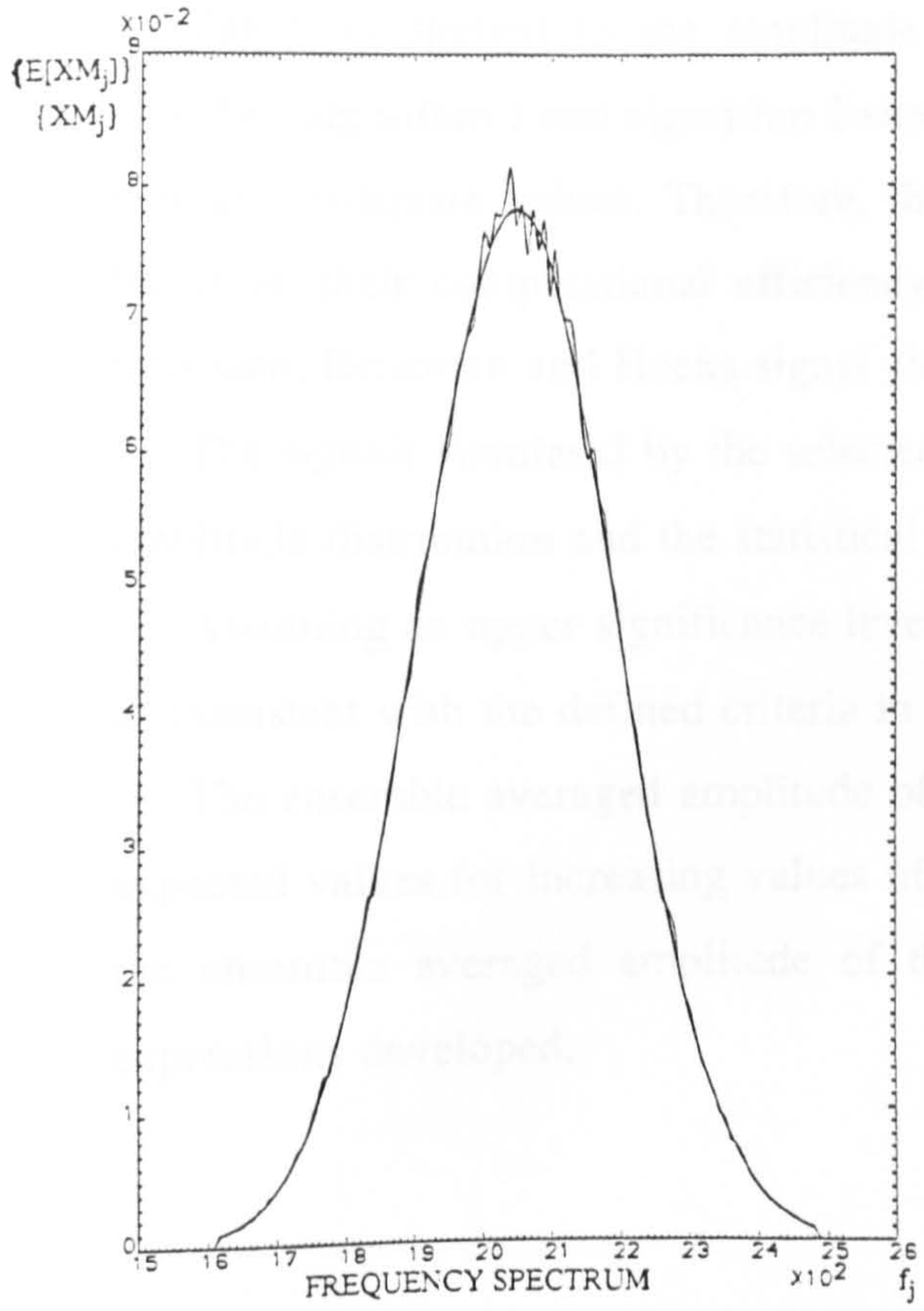


Fig. 3.6 (c)

Fig. 3.6: Theoretical ($\{XM_j\}$) (smooth) and ensemble averaged amplitude of the spectrum ($E[XM_j]$) when **(a)** 1, **(b)** 10, and **(c)** 500 samples, are considered for a Gaussian spectral profile.

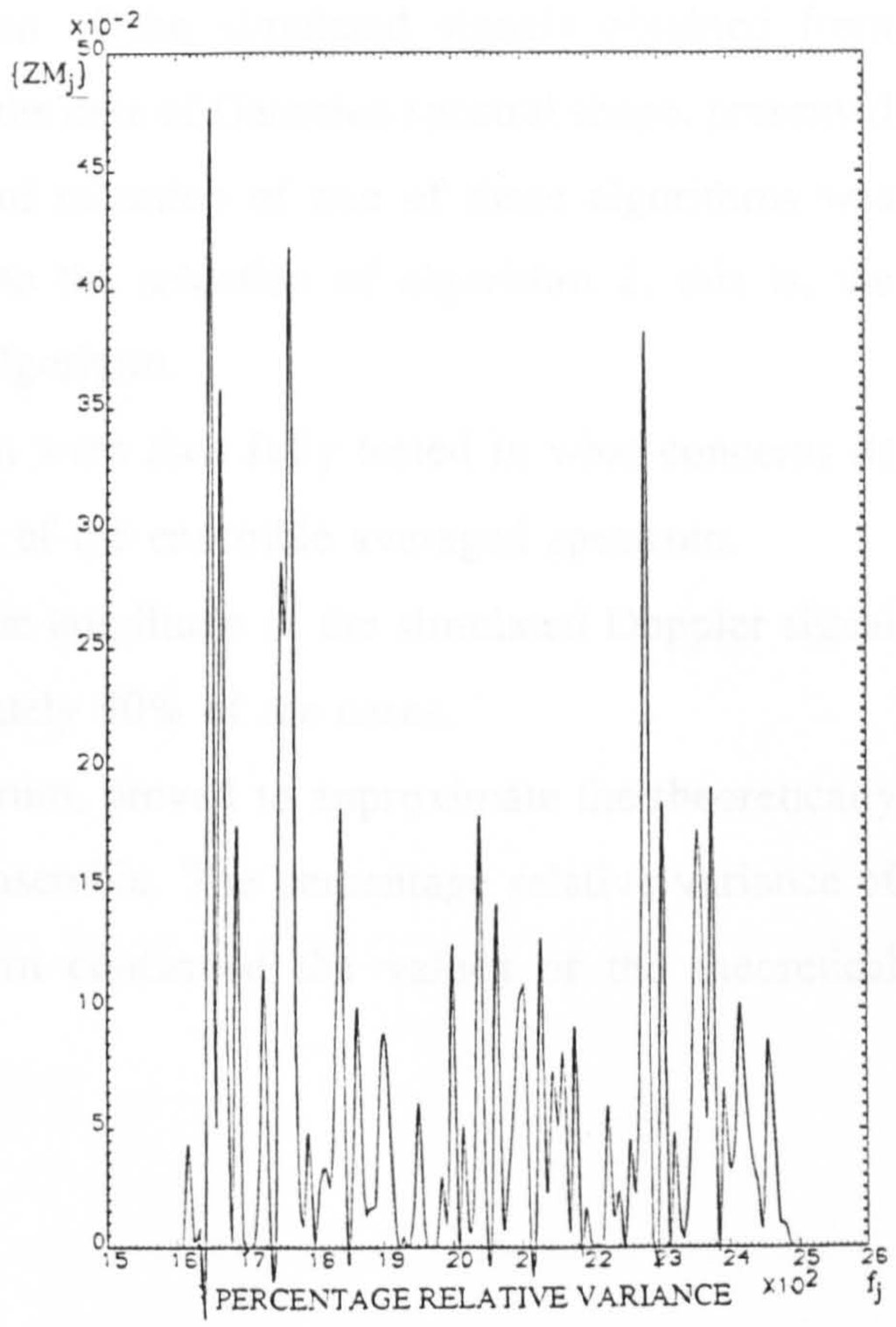


Fig. 3.7 (c)

Fig. 3.7: Plot of percentage relative variances ($\{ZM_j\}$) of the ensemble averaged amplitude of the spectrum when **(a)** 1, **(b)** 10, and **(c)** 500 samples, are considered for a Gaussian spectral profile.

3.5. CONCLUDING REMARKS

Two simulation algorithms have been considered in order to obtain an accurate simulation of a pulsed Doppler ultrasound signal. According to practical considerations about the statistical distribution of amplitudes of the simulated signal when a Gaussian spectral shape is considered, a modification to algorithm 1 has been performed.

The simulation algorithm 1 (original and modified versions) and simulation algorithm 2 were implemented on a software package, *DOPSYNA*.

The tests applied to the amplitude distribution of the simulated signals obtained from modified algorithm 1 and algorithm 2 simulators in the case of Gaussian spectral shape, presented similar Chi-square values. Therefore, the criteria of selection of one of these algorithms was based on their computational efficiency, leading to the selection of algorithm 2, this is, the Leuewen, Reneman and Hoeks signal simulation algorithm.

The signals simulated by the selected algorithm were then fully tested in what concerns its amplitude distribution and the statistical properties of the ensemble averaged spectrum.

Assuming an upper significance level of 5%, the amplitude of the simulated Doppler signal is consistent with the defined criteria in approximately 90% of the cases.

The ensemble averaged amplitude of the spectrum, proved to approximate the theoretically expected values for increasing values of size of ensemble. The percentage relative variance of the ensemble averaged amplitude of the spectrum confirmed the values of the theoretical expressions developed.

CHAPTER 4

CHAPTER 4

Spectral Statistics

4.1. INTRODUCTION

The previous chapter was devoted to the development of a pulsed Doppler ultrasound signal simulation. The amplitudes of these signal simulations were tested either in time and frequency domains to assure the validity of the simulation, and, to enable a better knowledge of the signal's characteristics.

In this chapter the statistical behaviour of the spectrum obtained by averaging an ensemble of Doppler spectra is studied. Estimators of the spectral mean frequency and half bandwidth parameters are developed. Analysis of the performance of the mean frequency and half bandwidth estimators with increasing sizes of ensemble is evaluated. Study of the effects of shifting the mean frequency parameter of the simulated signal on the bias of the spectral mean frequency estimator, for different sizes of ensemble, is also included.

4.2. SPECTRAL ANALYSIS LIMITATIONS

The power spectrum of the Doppler signal is conventionally obtained by applying a Fast Fourier Transform to short data segments, assuming that the Doppler signal is approximately stationary for the duration of each segment.

However, as a consequence of the random nature of the Doppler signal, the above procedure produces spectra which are themselves random, yielding only estimates of the true spectrum. Accuracy of spectral estimation may be improved by performing spectral ensemble averaging to reduce the variance of the estimate [30].

Earlier work on this field developed by Oppenheim and Schafer [77] showed that for a Gaussian random process the variance of an estimated spectrum is approximately proportional to the square of the true power spectrum for large N (the number of points in the data sequence).

The same authors proposed a method of averaging K spectrum of consecutive data segments, the so called Bartlett's method. However even using a large number of points in the Fourier transform the improvement on variance would still be of 1/K.

As Fourier transforms have to be applied to stationary data, and assuming that the Doppler signal may be considered as stationary over periods around 10-20ms of the cardiac cycle [30], the maximum frequency resolution that can be achieved with the Fourier based techniques is of the order of 100Hz.

A better way of effectively reducing the spectral variance maintaining the frequency resolution is to average the transforms of stationary portions of similar points of the cardiac cycle over several cardiac cycles [30]. This is the procedure considered on this study.

4.3. STATISTICAL CHARACTERIZATION OF SPECTRA

4.3.1. Spectral Distribution

In order to characterize the ensemble averaged spectrum obtained by simulation algorithm 2 (see Chapter 3), the expressions of the theoretical statistical distribution of the PSD of the signal are developed.

As has been stated in Chapter 3 (section 3.3.2.) the Fourier transform of the simulated signal has two components

$$XR_j = \sqrt{S_x[f_j]}\ .Y_R \qquad XI_j = \sqrt{S_x[f_j]}\ .Y_I \tag{4.1}$$

where the real (XR_j) and the imaginary (XI_j) components are computed by the product of a given PSD $S_x[f_j]$ by the statistically independent and zero mean Gaussian random variables Y_R and Y_I.

From now on the index j will be used to represent the j^{th} frequency of a spectrum with M points (j=1,...,M).

The magnitude of the amplitude of the spectrum, at frequency f_j is

$$X_j = \sqrt{XR_j^2 + XI_j^2} = \sqrt{S_x[f_j]} \cdot \sqrt{Y_R^2 + Y_I^2} \tag{4.2}$$

and has been proved to follow a Rayleigh distribution with mean μ_j and variance σ_j^2 (see expressions (3.31)) given by

$$\mu_j = \sqrt{S_x[f_j]} \cdot \sqrt{\frac{\pi}{2}} \qquad \sigma_j^2 = S_x[f_j] \cdot \frac{4-\pi}{2} \tag{4.3}$$

Now our interest is on the analysis of the power spectral density of the signal, so the squared magnitude of the Fourier transform of the simulated signal has to be computed, this is

$$SX_j = X_j^2 = S_x[f_j] \cdot [Y_R^2 + Y_I^2] \tag{4.4}$$

Regarding the computations necessary to the deduction of (4.3) (see section 3.3.2.), the variable $[Y_R^2 + Y_I^2]$ can be said to follow a Chi-square distribution with 2 degrees-of-freedom. Thus,

$$\begin{aligned} E[SX_j] &= S_x[f_j] \cdot E[\chi_2^2] \\ Var[SX_j] &= S_x^2[f_j] \cdot Var[\chi_2^2] \end{aligned} \tag{4.5}$$

were E[•] and Var[•] represent respectively the mean and variance of the random variable inside brackets, in this case a Chi-square variable. For ν degrees-of-freedom

$$\begin{aligned} E[\chi_\nu^2] &= \nu \\ Var[\chi_\nu^2] &= 2\nu \end{aligned} \tag{4.6}$$

[76], therefore expressions (4.5) may be rewritten in the form,

$$E[SX_j] = 2.S_x[f_j]$$
$$Var[SX_j] = 4.S_x^2[f_j] \qquad (4.7)$$

Thus, we will expect the estimated spectrum to present a magnitude of double the one of the theoretical spectral profile, and a variance four times greater than the magnitude of the theoretical curve.

4.3.2. Spectral Parameters Estimators

To assess performance analysis of the mean frequency and half bandwidth spectral estimators, statistical estimators for these parameters have to be defined. In this context, let the PSD of a simulated Doppler signal (stationary segment) be represented by SX_{ij} , where i represents the sample under consideration (i=1,...,L) and j the index of the j^{th} frequency (j=1,...,M for M = f_s.T/2, where f_s is the sampling frequency and T the data sequence duration).

For each sample of the ensemble, the mean frequency (f_{i0}) and the half bandwidth (σ_{i0}) are computed as follows

$$f_{i0} = \sum_{j=1}^{M} f_j \cdot \frac{SX_{ij}}{\sum_{j=1}^{M} SX_{ij}} \qquad i=1,\cdots,L \qquad (4.8)$$

$$\sigma_{i0} = \sqrt{\sum_{j=1}^{M} (f_j - f_{i0})^2 \cdot \frac{SX_{ij}}{\sum_{j=1}^{M} SX_{ij}}} \qquad i=1,\cdots,L \qquad (4.9)$$

The average value of f_{i0} over all i and identically the average value of σ_{i0} over all samples of the ensemble are the ensemble mean frequency (f_{00}) and ensemble half bandwidth (σ_{00}) respectively. They are

$$f_{00} = \frac{1}{L}\sum_{i=1}^{L} f_{i0} \qquad (4.10)$$

$$\sigma_{00} = \frac{1}{L}\sum_{i=1}^{L} \sigma_{i0} \qquad (4.11)$$

For each sample, the standard deviation observed for the estimates of mean frequency (f_{i0}) and half bandwidth (σ_{i0}) are

$$sd(f_{i0})_i = \sqrt{\frac{1}{L-1}\sum_{i=1}^{L}(f_{i0} - f_{00})^2} \qquad i = 1,...,L \tag{4.12}$$

$$sd(\sigma_{i0})_i = \sqrt{\frac{1}{L-1}\sum_{i=1}^{L}(\sigma_{i0} - \sigma_{00})^2} \qquad i = 1,...,L \tag{4.13}$$

The ensemble averaged spectrum represented by SX_{0j} is the average of all i=1,...,L spectra computed at each frequency bin j. This is,

$$SX_{0j} = \frac{1}{L}\sum_{i=1}^{L} SX_{ij} \qquad j = 1,..., M \tag{4.14}$$

and the standard deviation observed for each bin of frequency j is

$$sd(SX_{ij})_j = \sqrt{\frac{1}{L-1}\sum_{i=1}^{L}(SX_{ij} - SX_{0j})^2} \qquad j = 1,...,M \tag{4.15}$$

The mean frequency and the half bandwidth of the ensemble averaged spectra, $SX_{0j,}$ are also computed

$$\hat{f}_{00} = \sum_{j=1}^{M} f_j \cdot \frac{SX_{0j}}{\sum_{j=1}^{M} SX_{0j}} \tag{4.16}$$

$$\hat{\sigma}_{00} = \sqrt{\sum_{j=1}^{M} (f_j - \hat{f}_{00})^2 \cdot \frac{SX_{0j}}{\sum_{j=1}^{M} SX_{0j}}} \tag{4.17}$$

The above defined statistics are illustrated in Fig. 4.1.

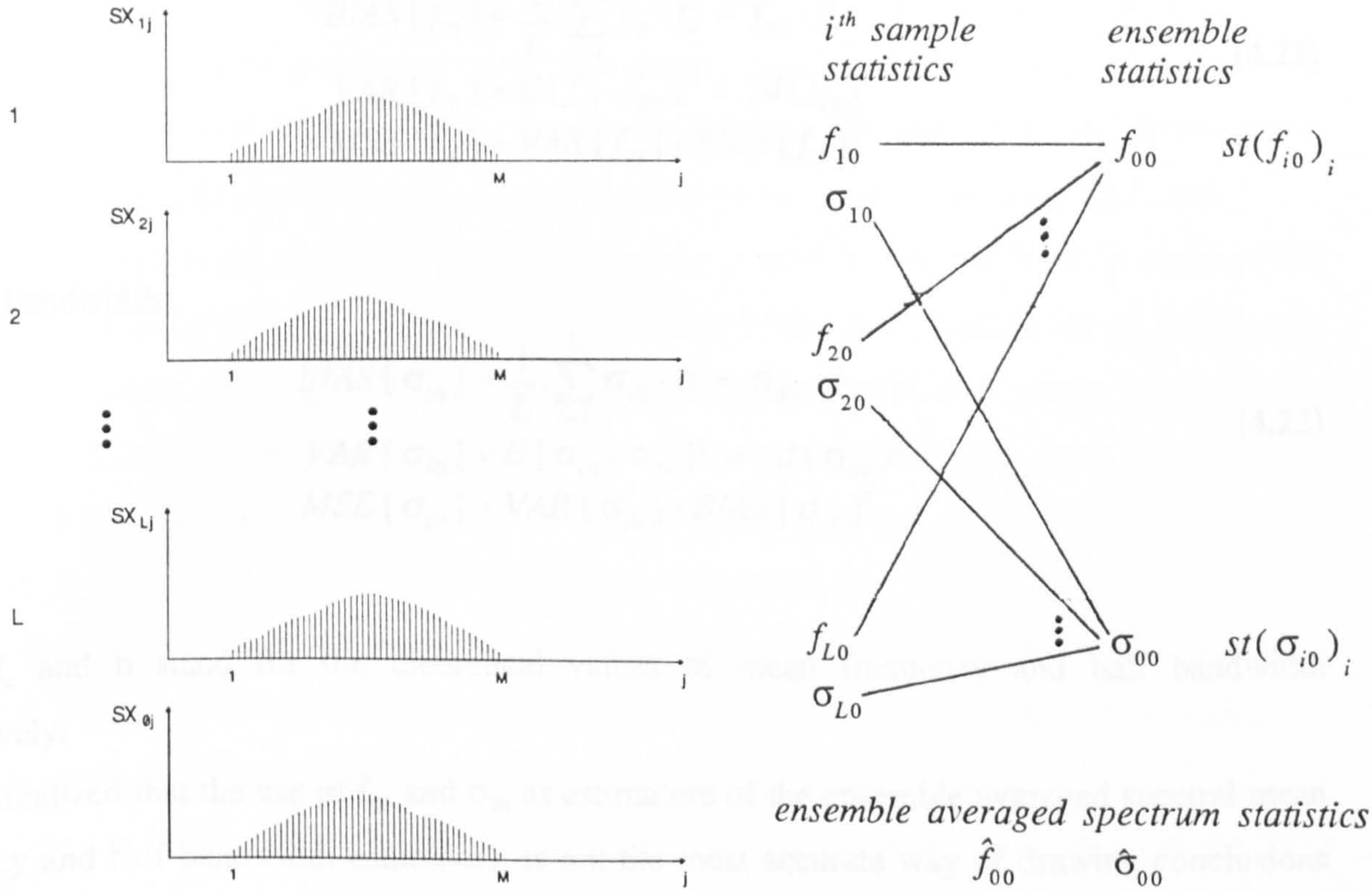


Fig. 4.1: Ensemble of Doppler spectra and the statistics employed.

After defining this set of estimators some ways of measuring the difference between the estimated and the exact values of the spectral parameters mean frequency and half bandwidth must be defined.

So, if we consider the definitions of bias, variance and mean squared error of an estimate $\hat{S}$ of a spectrum S(f) as given by,

$$BIAS[\,\hat{S}(f)\,] = E\,[\,\hat{S}(f)\,] - S(f) \tag{4.18}$$

$$VAR\,[\,\hat{S}(f)\,] = E[\,\hat{S}(f) - E\,[\,\hat{S}(f)\,]\,]^2 \tag{4.19}$$

$$\begin{aligned} MSE\,[\,\hat{S}(f)\,] &= E\,[\,\hat{S}(f) - S(f)\,]^2 \\ &= VAR\,[\,\hat{S}(f)\,] + BIAS^2\,[\,\hat{S}(f)\,] \end{aligned} \tag{4.20}$$

[12], we may think of computing the mean squared error of both the estimated mean frequency and half bandwidth as follows:

- mean frequency:

$$\begin{aligned} BIAS\,[f_{i0}] &= \frac{1}{L}\cdot\sum_{i=1}^{L} f_{i0} - f_c \;=\; f_{00} - f_c \\ VAR\,[f_{i0}] &= E\,[f_{i0} - f_{00}]^2 \;=\; sd\,(f_{i0})^2 \\ MSE\,[f_{i0}] &= VAR\,[f_{i0}] + BIAS\,[f_{i0}]^2 \end{aligned} \tag{4.21}$$

- half bandwidth:

$$\begin{aligned} BIAS\,[\sigma_{i0}] &= \frac{1}{L}\cdot\sum_{i=1}^{L} \sigma_{i0} - b \;=\; \sigma_{00} - b \\ VAR\,[\sigma_{i0}] &= E\,[\sigma_{i0} - \sigma_{00}]^2 \;=\; sd\,(\sigma_{00})^2 \\ MSE\,[\sigma_{i0}] &= VAR\,[\sigma_{i0}] + BIAS\,[\sigma_{i0}]^2 \end{aligned} \tag{4.22}$$

where f_c and b stand for the theoretical values of mean frequency and half bandwidth respectively.

It is realized that the use of f_{00} and σ_{00} as estimators of the ensemble averaged spectral mean frequency and half bandwidth estimators, is not the most accurate way of drawing conclusions about these population parameters. However the use of f_{00} and σ_{00} enabled the establishment of a clear relationship between the size of the ensemble and the parameters' behaviour (relationship not evident if $\hat{f}_{00}$ and $\hat{\sigma}_{00}$ were employed).

4.4. SOFTWARE IMPLEMENTATION

The generation of an ensemble (of variable size) of Doppler signal simulations, using algorithm 2 signal simulator (see Chapter 3) was software implemented in a program named *STENS*.

The software program *STENS*, was written in FORTRAN 77 [69] and made use of routines of the NAG software package [70]. It was implemented on a VAX 8600 with an operating system VMS V5.3.

The user is led to input the necessary parameters by following a menu. Two choices of signal spectral shapes are presented. If the parabolic plug profile is selected, the user is asked to input the maximum frequency of the spectrum and the profile constant (n=2). In case of a Gaussian

spectral shape, the mean frequency of the signal, the half bandwidth and the threshold spectrum value, are required.

For both choices of spectral shape, the user is asked to enter the sampling frequency and the data sequence duration. The software program computes the product $f_s.T$ and automatically adjusts the data sequence duration so that the number of points of the signal (N) becomes a power of two. If by mistake N is greater than the maximum value allowed to its correspondent variable, the program warns the user and enables him\her to reenter the values of f_s and T.

The user specifies the number of simulated Doppler signals that he\she is interested in. With this number, the software program computes the smallest power of two that is bigger than L (this is, the size of ensemble), say LC. Then, a 'long' signal with a number of points equal to NC= fs.T.LC = N.LC is created. Thus, the frequency resolution of this signal is at least L times smaller than the frequency resolution of the signals which are used to estimate the PSD (this is, f_s/N), and, at the same time, the effects of *window broadening* are reduced to negligible levels.

Each sample of the ensemble of Doppler signal simulations is stored in a data file. At the same time, a file containing the ensemble characteristics (size of ensemble, sampling frequency, mean frequency, half bandwidth, and data sequence length) is also created.

To study the influence of the size of ensemble on the estimated spectral variance, the PSD of the pulsed Doppler ultrasound signal simulations has been estimated using a Fourier transform.

This was implemented in a software package named *ANENS*, also written in FORTRAN 77 using the same operating system.

ANENS allows the user to input the Doppler signal simulations by selecting the code name of the data files (Eg. those created by *STENS*). The size of the ensemble also need to be input. For the purpose of validating the signal under analysis, the software program instructs the user to enter the signal's specifications, and checks their accordance with the data files.

At this stage of the work *ANENS* enabled the application of either Boxcar or Hanning window to the Doppler signal simulation, followed by Fourier transformation.

The signal simulations' estimated PSD - SX_{ij} (i=1,...,L ; j=1,...,M) - and the ensemble averaged spectrum - SX_{0j} (j=1,...,M) - are used by *ANENS* software package's statistical routine to compute the parameters presented in Table 4.1, as well as the measurements of error described on (4.21) and (4.22).

ensemble mean freq.	f_{00}	(4.10)
averaged spec.mean freq.	$\hat{f}_{00}$	(4.16)
st. deviation of f_{i0}	$sd(f_{i0})_i$	(4.12)
ensemble half bandwidth	σ_{00}	(4.11)
averaged spec.half band.	$\hat{\sigma}_{00}$	(4.17)
st. deviation of σ_{i0}	$sd(\sigma_{i0})_i$	(4.13)

Table 4.1: Statistical parameters computed by *ANENS* and respective equation numbers.

A graphics option using the GINOGRAF software package [71] is included in *ANENS*. A normalized theoretical spectral curve is plotted against a normalized ensemble averaged spectrum to allow comparison of both spectra. Both normalizations are performed with respect to the Euclidean norm of the respective vectors.

Subject to the choice of the user, the standard deviation of the ensemble averaged spectrum (see (4.15)) may or may not be plotted. If plotted, it is represented by two curves (normalized with respect to the 2-norm of the vector representing the ensemble averaged spectrum values), surrounding the ensemble averaged spectrum, as result of the operation

$$SX'_{0j} \pm sd\ (\ SX'_{0j}\ ) \tag{4.23}$$

where SX'_{0j} stands for the normalized ensemble averaged spectrum.

In summary, ensembles of Doppler signal simulations were created by the software package *STENS* and statistical parameters of their spectrum were computed by *ANENS*.

4.5. RESULTS

As a common feature, all simulations of the Doppler signal presented very low amplitudes. As an example, a signal simulation with mean frequency 1kHz and half bandwidth of 100Hz had amplitudes of the order of 10^{-04} - 10^{-05} which were 100 times smaller than the amplitudes of a signal simulation with same characteristics but created for only one sample (L=1).

This discrepancy resulted from the generation procedure of a 'long' signal. Recalling what has been said in section 3.2.2 (last chapter) algorithm 2 creates first the spectrum of the Doppler signal and then by application of an inverse Fourier transform the time-domain simulated signal is obtained. As by definition the inverse Fourier transform divided the time-domain signal by the

number of points that constitute the signal, which in the case of a 'long' signal is NC, each 10ms segment of the Doppler signal simulation (with N points) was scaled by a factor N/NC.

A maximum of 150 samples of each Doppler simulation case were created, and fifteen different cases were studied. The term 'simulation case' stands for a certain specification of mean frequency and half bandwidth. The correspondent sampling frequency is subject to the constraints stated earlier (chapter 3, section 3.2.4). The data sequence duration is always made equal to 10ms.

The allowed mean frequencies varied from 500Hz up to 8kHz in octaves. For each mean frequency there were three values of half bandwidth : 5% , 10% and 20%.

Each signal simulation case was analyzed using *ANENS*.

The joint graphical output of theoretical and ensemble averaged PSDs showed that the spectrum obtained presented mean and variance values following the mean and variance theoretical values (expressed in (4.7)) of a Chi-square distribution with two degrees of freedom. The theoretical spectral profile was multiplied by a factor of two (in accordance with (4.7)), so that the ensemble averaged and the theoretical spectra were appropriately scaled.

The upper curve of the standard deviation of the ensemble averaged spectrum, SX'_{0j} (see (4.23)), was found to be approximately double the one of the scaled theoretical spectrum, which again is in accordance with the relationship (4.7).

These features were evident in all signal simulation cases. As an example some *ANENS* graphical outputs are presented.

Fig. 4.2 (a) and (b) show that for a mean frequency of 1kHz and a half bandwidth of 100Hz only seven frequency bins constitute the standard deviation on the frequency range of the truncated Gaussian curve.

In the case of a simulated signal with 500Hz as mean frequency, the number of points inside the range $[f_{min} , f_{max}]$ is even smaller; for a half bandwidth of 5% it was just three points.

A spectral estimate build up from such a lower number of frequency bins is not recommended, specially if statistical conclusions have to be drawn.

On the other hand it was not possible to increase this number of frequency bins since they were dependent on the threshold value established (see Chapter 3) and a reduction of the threshold value would result in aliasing problems in the spectrum of the simulated signals with $b=0.2f_c$.

Therefore to assure the overall conditions of the Doppler signal simulation we considered only those with 9 or more frequency bins inside the range [f_{min} , f_{max}].

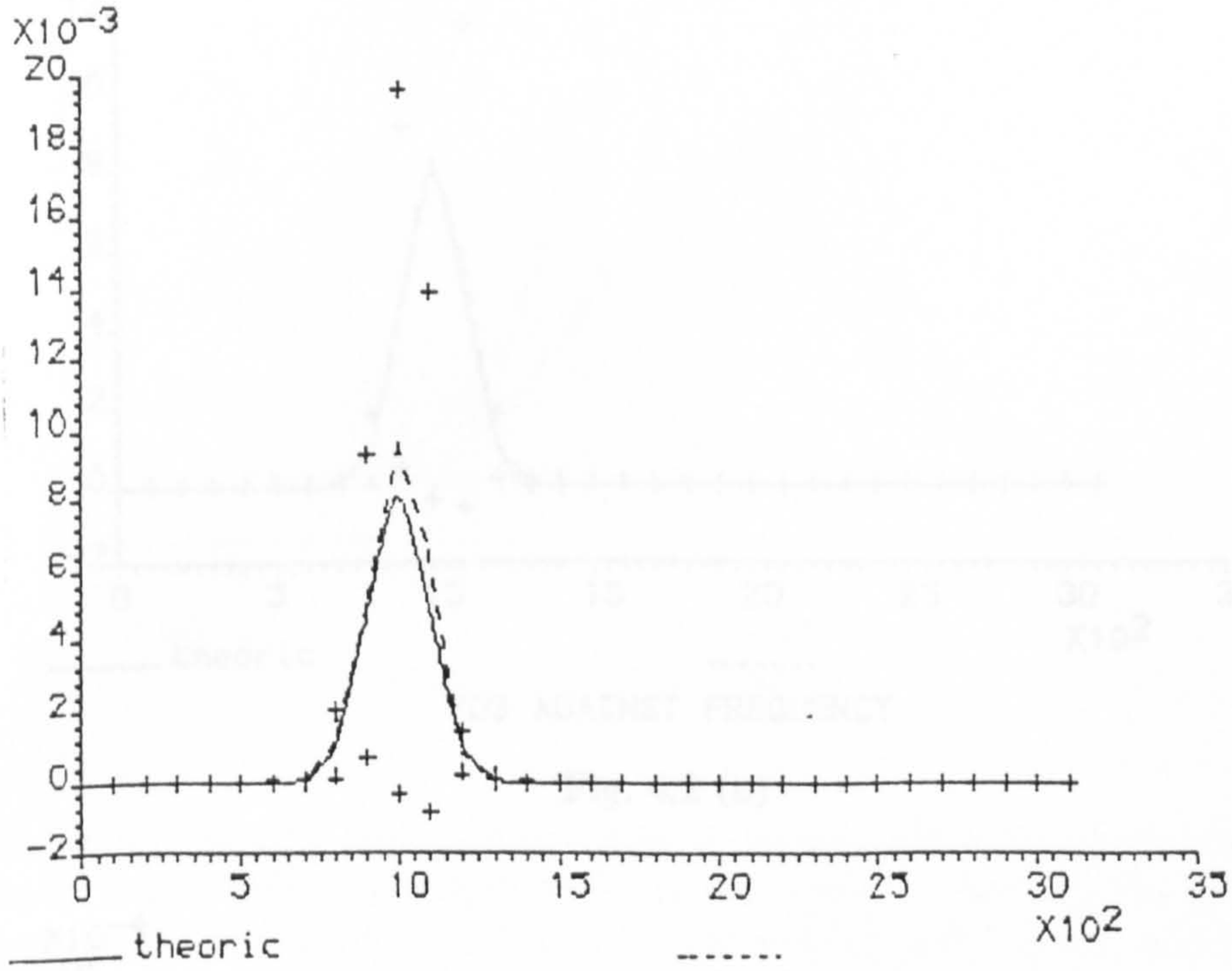


Fig. 4.2 (a)

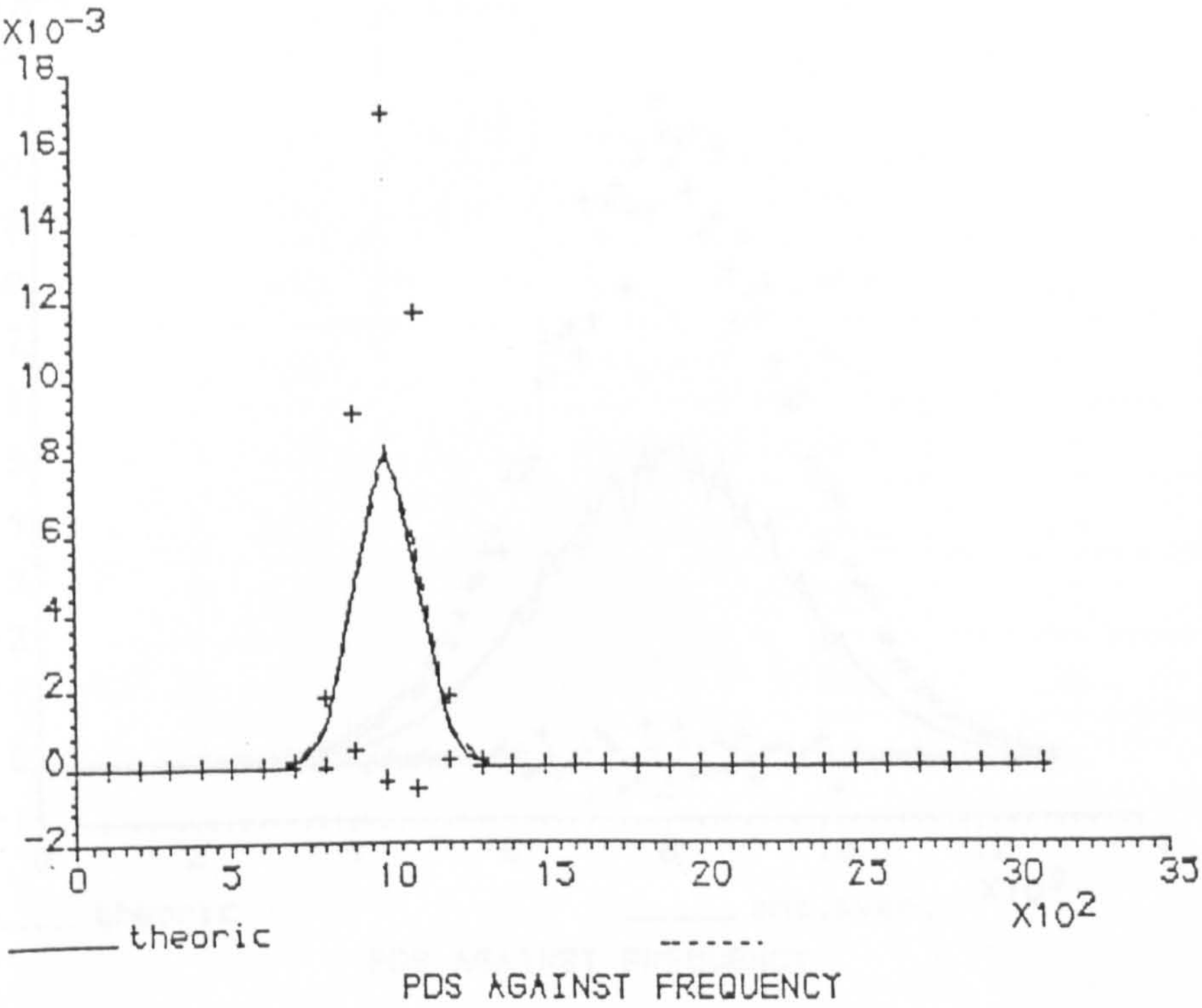


Fig. 4.2 (b)

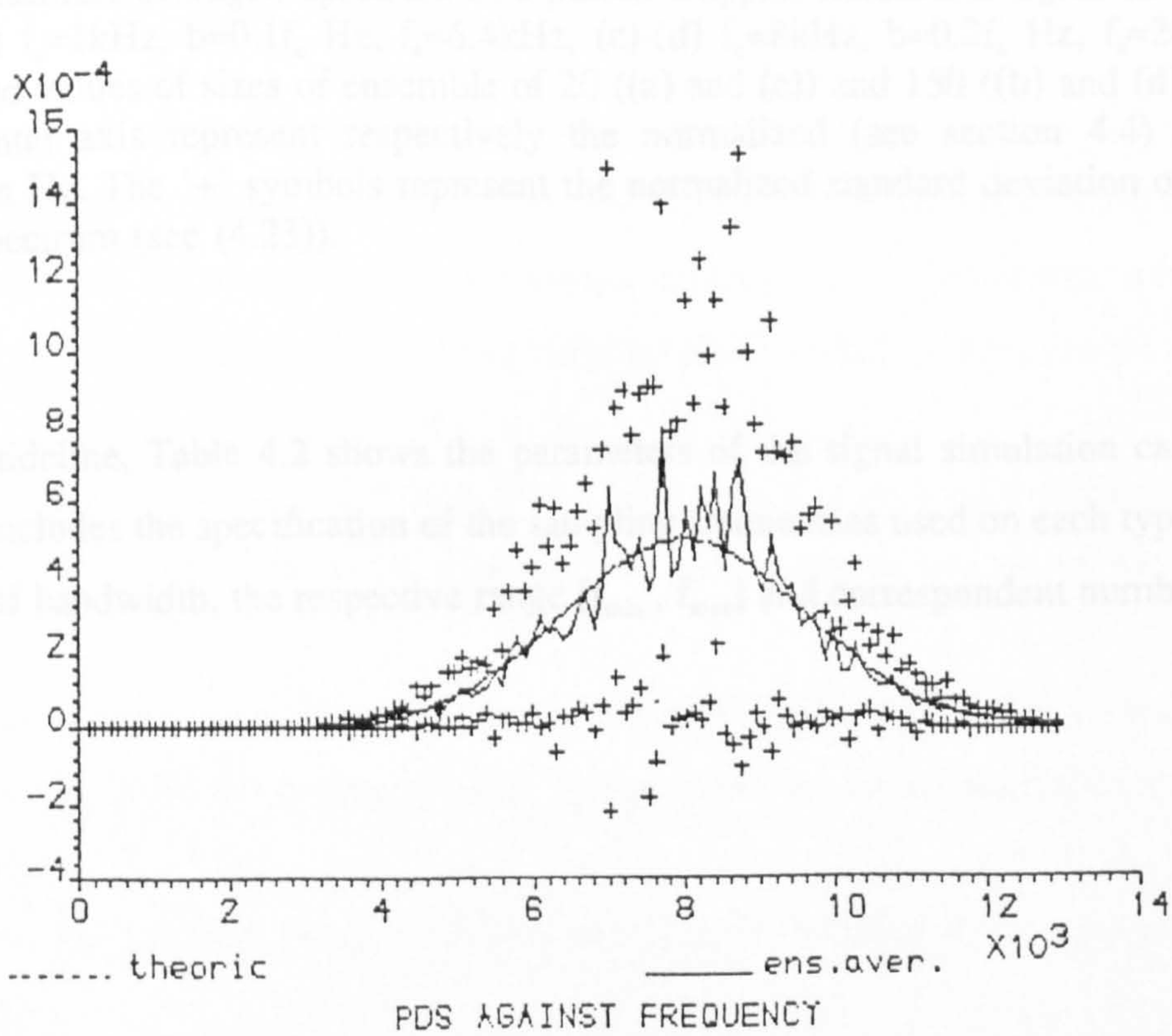


Fig. 4.2 (c)

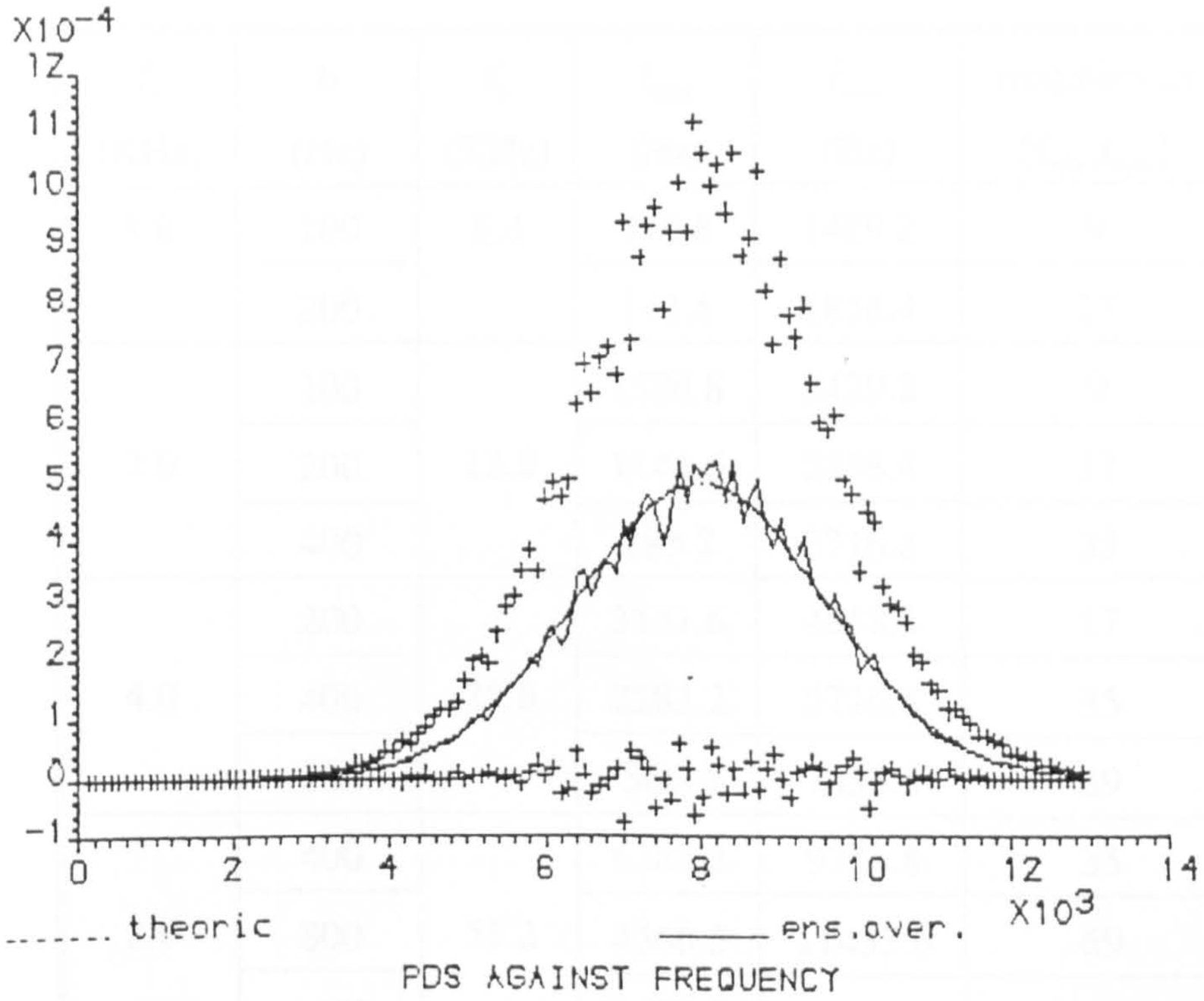


Fig. 4.2 (d)

Fig. 4.2: Ensemble averaged spectrum of a pulsed Doppler ultrasound signal simulation with (**a**)-(**b**) f_c=1kHz, b=0.1f_c Hz, f_s=6.4kHz, (**c**)-(**d**) f_c=8kHz, b=0.2f_c Hz, f_s=26kHz, all with T=10ms, and values of sizes of ensemble of 20 ((**a**) and (**c**)) and 150 ((**b**) and (**d**)). The vertical and horizontal axis represent respectively the normalized (see section 4.4) PSD, and the frequency in Hz. The '+' symbols represent the normalized standard deviation of the ensemble averaged spectrum (see (4.23)).

As a guideline, Table 4.2 shows the parameters of the signal simulation cases considered. This table includes the specification of the sampling frequencies used on each type of signal, and for each half bandwidth, the respective range [f_{min} , f_{max}] and correspondent number of frequency bins.

f_c (KHz)	b (Hz)	f_s (KHz)	f_{min} (Hz)	f_{max} (Hz)	freq.bins in $[f_{min},f_{max}]$
1.0	100	6.4	570.8	1429.2	9
	200		141.6	1858.4	17
2.0	100	12.8	1570.8	2429.2	9
	200		1141.6	2858.4	17
	400		283.2	3716.8	35
4.0	200	25.6	3141.6	4858.4	17
	400		2283.2	5716.8	35
	800		566.5	7433.6	69
8.0	400	51.2	6283.2	9716.8	35
	800		4566.5	11433.6	69
	1600		1132.9	14867.1	137

Table 4.2: Characteristics of pulsed Doppler ultrasound signal simulations to be employed.

The values of bias, variance and mean-squared error for both mean frequency and half bandwidth were analyzed for ensemble sizes from 10 to 150 in decade increments.

As an example, the statistical values obtained for a signal simulation with a mean frequency of 8kHz, a half bandwidth of 1.6kHz, sampled at 25.6kHz, and, T=9.8ms, are presented on Table 4.3.

By analysis of the numerical values of the Table 4.3, it is obviously difficult to determine a size of ensemble above which the measurement errors assume values of the same order of magnitude.

At the same time, comparison of the development of these statistics with L seemed to be more reasonable, if the measurement values presented to the user assumed the form of percentage relative values.

Therefore the software package *ANENS* was slightly modified to incorporate new statistical calculations.

size of ensemble	Bias of f_{i0}	Var of f_{i0}	MSE of f_{i0}	Bias of σ_{i0}	Var of σ_{i0}	MSE of σ_{i0}
10	9.7	1492.6	1586.7	-11.1	337.1	460.3
20	1.6	1422.3	1424.9	7.5	633.3	689.6
30	0.2	1344.8	1344.8	4.8	696.4	719.4
40	-2.1	1491.0	1495.4	4.6	613.4	634.6
50	2.5	1717.3	1723.6	3.9	548.4	563.6
60	-1.7	1516.8	1519.7	-5.4	530.3	559.5
70	-1.0	1355.6	1356.6	-6.6	551.1	594.7
80	0.7	1473.7	1474.2	-6.6	530.8	574.4
90	1.0	1532.2	1533.2	-5.1	579.7	605.7
100	0.1	1513.3	1513.4	-4.1	552.7	569.5
150	2.5	1515.8	1522.1	-4.7	593.9	615.9

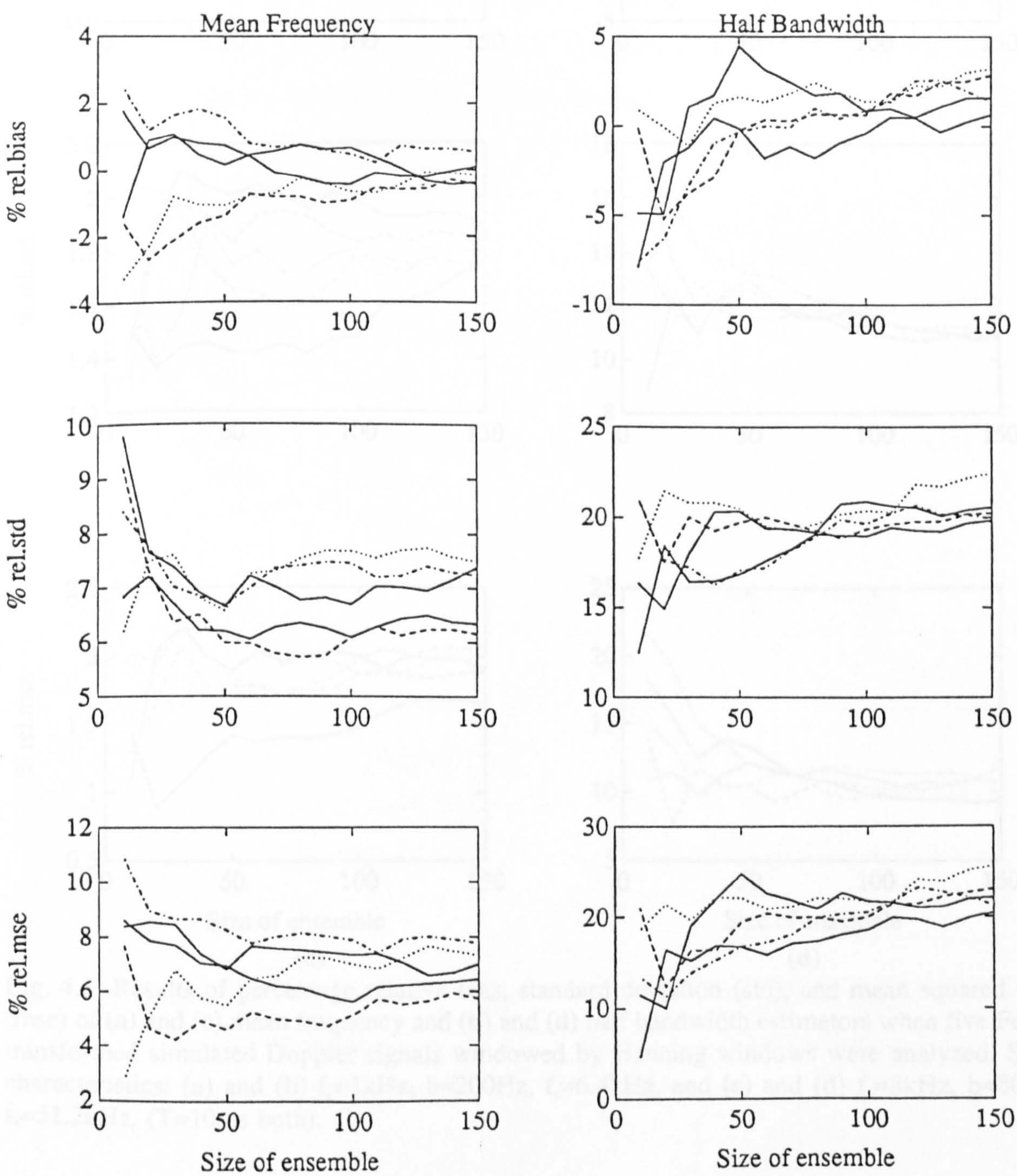
Table 4.3: Statistical characteristics of the spectrum of a simulated pulsed Doppler ultrasound signal windowed by a Boxcar window. The signal has Gaussian spectral profile with f_c=8kHz, b=1.6kHz, f_s=26kHz and T=9.8ms.

Equations (4.21) and (4.22) became measures of percentage relative values, calculated as shown in the following set of equations for the mean frequency estimator. For the half bandwidth estimator, symbols f_{i0} and f_c are substituted by σ_{i0} and b, respectively.

$$\begin{aligned} RBIAS\ [f_{i0}] &= \frac{BIAS\,[f_{i0}]}{f_c}.100 \\ RSTD\ [f_{i0}] &= \frac{\sqrt{VAR\,[f_{i0}]}}{f_c}.100 \\ RMSE\ [f_{i0}] &= RBIAS\,[f_{i0}] + RSTD\,[f_{i0}] \end{aligned} \qquad (4.24)$$

Attention must be drawn to the fact that the calculation of the measurement RMSE[f_{i0}] in (4.24) was achieved by simply adding the percentage relative values obtained for bias an standard deviation of the estimators. The relationship (4.20) is not maintained. However, since this measurement is representative of the joint contribution of bias and standard deviation, it will be used and named in the following text (for simplicity) by percentage relative mean squared error.

For each different signal simulation case and for the different values of L considered, the values of percentage relative bias, standard deviation and mean-squared error for both mean frequency and half bandwidth spectral estimators, were taken and stored in software data files. This procedure was repeated an arbitrary five times for each signal simulation case. The data files were imported into a software program written in MATLAB [78] on a SUN system, which enabled the output of graphs such as those presented on Fig. 4.3.



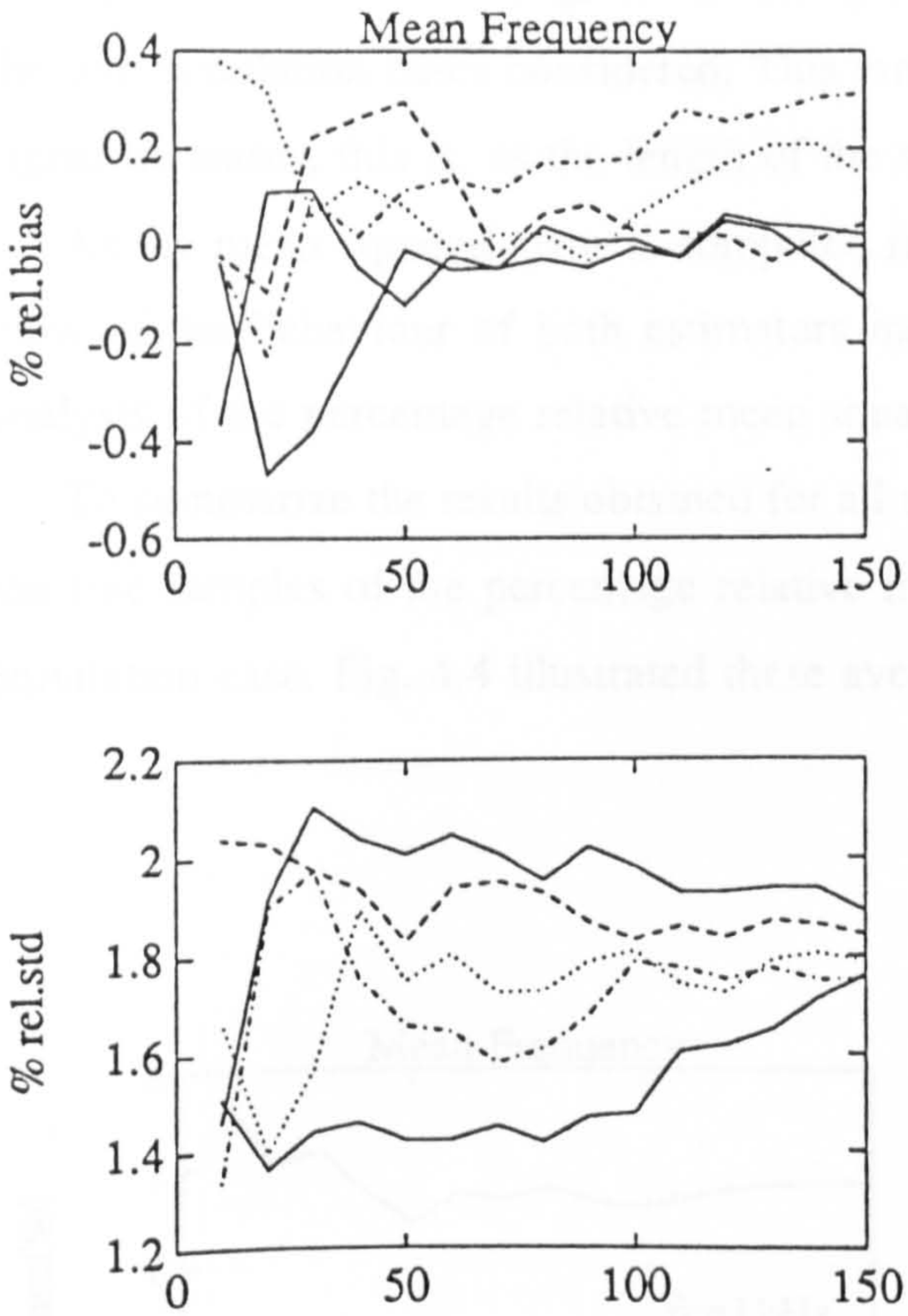
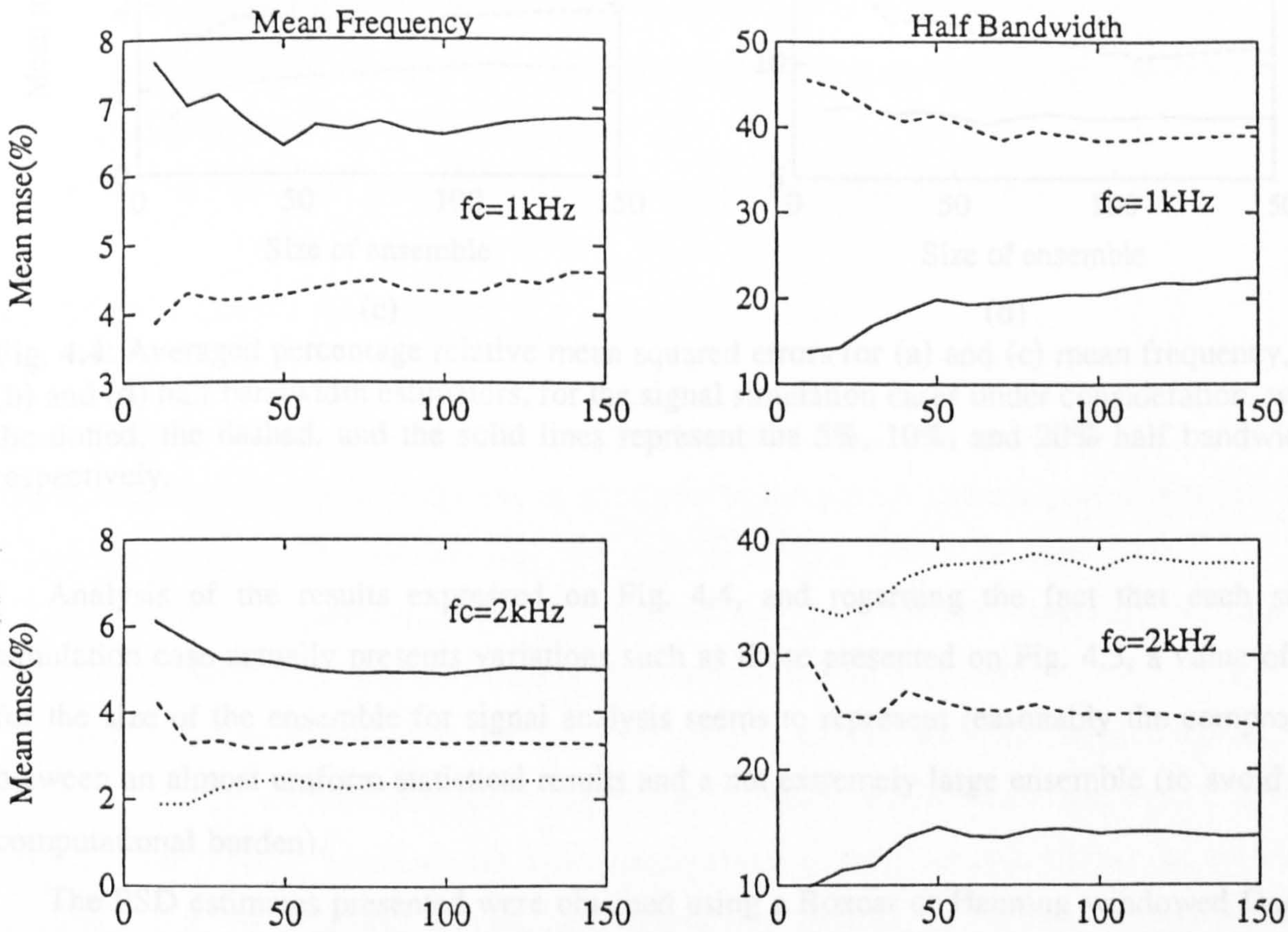


Fig. 4.3: Results of percentage relative bias, standard deviation (std), and mean squared error, (mse) of (**a**) and (**c**) mean frequency and (**b**) and (**d**) half bandwidth estimators when five Fourier transformed simulated Doppler signals windowed by Hanning windows were analyzed. Signal characteristics: (**a**) and (**b**) f_c=1kHz, b=200Hz, f_s=6.4kHz, and (**c**) and (**d**) f_c=8kHz, b=800Hz, f_s=51.2kHz, (T=10ms both).

Fig. 4.3 show a wide range of variability on the values of the statistical measurements among the five simulation cases considered. This range becomes smaller as the mean frequency of the signal increases, this is, as the length of the spectra increases.

As the mean squared error is computed from the bias and the standard deviation, a general view of the behaviour of both estimators mean frequency and half bandwidth is possible by analysis of the percentage relative mean squared error.

To summarize the results obtained for all signal simulation cases considered, an average over the five samples of the percentage relative mean squared errors was computed for each signal simulation case. Fig. 4.4 illustrated these average values.



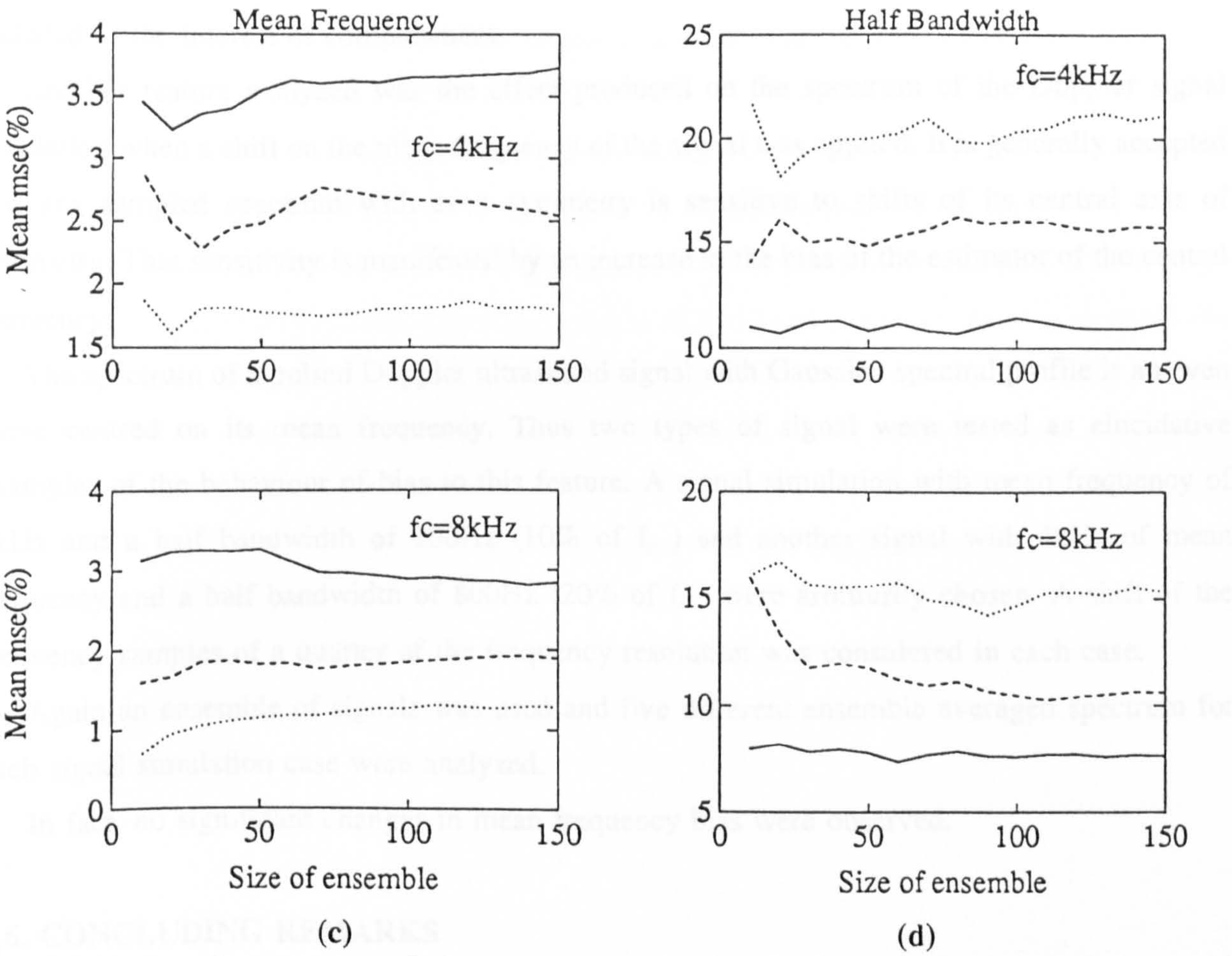


Fig. 4.4: Averaged percentage relative mean squared errors for (**a**) and (**c**) mean frequency, and (**b**) and (**d**) half bandwidth estimators, for the signal simulation cases under consideration, where the dotted, the dashed, and the solid lines represent the 5%, 10%, and 20% half bandwidths, respectively.

Analysis of the results expressed on Fig. 4.4, and regarding the fact that each signal simulation case actually presents variations such as those presented on Fig. 4.3, a value of 150 for the size of the ensemble for signal analysis seems to represent reasonably the compromise between an almost uniform statistical results and a not extremely large ensemble (to avoid high computational burden).

The PSD estimates presented were obtained using a Boxcar or Hanning windowed Doppler signal simulation. It is known that the energy spectrum of a Boxcar window is a *sinc* squared function, which is not integrable over the range $]-\infty,+\infty[$ and therefore the concept of a root mean squared bandwidth (see (3.6) in last chapter) for a Boxcar windowed signal is meaningless. However we obtain finite mean squared bandwidths since the frequency range of the analyzed

spectrum is truncated. Thus, the spectral results obtained with a Boxcar windowed signal are included in the interest of completeness.

Another feature analyzed was the effect produced on the spectrum of the Doppler signal simulation when a shift on the mean frequency of the signal was applied. It is generally accepted that any sampled spectrum with even symmetry is sensitive to shifts of its central axis of symmetry. That sensitivity is manifested by an increase in the bias of the estimator of the central frequency.

The spectrum of a pulsed Doppler ultrasound signal with Gaussian spectral profile is an even curve centred on its mean frequency. Thus two types of signal were tested as elucidative examples of the behaviour of bias to this feature. A signal simulation with mean frequency of 1kHz and a half bandwidth of 100Hz (10% of f_c) and another signal with 4kHz of mean frequency and a half bandwidth of 800Hz (20% of f_c) were arbitrarily chosen. A shift of the frequency samples of a quarter of the frequency resolution was considered in each case.

Again an ensemble of signals was used and five different ensemble averaged spectrum for each signal simulation case were analyzed.

In fact, no significant changes in mean frequency bias were observed.

4.6. CONCLUDING REMARKS

The procedure developed in Chapter 3 to create a simulation of a pulsed Doppler ultrasound signal, was adapted to enable the use of an ensemble averaged Doppler signal simulation in the following stages of this research work, the study of spectral estimation techniques.

A statistical characterization of the squared magnitude of the obtained spectrum was theoretically developed and confirmed by the practical results obtained.

Analysis of the number of frequency bins that effectively constitute the range of the estimated Gaussian spectral profile show that for signal simulations with low mean frequency values, this number is very small. Since we are interested in drawing statistical conclusions (as accurate as possible) from the estimated spectra, and, tacking into account the limitations imposed on the signals' specifications, only signal simulation cases with a number of frequency bins greater than 9, were considered. Thus, the signal simulation cases with mean frequencies of 500Hz, for all bandwidths, and, the case of mean frequency of 1kHz and half bandwidth of 5% were no longer considered on this study.

Measures of performance of the spectral parameter estimators mean frequency and half bandwidth were described, namely, bias, standard deviation and mean squared error, assuming the form of percentage relative values.

These metrics were software implemented and results were taken from spectra obtained by Fourier transforming windowed Boxcar and Hanning Doppler signal simulations.

The results obtained for several different signal simulation cases were analyzed. The average of these statistical measurements over five samples of each signal simulation case, enabled the selection of a size of ensemble of 150, as a suitable compromise between the computation burden of large ensemble number and resultant variance values from finite ensemble size.

With the spectra and sampling frequencies considered, the ensemble averaged spectrum is insensitive to small shifts (of a quarter of the frequency resolution) in the mean frequency value of the spectrum.

CHAPTER 5

CHAPTER 5

Spectral Estimation: Methods and Selection

5.1. INTRODUCTION

The primary objective of this research work is the selection of methods or if possible the selection of a method of spectral estimation which best estimates the spectrum of a pulsed Doppler ultrasound signal with Gaussian spectral profile.

Extensive research in the last few years has been carried out in the spectral estimation field, particularly investigating the applicability of parametric models.

Classical methods of spectral estimation based on the application of Fourier transforms are recognized in literature as limited. Accused of being very weak in frequency resolution, and of suffering from inherent leakage due to window effects, some of these methods will be considered in this study for comparison.

Parametric model estimation requires a previous knowledge of general behaviour of the signal to be estimated. Tacking in account the particular signal under investigation, some pre-selection of methods was carried. These methods were applied to the signal and their spectrum analyzed.

Another non-parametric but non-classical method of spectral estimation, the minimum variance method was also investigated.

Measures of bias and standard deviation of estimators spectral mean frequency and half bandwidth estimators were obtained for the different spectral estimation methods applied.

The meaning of 'best' spectral estimation was considered. A selection criteria has been developed to reflect the constraints and objectives of the user.

5.2. SPECTRAL ESTIMATION THEORY

5.2.1. Basics

Parametric methods basically consists of three steps. The first step is the selection of the suitable parametric time-series model to fit the measured data. Next, the parameters of the assumed model are estimated, and finally, an estimate of the spectrum is obtained by substitution of the estimated model parameters into the corresponding theoretical PSD.

Most of the discrete-time processes encountered in practice may be approximated by a time-series model represented by a filter linear difference equation

$$x[n] = -\sum_{k=1}^{p} a[k].x[n-k] + \sum_{k=0}^{q} b[k].u[n-k] \qquad (5.1)$$

where x[n] denotes the output sequence of the filter, with an input sequence u[n].

The z-transforms [37] of the output and of the input are related by the rational transfer function

$$H(z) = \frac{X(z)}{U(z)} = \frac{B(z)}{A(z)} \qquad (5.2)$$

where A(z) (considering without loss of generality a[0]=1) and B(z) are the polynomials

$$A(z) = 1 + \sum_{k=1}^{p} a[k].z^{-k} \qquad (5.3)$$

$$B(z) = \sum_{k=0}^{q} b[k].z^{-k} \qquad (5.4)$$

It is assumed that all zeros of A(z) are inside the unit circle of the z-plane to assure stability and causality of H(z). The parameters a[k] and b[k] represent the filter model parameters.

The output sequence may alternatively be represented by the convolution of the input sequence with h[n]

$$x[n] = \sum_{k=0}^{\infty} h[k] . u[n-k] \tag{5.5}$$

where h[k] is the impulse response defined as

$$h[k] = Z^{-1}[H(z)] \tag{5.6}$$

where $Z^{-1}[\cdot]$ is the inverse z-transform of function $\cdot$.

Writing (5.2) in the form

$$H(z) = \frac{\sum_{k=0}^{q} b[k] . z^{-k}}{1 + \sum_{k=1}^{p} a[k] . z^{-k}} \tag{5.7}$$

illustrates the idea of the pole-zero process that characterizes the autoregressive-moving average model. Thus it is generally referred to as ARMA(p,q).

If q=0 and b[0]=1,

$$H(z) = \frac{1}{1 + \sum_{k=1}^{p} a[k] . z^{-k}} \tag{5.8}$$

we have the all-pole model or, as usually termed, the autoregressive model of order p, AR(p).

A value of p=0, leads to the all-zero model or, moving-average model of order q, MA(q)

$$H(z) = \sum_{k=0}^{q} b[k] . z^{-k} \tag{5.9}$$

It is possible to establish the relationships between the filter parameters and the autocorrelation sequence either by taking the inverse of the z-transform of the power density spectrum, or directly from the time-series model [17].

The development of these relationships may be found in general books of spectral analysis, such as Proakis and Manolakis [37], Kay [17], or Marple [32]. The relationship between the ARMA parameters and the autocorrelation sequence $r_{xx}[k]$ of a process x[n] is

$$r_{xx}[k] = \begin{cases} r_{xx}^{*}[-k] & k<0 \\ -\sum_{i=1}^{p} a[i] \cdot r_{xx}[k-i] + \sigma_u^2 \cdot \sum_{i=k}^{q} b[i] \cdot h^{*}[i-k] & 0 \le k \le q \\ -\sum_{i=1}^{p} a[i] \cdot r_{xx}[k-i] & k>q \end{cases} \quad \textbf{(5.11)}$$

The equivalent equations for an AR process are obtained by setting q=0,

$$r_{xx}[k] = \begin{cases} r_{xx}^{*}[-k] & k<0 \\ -\sum_{i=1}^{p} a[i] \cdot r_{xx}[-i] + \sigma_u^2 & k=0 \\ -\sum_{i=1}^{p} a[i] \cdot r_{xx}[k-i] & k>0 \end{cases} \quad \textbf{(5.11)}$$

leading to the so called *Yule-Walker equations* [32].

Usually these linear relationships are written in the matrix form

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-(p-1)] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-(p-2)] \\ \vdots & \vdots & \cdots & \vdots \\ r_{xx}[p-1] & r_{xx}[p-2] & \cdots & r_{xx}[0] \end{bmatrix} \cdot \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} = -\begin{bmatrix} r_{xx}[1] \\ r_{xx}[2] \\ \vdots \\ r_{xx}[p] \end{bmatrix} \quad \textbf{(5.12)}$$

The square matrix (p*p) of autocorrelation values is designated the *Autocorrelation matrix*, $\mathbf{R_{xx}}$ and has the properties of being *Hermitian* ($[\mathbf{R_{xx}}]^H=\mathbf{R_{xx}}$), *Toeplitz* since the elements along any diagonal are equal, and *positive semidefinite* or even *positive definite* if x[n] does not consist purely of p-1 or fewer sinusoids [32].

These properties of $\mathbf{R_{xx}}$ provides us the facility of solving (5.12) using computational efficient algorithms such as the *Levinson algorithm*, with complexity (asymptotic number of flops required (see section 5.3.3)) $O(p^2)$ [17], [32].

If we now evaluate (5.10) for p=0 and noting that h[k]=b[k] for $1 \le k \le q$, the relationship between the ACS and the MA model parameters is given by (5.13) [32]

$$r_{xx}[k] = \begin{cases} r_{xx}^{*}[-k] & k<0 \\ \sigma_u^2 \sum_{i=k}^{q} b[i] \cdot b^{*}[i-k] & 0 \le k \le q \\ 0 & k>q \end{cases} \quad \textbf{(5.13)}$$

The above relationships between the ACS and the ARMA and MA filter parameters include nonlinearities due to the unknown input driving sequence u[n]. The *Wold Theorem* states that any ARMA or MA(AR) process may be represented uniquely by an AR(MA) model of infinite order [37], and may often be closely approximated by an AR (MA) model of an order not too much greater than that of the process being approximated.

Therefore, the choice of an AR model seems to be more attractive. Since its computational burden is less than that of an MA or an ARMA model, sometimes it is advisable to estimate the parameters of an AR model and afterwards compute the parameters of the desired model using those already obtained by the AR model as first approximation of the model parameters [32].

After computing the model parameters estimates the spectral estimate may be obtained by insertion of the estimated parameters into the power density spectrum expression appropriate for the chosen model.

The PSD of an ARMA process is given by

$$P(f_n) = \sigma_u^2 \cdot \left| \left(\frac{B(z)}{A(z)} \right)_{z=e^{j2\pi f_n}} \right|^2 \qquad (5.14)$$

where $f_n=f/f_s$ represent the normalized frequency ($f_n \in [-1/2,1/2]$).

5.2.2. Parametric Methods: Autoregressive spectral estimators

When dealing with AR processes the general modelling theory described in section 5.2.1 may be equivalently explained in terms of linear prediction theory and lattice filter theory.

Some of the AR spectral estimation methods employed are algorithmically developed taking advantage of the affinity between these three approaches. Therefore a brief explanation of the subject will help in understanding the spectral estimation methods investigated.

The linear prediction problem is concerned with the prediction of unobserved data x[n], assuming that x[n] is an AR process.

The *forward linear prediction estimate* is defined as

$$\hat{x}^f[n] = -\sum_{i=1}^{k} a^f[i] \cdot x[n-i] \tag{5.15}$$

where $a^f[i]$ are the *forward linear prediction coefficients*, and superscript f denotes forward prediction. The prediction $x^f[n]$ is based on the observed sequence {x[n-1], x[n-2], ..., x[n-k]} of the last k samples.

A measure of the quality of this estimate is given by the *forward linear prediction error*

$$e^f[n] = x[n] - \hat{x}^f[n] \tag{5.16}$$

which is frequently used in practice in the form of a mean squared error, computed as

$$\mathcal{E}[\,|e^f[n]|^2\,] = \mathcal{E}[\,|x[n] - \hat{x}^f[n]|^2\,] = \rho^f \tag{5.17}$$

The quantity ρ^f is termed the *power of the forward prediction error*. Analysis of the statistics of ρ^f allows its decomposition into the variance of the estimator $\hat{x}[n]$ and its bias [32].

If we can assume $\hat{x}^f[n]$ to be unbiased, ρ^f is only a measure of variance.

Our interest is to compute the parameters $a^f[i]$ that will produce the best approximation $\hat{x}^f[n]$ of the sequence x[n].

Matrix algebra theory suggests that in this situation the method of uniquely assigning values to $a^f[i]$ is the computation of the minimum of ρ^f [32].

Hence, developing (5.17) after substituting $\hat{x}^f[n]$ from (5.15), and assuming x[n] as a WSS process, we come to the conclusion that the vector $a^f[i]$ which minimizes the variance ρ^f is obtained by solution of the matrix system

$$\begin{bmatrix} r_{xx}[0] & r_{xx}^*[1] & \cdots & r_{xx}^*[k] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}^*[k-1] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[k] & r_{xx}[k-1] & \cdots & r_{xx}[0] \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a^f[1] \\ \vdots \\ a^f[k] \end{bmatrix} = \begin{bmatrix} \rho^f \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5.18}$$

where the equivalence $r_{xx}[-k]=r_{xx}^*[k]$ [32] is employed. This system presents similarities with the Yule-Walker equations for an AR process, and is very useful in understanding the subsequent algorithms.

It should however be noticed that distinct interpretations are given for the input and output sequences of the filter. Considering the modelling approach, u[n] is the input white noise process

of an autoregressive filter with x[n] as output, while considering the linear prediction theory, the error sequence $e^f[n]$ is the output of a forward linear prediction filter with x[n] as input. Only when x[n] is an AR(p) process, and k=p does $e^f[n]$ become a white noise process [79].

The *backward linear prediction estimate* $\hat{x}^b[n]$ is defined in a similar form as $\hat{x}^f[n]$ but basing the prediction at time index n on k later samples.

$$\hat{x}^b[n] = -\sum_{i=1}^{k} a^b[i] \cdot x[n+i] \tag{5.19}$$

The parameters $a^b[i]$ are called *backward linear prediction coefficients* evaluated at time i. The *backward linear prediction errors* $e^b[n]$ are defined by

$$e^b[n] = x[n-k] - \hat{x}^b[n-k] \tag{5.20}$$

and the *power of the backward prediction error* is

$$\rho^b = \mathcal{E}[\,|e^b[n]|^2] \tag{5.21}$$

Analogously, the computation of the sequence $a^b[i]$ that minimizes ρ^b, is obtained by solving the system

$$\begin{bmatrix} r_{xx}[0] & r_{xx}^*[1] & \cdots & r_{xx}^*[k] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}^*[k-1] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[k] & r_{xx}[k-1] & \cdots & r_{xx}[0] \end{bmatrix} \cdot \begin{bmatrix} a^b[k] \\ \vdots \\ a^b[1] \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \rho^b \end{bmatrix} \tag{5.22}$$

which again involves an Hermitian Toeplitz autocorrelation matrix [32].

When solving (5.18) and (5.22) the following properties are verified

$$\begin{aligned} \rho^b &= \rho^f \\ a^b[i] &= (a^f[i])^* \end{aligned} \tag{5.23}$$

for $1 \le i \le k$, which means that for prediction filters with same length k the backward linear prediction coefficients equal the complex conjugate of the forward linear prediction coefficients [32].

The other possible representation for an autoregressive process is the lattice filter representation.

In this approach either the forward and the backward prediction errors are functions of both prediction errors, and the parameters in use are the *reflection coefficients*, K_k.

The reflection coefficients equal the linear prediction coefficients $a_k[k]$ for $1 \leq k \leq p$, where the $a_p[k]$ are the final approximations of the linear prediction coefficients. The reflection coefficients are commonly used as the negative normalized correlation coefficient between the forward and the backward linear prediction errors with one unit of delay [32]

$$K_k = \frac{-\mathcal{E}[e_{k-1}^f[n] \cdot (e_{k-1}^b[n-1])^*]}{\sqrt{\mathcal{E}[|e_{k-1}^f[n]|^2]} \cdot \sqrt{\mathcal{E}[|e_{k-1}^b[n-1]|^2]}} \tag{5.24}$$

Development of the numerator of this equation enables the establishment of $|K_k| \leq 1$ for $1 \leq k \leq p$, which constitutes a necessary and sufficient condition for the autocorrelation matrix to be positive semidefinite [32].

The linear prediction errors are recursively computed as

$$e_k^f[n] = e_{k-1}^f[n] + K_k \cdot e_{k-1}^b[n-1] \tag{5.25}$$

$$e_k^b[n] = e_{k-1}^b[n-1] + K_k^* \cdot e_{k-1}^f[n] \tag{5.26}$$

with initial conditions $x[n]=e_0^f[n]=e_0^b[n]$.

So we may briefly summarize the possible AR(p) processes representations as:

a) the infinite-duration autocorrelation sequence. This sequence has in fact only p+1 samples, $\{r_{xx}[0], r_{xx}[1], ..., r_{xx}[p]\}$, as the other samples may be obtained recursively. This finite set of ACS values defines a positive definite sequence, enabling the application of fast computational algorithms to solve the Yule-Walker equations.

b) the finite sequence of autoregressive parameters $\{\rho_p, a_p[1], a_p[2], ..., a_p[p]\}$ obtained by minimization of the power of the prediction errors, which will produce a minimum phase polynomial

$$A(z) = 1 + \sum_{i=1}^{p} a_p[i] \cdot z^{-i} \tag{5.27}$$

c) the finite sequence of reflection coefficients $\{r_{xx}[0], k_1, k_2, ..., k_p\}$ which provide $|K_k| < 1$ and $\rho_k > 0$ for $1 \leq k \leq p$.

The description of the AR spectral estimation methods employed which follows, although commonly found in many publications in the area, seemed to be a useful complement in this study.

A. Yule-Walker Method

The Yule-Walker method is commonly termed the *Autocorrelation method* since it is based on the solution of the system of equations involving the autocorrelation matrix $\mathbf{R}_{xx}$.

Equation (5.12) is computed for estimated values

$$\begin{bmatrix} \hat{r}_{xx}[0] & \hat{r}_{xx}[-1] & \cdots & \hat{r}_{xx}[-(p-1)] \\ \hat{r}_{xx}[1] & \hat{r}_{xx}[0] & \cdots & \hat{r}_{xx}[-(p-2)] \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{xx}[p-1] & \hat{r}_{xx}[p-2] & \cdots & \hat{r}_{xx}[0] \end{bmatrix} \cdot \begin{bmatrix} \hat{a}[1] \\ \hat{a}[2] \\ \vdots \\ \hat{a}[p] \end{bmatrix} = - \begin{bmatrix} \hat{r}_{xx}[1] \\ \hat{r}_{xx}[2] \\ \vdots \\ \hat{r}_{xx}[p] \end{bmatrix} \tag{5.28}$$

where the biased ACS $\hat{r}_{xx}[k]$ for lag k is given by

$$\hat{r}_{xx}[k] = \begin{cases} \dfrac{1}{N} \cdot \displaystyle\sum_{n=0}^{N-1-k} x^*[n] \, . \, x[n+k] & k = 0,1,\cdots,p \\ \hat{r}^*_{xx}[-k] & k = -(p-1),-(p-2),\cdots,-1 \end{cases} \tag{5.29}$$

System (5.28) may be solved by various algorithms, but Levinson algorithm is usually the selected algorithm since it enables a computationally fast solution. This algorithm enables the computation of the reflection coefficients $\{k_1,k_2,...,k_p\}$ given the autocorrelation sequence values and the desired model order [17].

Regarding the application of the Levinson algorithm the solution of the Yule-Walker equations may be expressed in terms of the minimization of the prediction error power

$$\hat{\rho} = \frac{1}{N} \cdot \sum_{n=-\infty}^{\infty} \left| x[n] + \sum_{k=1}^{p} a[k] \, . \, x[n-k] \right|^2 \tag{5.30}$$

where for $n<0$ and $n>N$, $x[n]$ is made zero.

The estimate of the white noise variance σ^2 is

$$\hat{\sigma}^2 = \hat{\rho}_{min} = \hat{r}_{xx}[0] + \sum_{k=1}^{p} \hat{a}[k] \cdot \hat{r}_{xx}[-k] \tag{5.31}$$

which is given in the last step of the Levinson recursion as

$$\hat{\sigma}^2 = \hat{r}_{xx}[0] \prod_{i=1}^{p} (1 - |\hat{k}_i|^2) \tag{5.32}$$

The resolution of the spectral estimates obtained with this method are the poorest of this class of spectral estimation methods [17], specially if short data records are used [40].

The ACS values computed are biased estimates, producing in general model parameter estimates which lead to transfer functions with poles located inside the unit circle. The unbiased version of the ACS frequently produces singular autocorrelation matrices, and is not recommended [17].

B. Burg Method

This method, originally proposed by Burg as a maximum entropy algorithm, was later interpreted as a constrained least squares minimization algorithm [35].

The model parameter estimates are obtained by minimization, for each model order k, of the arithmetic mean of the forward ($\hat{\rho}^f_k$) and backward ($\hat{\rho}^b_k$) prediction error powers

$$\hat{\rho}^f_k = \frac{1}{N-k} \cdot \sum_{n=k}^{N-1} \left| x[n] + \sum_{i=1}^{k} a^f_k[i] \cdot x[n-i] \right|^2 = \frac{1}{N-k} \cdot \sum_{n=k}^{N-1} \left| \hat{e}^f_k[n] \right|^2 \tag{5.33}$$

$$\hat{\rho}^b_k = \frac{1}{N-k} \cdot \sum_{n=0}^{N-1-k} \left| x[n] + \sum_{i=1}^{k} a^{f*}_k[i] \cdot x[n+i] \right|^2 = \frac{1}{N-k} \cdot \sum_{n=k}^{N-1} \left| \hat{e}^b_k[n] \right|^2 \tag{5.34}$$

where the prediction errors, $\hat{e}_k^f[n]$ and $\hat{e}_k^b[n]$, computed over the available data only, are recursively obtained as functions of the k^{th} reflection coefficient,

$$\hat{K}_k = \frac{-2 \cdot \sum_{n=k}^{N-1} \hat{e}^f_{k-1}[n] \cdot \hat{e}^b_{k-1}[n-1]^*}{\sum_{n=k}^{N-1} \left(\left| \hat{e}^f_{k-1}[n] \right|^2 + \left| \hat{e}^b_{k-1}[n-1] \right|^2 \right)} \tag{5.35}$$

by the relationship,

$$\begin{aligned} \hat{e}^f_k[n] &= \hat{e}^f_{k-1}[n] + \hat{K}_k \cdot \hat{e}^b_{k-1}[n-1] \qquad n = k+1,k+2,\cdots,N-1 \\ \hat{e}^b_k[n] &= \hat{e}^b_{k-1}[n-1] + \hat{K}^*_k \cdot \hat{e}^f_{k-1}[n] \qquad n = k,k+1,\cdots,N-2 \end{aligned} \tag{5.36}$$

The averaged prediction error power, $\rho_k=(\rho^f_k+\rho^b_k)/2$, may be simply expressed in terms of the last estimate $\hat{\rho}_{k-1}$ as

$$\hat{\rho}_k = (1 - |\hat{K}_k|^2) \cdot \hat{\rho}_{k-1} \tag{5.37}$$

making clear the fact that the parameter estimates are constrained by the previously obtained parameter estimates [17].

The AR model parameter estimates are updated recursively as functions of the reflection coefficient estimates $\hat{K}_k$,

$$\hat{a}_k[i] = \begin{cases} \hat{a}_{k-1}[i] + \hat{K}_k \cdot \hat{a}^*_{k-1}[k-i] & i=1,2,\cdots,k-1 \\ \hat{K}_k & i=k \end{cases} \tag{5.38}$$

where for k=1 $\hat{a}_1[1]=\hat{K}_1$.

The final AR parameter estimates are $\{\hat{a}_p[1], \hat{a}_p[2],\ldots,\hat{a}_p[p]\}$ and the white noise variance is made equal to the prediction error power for lag p.

The constraint imposed by Burg ensures that the AR filter poles are within the unit circle (to achieve stability of the filter) [35], however the constraint seems to produce problems such as spectral line splitting, specially for narrowband signals [11], and bias of the frequency estimate, which do not occur when a non-constrained minimization of the same expression of averaged prediction error power is used [32].

C. Modified Covariance Method

The modified covariance method as the name suggests results from a modification on the *covariance method.*

The covariance method computes the model parameter estimates by minimizing the following expression of prediction error power

$$\hat{\rho} = \frac{1}{N-p} \cdot \sum_{n=p}^{N-1} \left| x[n] + \sum_{k=1}^{p} a[k] \cdot x[n-k] \right|^2 \qquad (5.39)$$

This expression, coincident with the forward prediction error power used in the modified covariance method, differs from the one employed in Yule-Walker method only on the range of the outer summation. In (5.39) only (N-p) lag products are used to compute the ACS, although there are more, and in (5.30) zeroing of data is assumed for n<0 and n>N. So, both Yule-Walker and covariance methods make use of a truncated autocorrelation sequence which is the cause of some smearing of the spectra.

The modified covariance method makes use of all the available data samples by considering, as the Burg method, the minimization of the arithmetic average of the forward and backward prediction error powers

$$\hat{\rho} = \frac{1}{2} \cdot (\hat{\rho}^f + \hat{\rho}^b) \qquad (5.40)$$

where ρ^f is given by (5.39) and ρ^b is

$$\hat{\rho}^b = \frac{1}{N-p} \cdot \sum_{n=0}^{N-1-p} \left| x[n] + \sum_{k=1}^{p} a^*[k] \cdot x[n+k] \right|^2 \qquad (5.41)$$

This approach involves the optimal forward and backward prediction estimates as expressed in (5.15) and (5.19) respectively.

ρ_{min} is obtained by solution of the *modified covariance equations*

$$\begin{bmatrix} c_{xx}[1,1] & c_{xx}[1,2] & \cdots & c_{xx}[1,p] \\ c_{xx}[2,1] & c_{xx}[2,2] & \cdots & c_{xx}[2,p] \\ \vdots & \vdots & \ddots & \vdots \\ c_{xx}[p,1] & c_{xx}[p,2] & \cdots & c_{xx}[p,p] \end{bmatrix} \cdot \begin{bmatrix} \hat{a}[1] \\ \hat{a}[2] \\ \vdots \\ \hat{a}[p] \end{bmatrix} = - \begin{bmatrix} c_{xx}[1,0] \\ c_{xx}[2,0] \\ \vdots \\ c_{xx}[p,0] \end{bmatrix} \qquad (5.42)$$

where the matrix, called the *covariance matrix*, is Hermitian and positive definite [17]. Each element of the matrix is computed as

$$c_{xx}[i,j] = \frac{1}{2.(N-p)} \cdot \left(\sum_{n=p}^{N-1} x^*[n-i] \,.\, x[n-j] + \sum_{n=0}^{N-1-p} x[n+i] \,.\, x^*[n+j] \right) \qquad (5.43)$$

The AR model parameter estimates are obtained by solution of the linear system (5.42) using for instance Cholesky decomposition [17] [32].

The estimate of the white noise variance is obtained as

$$\hat{\sigma}^2 = \hat{\rho}_{min} = c_{xx}[0,0] + \sum_{k=1}^{p} \hat{a}[k] \,.\, c_{xx}[0,k] \qquad (5.44)$$

Phenomena such as spectral line splitting, false shifting of frequency peaks and peak location dependence on initial phase have not been observed on spectral estimations obtained with the modified covariance method [17], and in particular this method is better than the Burg method which does exhibit spectral line splitting. A possible cause of such a behaviour is the fact that this method performs a minimization of the prediction error power ρ_k with respect to all prediction coefficients rather than just to the $a_{k-1}[k-1]$ as does the Burg method [32].

However the removal of the constraint no longer guarantees the filter stability, although most of the time the method is stable [80].

Another advantage of the modified covariance method is the fact that for a model order much smaller than the number of data points (N), the computational efficiency of the modified covariance algorithm approximates that of a Burg algorithm with the same model order [32].

After having computed the model parameter estimates, using any of the above methods, Yule-Walker, Burg, or, modified covariance, the estimates of the power spectrum are obtained using the estimates â[k] for k=1,2,...,p in the expression

$$\hat{P}_{AR}(f_n) = \frac{\hat{\sigma}^2}{|1 + \hat{a}[1] \,.\, e^{-j2\pi f_n} + \cdots + \hat{a}[p] \,.\, e^{-j2\pi f_n p}|^2} \qquad (5.45)$$

where f_n represents the normalized frequency f/f_s.

5.2.3. Parametric Methods: Moving Average Spectral Estimators

The power density spectrum of a moving average process may be represented by (5.46).

$$P_{MA}(f_n) = \sum_{k=-q}^{q} r_{xx}[k] \cdot e^{-j2\pi f_n k}$$

$$for \quad r_{xx}[k] = \begin{cases} \sigma^2 \cdot \sum_{k=-q}^{q} b^*[l] \cdot b[l+k] & k = 0,1,\cdots,q \\ r_{xx}^*[-k] & k = -q,-(q-1),\cdots,-1 \end{cases} \tag{5.46}$$

$P_{MA}(f)$ is obtained by Fourier transforming the ACS evaluated at lags $-q \leq k \leq q$, which makes this approach very similar to a Blackman Tukey spectral estimation. The process is assumed to be a MA process with zero theoretical ACS values for lags outside the range [-q,q], while in the Blackman Tuckey periodogram the unmeasured ACS samples are forced to zero [17].

The difficulty of solving the nonlinear system to obtain the ACS lead to other means of computing the MA spectral estimate.

The Durbin's method exploits the possibility of computing the MA parameters by an AR process of very high order, as stated by the Wold's theorem [17].

Durbin Method

A MA(q) process may be represented by the data sequence

$$x[n] = \sum_{k=0}^{q} b[k] \cdot u[n-k] \tag{5.47}$$

which is equivalent to

$$x[n] = -\sum_{k=1}^{\infty} a[k] \cdot x[n-k] + u[n] \tag{5.48}$$

of an AR(∞) process, if a[k] constitute the impulse response of A(z)=1/B(z).

The general idea is to replace the MA(q) process by an AR(L), with $q \ll L \ll N$, that approximates the impulse response of 1/B(z), which decays to zero for index values greater than

L. This will lead to AR(L) parameter estimates {â[1], â[2],..., â[L]}, and an estimate of the white noise variance computed as

$$\hat{\sigma}^2 = \hat{r}_{xx}[0] + \sum_{k=1}^{L} \hat{a}[n].\hat{a}[n+i] \tag{5.49}$$

The parameter estimates are fed to an AR(q) filter as input data and the resulting parameter estimates are the MA(q) parameter estimates, this is, {b̂[1],b̂[2],...,b̂[q]}.

This way the MA parameter estimates represented as $\hat{\mathbf{b}}$ are obtained by the product

$$\hat{b} = -\hat{R}_{aa}^{-1} . \hat{r}_{aa} \tag{5.50}$$

where $\hat{R}_{aa}^{-1}$ is the inverse of the L×L matrix of autocorrelation between the estimated AR(L) parameters with elements computed as

$$[\hat{R}_{aa}]_{ij} = \frac{1}{L+1} . \sum_{n=0}^{L-|i-j|} \hat{a}[n] . \hat{a}[n+|i-j|] \qquad i,j = 1,2,\cdots,q \tag{5.51}$$

and $\hat{\mathbf{r}}_{aa}$ is the L×1 vector of these ACS values with elements given by

$$[\hat{r}_{aa}]_i = \frac{1}{L+1} . \sum_{n=0}^{L-i} \hat{a}[n] . \hat{a}[n+i] \qquad i = 1,2,\cdots,q \tag{5.52}$$

The MA parameter estimation method just stated makes use of the Yule-Walker or autocorrelation method. However any other method of last section would be applicable [32].

The spectral estimate is obtained by substitution of {b̂[1],b̂[2],...,b̂[q],σ̂²} into

$$\hat{P}_{MA}(f_n) = \hat{\sigma}^2 \left| \hat{B}(f_n) \right|^2$$
$$for \quad \hat{B}(f_n) = 1 + \sum_{k=1}^{q} \hat{b}[k] . e^{-j2\pi f_n k} \tag{5.53}$$

5.2.4. Parametric Methods: Autoregressive Moving Average Spectral Estimators

The model ARMA is suitable to approximate processes whose PSDs in general exhibit both spectral peaks and spectral valleys.

For reliable estimates of the ARMA parameters maximum likelihood estimate techniques should be applied. For large data records these estimation methods produce unbiased and efficient estimates. However the solution of their highly nonlinear set of equations make them difficult to implement and computational time consuming [17].

For this reasons alternative methods have been developed. They rely on the theoretical relationship between the ACS and the ARMA parameters stated in equation (5.10).

With this approach faster algorithms are achieved at the expense of suboptimal solutions.

Akaike method is concerned with the maximization of the likelihood function, therefore it has not been included in this study.

The *least squares modified Yule-Walker equations method* (LSMYWE method) is an extended version of the *modified Yule-Walker equations method.* It considers more equations than unknowns resulting in a least squares problem.

The LSMYWE method appear to provide best performance with ARMA processes whose poles are dominant [17]. Since the PDU signal spectrum usually has a single peak, the LSMYWE method has been studied.

For the same reason another suboptimal method the *Mayne-Firoozan* method has been investigated.

A. Least Square Modified Yule-Walker Equations Method

This method is based on the modified Yule-Walker equations method. A brief description of the underlying theory is therefore explained.

For an ARMA(p,q) process the ACS may be expressed as

$$r_{xx}[k] = -\sum_{i=1}^{q} a[i] \cdot r_{xx}[k-i] \qquad k \geq q+1 \tag{5.54}$$

where the choice of the p equations corresponding to k=q+1,q+2,...,q+p is arbitrary [17]. Writing these equations in matrix form the *modified Yule-Walker equations* are obtained

$$\begin{bmatrix} r_{xx}[q+1] \\ r_{xx}[q+2] \\ \vdots \\ r_{xx}[q+p] \end{bmatrix} = - \begin{bmatrix} r_{xx}[q] & r_{xx}[q-1] & \cdots & r_{xx}[q-p+1] \\ r_{xx}[q+1] & r_{xx}[q] & \cdots & r_{xx}[q-p+2] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[q+p-1] & r_{xx}[q+p-2] & \cdots & r_{xx}[q] \end{bmatrix} \cdot \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} \tag{5.55}$$

Solving these equations with the theoretical ACS values replaced by estimates brings up the AR parameters {â[1], â[2], ...,â[p]} of the ARMA process.

The remaining parameters {b̂[1], b̂[2], ...,b̂[p], σ̂²}, may be found by filtering the sequence x[n] with

$$\hat{A}(z) = 1 + \sum_{i=1}^{p} \hat{a}[i] \cdot z^{-i} \tag{5.56}$$

For good approximations Â(z) of A(z) the output of the filter will approximate the MA(q) process, for which Durbin method produces the parameter estimates.

Problems arise when the matrix of ACS samples in (5.55) is nearly singular. The solution proposed on LSMYWE method is to consider more than p equations, say M, and solve the system in a least squares sense [17].

The number of equations considered must be sufficiently high to take advantage of the information on the high-order lags, but not so large that the highest sample of the ACS becomes unreliable.

Equation (5.55) may be rewritten considering M equations, and including an error vector introduced by the overdetermination imposed

$$\begin{bmatrix} r_{xx}[q+1] \\ r_{xx}[q+2] \\ \vdots \\ r_{xx}[M] \end{bmatrix} = - \begin{bmatrix} r_{xx}[q] & r_{xx}[q-1] & \cdots & r_{xx}[q-p+1] \\ r_{xx}[q+1] & r_{xx}[q] & \cdots & r_{xx}[q-p+2] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[M-1] & r_{xx}[M-2] & \cdots & r_{xx}[M-p] \end{bmatrix} \cdot \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix} + \begin{bmatrix} e[q+1] \\ e[q+2] \\ \vdots \\ e[M] \end{bmatrix} \tag{5.57}$$

or in matrix notation

$$\hat{r} = -\hat{R}.a + e \tag{5.58}$$

where r̂ and R̂ denote respectively the vector and matrix of ACS samples in (5.57), when computed for estimated values.

The ACS estimates to be used in (5.58) should be unbiased estimates so that the average of this error equation ($\mathscr{E}[\mathbf{e}]=\mathscr{E}[\hat{\mathbf{r}}]+\mathscr{E}[\hat{\mathbf{R}}]\mathbf{a}=\mathbf{r}+\mathbf{R}\mathbf{a}$) produces nil values [17].

The least-squares solution of (5.58), as in general least squares problems, is given by

$$\hat{a} = -(\hat{R}^H . \hat{R})^{-1} . \hat{R}^H . \hat{r} \tag{5.59}$$

where $\hat{\mathbf{R}}^H\hat{\mathbf{R}}$ is usually positive definite, thus invertible by Cholesky decomposition [32].

Having these AR parameter estimates, the MA parameters are obtained in the same way as in the Modified Yule-Walker method.

The major problem of the LSMWYE method is the selection of a suitable M. If M is too large $\hat{\mathbf{R}}^H\hat{\mathbf{R}}$ will tend to be singular; however the variance of the estimator $\hat{\mathbf{a}}$ decreases with increasing number of equations (M-q). A compromise value of M=N/5 for N representing the number of frequency points is suggested by Kay [17].

B. Mayne-Firoozan Method

Another possible approach to avoid the computation of maximum likelihood estimators and consequent nonlinear systems of equations is to exploit the relationship between the data sequence x[n] and the unknown input u[n]. In fact the observed nonlinearity is due to the unknown crosscorrelation function between x[n] and the input u[n] [17].

In this method, also known as the *three-stage least square method*, an estimate of u[n] is computed enabling (5.1) to be written as

$$x[n] = -\sum_{k=1}^{p} a[k] . x[n-k] + \sum_{k=1}^{q} b[k] . \hat{u}[n-k] + \hat{u}[n] \qquad n=0,1,\cdots,N-1 \tag{5.60}$$

where {x[-p],...,x[-1],u[-q],...,u[-1]} are set to zero, or equivalently in matrix notation

$$\hat{x} = H . \theta + \hat{u} \tag{5.61}$$

where

$$\begin{aligned} \boldsymbol{x} &= [x[0] \;\; x[1] \;\cdots\; x[N-1]]^T \\ \hat{\boldsymbol{u}} &= [\hat{u}[0] \;\; \hat{u}[1] \;\cdots\; \hat{u}[N-1]]^T \\ \theta &= [-a[1] \;\; -a[2] \;\cdots\; -a[p] \;\; b[1] \;\; b[2] \;\cdots\; b[q]]^T \end{aligned} \tag{5.62}$$

and

$$\boldsymbol{H} = \begin{bmatrix} x[-1] & x[-2] & \cdots & x[-p] & \hat{u}[-1] & \hat{u}[-2] & \cdots & \hat{u}[-q] \\ x[0] & x[-1] & \cdots & x[-p+1] & \hat{u}[0] & \hat{u}[-1] & \cdots & \hat{u}[-q+1] \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x[N-2] & x[N-3] & \cdots & x[N-p-1] & \hat{u}[N-2] & \hat{u}[N-3] & \cdots & \hat{u}[N-q-1] \end{bmatrix} \tag{5.63}$$

System (5.61) has now the structure of a linear system whose parameters are given by the least squares solution

$$\hat{\theta} = (\boldsymbol{H}^H.\boldsymbol{H})^{-1}.\boldsymbol{H}^H.\boldsymbol{x} \tag{5.64}$$

which might be obtained by Cholesky decomposition [17]. Thus, the estimates of the input sequence $\hat{u}[n]$ are obtained by filtering the data sequence x[n] with $\hat{A}(z)$, whose parameters are estimates obtained by application of one of the AR estimation algorithms to the data x[n], considering a large order model, say M.

The MA parameter estimates $\tilde{b}[k]$ obtained are used to form the filtered sequences y[n] and z[n], which will lead to the final estimates of the ARMA parameters by minimization of

$$\sum_{n=0}^{N-1} \left| \sum_{k=0}^{p} a[k].y[n-k] - \sum_{k=0}^{q} b[k].z[n-k] \right|^2 \tag{5.65}$$

following an identical procedure as indicated in (5.64) where **x** is replaced by (**y-z**), **H** has the same form as in (5.63) with x replaced by y, and, u is replaced by z [17].

The estimate of the white noise variance is obtained by [17]

$$\hat{\sigma}^2 = \frac{1}{N} \cdot \int_{-\frac{1}{2}}^{\frac{1}{2}} \frac{|X(f_n)|^2 . |\hat{A}(f_n)|^2}{|\hat{B}(f_n)|^2} \, df_n \tag{5.66}$$

This method, characterized by making simultaneous estimate of both the autoregressive and moving average parameters in an interactive way, does not always guarantee the convergence of

the iterative procedure and the conditions for convergence are not generally known [32], which is clearly a deficiency of the method.

5.2.5. Minimum Variance Method

The minimum variance method, sometimes referred to as the *Capon method*, or the *maximum likelihood method*, is neither a parametric nor a classical method of spectral estimation.

This method provides an estimate of the PSD by measuring the power out of a set of narrowband filters like the periodogram estimate. But in the minimum variance method the shape of the filters are, in general, different for each frequency, and adapted to the process for which the PSD is sought [81], while in the periodogram estimate procedure the filter shapes are fixed [40].

In this method the PSD is expressed in terms of the impulse response of a linear shift-invariant filter h[n], whose frequency response $H(\omega)$ ($\omega=2\pi f_n$) is equivalent to a bandpass filter with centre frequency $\omega=\omega_0$, producing a filter output power due only to the PSD in the vicinity of ω_0 [81].

To alleviate the problem of sidelobe leakage into the neighbourhood frequencies which distorts the resulting PSD, the power at the output of each filter

$$\rho = \int_{-\pi}^{\pi} |H(\omega)|^2 . P_x(\omega) d\frac{\omega}{2\pi} \tag{5.67}$$

is minimized subject to the constraint that the filter frequency response should be unitary, this is $H(\omega_0)=1$.

Development of this expression enables the power ρ to be expressed in the matrix form

$$\rho = h^H R_{xx} h \tag{5.68}$$

$$when \quad h = [h[0] \;\; h[-1] \;\; \cdots \;\; h[-(N-1)]]^T$$

and $\mathbf{R_{xx}}$ is the N×N matrix of autocorrelation samples, presenting a minimum value of [81]

$$\rho = \frac{1}{e^H R_{xx}^{-1} e} \tag{5.69}$$

The spectral estimate is defined to be

$$\hat{P}_{MV} = \frac{p}{e^{H} R_{xx}^{-1} e} \tag{5.70}$$

for

$$e = \left[1 \quad e^{j\omega} \quad \cdots \quad e^{j\omega(p-1)}\right]^{T} \tag{5.71}$$

where the factor p is included to yield a power spectral density estimate [81]. As in the other nonparametric methods a tradeoff between bias and variance of the estimator is observed in the selection of p. In this method, too large values of p increases the variance while too small values present high bias [81].

Although the minimum variance spectral estimates exhibit better resolution than the periodogram estimates they are still poorer than these of an AR spectral estimator [40].

5.3. CRITERIA OF SELECTION OF SPECTRAL ESTIMATION METHODS

This project is concerned with comparing several spectral estimation techniques to select a method that best estimates the spectrum of the PDU signal simulation under consideration.

The definition of 'best' estimation is vague and we should contemplate several aspects. It is intended that the spectral estimate produce rapid efficient estimates of the mean frequency and bandwidth of the signal's spectrum.

When parametric spectral estimation is performed an additional difficulty is encountered. The order of the assumed model may vary leading to different statistical results for the estimators studied.

Selection of the best model order for each of the parametric methods employed would help in the comparison between the different spectral estimation methods.

Some model order selection methods are available in literature and the most commonly referred are described in next section.

These accepted model order selection criteria compute model order by analysis of the prediction error power, reflecting the goodness-of-fit of the parametric model assumed and the signal. This may not lead to an accurate estimate of spectral parameters [82]. In addition these criteria often lead to poor estimates of the correct order when used with signals whose order is known [16] [17].

These difficulties indicated the desirability of a new method of estimator selection (including the order of model based estimators) in which both accuracy of estimation of selected spectral parameters and the computational complexity of the estimation were taken into account.

For this reason a new selection criteria has been developed. It enables the joint comparison of parametric and nonparametric methods, given the statistical results obtained with each method and the user's requirement of accuracy in mean frequency and bandwidth estimation.

Each considered method may be weighted according to the user's requirement. Bias and standard deviation of both estimators (mean frequency and bandwidth) have also adjustable contributions and computational complexity of the algorithms employed may be considered in this new criteria.

5.3.1. Parametric Model Order Selection

Selection of the model order(s) of a parametric model may be achieved by systematic analysis of practical experiments with successive model orders or may be accomplished by application of a criteria of selection.

Since the use of a too low a model order implies, in general, a smooth, poor resolution, spectral estimate and conversely a too large value produces spurious spectral peaks [40], a reliable method of model order selection is of interest.

There are three common methods of computing the AR model order for real data sets.

The *Final Prediction Error* (FPE) and the *Akaike Information Criterion* (AIC) proposed by Akaike are based on the prediction error power's property of decreasing its value with increasing model order. Both include information about the number of data samples and the model order itself, such that the joint contribution of these three factors results in a curve with a minimum denoting the optimal model order.

More precisely, the FPE criteria is defined by the function

$$FPE(k) = \frac{N+k}{N-k}.\hat{\rho}_k \qquad (5.72)$$

[17] where $\hat{\rho}_k$ is the estimate of the prediction error power of the k^{th} order AR model and N is the number of data samples considered.

The AIC function is given by

$$AIC(k) = N\log\hat{\rho}_k + 2k \tag{5.73}$$

This last criteria, assumes that the process has Gaussian statistics [83], and is developed from statistical considerations on the difference between the assumed and the actual probability density functions of the sequence x[n].

For short data records the AIC criterion is recommended [17], however as N→∞ both AIC and FPE estimators yield identical model order estimates [41].

The third well known criterion termed the *Criterion Autoregressive Transfer function* (CAT) was proposed by Parzen.

$$CAT(k) = \frac{1}{N}\sum_{i=1}^{k}\frac{1}{\tilde{\rho}_i} - \frac{1}{\tilde{\rho}_k} \tag{5.74}$$

$$where \quad \tilde{\rho}_i = \frac{N}{N-i}\cdot\hat{\rho}_i$$

The optimum k is given when the estimate of the mean square error between the true filter and the estimated filter is a minimum [84].

The estimates of MA and ARMA model orders cannot be achieved by CAT or FPE criteria since for these models the prediction error power is monotonically decreasing with the order of the linear predictor [17].

So the Akaike information criterion assumes the form

$$AIC(j) = N\log\hat{\sigma}_j^2 + 2j \tag{5.75}$$

for an MA model, where σ_j^2 is the maximum likelihood estimate of the white noise variance of the j^{th} MA order model.

For an ARMA model the AIC function is

$$AIC(k,j) = N\log\hat{\sigma}_{kj}^2 + 2(k+j) \tag{5.76}$$

where k and j are the AR and MA model orders respectively, and the σ_{kj}^2 is the maximum likelihood estimate of the white noise variance of an ARMA(k,j).

These techniques present some weakness. It has been observed for instance that although producing the most accurate model order among the three criteria, AIC produces in general overestimated model orders when tested with a true AR-model input data sequence [17].

Schlindwein and Evans have also shown that these methods produce estimates of the model order of a true AR process different than the true order when short data sequences are considered [15].

Nevertheless, these or similar model order estimation methods may be applied to determine first approximations to the model order, specially when the process studied does not fit a pure parametric model.

5.3.2. New Selection Criterion: Cost/Benefit Criterion

In this study we were interested in determining the best spectral estimation method (or methods) among a variety of techniques applied to PDU signal simulations characterized with Gaussian spectral profile. In the case of parametric methods, the specification of the best method had also to include the optimal model order.

We were interested in determining a 'good' spectral estimate, in the sense that the mean frequency and half bandwidth estimates obtained were as close as possible to their true values.

As stated earlier, the available FPE, AIC and CAT are model order selection criteria applicable to a particular parametric spectral estimation method. Besides, they select the model order based on the study of the prediction error power. Thus these criteria are inappropriate and insufficient for the problem dealt here.

The new criterion enables simultaneous comparison of parametric or nonparametric estimator methods resulting in an indication of the method presenting the best performance.

In the case of the parametric techniques the model order that enabled a best fit of the assumed model to the data is also calculated.

The specification of what constitutes a good estimate is furnished to the criterion by means of weights attributed to the criterion parameters.

The parameters weighted according to the user are the mean squared errors of the estimators mean frequency and half bandwidth, the signal simulation cases, and the complexity of the algorithm employed.

Formalizing the criterion, the best spectral estimation method is therefore selected among m=1 up to m=M methods under comparison, as the one which presents a minimum value for an optimal criterion function $c_o(m)$, that is

$$\min_m \{ c_o(m), m = 1,\cdots,M \} \tag{5.77}$$

The optimal criterion function $c_o(m)$ represents by itself the result of the application of the criterion to a particular method m.

Let i and j be the indicators of the AR and MA model orders respectively, so that i=j=0 is representative of a nonparametric method, then the optimal criterion function is obtained by

$$c_o(m) = \min_{i,j} \left\{ \sum_{k=1}^{K} w_k . c(m,k,i,j), \quad i = 0,1,\cdots,I, \quad j = 0,1,\cdots,J \right\} \tag{5.78}$$

where k=1,...,K represent the K different simulation signals considered. The usefulness of each of these signals to the criterion may be expressed by weighting function w_k this is, a function of the probability of occurrence of the signal k in practice.

In (5.78) the minimization is performed for all values of model orders up to I and J which are the maximum values of model orders common to all cases K.

The function c(m,k,i,j) is a cost-benefit function

$$c(m,k,i,j) = \frac{cost(m,N(k),i,j)}{benefit(W,B_f,S_f,B_b,S_b)}$$

where

$$\begin{aligned} B_f &= bias_f(m,k,i,j) \\ S_f &= std_f(m,k,i,j) \\ B_b &= bias_b(m,k,i,j) \\ S_b &= std_b(m,k,i,j) \end{aligned} \tag{5.79}$$

were 'std' stands for standard deviation, and the subscripts 'f' and 'b' represent respectively the mean frequency and the half bandwidth estimators.

On the above equation the numerator represents the cost of applying a specific spectral estimation method m to a certain simulation signal k, characterized by N sample points and i and j represent the model order parameters in case of a parametric method.

The cost function is a measurement of the complexity of the algorithm employed. Factors such as the input of the data to the software program or the display of data were not taken into account since they occur for all methods and they are device dependent, rather than determined by algorithm characteristics.

This cost function is described in more detail in the next section.

The denominator of (5.79) is a measure of the quality of spectral estimation.

The benefit function weights the separate contribution of the estimators mean frequency and half bandwidth by means of their mean squared errors. This way the performance of the spectral estimation may be evaluated giving more emphasis to the clinical feature mean blood velocity, by attribution of higher weights to the statistics related with the mean frequency estimator, or, conversely by considering the blood flow disturbance as a more relevant feature by giving higher weights to the half bandwidth estimator.

A more precise characterization of the spectral study is provided by allowing the attribution of weights to both bias and standard deviation of mean frequency and half bandwidth estimators.

The benefit function is

$$benefit(W, B_f, S_f, B_b, S_b) = \frac{1}{w_{bf}.B_f^2 + w_{sf}.S_f^2 + w_{bb}.B_b^2 + w_{sb}.S_b^2} \qquad (5.80)$$

The symbol W on the left hand-side of equation (5.80) represents the set of weights w_{bf}, w_{sf}, w_{bb}, and w_{sb} attributed to the statistical performance measurements.

5.3.3. Complexity of the Algorithms

The cost function indicated as the numerator of (5.79) should quantify the computational cost of the algorithms employed to obtain the spectral estimates.

Usually assessment of algorithmic performance involves subjective considerations such as simplicity, clarity and appropriateness of the algorithm for the expected data, and run time efficiency [85].

Simplicity and clarity were considered when developing the algorithm software and therefore affect complexity and consequently the cost function.

The third issue (the appropriateness of the algorithm) was achieved in successive steps. A priori knowledge of the signal's characteristics had already influenced the selection of methods for consideration, specially when parametric spectral estimation was to be performed. On the other hand, the selection criterion developed aimed to select the algorithm which most appropriately fitted the input data.

The run time efficiency of an algorithm may be determined by measurement of the execution time of a particular implementation on a specific computer for a certain input data.

However this procedure would result in highly device dependant measures as well as dependence on the data set used.

For this reason asymptotic time complexity measures are preferred. They should account the worst-case time complexity, ignoring constant factors in intermediate calculations.

The efficiency of an algorithm is usually represented in *big-Oh* notation, this is to say O(n). O(n) represents the function of n that gives the maximum, over all inputs of length n, of the number of mathematical operations taken by the algorithm on that input [85].

In order to calculate the complexity of an algorithm running time in big-Oh notation some rules of algorithm analysis have to be considered.

Let P_1 and P_2 represent two program fragments, each with complexity O(f(n)) and O(g(n)) respectively. The complexity of P_1 executed in sequence by P_2 is given by the overall complexity

$$O(\max\{f(n), g(n)\}) \tag{5.81}$$

The above rule is known as the *rule of sums* [85].

If P_1 and P_2 were for instance a loop and its body, the *rule of products* would state that the complexity achieved was

$$O(f(n).g(n)) \tag{5.82}$$

These basic rules are used to analyze more complex structures of the algorithm as illustrated in the following examples:

a) the complexity of a sequence of statements is computed by the sum rule,

b) the cost of an if-statement will be the maximum of the complexities of the if-conditioning statement and the longest conditional sequence of statements, where the sequence of statements has complexity evaluated as in a),

c) the complexity to execute a loop is ruled by (5.82), but the body of the loop, interpreted as a sequence of statements, may by itself include if-statements or other loops, whose complexities are computed following the above rules.

The spectral estimation algorithms used on the study of the PDU signal simulation were also analyzed under the above rules. Table 5.1 presents the resulting expressions of complexity in terms of floating point operations.

FFT/Boxcar(*)	$A=(N/2)\log_2 N$
FFT/Hanning	$4N+A$
AR/Yule-Walker	$3p^2/2+2Np+6N+A$
AR/Burg	$-4p^2+10Np+6N+A$
AR/mod. covar.(*)	$6p^2+Np+A$
ARMA/LSMYWE	$2p^2(N/5-q)+$ $2p(N-q)+N^2+7N+2A$

Table 5.1: Complexities of the spectral estimation methods employed (*[32]).

This table do not refer the complexity of both ARMA Mayne-Firoozan and minimum variance methods for reasons explained in next section.

The complexities of the AR and ARMA methods are represented in Fig. 5.1. as functions of the model order. Each graph illustrates the behaviour of the respective expressions of complexity for four different values of data sequence lengths.

A comparison of the complexities of the AR methods algorithms is presented in Fig. 5.2. for the particular data sequence length of 512 points. The complexity of the AR modified covariance algorithm (Marple's version [32]), is the smallest among the AR methods considered.

From figures 5.1. and 5.2. we may notice that the order of magnitude of the values of the complexity functions is very high, fact that has to be taken into account when acting in conjunction with the values of the benefit function to form (5.79).

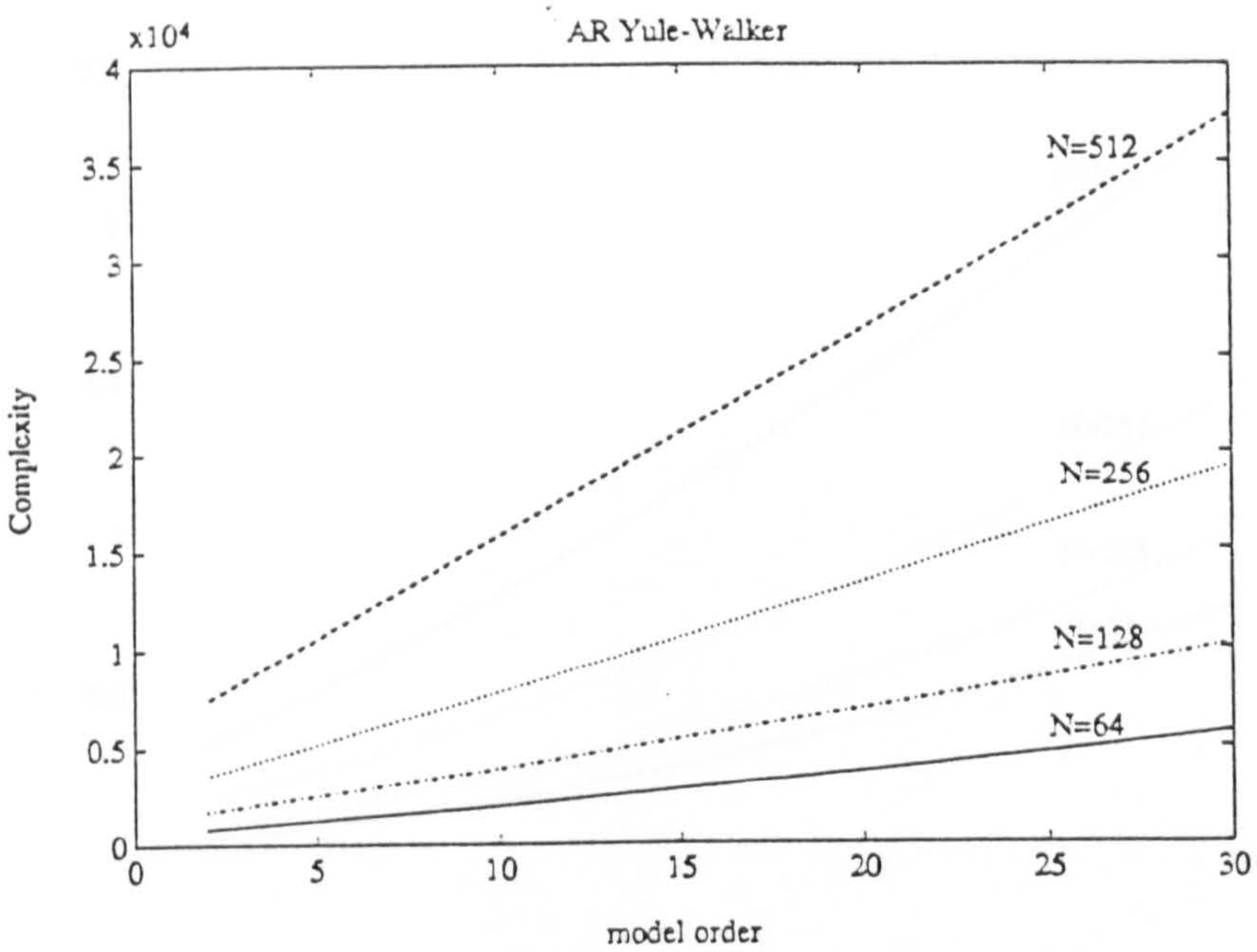


Fig. 5.1(a)

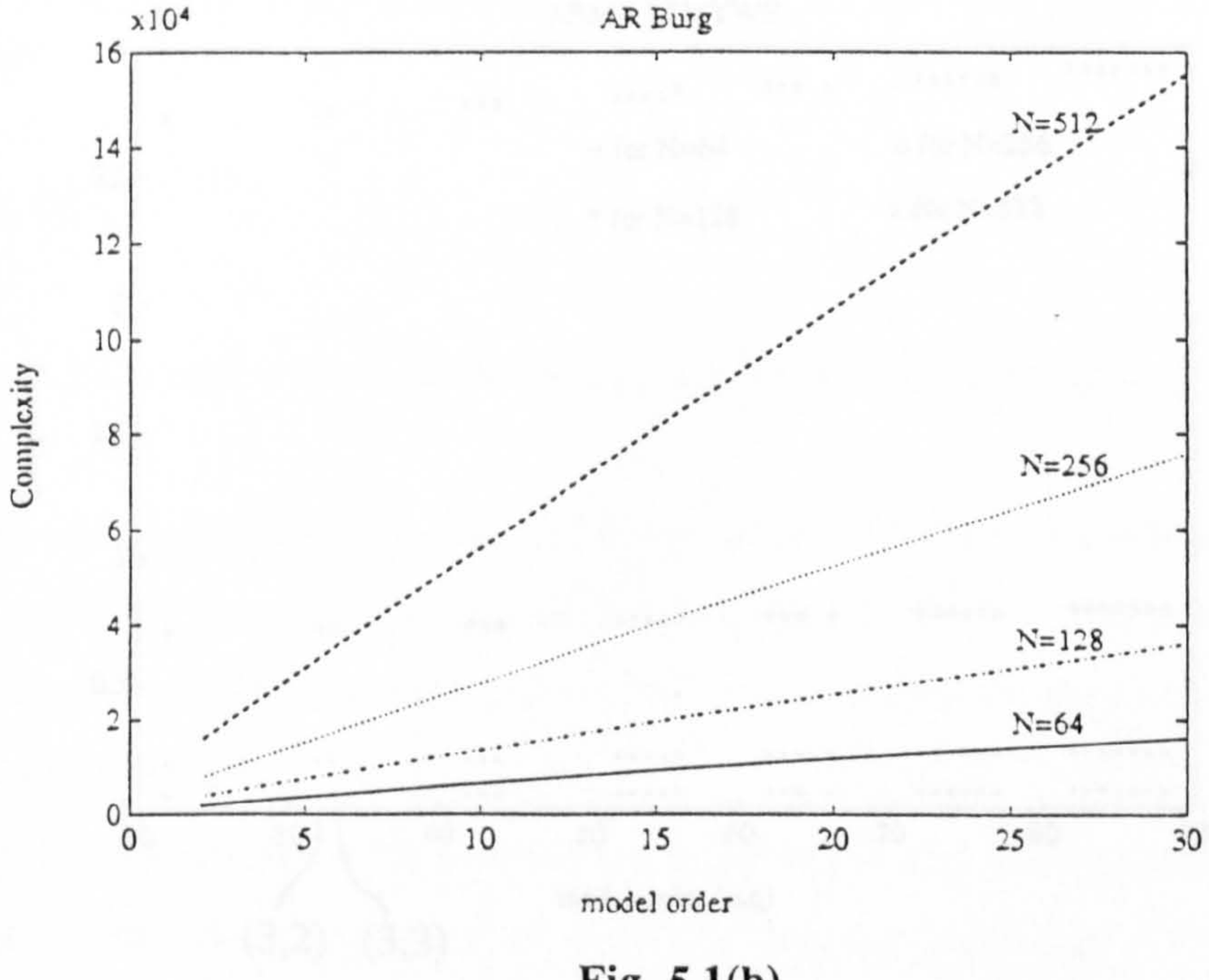


Fig. 5.1(b)

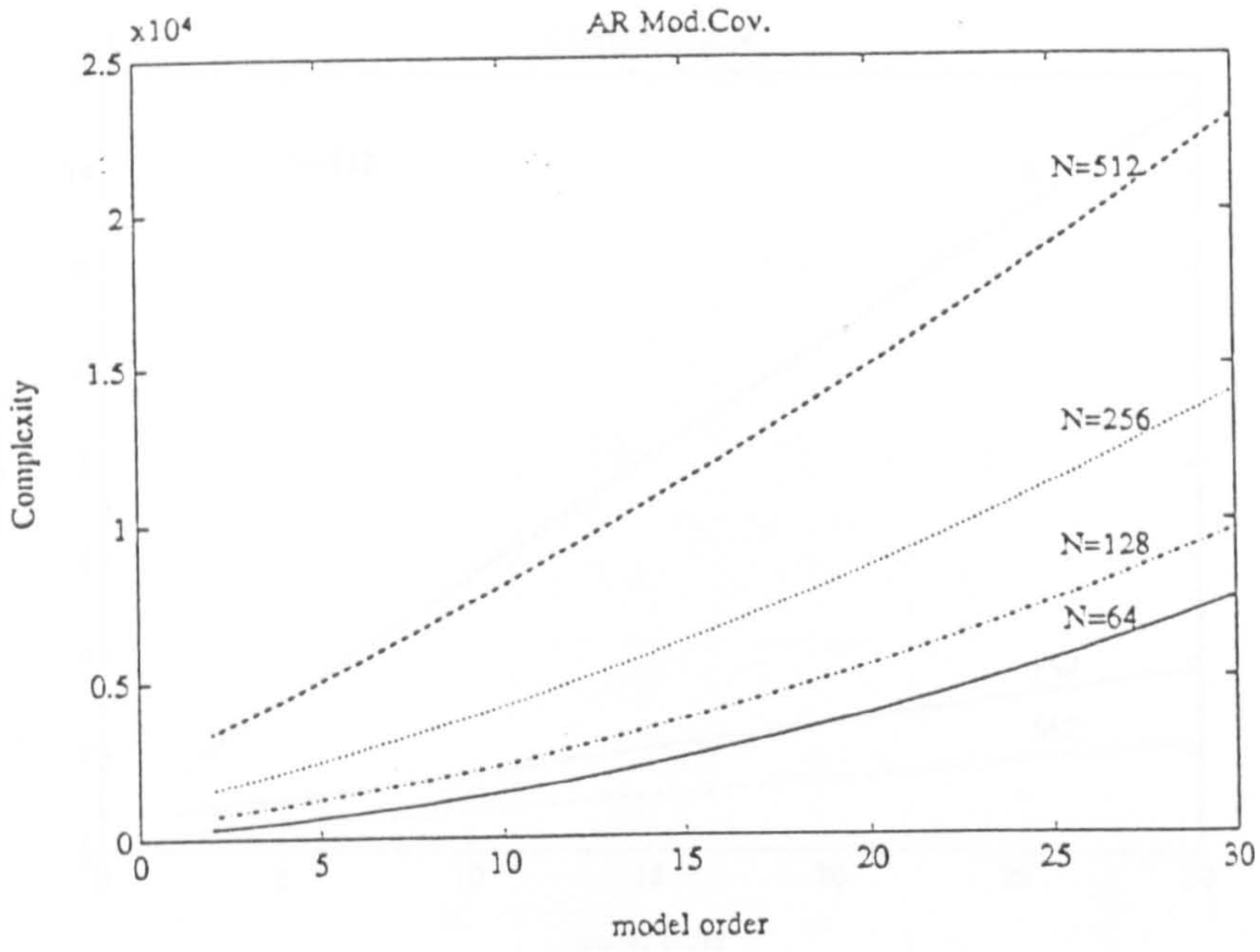


Fig. 5.1(c)

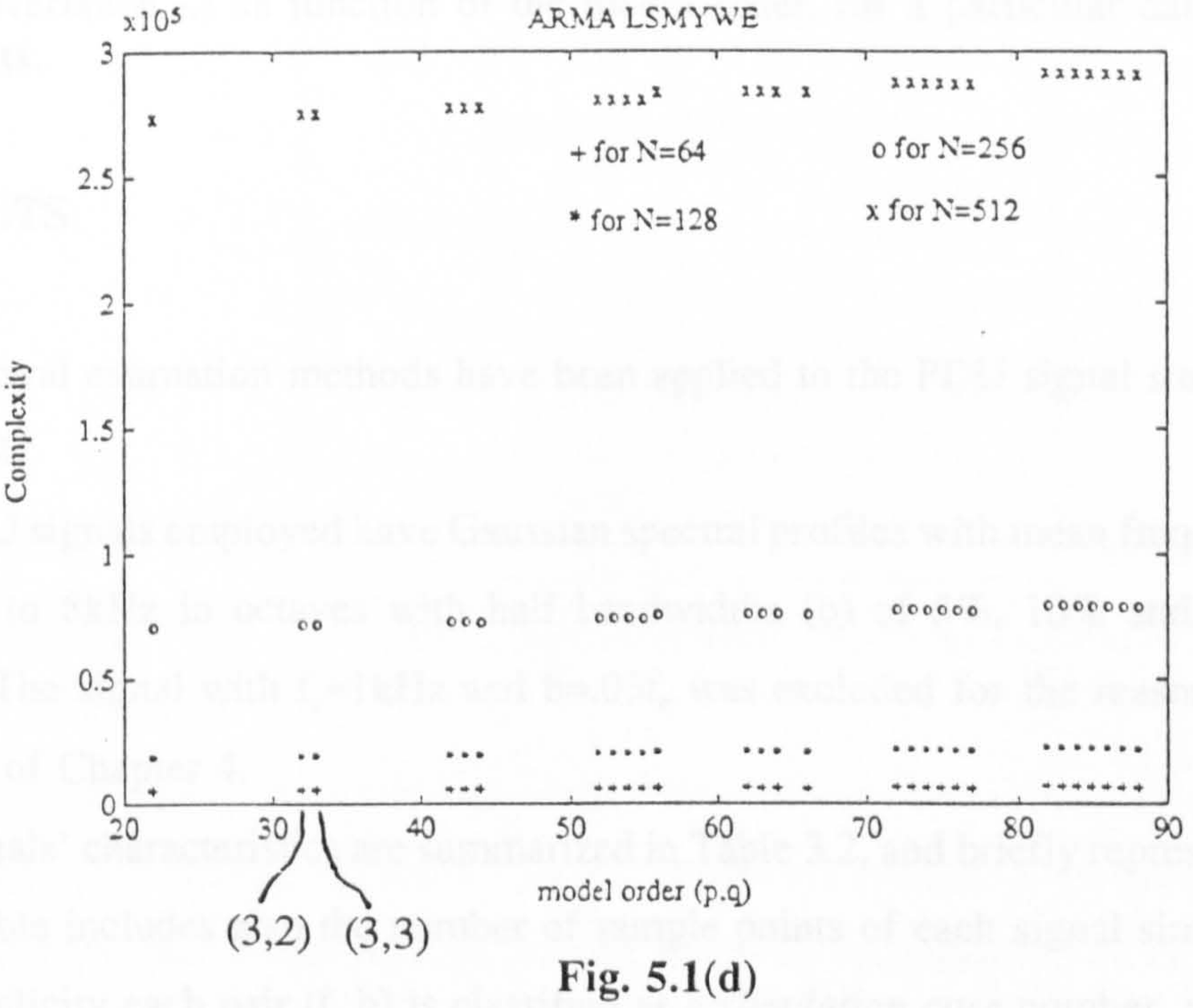


Fig. 5.1(d)

Fig. 5.1: Complexity of the algorithms of AR Yule-Walker (**a**), AR Burg (**b**), AR modified covariance (**c**), and ARMA LSMYWE methods (**d**), as functions of the model order(s) for data sequence lengths of 64, 128, 256, and 512 points.

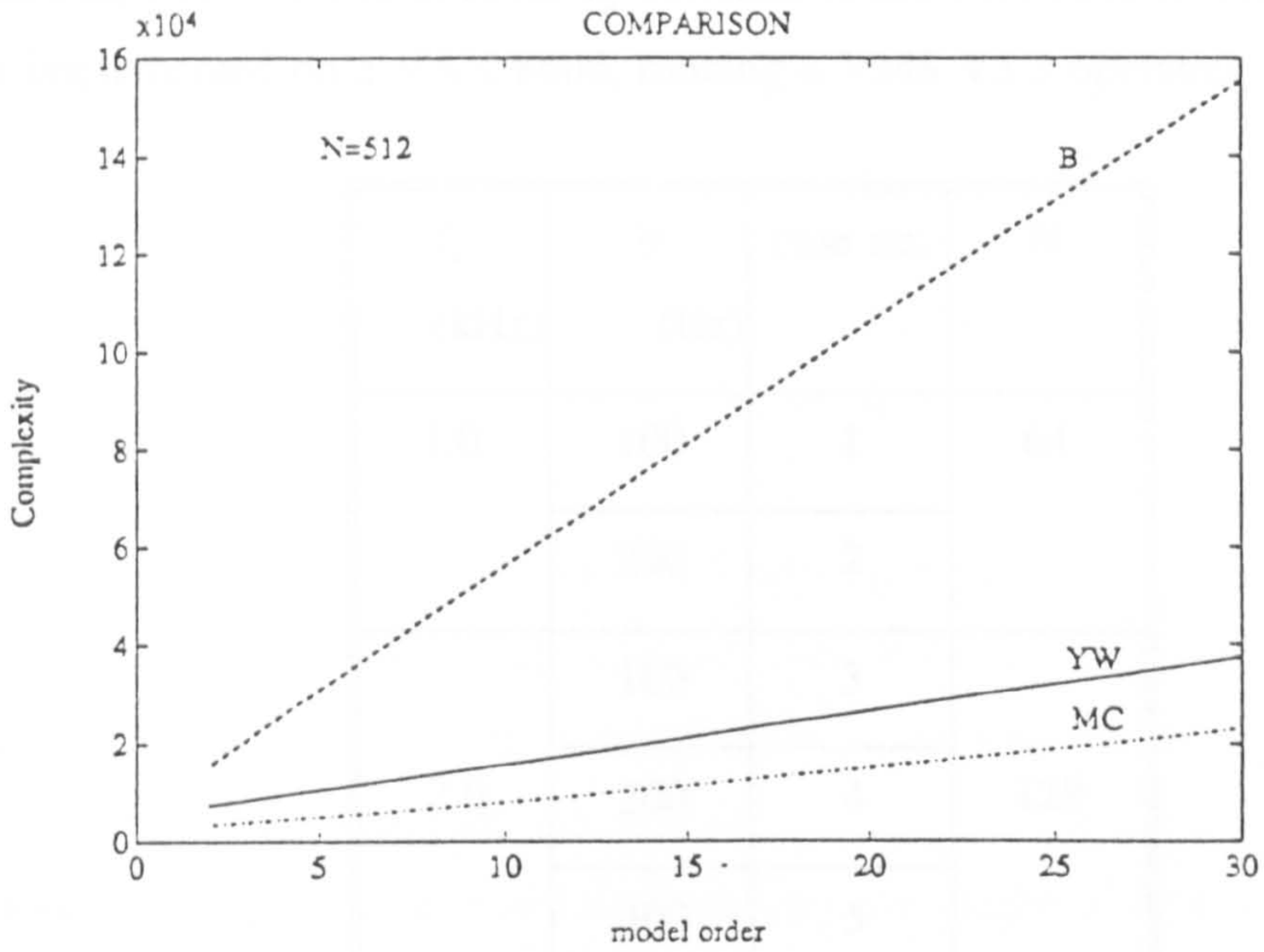


Fig. 5.2: Comparison of complexities of the AR algorithms considered (Yule-Walker -, Burg --, modified covariance -.) as function of the model order, for a particular data sequence length, N=512 points.

5.4. RESULTS

Several spectral estimation methods have been applied to the PDU signal simulations described in Chapter 4.

The PDU signals employed have Gaussian spectral profiles with mean frequencies (f_c) varying from 1kHz to 8kHz in octaves with half bandwidths (b) of 5%, 10% and 20% of the mean frequency. The signal with f_c=1kHz and b=.05f_c was excluded for the reasons explained in the last section of Chapter 4.

The signals' characteristics are summarized in Table 3.2, and briefly represented now in Table 5.2. This table includes also the number of sample points of each signal simulation.

For simplicity each pair (f_c,b) is classified as a *simulation case* number, which from now on will be used to reference each type of signal simulation.

These simulated signals constitute the input of the software program *ANENS* written in FORTRAN 77 [69]. This program makes use of routines from the NAG software package [70],

and presents graphics facilities enabled by routines of the GINOGRAF software package [71]. It has been implemented on a VAX 8600, running a VMS V5.3 operating system.

f_c (kHz)	b (Hz)	case no.	N
1.0	100	1	64
	200	2	
2.0	100	3	128
	200	4	
	400	5	
4.0	200	6	256
	400	7	
	800	8	
8.0	400	9	512
	800	10	
	1600	11	

Table 5.2: PDU signal simulations employed in spectral estimation analysis.

ANENS, whose constituent software block structures have been already mentioned in Chapter 4, was developed to include far more spectral estimation methods than the classical ones.

ANENS' statistical routine was maintained, enabling the computation of percentage relative bias and standard deviation of both estimators mean frequency and half bandwidth of the ensemble averaged spectrum. These statistical measures are evaluated by (4.24) for the mean frequency estimator, and similarly for the half bandwidth estimator, as stated in last Chapter. For simplicity, these percentage relative bias, standard deviation, and mean squared error measures will also be shortly called by bias, standard deviation, and mean squared error, respectively.

Each ensemble averaged spectrum is obtained by considering 150 simulations of the PDU signal.

ANENS produces graphical outputs of the normalized ensemble average spectrum and theoretical spectrum. Numerical outputs of the statistical measurements of performance are presented. Fig. 5.3 is an example of the output produced by this program, for case 4 when the spectrum of the simulated signal is obtained by an AR(6) model estimated by Yule-Walker method.

The latest version of *ANENS* included the following spectral estimation methods: FFT with Boxcar window, FFT with Hanning window, AR Yule-Walker, AR Burg, AR modified covariance, ARMA Mayne-Firoozan, ARMA least squares modified Yule-Walker equations (LSMYWE), and the minimum variance method.

The ARMA Mayne-Firoozan and minimum variance methods were excluded from further consideration at this stage.

The Mayne-Firoozan method, implemented with a modified covariance method to estimate the large order AR filter, produced forced exiting of the software program when estimating the signal simulation case 1. This result agreed with the recognized limitation of this method, this is, the production of unstable MA filters [17].

The minimum variance method was applied to the signal simulation cases 1 and 6, presenting high biases (specially for the half bandwidth estimator), therefore was also excluded.

Each of these spectral estimation methods was applied to each simulation case mentioned in Table 5.2. When the parametric methods were considered, increasing values of model order were tried till the equations to be solved became ill-conditioned, or the arbitrary limit of p=30 was attained, as in the case of the Yule-Walker method.

The statistical measurements obtained with the AR Yule-Walker and Burg methods were processed by a program written in MATLAB [78] to allow graphical presentation of the variation of bias and standard deviation of both estimators mean frequency and half bandwidth with the model order.

```
spectral estimator                  : AR estimator:Yule-Walker
spectral function                   :  Gaussian
size of ensemble                    =    150
mean frequency                      =  2000.000 Hz
half bandwidth ("sigma")            =   200.000 Hz
maximum frequency                   =  2858.386 Hz
data sequence duration               =0.0100 S
sampling rate                       = 12800.0 Hz

RESULTS***
CENTER FREQUENCY                BIAS                STD
AECF= 2007.217041 (f00)
ASCF= 2008.985962 f̂00             7.217               56.929
                                0.361%              2.846%
SDAECF=   56.928871 st(fi0)

 HALF BANDWIDTH
AEBW=  307.005310 (σ00)
ASBW=  308.725800 σ̂00           107.005              64.305
                               53.503%             32.153%
SDAEBW=   64.305193 st(σi0)

 Press <RET>
```

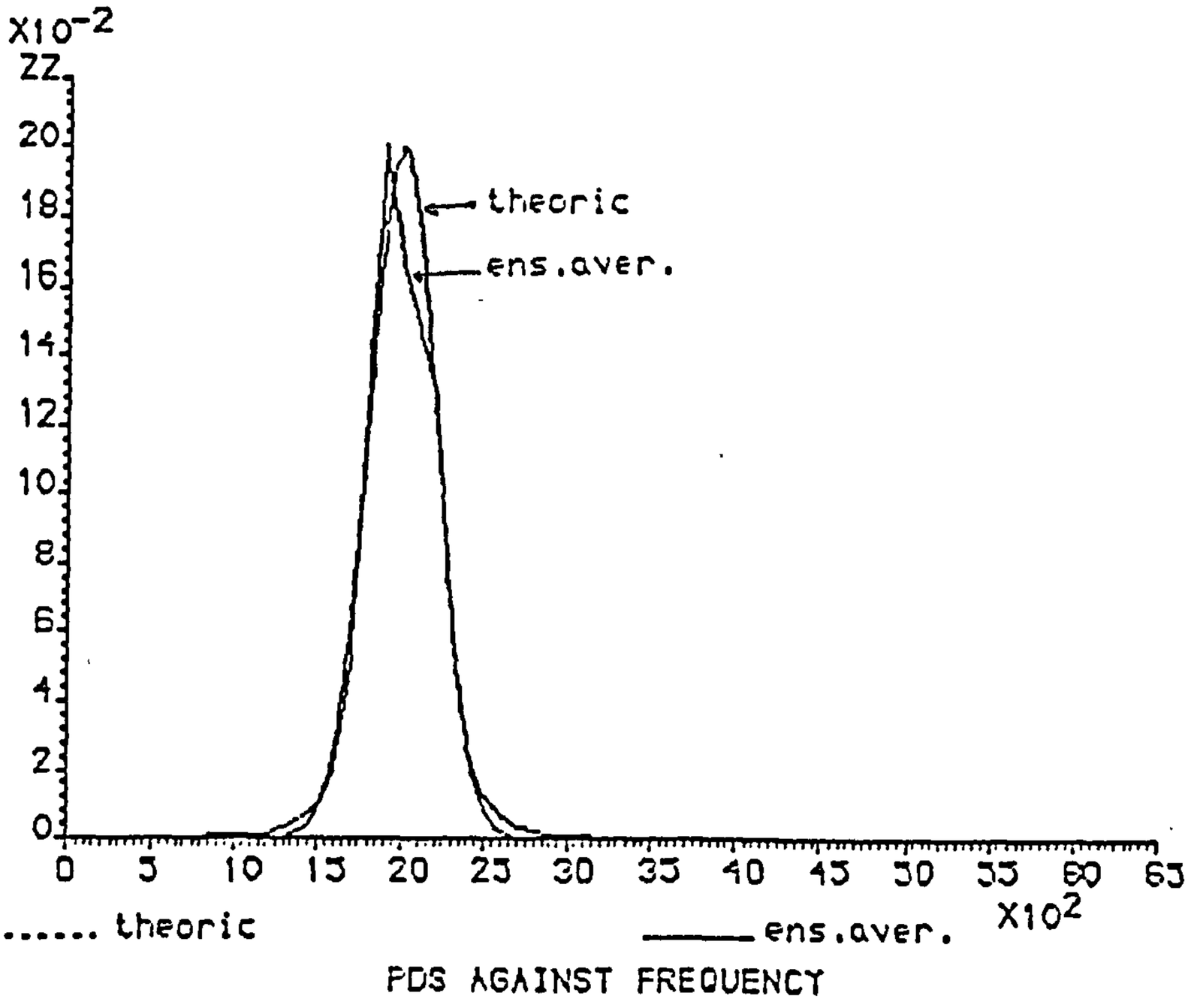


Fig. 5.3: Signal's characteristics, statistical results computed from the estimated spectrum, and curves of estimated against theoretical spectra are obtained. Case 4, using AR Yule-Walker with p=6 as spectral estimator. The vertical and horizontal axis represent the normalized (see section 4.4) PSD, and the frequency in Hz.

Examples of the graphs obtained with the Yule-Walker method are presented in Figures 5.4 (a) to (c) with different values of mean frequencies, each one with half bandwidths of 5%, 10% and 20% of the respective mean frequency, to allow a general view of the behaviour of the statistical measurements with the model order.

From these figures we may notice that the range of values of the bias and standard deviation are of a different order of magnitude for the estimator mean frequency, while for the estimator half bandwidth, bias and standard deviation may or may not be of the same order of magnitude depending on the simulation case.

Another feature occurring for almost all the graphs is the tendency of the curve to show very small or even negligible variations in the statistic after a certain value of model order. However there are some exceptions; for example in the case of mean frequency standard deviation in Fig. 5.4 (b), and also in the cases of bias and standard deviation of mean frequency estimator of Fig. 5.4 (a), for which the curves increase continuously after p=10.

Similar curves were obtained for the other parametric methods, revealing a difficulty of determining the order producing the best statistical performance for either mean frequency or half bandwidth estimators based solely on minimum error. Thus it is even more difficult to empirically select a model order which corresponds to a best performance over both spectral estimators and all simulation cases.

Nevertheless, a first comparative study (based on subjective judgements) of the behaviour of the parametric spectral estimation methods for each signal simulation case was performed. The criterion used to select the model orders considered firstly the individual performance of the statistics bias and standard deviation of each spectral estimator. The lowest model order of the range of model orders over which these were negligible variations of the statistics was indicated. Then, the model order that seemed to represent the best compromise between the four model orders pointed out and their respective statistical behaviour was selected.

Table 5.3 summarizes the empirically selected model orders for the parametric methods applied to the PDU signal simulations, using the above described procedure.

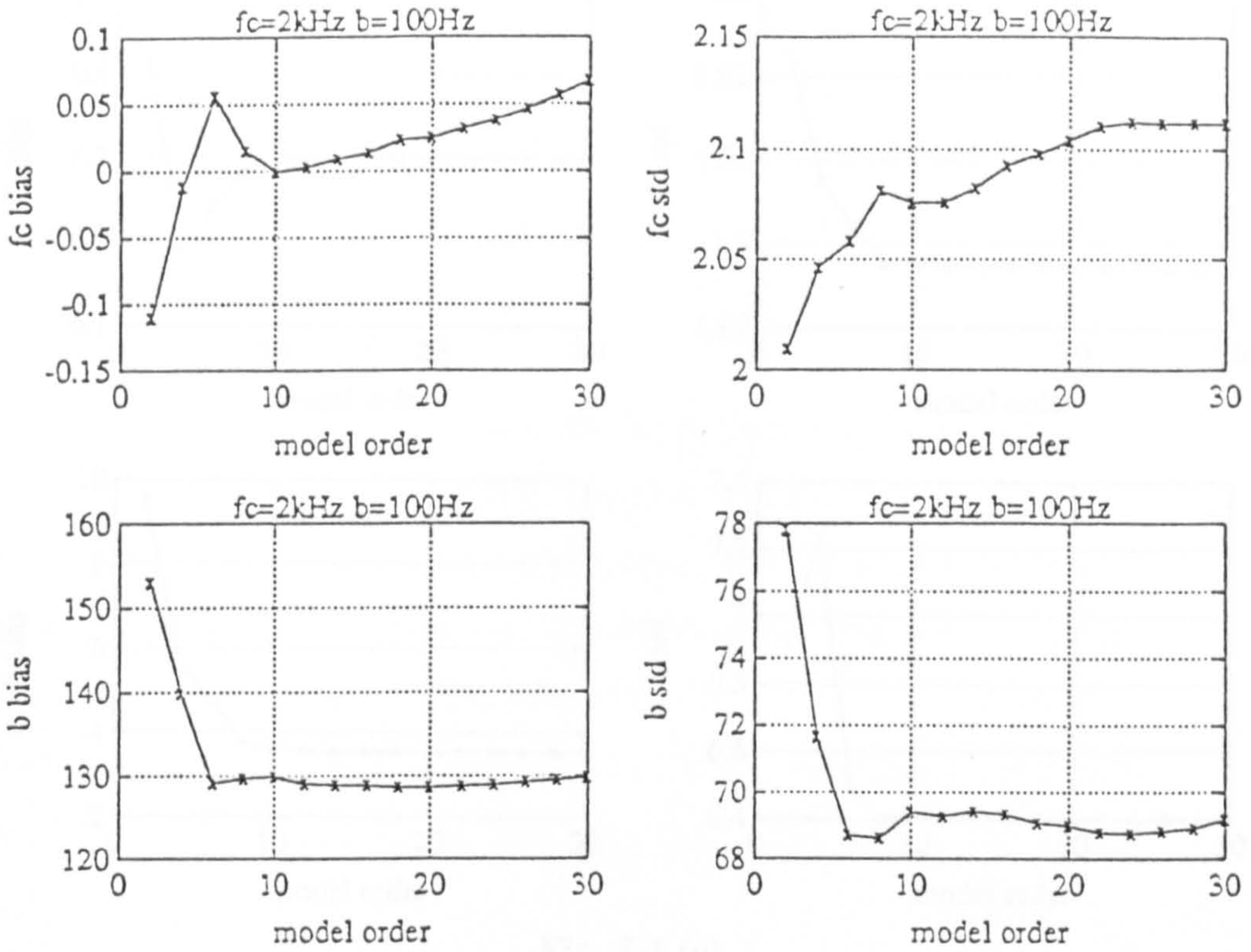


Fig. 5.4 (a)

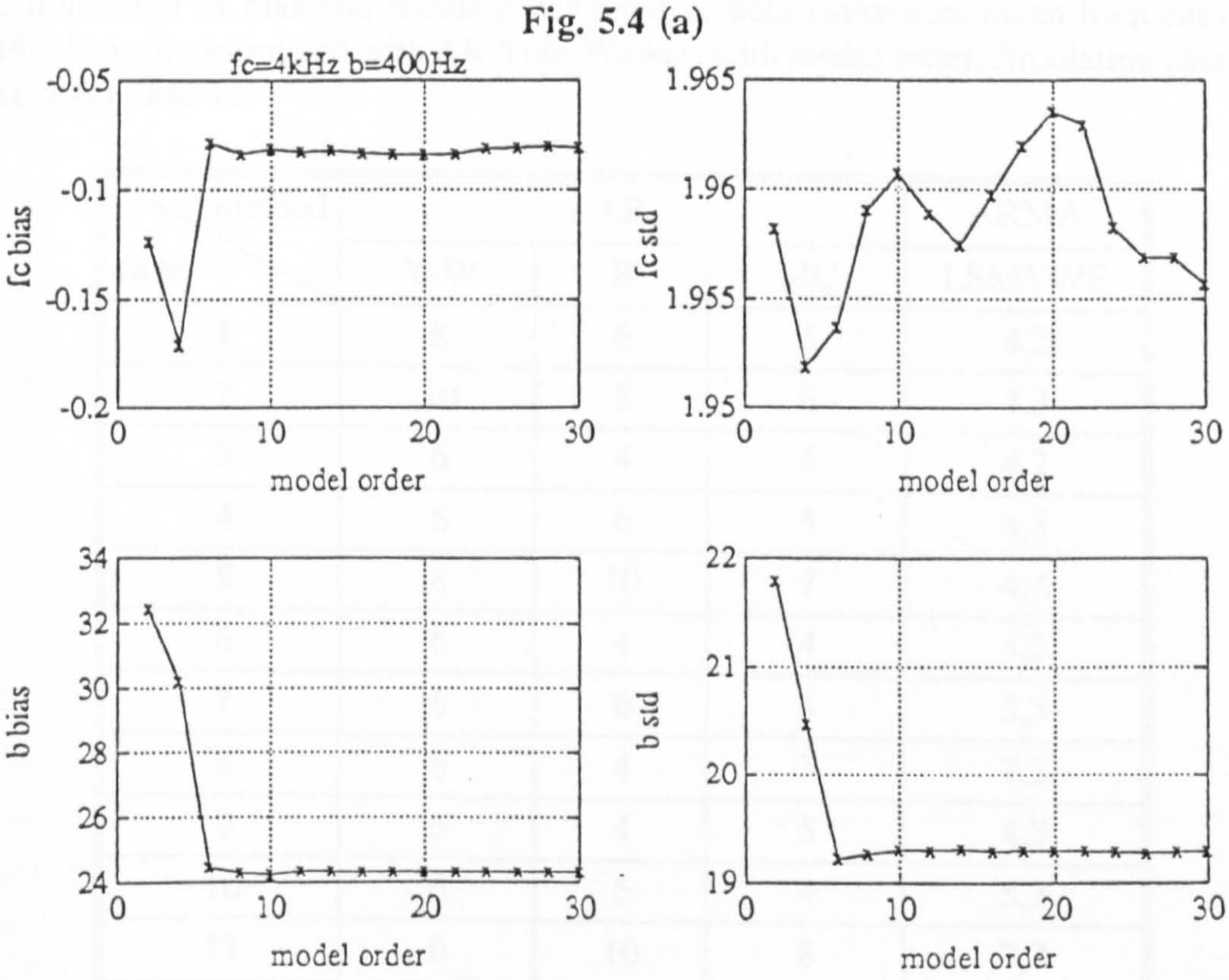


Fig. 5.4 (b)

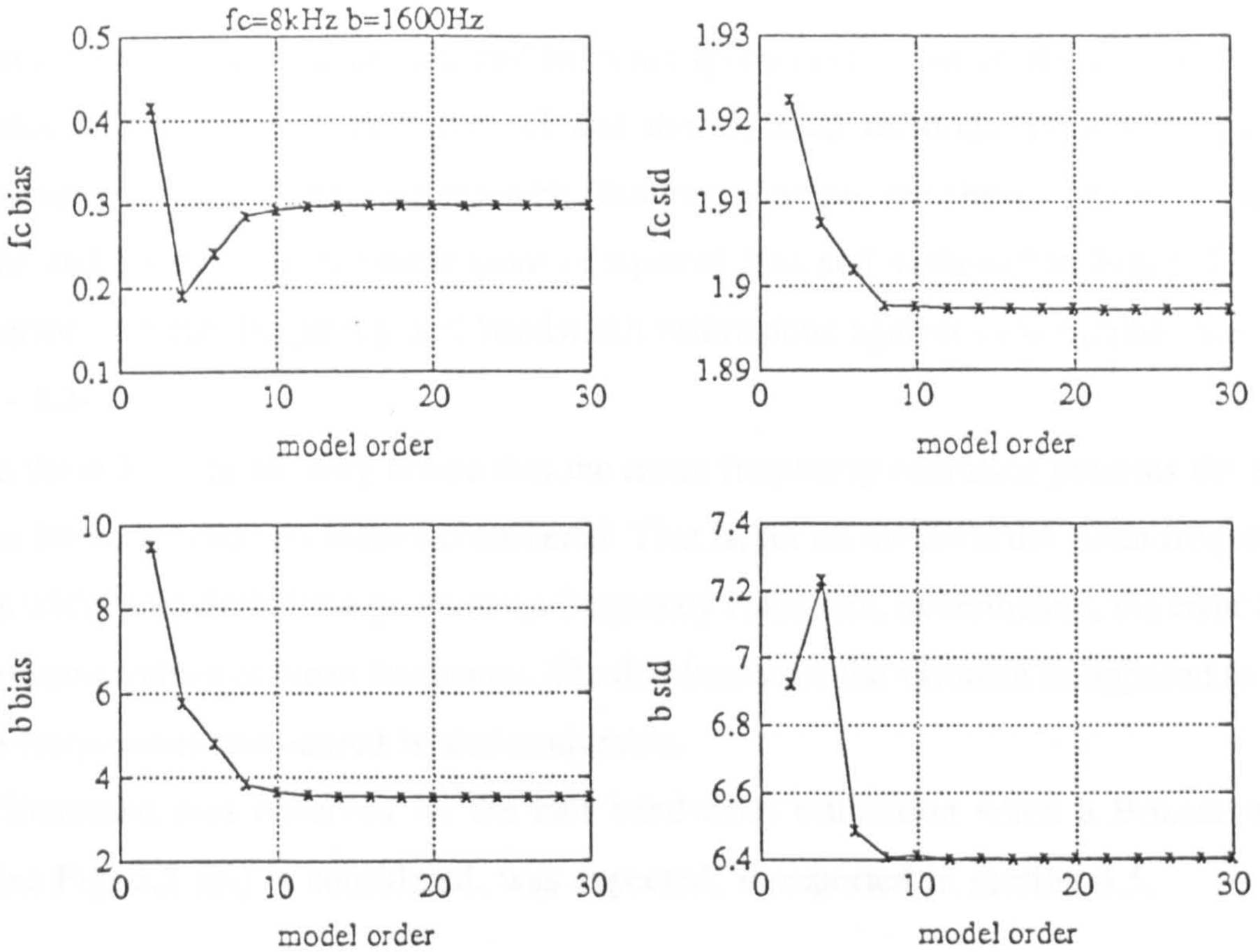


Fig. 5.4 (c)

Fig. 5.4: Behaviour of bias and standard deviation of both estimators mean frequency and half bandwidth of spectra estimated with AR Yule-Walker, with model order;Simulation cases:(**a**) case 4,(**b**) case 7,(**c**) case 11.

method / case	AR			ARMA
	Y-W	B	MC	LSMYWE
1	8	6	5	4,2
2	10	8	6	4,3
3	6	4	4	4,2
4	6	6	5	4,3
5	8	10	7	4,4
6	6	4	4	4,2
7	6	6	4	5,5
8	6	4	7	7,3
9	6	4	5	4,3
10	6	6	4	5,3
11	6	10	8	7,7

Table 5.3: Empirically chosen model orders for the parametric methods.

The statistical measurements obtained from the spectra estimated by the parametric methods with model orders specified in Table 5.3 and the classical methods, Fourier transform with Boxcar window and Fourier transform with Hanning window, are shown below in the form of a variance and a mean squared error (sum of squared bias and variance) in Fig. 5.5 - 5.12.

The error in mean frequency and bandwidth estimations against case number are shown in Fig. 5.5 - 5.10.

From these Figures we may notice that the mean frequency estimator presents the same sort of pattern for all spectral estimators considered. That is, for all methods the mean frequency error increases with bandwidth, for a given mean frequency value, but, nevertheless, the error decreases with increasing values of mean frequency. The dominance of the variance as opposed to bias over all mean frequencies considered is also noticeable.

The increased bias observed on the half bandwidth estimation when a Boxcar windowed signal (see Fig. 5.5 (b)) is considered, was expected, as reported in section 4.5.

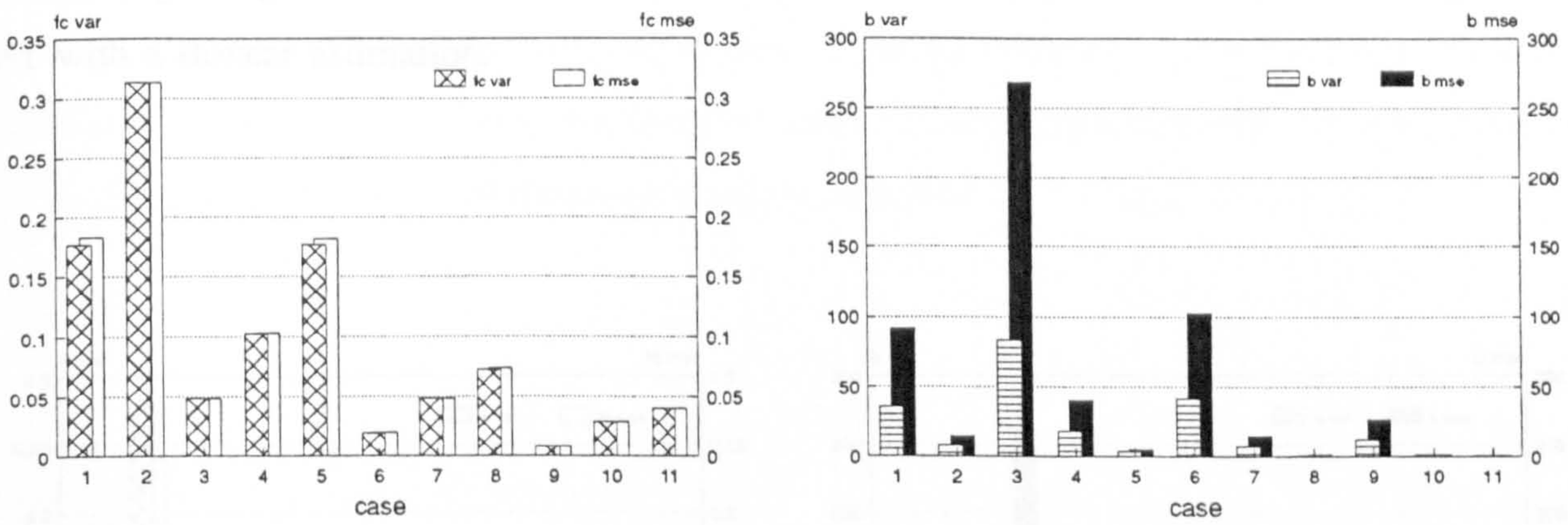
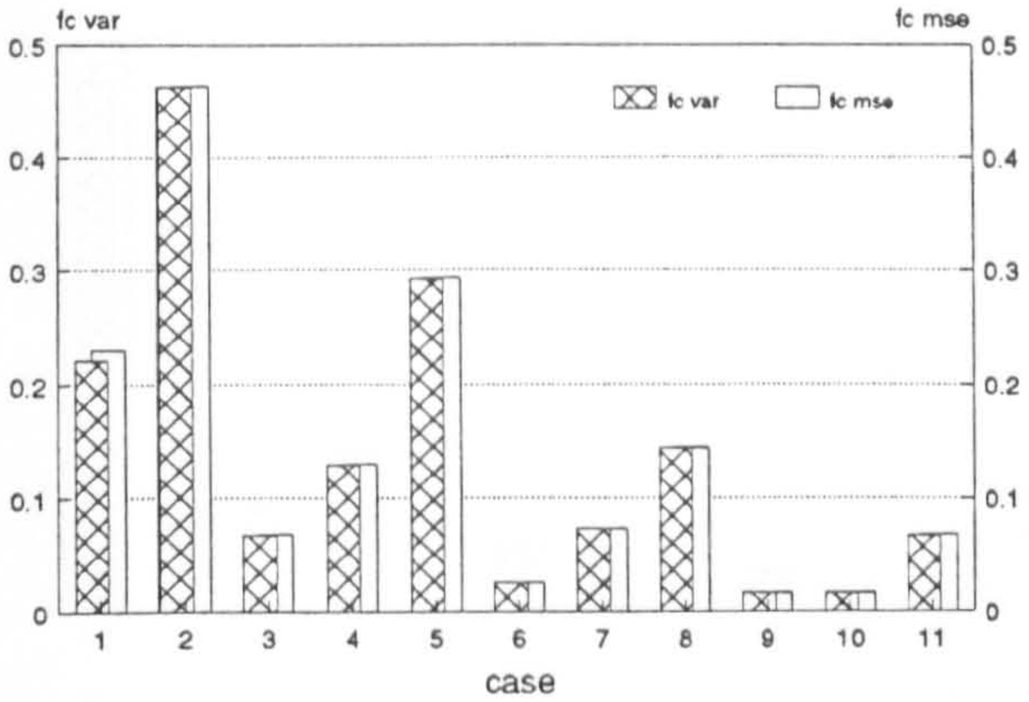
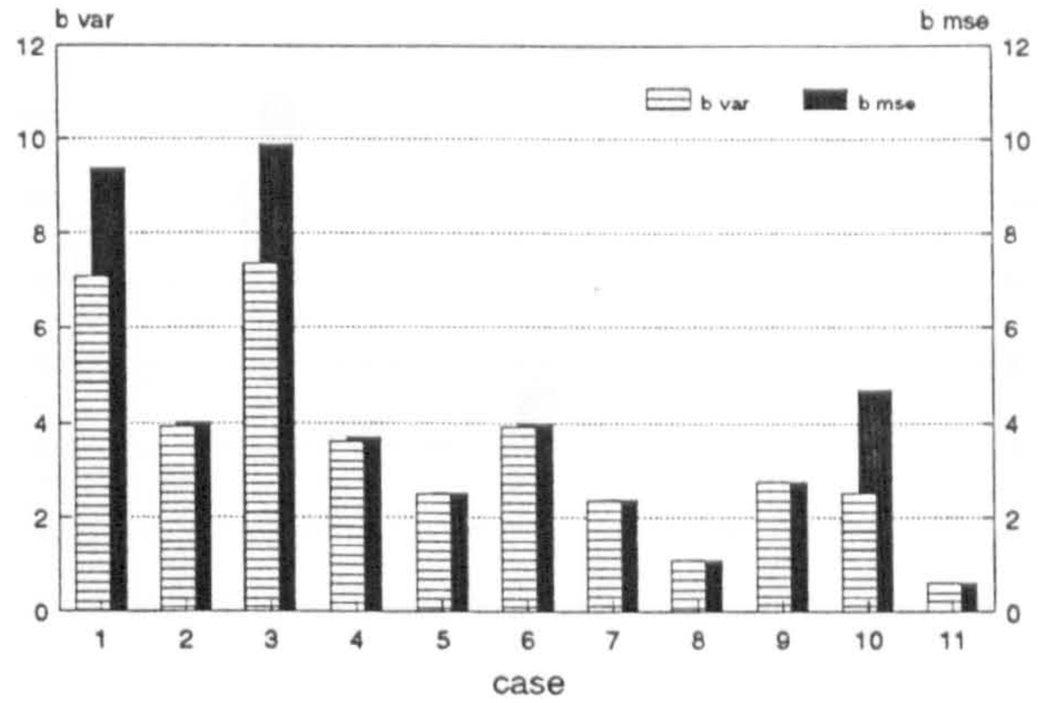


Fig. 5.5: Statistical performance of FFT with Boxcar window spectral estimator.Variance (var) and mean squared error (mse) of estimators: (**a**) mean frequency, (**b**) half bandwidth.

If instead a Hanning window is used (Fig. 5.6), the contribution of bias to the half bandwidth mean squared error is much lower, being generally noticeable when the simulation case has small half bandwidths (5% of the respective mean frequency), as may be seen in Fig. 5.6 (b).



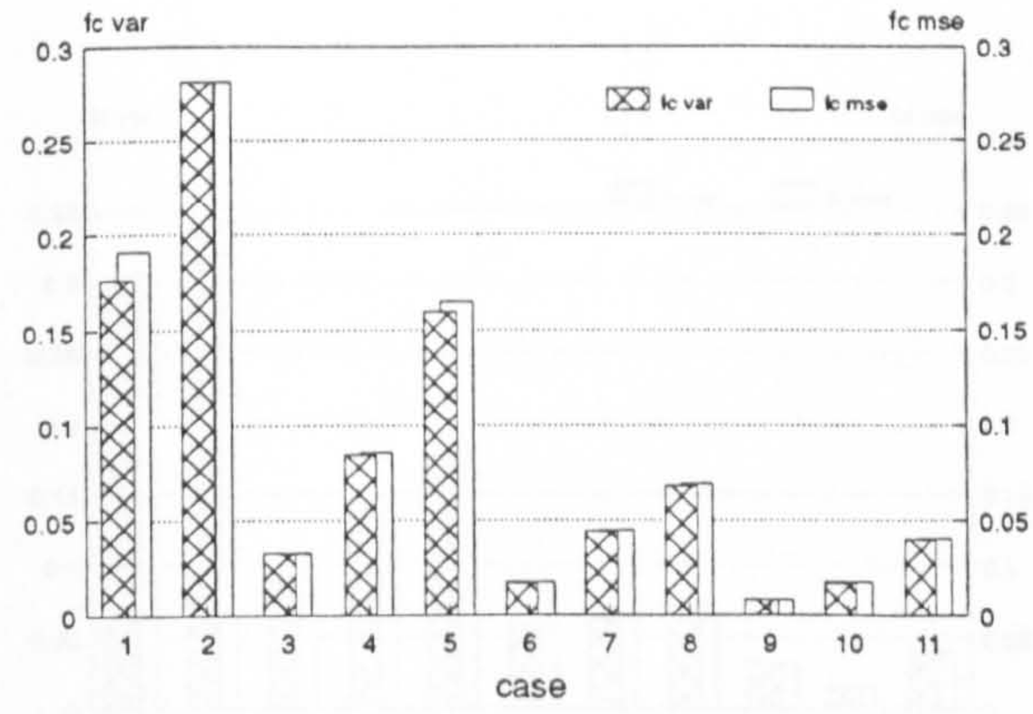
(a)



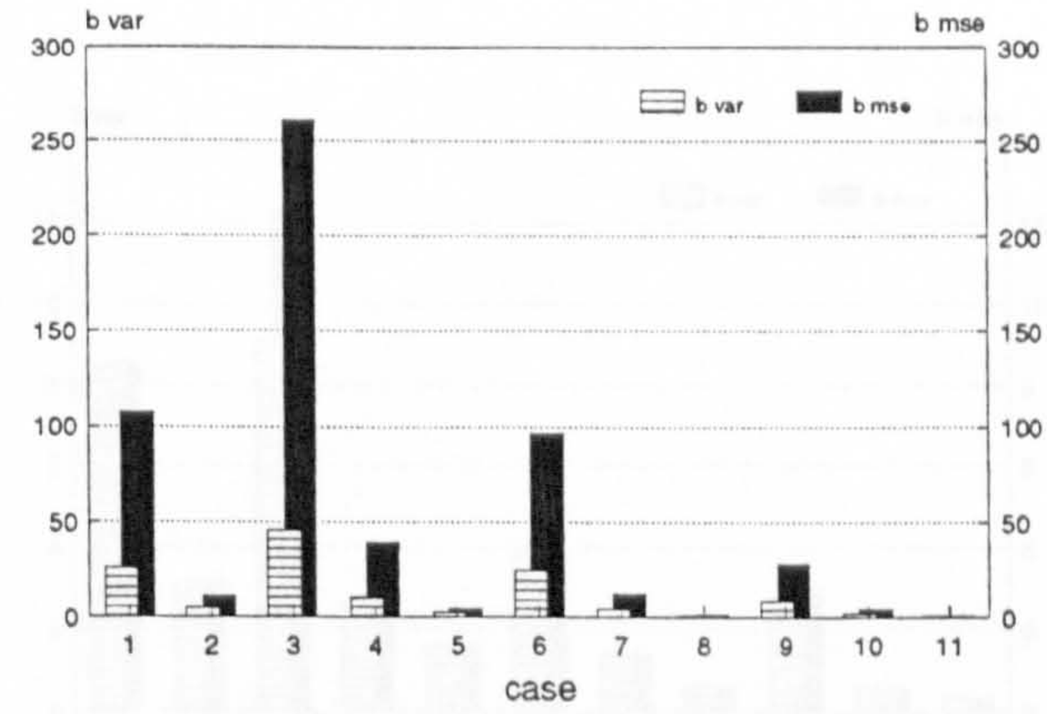
(b)

Fig. 5.6: Statistical performance of FFT with Hanning window spectral estimator.Variance (var) and mean squared error (mse) of estimators: **(a)** mean frequency, **(b)** half bandwidth.

Looking to figure 5.7 we note that the AR Yule-Walker estimation behaves similarly to the FFT with a Boxcar estimation.

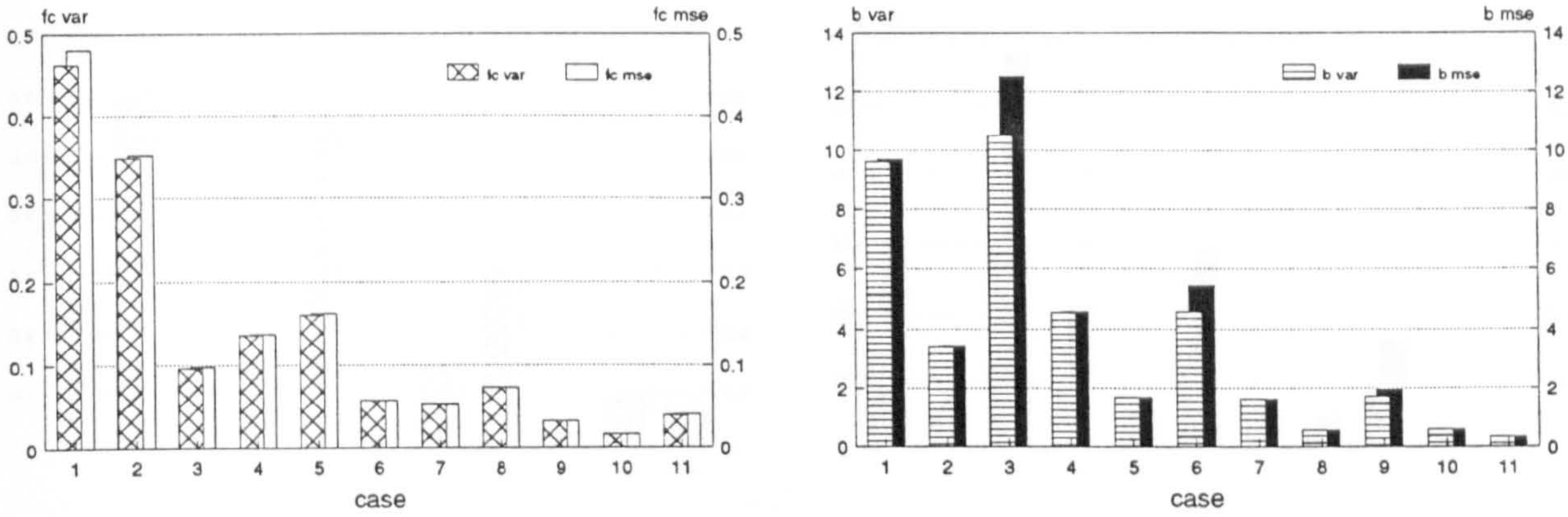


(a)



(b)

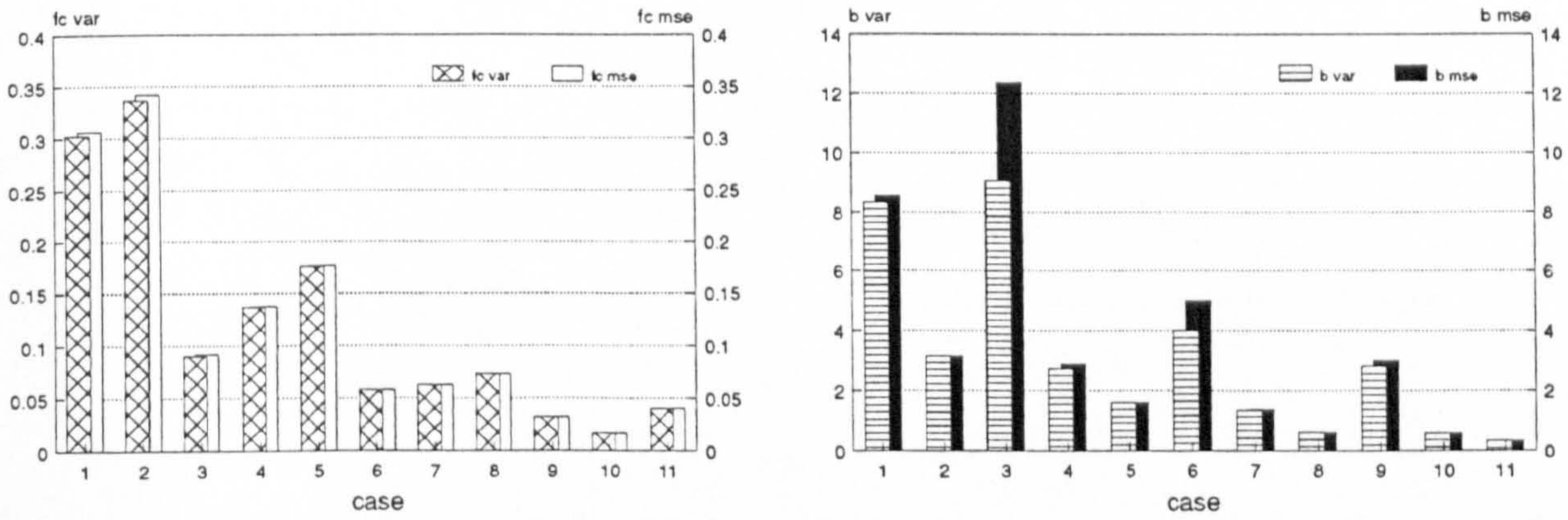
Fig. 5.7: Statistical performance of AR Yule-Walker spectral estimator.Variance (var) and mean squared error (mse) of estimators: **(a)** mean frequency, **(b)** half bandwidth.



(a) **(b)**

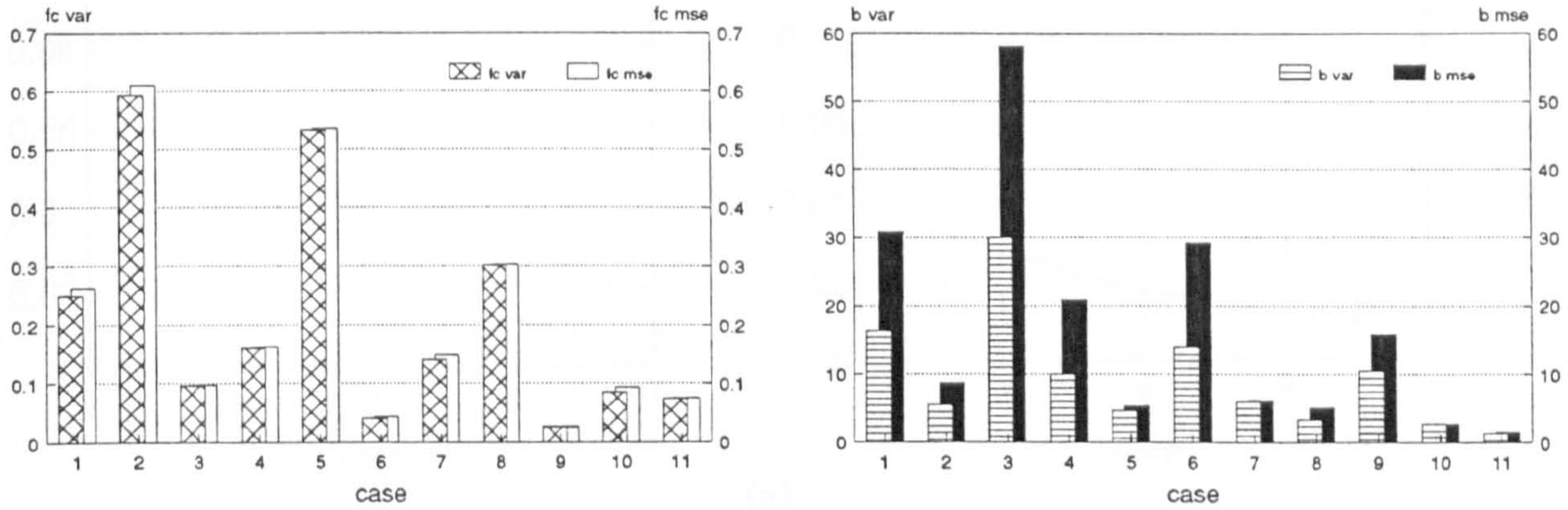
Fig. 5.8: Statistical performance of AR Burg spectral estimator.Variance (var) and mean squared error (mse) of estimators: (**a**) mean frequency, (**b**) half bandwidth.

For spectra estimated by AR Burg and AR modified covariance methods, we may see that the contribution of bias to the mean squared error for both parameters in study (see figures 5.8 and 5.9) becomes less important than the observed in previous figures. Also, the range of values assumed by the mean squared error for these two methods are comparable with the one presented in Fig. 5.6 (b) by the FFT with Hanning window estimator.



(a) **(b)**

Fig. 5.9: Statistical performance of AR modified covariance spectral estimator.Variance (var) and mean squared error (mse) of estimators: (**a**) mean frequency, (**b**) half bandwidth.



(a) **(b)**

Fig. 5.10: Statistical performance of ARMA LSMYWE spectral estimator.Variance (var) and mean squared error (mse) of estimators: **(a)** mean frequency, **(b)** half bandwidth.

The ARMA LSMYWE method presents the highest values of mean frequency and mean squared error of this comparison, demonstrating high sensitivity to variations in bandwidth as shown in Fig. 5.10. Half bandwidth mean squared error has also a strong bias component.

The variation of mean squared error of the centre frequency and half bandwidth estimators against signal case, for the three different fractional bandwidths, are shown in Figures 5.11 and 5.12.

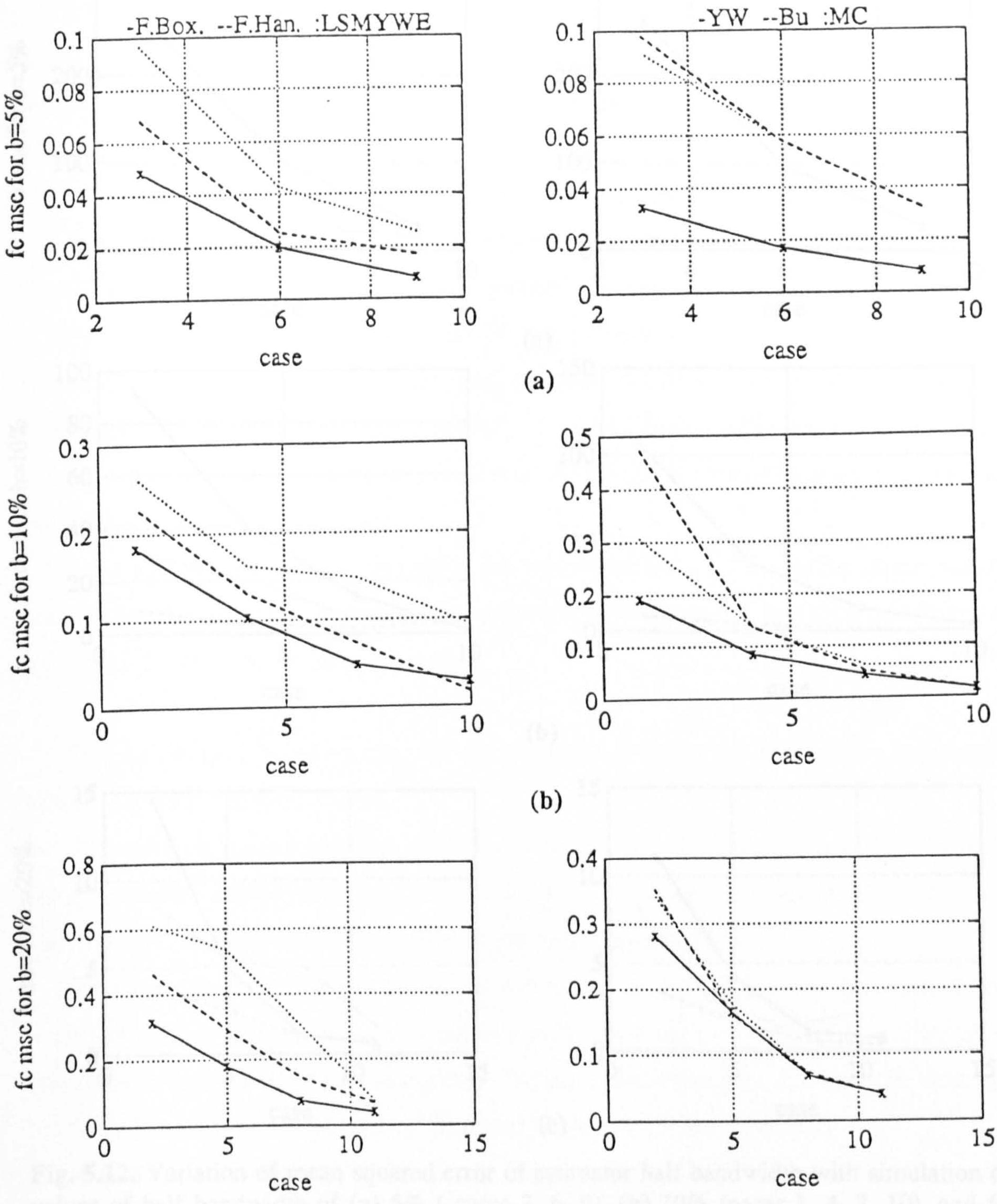


Fig. 5.11: Variation of mean squared error of estimator mean frequency with simulation case for values of half bandwidth of **(a)** 5% (cases 3, 6, 9), **(b)** 10% (cases 1, 4, 7, 10), and **(c)** 20% (cases 2, 5, 8, 11) of respective mean frequency.

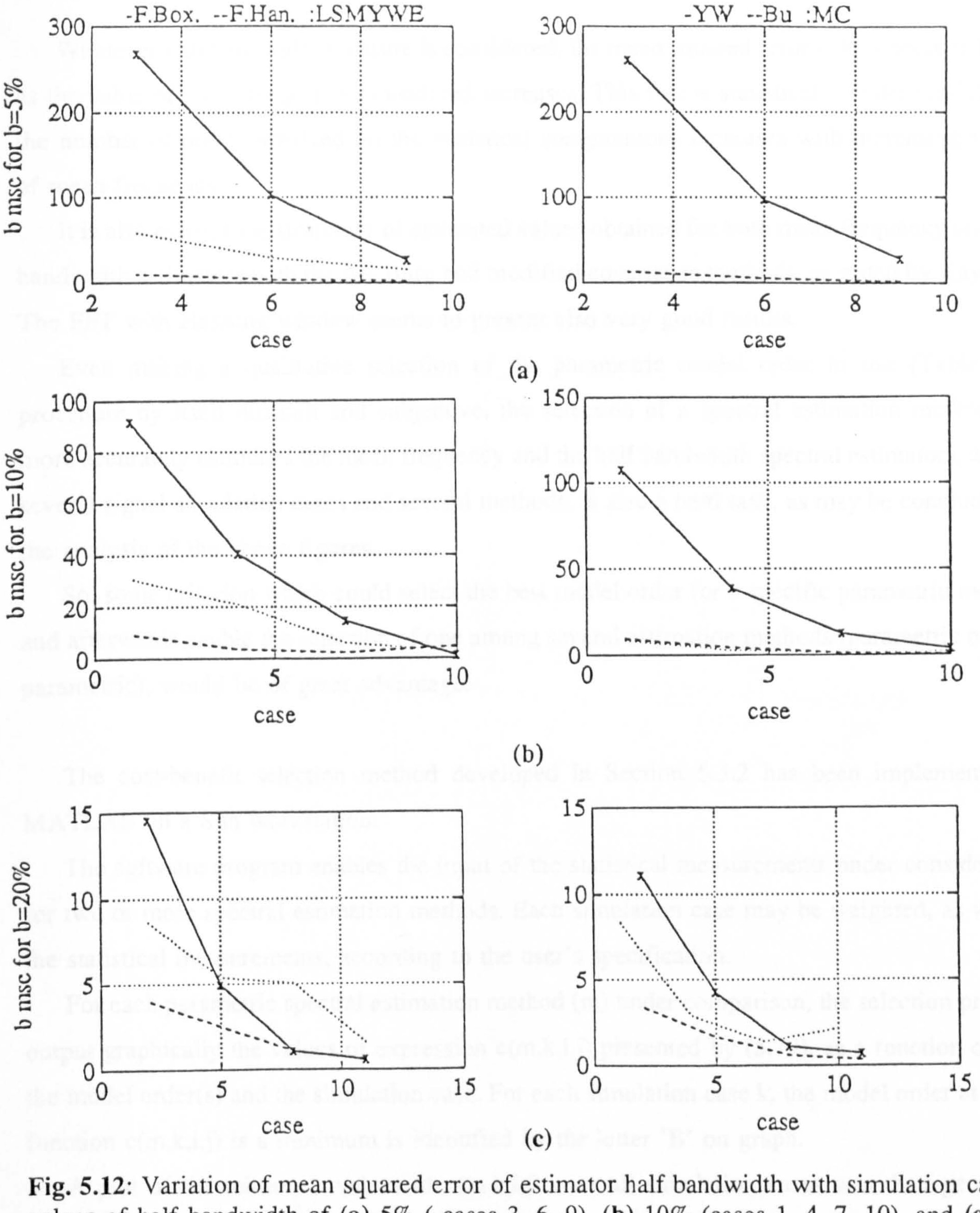


Fig. 5.12: Variation of mean squared error of estimator half bandwidth with simulation case for values of half bandwidth of (**a**) 5% (cases 3, 6, 9), (**b**) 10% (cases 1, 4, 7, 10), and (**c**) 20% (cases 2, 5, 8, 11) of respective mean frequency.

From these figures we may notice that when the mean frequency estimator is considered, the larger the bandwidth of the signal, the larger the values of the mean squared error becomes. Conversely, as the bandwidth increases the half bandwidth parameter becomes better estimated.

Whatever half bandwidth measure is considered, the mean squared error values become better as the value of mean frequency considered increases. This fact is statistically understandable as the number of points involved on the statistical computations increases with increasing values of mean frequency.

It is also evident the similarity of estimated values obtained for both mean frequency and half bandwidth estimators with the AR Burg and modified covariance methods, as stated by Kay [17]. The FFT with Hanning window seems to present also very good results.

Even making a qualitative selection of the parametric model order to use (Table 5.3), procedure by itself difficult and subjective, the selection of a spectral estimation method that more accurately estimates the mean frequency and the half bandwidth spectral estimators, among several signal simulation cases and several methods, is also a hard task, as may be concluded by the analysis of the above figures.

So, some criterion which could select the best model order for a specific parametric method, and afterwards enable the selection of one among several estimation methods (parametric or non-parametric), would be of great advantage.

The cost-benefit selection method developed in Section 5.3.2 has been implemented in MATLAB on a Sun workstation.

The software program enables the input of the statistical measurements under consideration for two or more spectral estimation methods. Each simulation case may be weighted, as well as the statistical measurements, according to the user's specification.

For each parametric spectral estimation method (m) under comparison, the selection program output graphically the values of expression c(m,k,i,j) presented by (5.79), as a function of both the model order(s) and the simulation case. For each simulation case k, the model order at which function c(m,k,i,j) is a minimum is identified by the letter 'B' on graph.

A plot of the minimum values of c(m,k,i,j) over all simulation cases was then generated, followed by calculation and display of $c_o(m)$ for all the estimators under consideration and identification of the optimum estimator m given by $\min\{c_o(m)\}$.

In this example section all simulation signals were given equal weight ($w_k=1$ for k=1,...,11).

In a first case study, equal importance was assigned to the accuracy of mean frequency and bandwidth - reflecting an equal interest in blood velocity estimation and flow disturbance detection - by putting $w_{bf}=w_{sf}=w_{bb}=w_{sb}=1$. Since the bandwidth estimation error is always very

much greater than that for mean frequency estimation this means that the selection of a spectral estimation method was mainly determined by the behaviour of the bandwidth estimator.

Fig. 5.13 is an example of the output of the criterion program after evaluation of the function c(m,k,i,j). It shows the behaviour of the AR modified covariance method when applied to the different signal simulation cases, estimating both mean frequency and half bandwidth estimators, as function of the model order i. The letters 'B' on the graphs identify the model order for which c(m,k,i,j) is a minimum for that particular simulation case k.

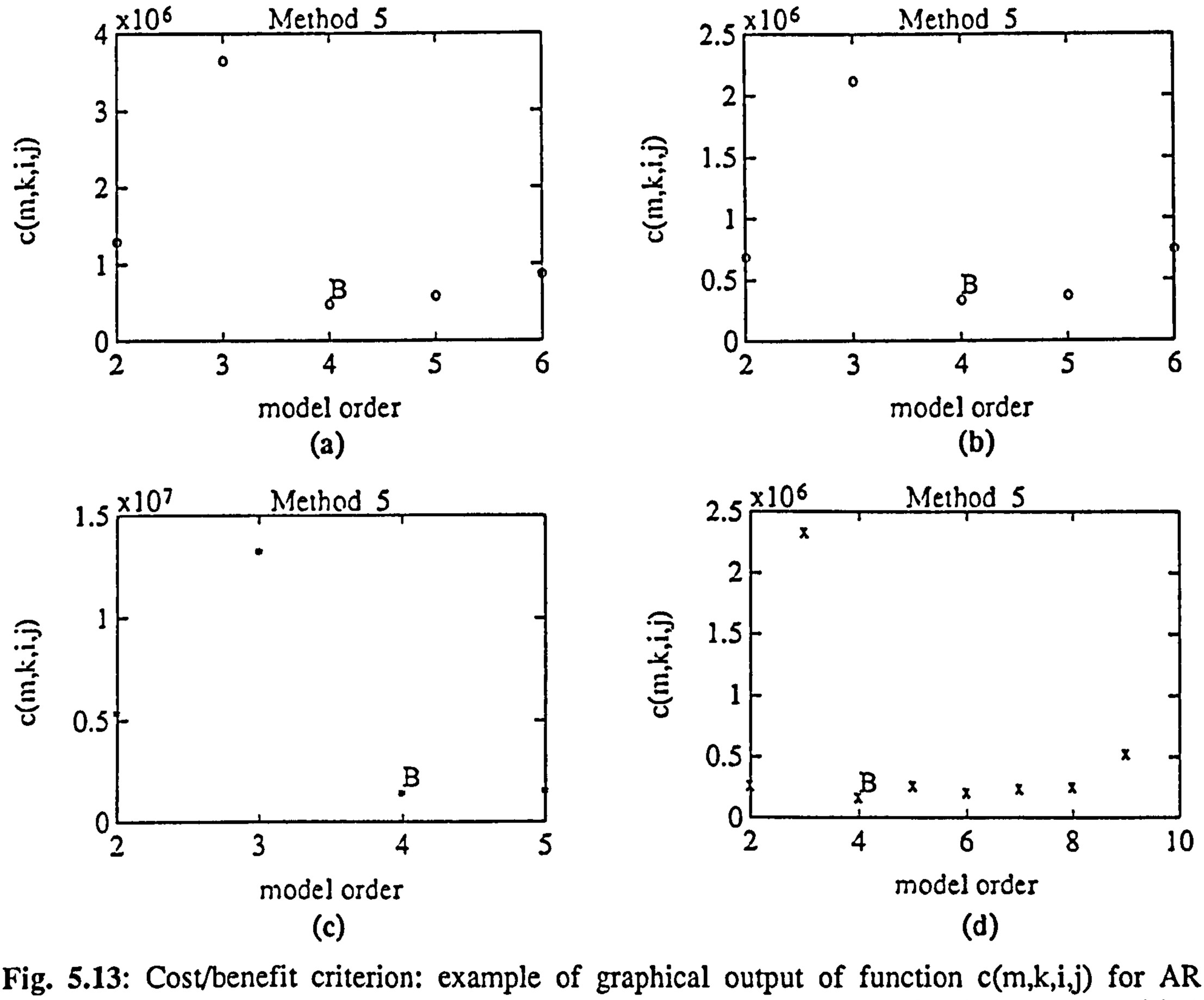


Fig. 5.13: Cost/benefit criterion: example of graphical output of function c(m,k,i,j) for AR modified covariance method when estimating both f_c and b, for the simulation cases: **(a)** 1 (f_c=1kHz, b=100Hz), **(b)** 4 (f_c=2kHz, b=200Hz), **(c)** 6 (f_c=4kHz, b=200Hz), **(d)** 11 (f_c=8kHz, b=1.6kHz).

A summary of the selected model orders at this stage of the application of the criterion, for each simulation case, considering all the parametric methods studied, is presented on Table 5.4.

method	AR			ARMA
f_c|b (Khz)	YW	B	MC	LSMYWE p|q
1|0.1	2	4	4	4|2
1|0.2	2	2	2	4|3
2|0.1	2	2	4	4|2
2|0.2	2	2	4	5|2
2|0.4	2	2	2	4|2
4|0.2	2	4	4	4|2
4|0.4	2	4	4	5|2
4|0.8	6	2	4	4|2
8|0.4	2	4	4	4|2
8|0.8	6	4	4	4|2
8|1.6	6	4	4	7|7
best p	*2*	*4*	*4*	*4|2*

Table 5.4: Model orders selected by the first stage of the cost/benefit criterion when estimating both f_c and b. The last row indicates the model order selected by the intermediate stage of the criterion.

The last row of this table indicates the result of the intermediate stage of this selection method, this is, the minimum values of c(m,k,i,j) over all simulation cases - values of $c_o(m)$ for different m. In the case of parametric spectral estimation the function $c_o(m)$ enables the selection of the best model order.

Fig. 5.14 is representative of this stage of the selection when the modified covariance method is employed. Again, letter 'B' on graph indicates the model order to select when considering the whole range of simulation signals indicated by k, for a particular spectral estimation method.

With knowledge of all $c_o(m)$ values for the methods in comparison, the software program presents a last graph, of $c_o(m)$ values for the different methods, indicating the method which performs best overall.

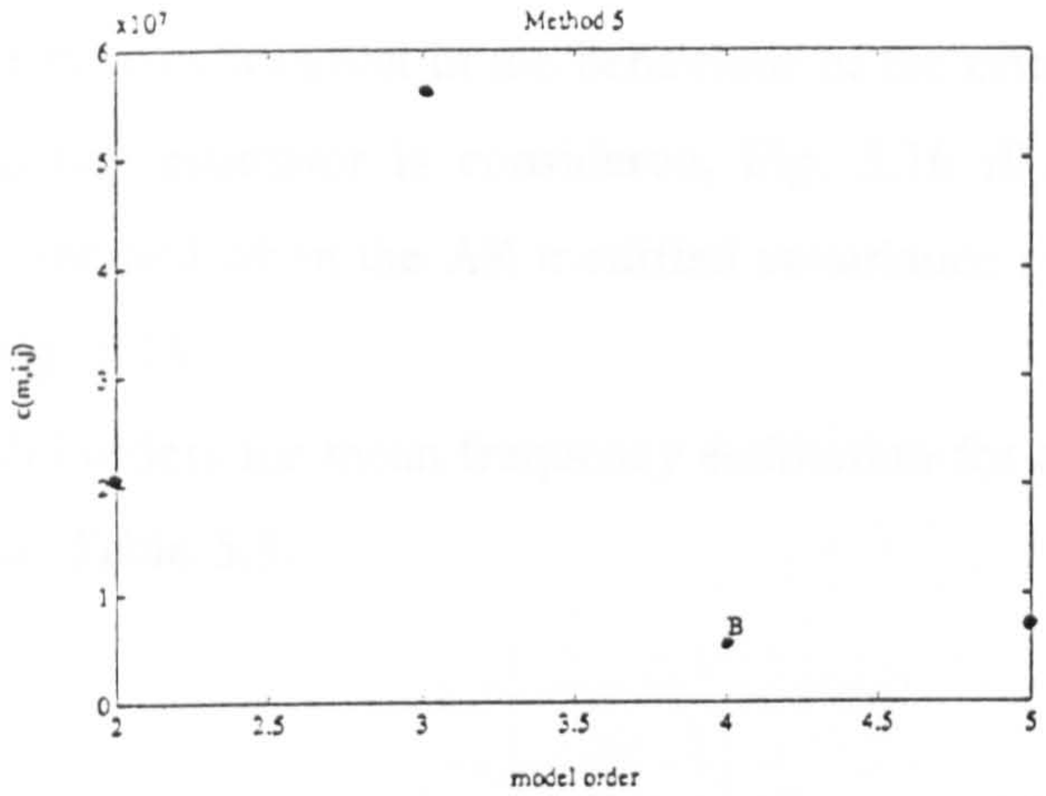


Fig. 5.14: Cost/Benefit criterion: minimum values of c(m,i,j) for different model orders, when estimating both f_c and b. B identifies the method's model order selected.

The results of the final stage of the selection procedure taking all methods in consideration, are presented in Fig. 5.15, indicating the selection of the AR modified covariance method with a model order of 4 when both estimators are equally weighted.

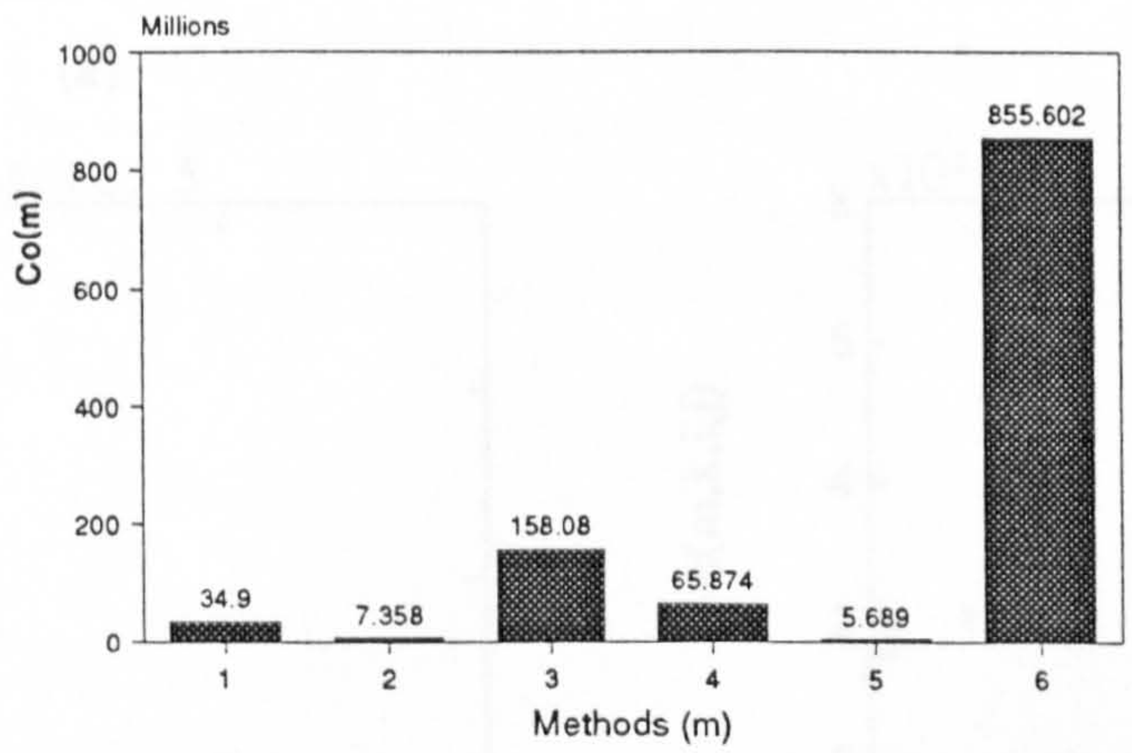


Fig. 5.15: Final stage of the cost/benefit criterion versus spectral estimator (1-FFT/Boxcar, 2-FFT/Hanning, 3-AR/Yule-Walker, 4-AR/Burg, 5-AR/modified covariance, 6-ARMA/LSMYWE) when estimating both f_c and b.

In a second case study, only spectral mean frequency estimation was considered, denoting more interest in blood velocity estimation. In this situation the statistical measurements' weights were $w_{bf}=w_{sf}=1$ and $w_{bb}=w_{sb}=0$.

As an example of the modification in the behaviour of the criterion function c(m,k,i,j) when only the mean frequency estimator is considered, Fig. 5.16 illustrates the first stage of the cost/benefit selection method when the AR modified covariance method is considered, enabling a comparison with Fig. 5.13.

The selected model orders for mean frequency estimation for all signal cases and estimation methods are shown in Table 5.5.

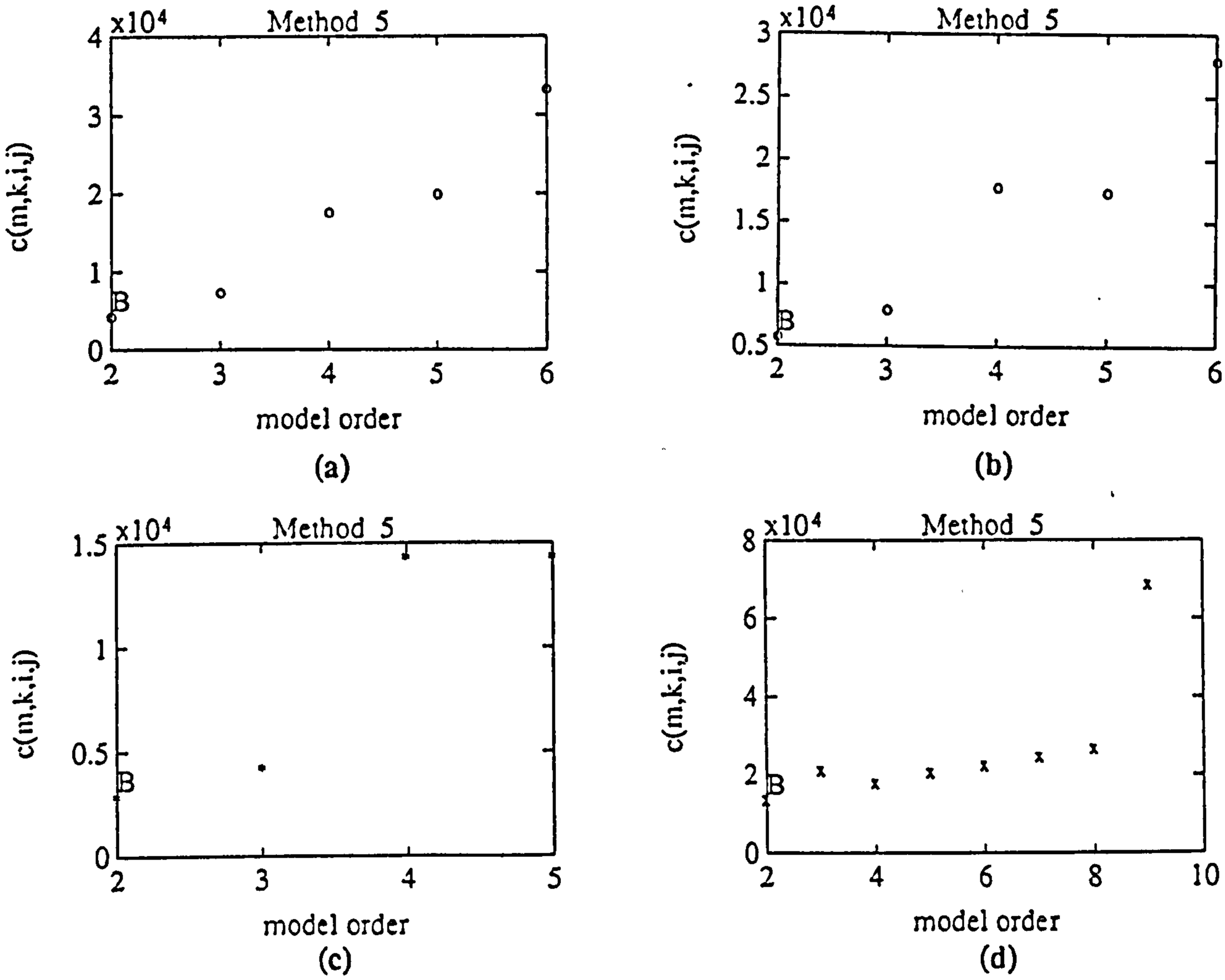


Fig. 5.16: Cost/benefit criterion: example of graphical output of function c(m,k,i,j) for AR modified covariance method when estimating f_c only, for the simulation cases: (a) 1 (f_c=1kHz, b=100Hz), (b) 4 (f_c=2kHz, b=200Hz), (c) 6 (f_c=4kHz, b=200Hz), (d) 11 (f_c=8kHz, b=1.6kHz).

Fig. 5.17 presents the results of the last stage of the proposed criterion, showing the Fourier transform with Boxcar window as the selected method when spectral mean frequency estimation of the pulsed Doppler ultrasound signal is required.

method	AR			ARMA
f_c|b (kHz)	YW	B	MC	LSMYWE p|q
1|0.1	2	2	2	4|3
1|0.2	2	2	2	4|2
2|0.1	2	2	2	4|3
2|0.2	2	2	2	4|3
2|0.4	2	2	2	3|3
4|0.2	2	2	2	4|3
4|0.4	2	2	2	4|3
4|0.8	2	2	2	6|6
8|0.4	2	2	2	4|3
8|0.8	2	2	2	4|3
8|1.6	2	2	2	7|7
best p	*2*	*2*	*2*	*4/2*

Table 5.5: Model orders selected by the first stage of the cost/benefit criterion when estimating f_c only. Last row indicates the model order selected by the intermediate stage of the criterion.

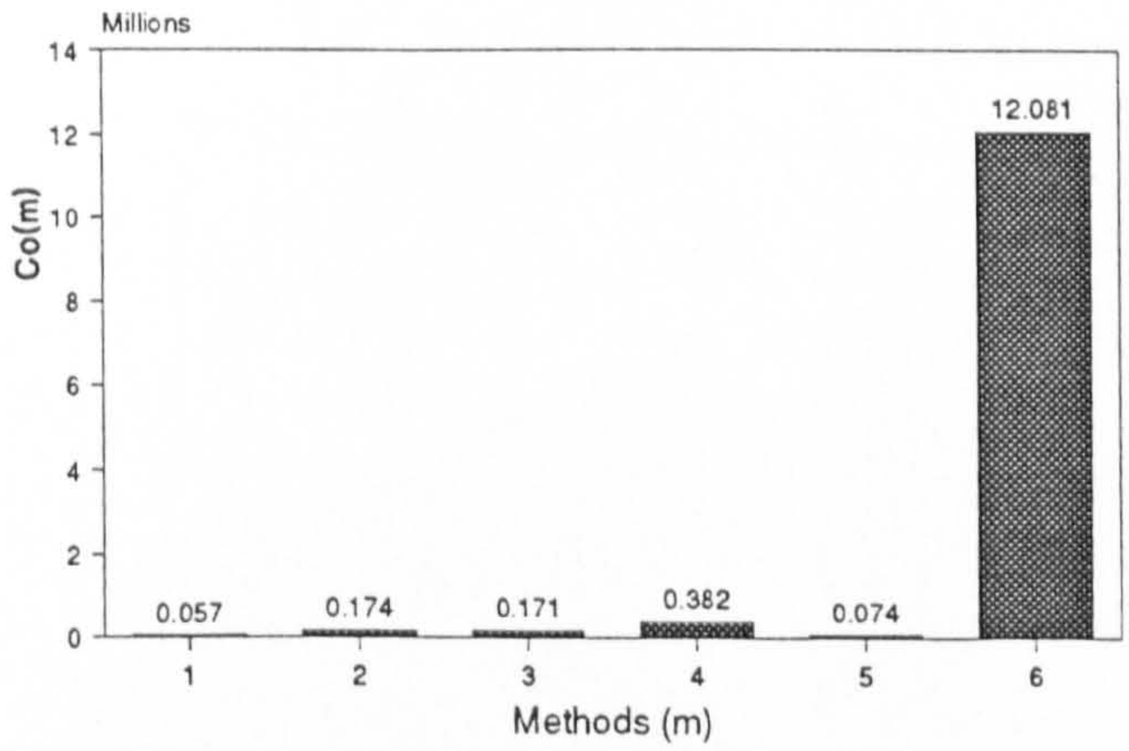


Fig. 5.17: Final stage of the cost/benefit criterion versus spectral estimator (1-FFT/Boxcar, 2-FFT/Hanning, 3-AR/Yule-Walker, 4-AR/Burg, 5-AR/modified covariance, 6-ARMA/LSMYWE) when estimating f_c only.

If the user is concerned only with the estimation of the blood flow disturbance, the mean frequency statistics' weights w_{bf} and w_{sf} are set to 0, and the modified covariance method with model order 4 is also selected by the cost/benefit criterion program.

The model order selection for each simulation case was compared with the selected orders obtained by application of previously published model order selectors, when AR parametric spectral estimators (j=0) were considered.

To do so, some adaptations of the cost/benefit criterion algorithm had to be performed. The criterion algorithm had to be modified to compute only the best model order for a certain simulation case, considering each spectral estimation method separately.

Graphical presentations of the function c(m,k,i,j) versus model order i for particular method (m) and signal case were compared with FPE, AIC and CAT values and show similarly shaped curves with a single minimum. The curves with the most noticeable minimum are those from the cost/benefit criterion, followed by those from the CAT criterion.

As an example, these two curves are shown in Fig. 5.18, when the AR Yule-Walker method is considered and even values of model order up to 30 are employed.

Two simulation cases are presented to show that as the simulation case numbers increased, that is, as the simulated signal mean frequency increased, the minima became less pronounced with the accepted criteria, while it remained clearly noticeable with the cost/benefit criterion.

Fig. 5.19 shows a summary of the model orders selected when all simulation cases were considered, by the four model order selector criteria for the AR spectral estimators considered in this study.

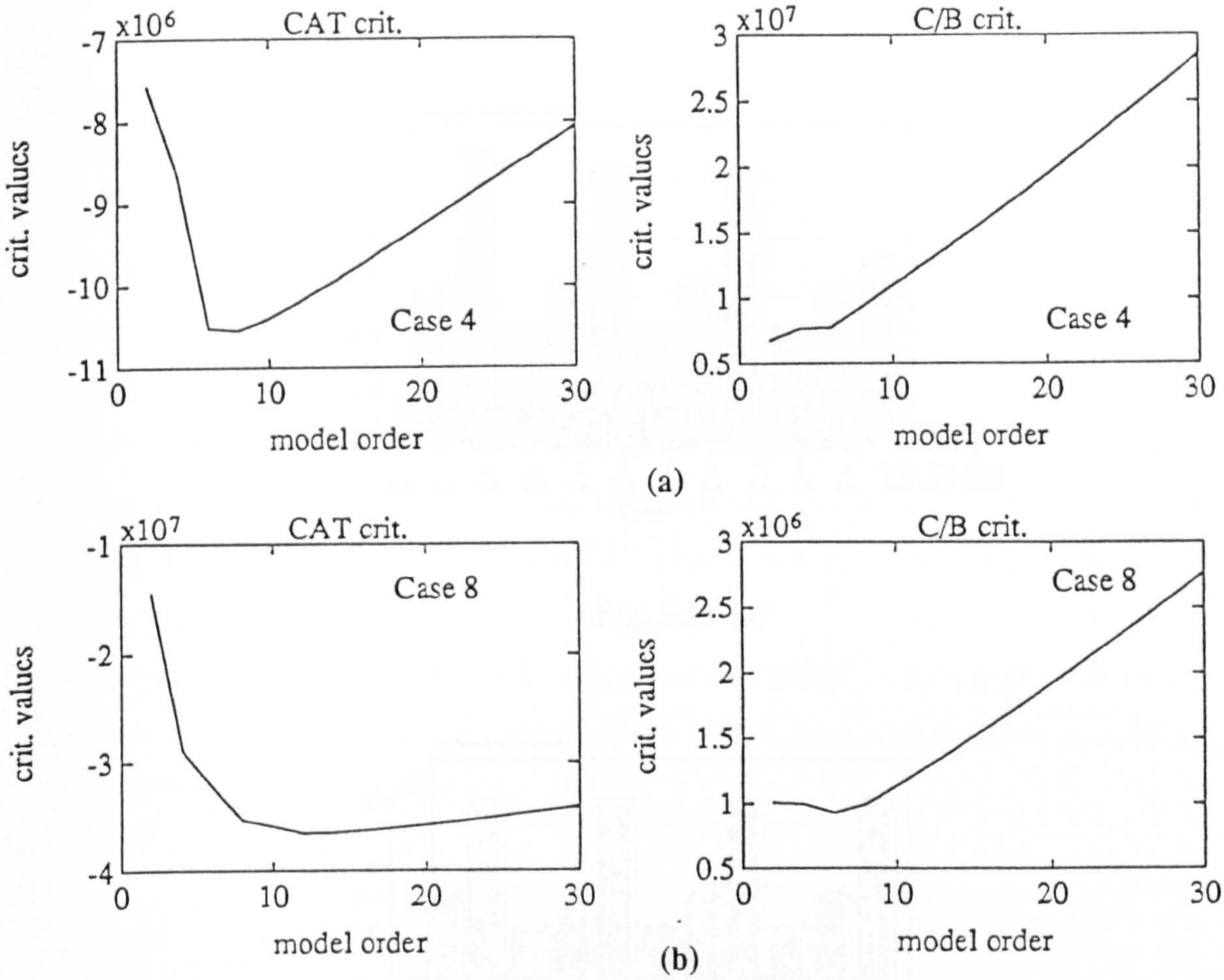


Fig. 5.18: Model order selection when AR Yule-Walker method is applied to simulation case (**a**) 4 (f_c=2kHz, b=200Hz) (**b**) 8 (f_c=4kHz, b=800Hz), when CAT and the first stage of the cost/benefit criterion are compared.

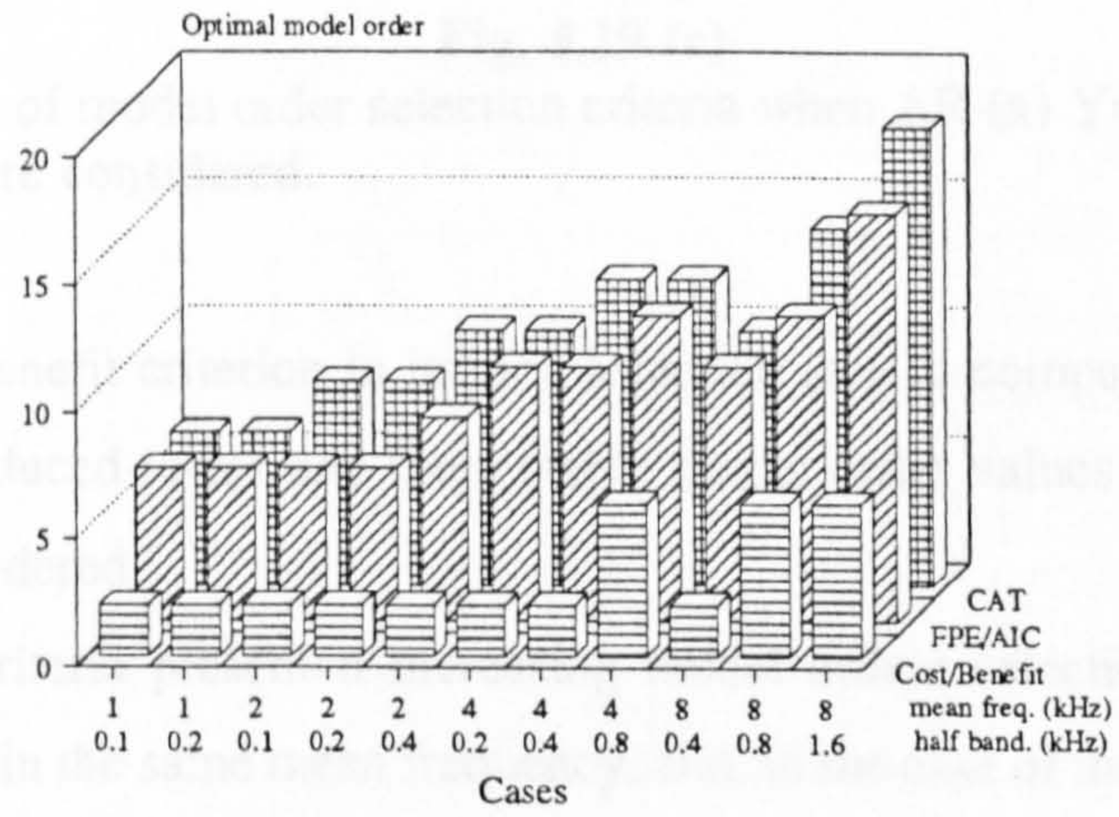


Fig. 5.19 (a)

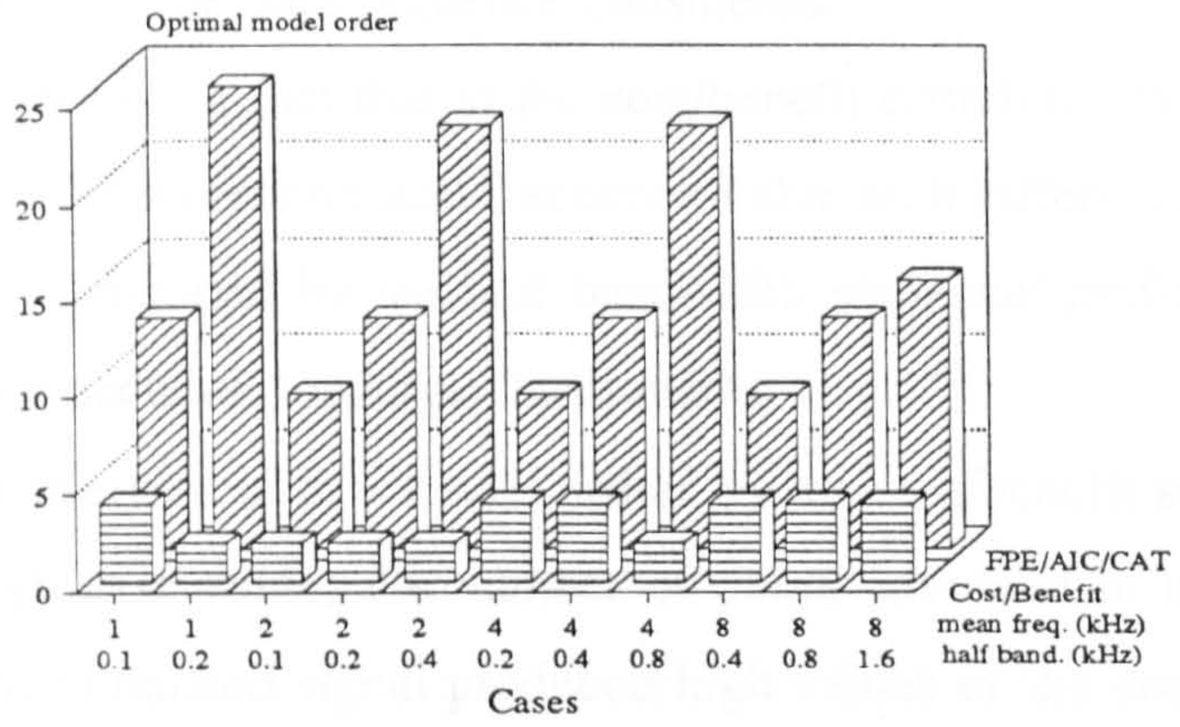


Fig. 5.19 (b)

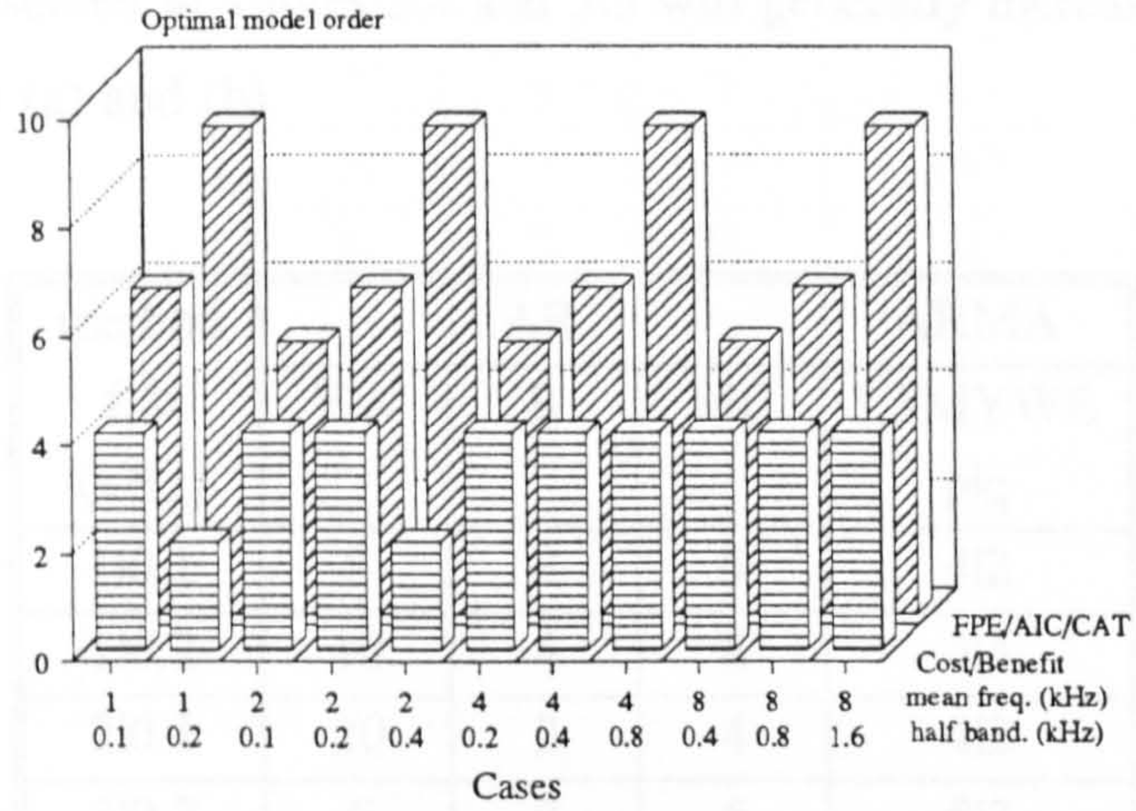


Fig. 5.19 (c)

Fig. 5.19: Comparison of model order selection criteria when AR **(a)** Yule-Walker, **(b)** Burg, **(c)** modified covariance are considered.

Clearly, the cost/benefit criterion in its first selection step in comparison with the FPE, AIC and CAT criteria, produced lower and more stable model order values over the range of signal simulation cases considered.

While the other criteria presented increasing model orders selections with increasing half bandwidth values within the same mean frequency, and, in the case of the Yule-Walker estimator, even increasing model orders with increasing mean frequency values, the cost/benefit criterion,

dependent on the spectral estimator, tended to select smaller model orders for higher bandwidths, independently of the size of the data sequence considered.

This feature results from the fact that in the cost/benefit criterion although the cost function does not vary greatly within a certain mean frequency value with different half bandwidth values, the benefit function is dominated by the half bandwidth statistical performance values, which have proven to be more accurate for larger bandwidths.

The reason for the selection of low model orders by the cost/benefit selection method is due to the fact that for high case values the number of points involved in the computation of the spectral estimate of the simulated signal produced high values of the cost function, forcing the cost/benefit criterion algorithm to prefer smaller model orders at the cost of a small loss in statistical accuracy.

If the cost function in the cost/benefit selection method is put equal to unity, the selected model order values presented in Tables 5.4 and 5.5 will generally increase, achieving the values indicated in Tables 5.6 (a) and (b).

method	AR			ARMA
$f_c$lb (kHz)	YW	B	MC	LSMYWE plq
110.1	8	8	5	412
110.2	10	4	8	413
210.1	20	2	4	412
210.2	6	2	5	512
210.4	30	6	6	512
410.2	6	4	5	412
410.4	10	4	5	512
410.8	26	12	6	412
810.4	6	4	4	412
810.8	10	4	4	412
811.6	30	12	6	717
best p	*16*	*4*	*4*	*4/2*

Table 5.6 (a)

When the joint estimation of spectral mean frequency and half bandwidth parameters is performed (see Tables 5.4 and 5.6(a)), the benefit function reveals more sensitivity to the half

bandwidth values of by each simulation case than the cost/benefit function, particularly when the AR Yule-Walker spectral estimator is used. This sensitivity is less pronounced when only mean frequency estimation is considered (see Tables 5.5 and 5.6 (b)).

The best model orders for each parametric method (indicated in the last line of Tables 5.4 - 5.6) are maintained, except for the Yule-Walker method. This is probably due to the fact that the cost/benefit selection method includes a rather simple cost function.

method	AR			ARMA
f_c|b (kHz)	YW	B	MC	LSMYWE p|q
1|0.1	2	2	2	4|3
1|0.2	4	4	2	4|2
2|0.1	2	2	2	4|3
2|0.2	26	2	2	4|3
2|0.4	4	4	6	3|3
4|0.2	2	2	2	4|3
4|0.4	6	2	2	4|3
4|0.8	14	14	8	6|6
8|0.4	2	2	2	4|3
8|0.8	4	12	5	4|3
8|1.6	4	18	8	7|7
best p	*4*	*2*	*2*	*4|3*

Table 5.6 (b)

Table 5.6: Model orders selected by the first stage of the cost/benefit criterion when considering cost=1, when (a) f_c and b, or, (b) f_c only, are the spectral estimators considered. Last row indicates the model order selected by the intermediate stage of the criterion.

5.5. CONCLUDING REMARKS

The simulation of the PDU signal characterized by Gaussian spectral profile developed in Chapter 4 has now been subject of study in the frequency domain by application of some spectral estimation methods available from the literature.

The so called classical methods of Fourier transforming windowed signals were applied, using a Boxcar and a Hanning window. Among the parametric methods, the AR Yule-Walker, Burg, and modified covariance, and two ARMA methods, the least square modified Yule-Walker equations and the Mayne-Firoozan methods were employed. The minimum variance method, a non-parametric method, was also tried.

Both the ARMA Mayne-Firoozan and the minimum variance methods were eliminated from consideration since their results proved to produce either non-minimum-phase filter estimates (case of the ARMA method) and high spectral variances (case of the minimum variance method), presenting clear disadvantages in estimating the PDU signal with a Gaussian spectral profile.

An initial study of the statistical results obtained with the selected spectral estimation methods has been performed, basing model order selection and comparative judgements on qualitative decisions. This first study, motivated the development of an objective criterion of spectral estimator and parametric model order selection, the cost/benefit criterion.

The cost/benefit criterion enables the selection of spectral estimators by taking into account the cost (or computational complexity) and allowing varying weighting to be given to mean frequency and bandwidth estimation errors across the range of these signal characteristics. This allows the possibility of weighting appropriate to estimator (mean frequency and bandwidth) and signal frequency importance in disease detection. It also enables more appropriate order selection for parametric estimators.

This selection criterion proved to produce justified choices both of the spectral estimation methods under consideration and the selected model orders. It is shown that when estimating both mean frequency and bandwidth, or only the latter, the AR modified covariance method is selected.

However, when the mean frequency only is the estimator of interest, the improvement in statistical performance by the use of a parametric model does not compensate for the cost of applying a model to the data instead of a FFT algorithm, and a non-tapered window allows maximum contribution of the available data to the estimation.

The results of the first selection step of the cost/benefit criterion has been compared with those obtained for the same simulation cases with accepted model order selectors, the FPE, AIC, and, CAT criteria.

The parametric model orders selected by the cost/benefit criterion are lower than the selected by accepted criteria.

Two factors influence this. Firstly, the choice of the spectral parameters of interest (the mean frequency and bandwidth) involve only low order spectral moments and impose small control on model spectral shape and therefore allows the use of low model-orders as has been noted elsewhere [19][86].

Secondly the rapid increase in cost with model order tends to constrain the selected model order to low values whereas the other criteria have slower rates of increase with model order leading to shallow minima and the selection of orders higher than may be adequate for the spectral accuracy required.

It was expected that the cost function would have a noticeable effect on model order selection in all signal cases. The cost function was fairly simple in this study and this situation might change when the cost function is developed further.

CHAPTER 6

CHAPTER 6

Implementation of AR-Modified Covariance Spectral Estimator Using Parallel Processing Techniques

6.1. INTRODUCTION

The performance of several spectral estimators with respect to the statistical accuracy of the spectral estimators mean frequency and half bandwidth was investigated (Chapter 5), when applied to ensembles of Doppler signal simulation (Chapter 4). The simulated signals were characterised by Gaussian probability density function, Gaussian spectral profiles (Chapter 3) with mean frequencies ranging from 1kHz to 8kHz in octaves, and half bandwidths of 5% (excepting the 1kHz case), 10% and 20% of the mean frequency. The cost/benefit criterion [87] [88] enabled the selection of the AR modified covariance method with a model order of 4 as the spectral estimator which most accurately estimated (under general conditions) the spectral parameters of interest.

The final objective of this thesis is the implementation of the chosen spectral estimator in real-time, enabling its spectral resolution benefits to be applied to the diagnosis of cardiovascular disease progression and treatment.

With such an objective in mind, the implementation of the previously selected spectral estimator (the AR-modified covariance algorithm) using fast computer processing techniques was developed and is reported in this Chapter. General background theory in the field of parallel processing was included in Chapter 2, where the definition of typical terms and procedures were described.

Two different approaches to the parallelization of the modified covariance spectral estimation method are reported. The first considered the application of the method to a single Doppler data segment, while the other, a more general approach, regarded the concurrent spectral estimation of a number of Doppler data segments. The latter is being currently developed in collaboration with a research group colleague, and is briefly described in section 6.8.

The new version of the modified covariance algorithm [89], based on the algorithm published by Kay [17], is described in the section 6.2. A study of the computational complexity of the new algorithm in comparison with the published versions of Kay and of Marple [32] is also presented. The strategy employed to achieve the parallelization of the algorithm (this is the separate treatment of each major computational block and then the connection of all blocks) is reported in more detail in section 6.2.4.

The following sections (6.3 - 6.6) describe the development of each computational block of the modified covariance algorithm, the parallel partitioning schemes followed, the parallel implementation topologies employed and the results obtained. Some comments on the performance of the parallel implementations are also made.

Section 6.7 deals with the overall implementation of the algorithm, after connection of all the computational blocks. General remarks on the performance achieved are included.

Section 6.9 summarizes the most relevant features of this chapter.

6.2. THE AR MODIFIED COVARIANCE ALGORITHM

The general theory behind the modified covariance spectral estimation method has already been described in section 5.2.2.C., however its most important mathematical expressions are repeated here to aid a clear understanding of the simplifications made on them to achieve a parallel algorithm.

Kay [17] and Marple [32], describe two different algorithms to implement the modified covariance method. The former, follows the usually given interpretation of the method, while the latter, corresponds to a modification introduced by Marple on the general algorithm [80].

Although the algorithm employed in Chapter 5 was the Marple's version of the modified covariance method (choice due to its fast sequential computational times and capacity to determine matrix ill-conditioning situations), its high degree of recursion and nested structure lead to the usage of Kay version as the basis of the simplifications here developed.

6.2.1. General Algorithm

The modified covariance method minimizes the average of the forward and backward prediction error powers (see (5.39 to (5.41)) by solution of the modified covariance equations

$$\begin{bmatrix} c_{xx}[1,1] & c_{xx}[1,2] & \cdots & c_{xx}[1,p] \\ c_{xx}[2,1] & c_{xx}[2,2] & \cdots & c_{xx}[2,p] \\ \vdots & \vdots & \ddots & \vdots \\ c_{xx}[p,1] & c_{xx}[p,2] & \cdots & c_{xx}[p,p] \end{bmatrix} \cdot \begin{bmatrix} \hat{a}[1] \\ \hat{a}[2] \\ \vdots \\ \hat{a}[p] \end{bmatrix} = - \begin{bmatrix} c_{xx}[1,0] \\ c_{xx}[2,0] \\ \vdots \\ c_{xx}[p,0] \end{bmatrix} \tag{6.1}$$

where the covariance matrix, is Hermitian and positive definite.

Each element of the matrix is computed as

$$c_{xx}[j,k] = \frac{1}{2.(N-p)} \cdot \left(\sum_{n=p}^{N-1} x^*[n-j] \,.\, x[n-k] + \sum_{n=0}^{N-1-p} x[n+j] \,.\, x^*[n+k] \right) \tag{6.2}$$

and the solution of the system is typically obtained by Cholesky decomposition.

The white noise variance estimate is computed as

$$\hat{\sigma}^2 = \hat{\rho}_{min} = c_{xx}[0,0] + \sum_{k=1}^{p} \hat{a}[k] \;.\; c_{xx}[0,k] \tag{6.3}$$

The approximate number of computational operations required for the implementation of each major algorithmic block in this method is represented in Table 6.1. Computational burden is measured in terms of floating point operations (flops).

In the situation under consideration, where N is large compared with p, the most computational time-consuming block is that concerned with the computation of the covariance

matrix elements, while the solution of the linear system of equations is the action that requires the least number of flops.

Therefore the block of calculation of the covariance matrix elements is the one requiring most attention in order to reduce computational times.

BLOCK	FLOPS
covariance matrix	$3p^2N$
right-hand-side vector	$4pN$
solution of system (Cholesky)	p^3
white noise variance	$4pN$

Table 6.1: Modified covariance algorithm blocks and respective complexities (in flops).

6.2.2. Development of Modified Covariance Algorithm

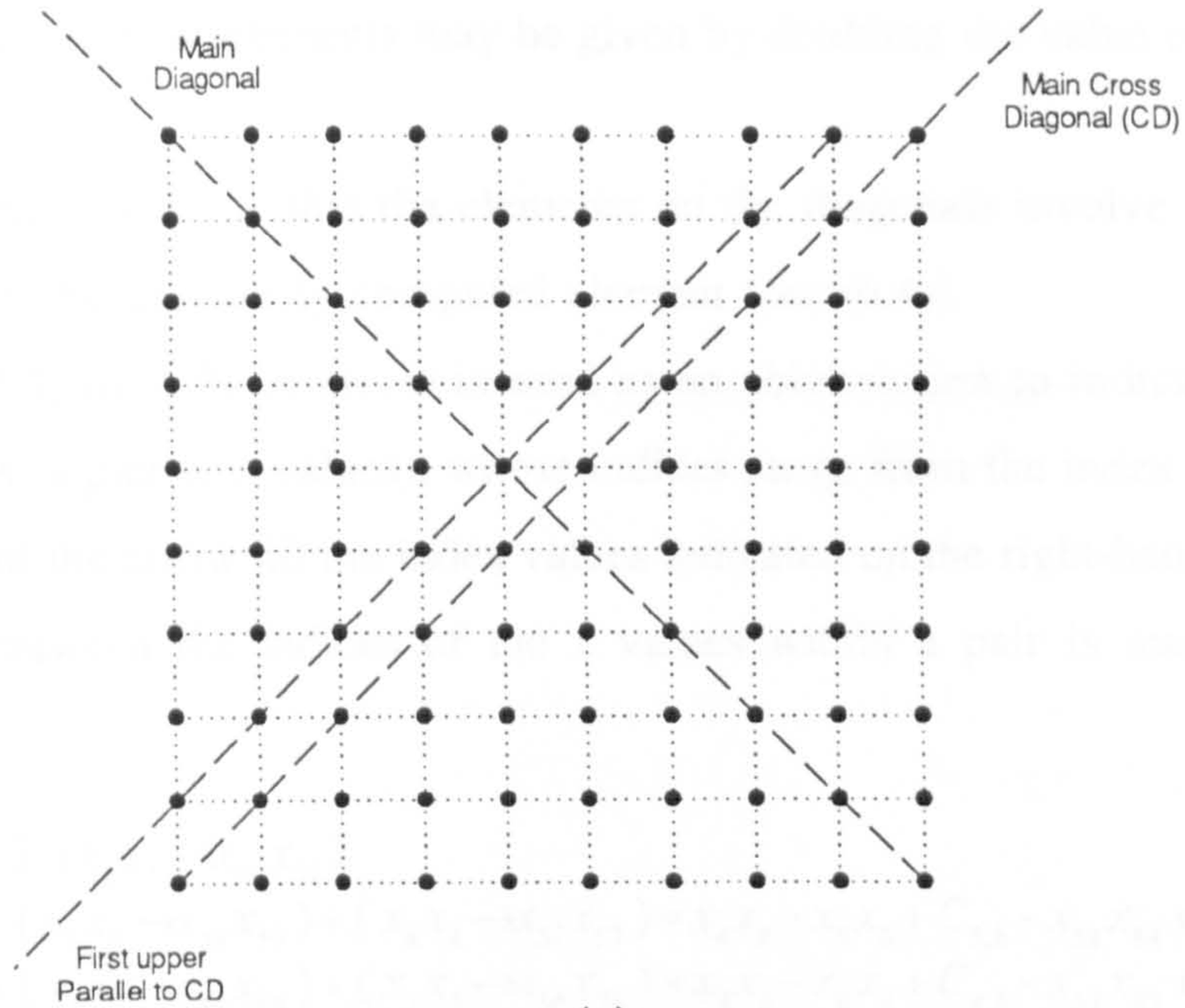
The development of the modified covariance algorithm will be achieved through an example where a data sequence length of 64 is considered and a model order of 10 is assumed.

Since the covariance matrix is Hermitian (complex conjugate symmetry) we need to compute only half the elements of the matrix.

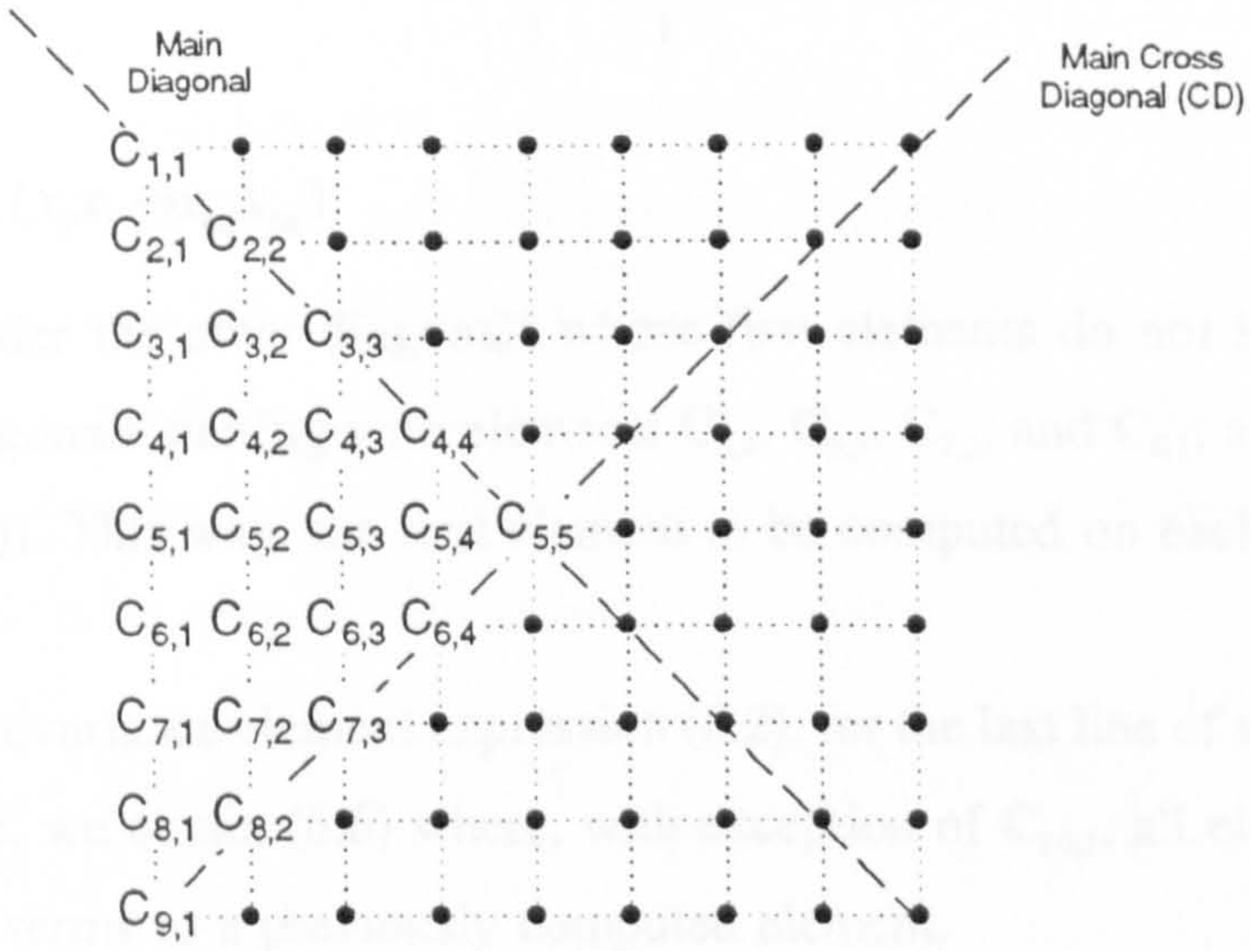
Developing equation (6.2) for different values of line and column indices we conclude that the matrix obtained by omitting the last line and last column of the covariance matrix is also symmetric about its main cross diagonal. This matrix is called the *remaining matrix*. Fig. 6.1 (a) shows the original matrix, where the main diagonal, the main cross diagonal and one of the parallels to the main cross diagonal are indicated. The elements that need to be computed to calculate the whole covariance matrix are highlighted in Fig. 6.1 (b), where the remaining matrix is illustrated.

In the following development, element $C_{5,5}$, the middle element of the remaining matrix, will be considered as the starting point for the analysis. The "walking lines" for computation of the remaining matrix elements are the main diagonal and its parallels. The starting point of each walking line is the element located on the intersection of that particular diagonal with either the main cross diagonal of the remaining matrix - called *elements lying on the main cross diagonal*

(or more briefly, on the cross diagonal) - or, the first upper parallel to the main cross diagonal of the remaining matrix - denoted as *elements not lying on the main cross diagonal.*



(a)



(b)

Fig. 6.1: (**a**) Original covariance matrix, and, (**b**) remaining matrix for a 10^{th} order model.

Thus, starting by developing the middle element $C_{5,5}$ of the remaining matrix we see that both summations in (6.2) are equivalent. This feature occurs for all elements on the cross diagonal, that is $C_{6,4}$, $C_{7,3}$, $C_{8,2}$, and $C_{9,1}$. Therefore instead of computing two summations, each one involving 2(N-p) flops, these elements may be given by doubling the value obtained on just one summation.

Another important aspect is that the elements on the diagonals involve a summation range overlap with that of the previously computed element (see (6.4)).

In equations (6.4) to (6.7) an arrow is used as an abbreviation to indicate repetition of the product of x values (a pair of x values), whose indices range from the index values indicated on the left-hand-side of the arrow till the index values indicated on the right-hand-side of the arrow. The relationship between the indices of the x values within a pair is maintained during the repetition.

$$
\begin{aligned}
&C_{5,5} = 2.(x_5x_5 \rightarrow x_{58}x_{58}) \\
&C_{4,4} = (x_6x_6 \rightarrow x_{59}x_{59}) + (x_4x_4 \rightarrow x_{57}x_{57}) = x_4x_4 - x_5x_5 + C_{5,5} - x_{58}x_{58} + x_{59}x_{59} \\
&C_{3,3} = (x_7x_7 \rightarrow x_{60}x_{60}) + (x_3x_3 \rightarrow x_{56}x_{56}) = x_3x_3 - x_6x_6 + C_{4,4} - x_{57}x_{57} + x_{60}x_{60} \\
&\qquad\qquad \vdots \qquad\qquad \vdots \\
&C_{6,4} = 2.(x_6x_4 \rightarrow x_{59}x_{57}) \\
&C_{5,3} = (x_7x_5 \rightarrow x_{60}x_{58}) + (x_5x_3 \rightarrow x_{58}x_{56}) = x_5x_3 - x_6x_4 + C_{6,4} - x_{59}x_{57} + x_{60}x_{58} \\
&C_{4,2} = (x_8x_6 \rightarrow x_{61}x_{59}) + (x_4x_2 \rightarrow x_{57}x_{55}) = x_4x_2 - x_7x_5 + C_{5,3} - x_{58}x_{56} + x_{61}x_{59} \\
&\qquad\qquad \vdots \qquad\qquad \vdots \\
&\vdots \\
&C_{9,1} = 2.(x_9x_1 \rightarrow x_{62}x_{54})
\end{aligned}
\tag{6.4}
$$

If we now consider the other diagonals whose first elements do not lie on the main cross diagonal, i.e. the diagonals starting with elements $C_{5,4}$, $C_{6,3}$, $C_{7,2}$, and $C_{8,1}$, a similar structure can be devised (see (6.5)). This way, the first element to be computed on each diagonal requires 2(N-p-1)+5 flops.

Developing the covariance element expression (6.2), for the last line of the covariance matrix, j=10 in this example, we obtain (6.6) where, with exception of $C_{10,1}$, all elements are computed by adding four new terms to a previously computed element.

$$
\begin{aligned}
C_{5,4} &= (x_6x_5 \rightarrow x_{59}x_{58}) + (x_5x_4 \rightarrow x_{58}x_{57}) = x_5x_4 + 2.(x_6x_5 \rightarrow x_{58}x_{57}) + x_{59}x_{58} \\
C_{4,3} &= (x_7x_6 \rightarrow x_{60}x_{59}) + (x_4x_3 \rightarrow x_{57}x_{56}) = x_4x_3 - x_6x_5 + C_{5,4} - x_{58}x_{57} + x_{60}x_{59} \\
C_{3,2} &= (x_8x_7 \rightarrow x_{61}x_{60}) + (x_3x_2 \rightarrow x_{56}x_{55}) = x_3x_2 - x_7x_6 + C_{4,3} - x_{57}x_{56} + x_{61}x_{60} \\
&\vdots \qquad\qquad \vdots \\
C_{6,3} &= (x_7x_4 \rightarrow x_{60}x_{57}) + (x_6x_3 \rightarrow x_{59}x_{56}) = x_6x_3 + 2.(x_7x_4 \rightarrow x_{59}x_{56}) + x_{60}x_{57} \\
C_{5,2} &= (x_8x_5 \rightarrow x_{61}x_{58}) + (x_5x_2 \rightarrow x_{58}x_{55}) = x_5x_2 - x_7x_4 + C_{6,3} - x_{59}x_{56} + x_{61}x_{58} \\
C_{4,1} &= (x_9x_6 \rightarrow x_{62}x_{59}) + (x_4x_1 \rightarrow x_{57}x_{54}) = x_4x_1 - x_8x_5 + C_{5,2} - x_{58}x_{55} + x_{62}x_{59} \\
&\vdots \qquad\qquad \vdots \\
C_{8,1} &= (x_9x_2 \rightarrow x_{62}x_{55}) + (x_8x_1 \rightarrow x_{61}x_{54}) = x_8x_1 + 2.(x_9x_2 \rightarrow x_{61}x_{54}) + x_{62}x_{55}
\end{aligned}
\tag{6.5}
$$

$$
\begin{aligned}
C_{10,1} &= x_9x_0 + 2.(x_{10}x_1 \rightarrow x_{62}x_{53}) + x_{63}x_{54} \\
C_{10,2} &= (x_8x_0 \rightarrow x_9x_1) + 2.(x_{10}x_2 \rightarrow x_{61}x_{53}) + (x_{62}x_{54} \rightarrow x_{63}x_{55}) = x_0x_8 - x_1x_9 + C_{9,1} - x_{54}x_{62} + x_{55}x_{63} \\
C_{10,3} &= (x_7x_0 \rightarrow x_9x_2) + 2.(x_{10}x_3 \rightarrow x_{60}x_{53}) + (x_{61}x_{54} \rightarrow x_{63}x_{56}) = x_0x_7 - x_2x_9 + C_{8,1} - x_{54}x_{61} + x_{56}x_{63} \\
&\vdots \\
C_{10,8} &= (x_2x_0 \rightarrow x_9x_7) + 2.(x_{10}x_8 \rightarrow x_{55}x_{53}) + (x_{56}x_{54} \rightarrow x_{63}x_{61}) = x_0x_2 - x_7x_9 + C_{3,1} - x_{54}x_{56} + x_{61}x_{63} \\
C_{10,9} &= (x_1x_0 \rightarrow x_9x_8) + 2.(x_{10}x_9 \rightarrow x_{54}x_{53}) + (x_{55}x_{54} \rightarrow x_{63}x_{62}) = x_0x_1 - x_8x_9 + C_{2,1} - x_{54}x_{55} + x_{62}x_{63} \\
C_{10,10} &= (x_0x_0 \rightarrow x_9x_9) + 2.(x_{10}x_{10} \rightarrow x_{53}x_{53}) + (x_{54}x_{54} \rightarrow x_{63}x_{63}) = x_0x_0 - x_9x_9 + C_{1,1} - x_{54}x_{54} + x_{63}x_{63}
\end{aligned}
\tag{6.6}
$$

The elements of the right-hand-side vector of (6.1) are computed by setting k in (6.2) equal to zero and considering x[n] to be a real data sequence. Developing the resultant expressions we obtain (6.7) where it can be seen that all the elements , except the last, are already computed. The last element, $C_{0,10}$ needs to be completely calculated. However its computation is easily incorporated in an already existing algorithm loop since it has a structure similar to that of the elements on the main cross diagonal of the remaining matrix.

$$
\begin{aligned}
C_{0,1} &= (x_0x_1 \rightarrow x_8x_9) + 2.(x_9x_{10} \rightarrow x_{53}x_{54}) + (x_{54}x_{55} \rightarrow x_{62}x_{63}) = C_{10,9} \\
C_{0,2} &= (x_0x_2 \rightarrow x_7x_9) + 2(x_8x_{10} \rightarrow x_{53}x_{55}) + (x_{54}x_{56} \rightarrow x_{61}x_{63}) = C_{10,8} \\
&\vdots \\
C_{0,9} &= x_0x_9 + 2.(x_6x_4 \rightarrow x_{59}x_{57}) + x_{54}x_{63} = C_{10,1} \\
C_{0,10} &= 2.(x_0x_{10} \rightarrow x_{53}x_{63})
\end{aligned}
\tag{6.7}
$$

Thus, all the elements of the covariance matrix and right-hand-side vector arise from a computation of only a few elements as shown in Figures 6.2 and 6.3 where the double arrows denote that an element is calculated from the previously computed element.

$$C_{5,5} \Rightarrow C_{4,4} \Rightarrow C_{3,3} \Rightarrow C_{2,2} \Rightarrow C_{1,1} \Rightarrow C_{10,10}$$
$$C_{6,4} \Rightarrow C_{5,3} \Rightarrow C_{4,2} \Rightarrow C_{3,1} \Rightarrow C_{10,8}$$
$$C_{7,3} \Rightarrow C_{6,2} \Rightarrow C_{5,1} \Rightarrow C_{10,6}$$
$$C_{8,2} \Rightarrow C_{7,1} \Rightarrow C_{10,4}$$
$$C_{9,1} \Rightarrow C_{10,2}$$
$$C_{0,10}$$

Fig. 6.2: Remaining matrix elements whose first element lie on the main cross diagonal.

$$C_{5,4} \Rightarrow C_{4,3} \Rightarrow C_{3,2} \Rightarrow C_{2,1} \Rightarrow C_{10,9}$$
$$C_{6,3} \Rightarrow C_{5,2} \Rightarrow C_{4,1} \Rightarrow C_{10,7}$$
$$C_{7,2} \Rightarrow C_{6,1} \Rightarrow C_{10,5}$$
$$C_{8,1} \Rightarrow C_{10,3}$$
$$C_{10,1}$$

Fig. 6.3: Remaining matrix elements whose first element do not lie on the main cross diagonal.

The other elements of the matrix are obtained by software attribution expressions, taking into consideration the matrix symmetries.

The computational burden of calculating the white noise variance may also be reduced. The value of $C_{0,0}$, initially computed by (6.2), is coincident with $C_{10,10}$ reducing to 2p+1 flops the computation of the last block of Table 6.1.

The example used to explain the development of the method uses an even model order. Although the symmetry of an odd order covariance matrix differs from that presented, a similar simplification procedure may be adopted.

The solution of the linear system of equations (another block of Table 6.1) and the computation of the power spectral density (PSD) will be described in sections 6.4 and 6.6 respectively.

6.2.3. Comparison of the computational complexities of the Modified Covariance versions

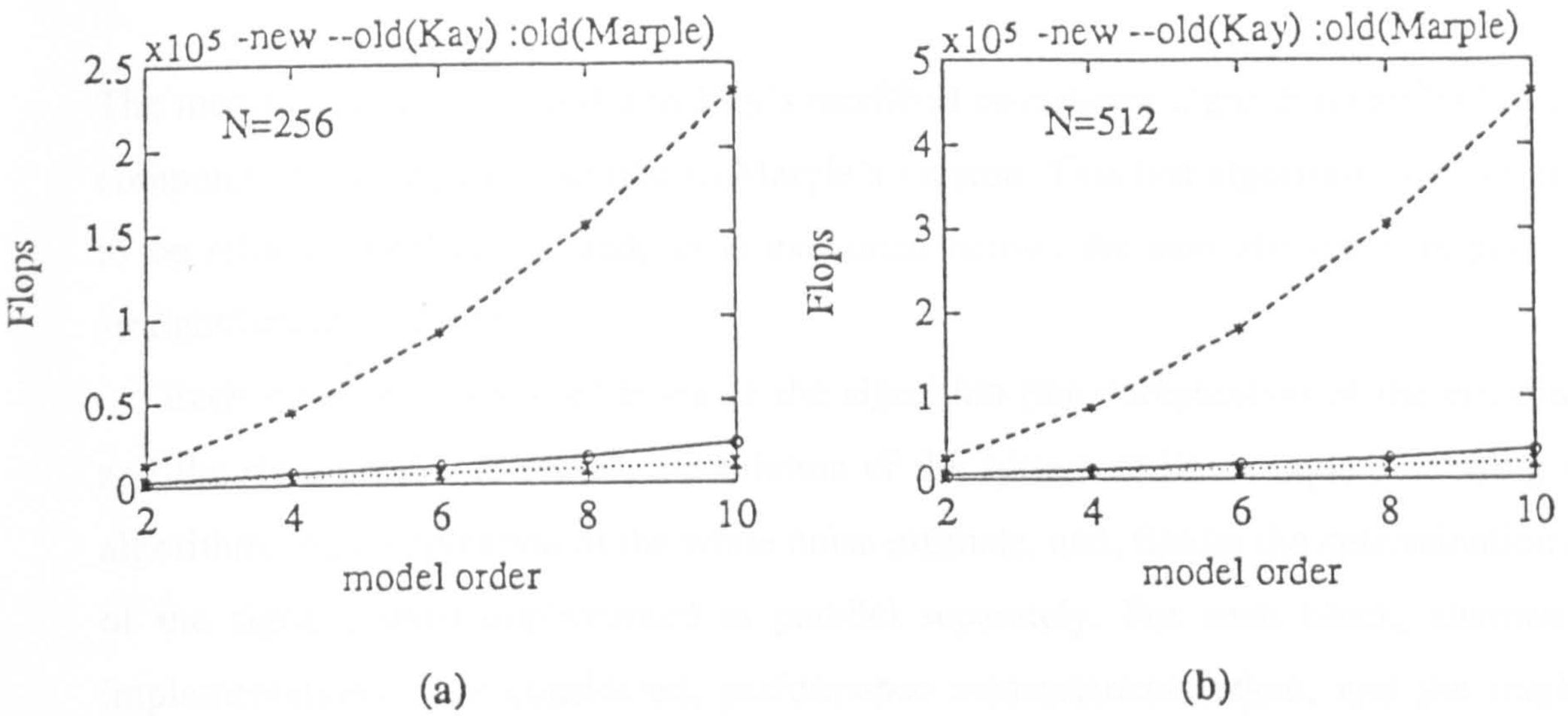
The new modified covariance algorithm was evaluated in terms of computational complexity in comparison with the other two existing versions.

To achieve this, the three versions were implemented in MATLAB [78]. The solution of the system of equations was performed making use of MATLAB functions.

The complexity of the algorithms was measured in flops by making use of the MATLAB floating point operations counting facilities. Data sequence lengths (N) of 64, 128, 256 and 512 were considered, as those obtained with the Doppler signal simulations in study. Also having in mind the application of these methods to the Doppler signal, the model order values considered were even numbers, ranging only from 2 to 10, as typically occurred in Chapter 5.

Analyzing the variation of the number of flops with the model order in the graphs (Fig. 6.4) obtained for the different data sequence lengths (N), it is noted that the new algorithm present a computational complexity of the same order of magnitude as Marple's version, both presenting a much lower number of flops than the Kay version. This result is shown in Fig. 6.4, where data sequence lengths of 256 and 512 are considered as examples, and the new, Kay, and Marple methods are quoted as "new", "old(Kay)" and "old(Marple)".

The development of the number of flops with the length of the data sequence for a particular model order (p) value also proves the similarity of computational complexity of both the new and the Marple's algorithms (see Fig. 6.5 for particular case of p=4).



(a) (b)

Fig. 6.4: Comparison of computational complexity between new (-) and old versions of Kay (--) and Marple (:) of the modified covariance method versus model order, for data sequences of **(a)** 256 and **(b)** 512 points.

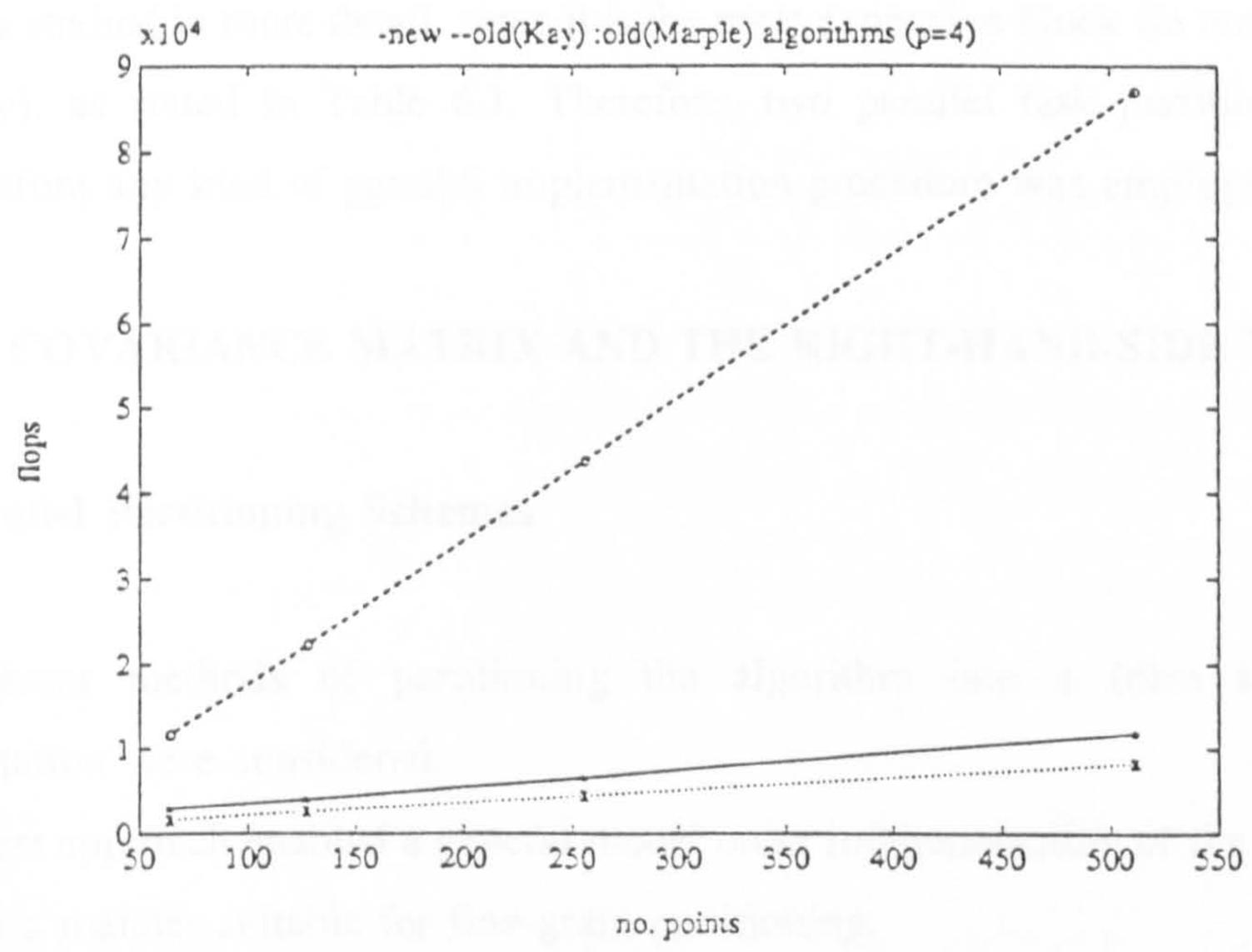


Fig. 6.5: Comparison of computational complexity between new (-), Kay (--) and Marple (:) versions of the modified covariance method versus no. of points of the data sequence (p=4).

6.2.4. Strategy for the Parallel Implementation of the New Modified Covariance Algorithm

The modifications introduced into Kay's modified covariance algorithm resulted in a sequential computational complexity similar to Marple's version. This last algorithm is much too complex to be reliably parallelized, and, as is explained below, the new algorithm is parallelized in a straightforward manner.

Each main computational block of the algorithm (the computation of the covariance matrix and the right-hand-side vector, the solution of the system of linear equations using a Cholesky algorithm, the computation of the white noise estimate, and, finally the determination of the PSD of the signal), were implemented in parallel separately. For each block, alternative parallel implementations were considered, performance measurements taken, and the implementation presenting the better performance identified.

These best implementations (of each computational block) were put together to form a single parallel program and final performance results taken.

The modified covariance algorithm block concerned with the computation of the covariance matrix was studied in more detail, since it is the most expensive block (in terms of computational complexity), as stated in Table 6.1. Therefore, two parallel task partitioning schemes were studied, before any kind of parallel implementation procedure was employed.

6.3. THE COVARIANCE MATRIX AND THE RIGHT-HAND-SIDE VECTOR

6.3.1. Parallel Partitioning Schemes

Two different methods of partitioning the algorithm into a form suitable for parallel implementation were considered.

The first approach enabled a general model order implementation of the modified covariance method in a manner suitable for fine-grain partitioning.

The other approach was specifically oriented for the Doppler spectrum under consideration and the algorithm was partitioned assuming a medium-grain parallelism.

A. Fine-grain Partitioning

Observing from (6.4) to (6.6) that the major computational times, for each element of the matrix, were produced during the calculation of the summations, each summation was partitioned into four smaller summations, assuming that at this stage a maximum of four processors were available. Each small summation was computed in parallel by one of the four processors. The terms before and after the summations were distributed to the processors in an organized way to avoid idle times and reduce the communications among the processors.

Table 6.2 represents a possible way of performing such a distribution of tasks for the computation of the elements shown in Fig. 6.2.

P_1	P_2	P_3	P_4	FLOPS
$C_{5,5}$ | 4	$C_{5,5}$ | 4	$C_{5,5}$ | 4	$C_{5,5}$ | 4	α
$C_{4,4}$ | 2	⇨	⇨	⇨	τ
$C_{5,5}$	$C_{4,4}$ | 2	$C_{3,3}$ | 2	$C_{2,2}$ | 2	τ
~	⇨	$C_{3,3}$ | 2	$C_{2,2}$ | 2	τ
$C_{4,4}$	$C_{1,1}$ | 2	⇨	⇨	τ
$C_{3,3}$	$C_{1,1}$ | 2	$C_{10,10}$ | 2	$C_{10,10}$ | 2	τ
$C_{2,2}$	⇨	⇨	⇨	τ
$C_{1,1}$	~	~	~	τ
$C_{10,10}$	~	~	~	τ
$C_{6,4}$ | 4	$C_{6,4}$ | 4	$C_{6,4}$ | 4	$C_{6,4}$ | 4	α
⋮	⋮	⋮	⋮	⋮
$C_{6,4}$ | 4	$C_{6,4}$ | 4	$C_{6,4}$ | 4	$C_{6,4}$ | 4	α
⇨	~	⇨	⇨	τ
~	$C_{0,10}$	~	~	τ

Table 6.2: Fine-grain partitioning for elements of Fig. 6.2.

In Table 6.2 the first time slice for each processor is spent computing a quarter (approximately) of the summations to determine element $C_{5,5}$. This is represented by $C_{5,5}$ / *4* and takes α flops. The next time slice is used by processors P_2,P_3,and P_4 to communicate to P_1 the

real result of the partial summation - indicated by the symbol ⇨. Meanwhile P_1 spends the same amount of time computing the first two terms of element $C_{4,4}$, which is represented in the table by $C_{4,4}$ / 2. Element $C_{4,4}$ may only be determined after $C_{5,5}$ and its additional terms have been completely calculated and the results communicated to the processor that is going to perform the final sum of partial results. In this scheme processor P_1 is idle for a time slice corresponding to τ flops. Idle times are represented in Table II by the symbol ⁓.

A similar procedure is repeated for all elements shown in figures 6.2 and 6.3.

B. Medium-grain Partitioning

The fine-grain parallelism described above requires a considerable amount of communications, as is evident from an analysis of Table 6.2. The overhead introduced by communications and idle times will reduce the profitability of the parallelism.

Therefore, another possible partitioning of the algorithm - computing the elements on the diagonal and its adjacent sub-diagonals on separate processors was considered.

To illustrate this alternative partitioning scheme, a 4^{th} order model will be used as an explanatory example, for reasons of simplicity.

In this new example, the elements to be computed are shown in Fig. 6.6. Elements $C_{2,2}$, $C_{3,1}$, and $C_{0,4}$ are elements lying on the main cross diagonal, requiring 2(N-p) flops, while $C_{2,1}$ and $C_{4,1}$ lie out of the main cross diagonal, requiring 2(N-p-1)+5 flops, as happened for corresponding elements in the 10^{th} order model given earlier. The other elements are obtained from those previously computed with an additional cost of 8 flops each.

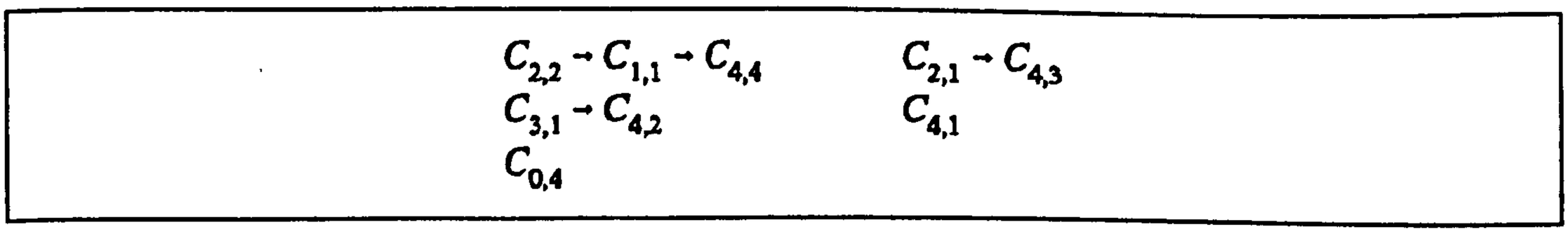


Fig. 6.6: Elements to compute the covariance matrix and right-hand-side vector for a 4^{th} order model.

In this approach, each horizontal line of the two groups of elements represented on Fig. 6.6 is computed on a separate processor. The first time-slice, each of the five (for this particular example) processors in use, is computing the element which needs to be obtained by computation

of a summation, that is, $C_{2,2}$, $C_{3,1}$, $C_{0,4}$, $C_{2,1}$ and $C_{4,1}$. In the next time slice all processors with exception of the third and fifth - which do not have subsequent elements - are calculating $C_{1,1}$, $C_{4,2}$, and $C_{4,3}$ respectively, in parallel. The last element calculated - $C_{1,1}$ - is computed in the first processor, using 8 flops, while all the others are already free. Table 6.3 elucidates this. In this table, the elements marked with * require 3 more flops than the number indicated.

P_1	P_2	P_3	P_4	P_5	FLOPS
$C_{2,2}$	$C_{3,1}$	$C_{0,4}$	$C_{2,1}$ *	$C_{4,1}$ *	2(N-p)
$C_{1,1}$	$C_{4,2}$	~	$C_{4,3}$	~	8
$C_{4,4}$	~	~	~	~	8

Table 6.3: Medium-grain partitioning for a 4th order model.

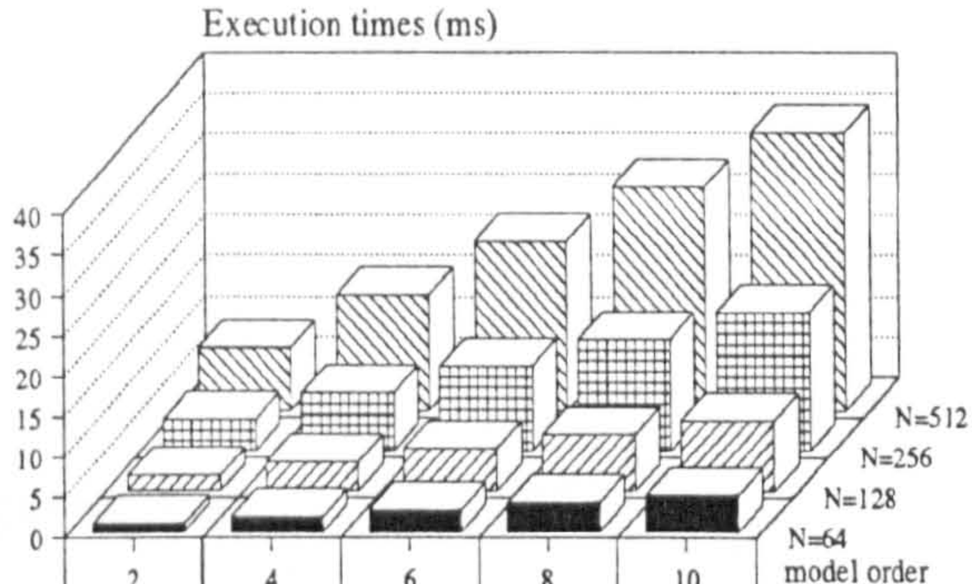
6.3.2. Parallel Implementation

This section describes the strategies used to implement the parallel realization for computing the coefficients of the modified covariance equations, on a transputer-based system. This involved two main aspects: identifying a suitable partitioning scheme to decompose the algorithm into a number of the parallel tasks, and utilising a suitable task allocation procedure to evenly distribute the resulting tasks onto the target parallel processing system.

Owing to its coarser granularity, the medium-grain partitioning scheme, described in section 6.3.1.B., proved to be the most straightforward for mapping parallel tasks onto the target processing system.

Typical measurements of performance such as execution time, speedup and efficiency [90] of each topology were determined by mapping the parallel version of the algorithm onto a T800-25 transputer platform [91], hosted by a SUN system, and programmed in OCCAM.

Before implementing the parallel realization of the covariance matrix and right-hand-side vector on a multiple transputer-based system, a sequential version of this computational block was implemented using a single transputer. The execution time of this sequential version was measured using the real-time clock facility available on the transputer architecture. Fig. 6.7 illustrates the execution times (in ms) for this sequential version for a range of p and N values.



	2	4	6	8	10
N=512	7.855	14.409	20.984	27.579	34.196
N=256	3.946	7.245	10.565	13.906	17.267
N=128	1.992	3.664	5.356	7.07	8.805
N=64	1.014	1.872	2.752	3.651	4.573

Fig. 6.7: Execution times obtained for the sequential version implementation of the computation of the covariance matrix and right-hand-side vector versus model order (p) and data sequence length (N).

These execution times acted as reference for the values obtained for the multiple-transputer implementations.

The transputer-based topologies adopted were:

- a linear processor farm topology,
- a tree topology (depth 3), where two different task allocation strategies (dynamic and static) were implemented, and,
- a tree topology (depth 2).

To simplify the explanation of how the allocation of the computation of the matrix and the right-hand-side vector elements onto each processor for each one of these topologies was carried out, Fig. 6.8 illustrates some abbreviations used, when a 10^{th} order model is employed. It considers the lower quarter part of the covariance matrix, resulting from the division of the remaining matrix by its main diagonal and main cross diagonal. This way, the elements of the remaining matrix, the elements of the last line of the covariance matrix, and also, the element of the right-hand-side vector (the element located at the bottom left side of the covariance matrix) to be fully computed, are contemplated. The elements whose calculation is based on elements lying on the cross diagonal or on the first parallel to the main cross diagonal (identified as elements not lying on the cross-diagonal in Fig. 6.3 for p=4), are also pointed out.

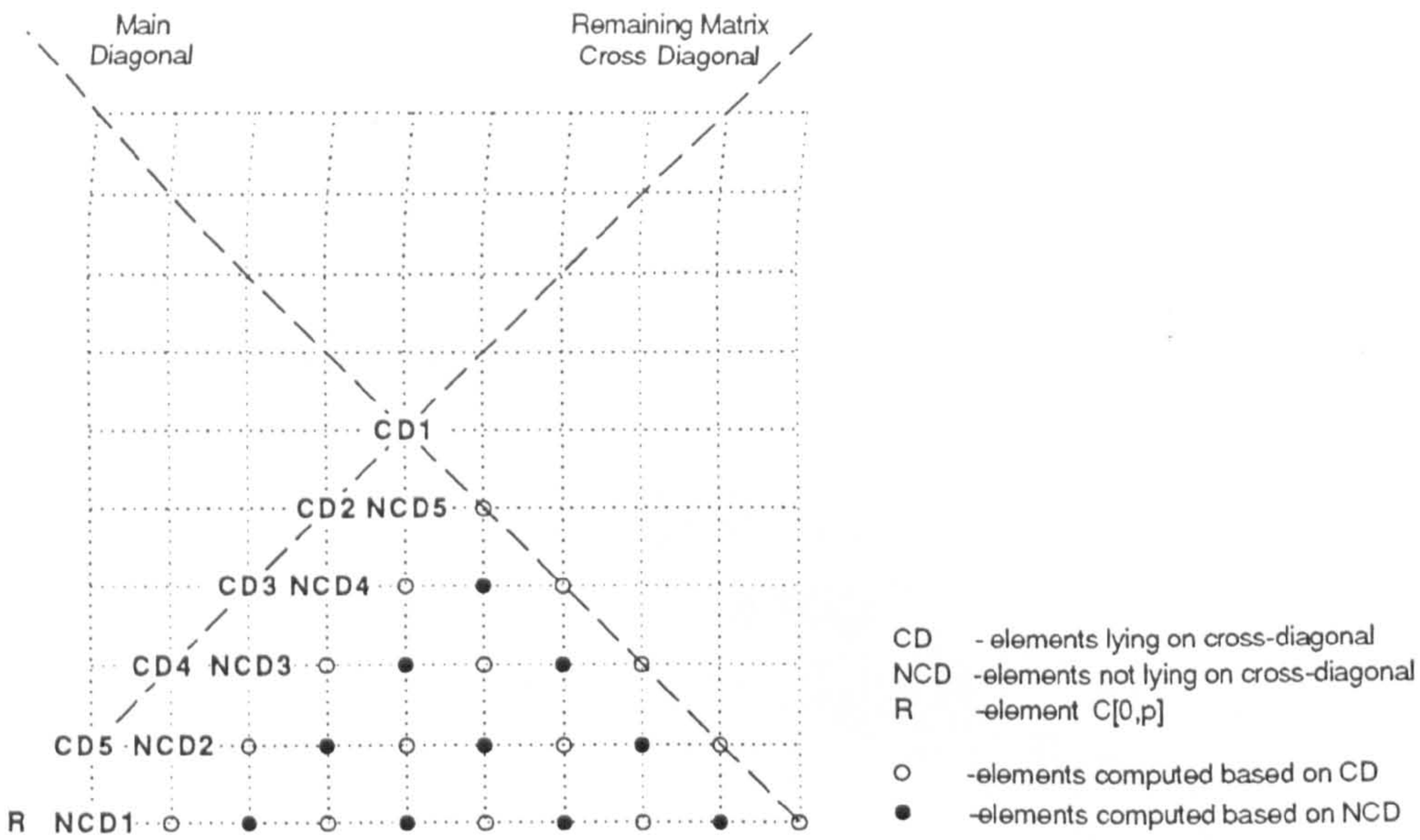


Fig. 6.8: Abbreviations used to identify the elements of the covariance matrix and right-hand-side vector of a 10^{th} (p=10).

The parallel implementation topologies used will now be described and their respective performance results presented.

A. Processor Farm Linear Topology

The most straightforward version of the processor farm model uses a single master processor for scheduling tasks to one or more worker processors connected in a line.

The computation task is the calculation of some of the elements of the covariance matrix and the right-hand-side vector (see Fig. 6.8 for a particular example). This task runs on each of the worker processors, and *routers* are used to route data and results through the network. Task identifiers, to define which tasks must be executed, and input data, are passed to the network as they are required, in order to utilise the full processing potential of the system. This provides an automatic mechanism for dynamic load balancing. Results are collected by the master as they are produced from the network. Finally, these results are used to generate the remaining elements. The processor farm linear model is shown in Fig. 6.9.

The execution time measurements of this topology were carried out varying N (the length of the input data) and the number of workers in each farm. The number of workers used was

equal to the model order plus one. Fig. 6.10 shows the resulting speedup and efficiency for the linear processor farm scheme employed.

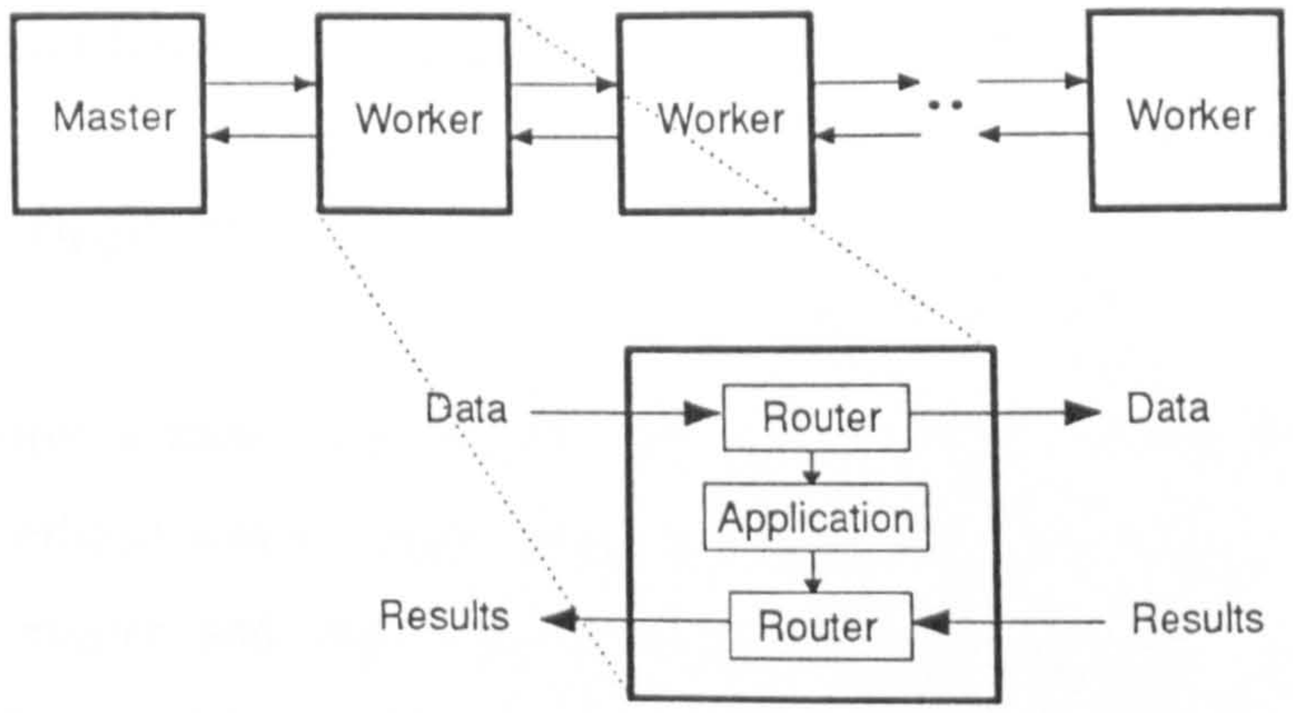
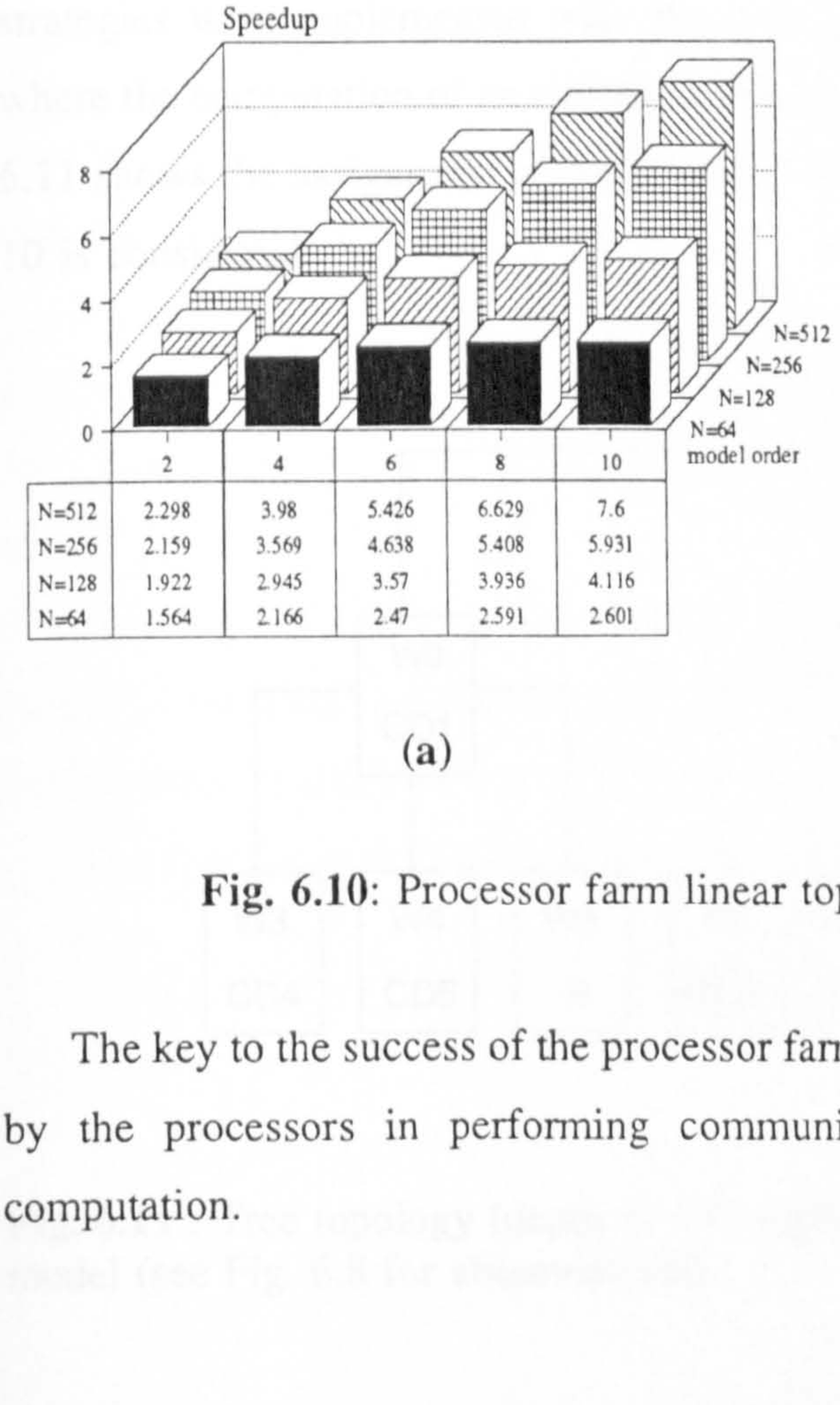
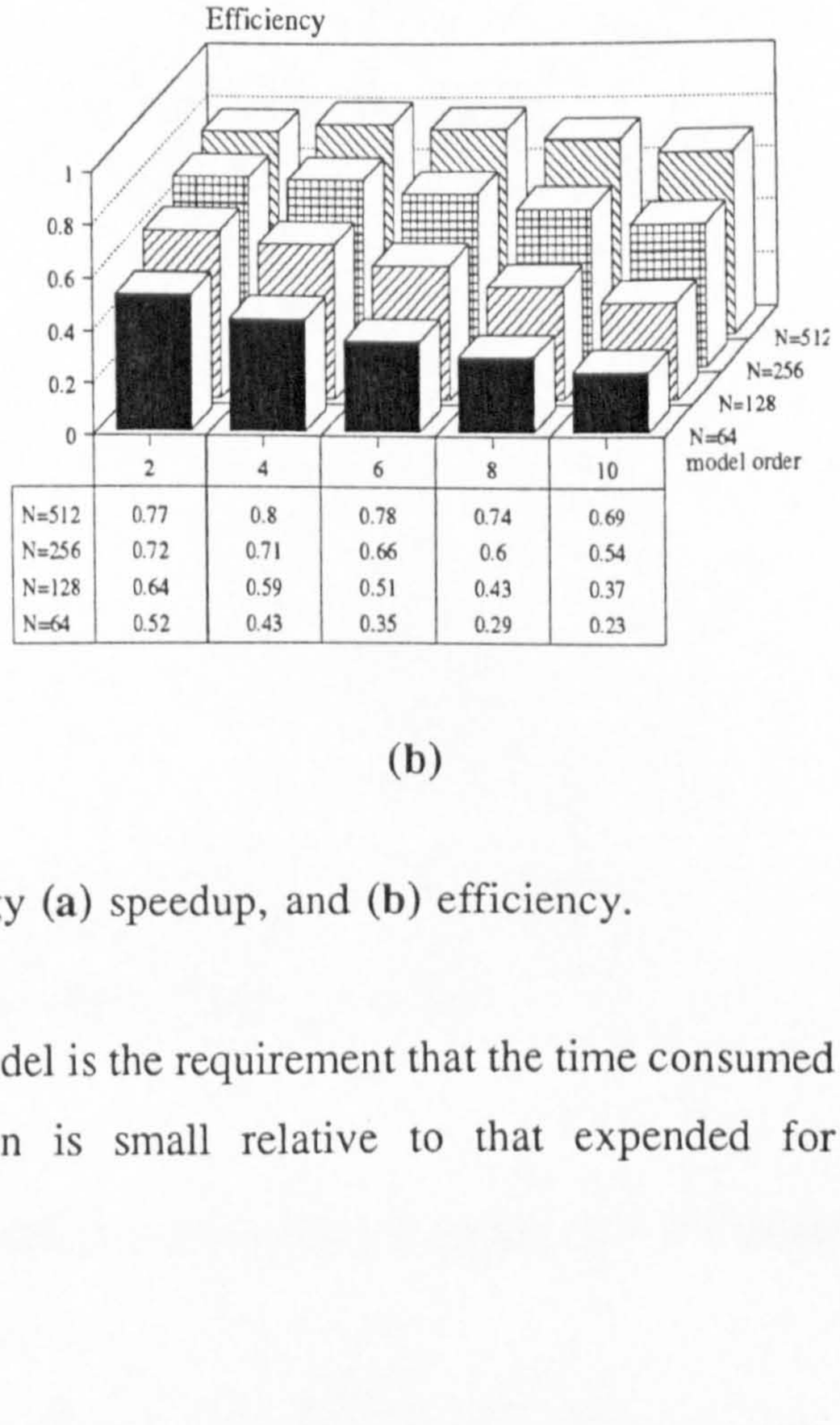


Fig. 6.9: Processor farm linear topology.



	2	4	6	8	10
N=512	2.298	3.98	5.426	6.629	7.6
N=256	2.159	3.569	4.638	5.408	5.931
N=128	1.922	2.945	3.57	3.936	4.116
N=64	1.564	2.166	2.47	2.591	2.601

(a)



	2	4	6	8	10
N=512	0.77	0.8	0.78	0.74	0.69
N=256	0.72	0.71	0.66	0.6	0.54
N=128	0.64	0.59	0.51	0.43	0.37
N=64	0.52	0.43	0.35	0.29	0.23

(b)

Fig. 6.10: Processor farm linear topology **(a)** speedup, and **(b)** efficiency.

The key to the success of the processor farm model is the requirement that the time consumed by the processors in performing communication is small relative to that expended for computation.

However, when this requirement is not matched and a significant overhead on communicating data items exists, this linear array may be unsuitable for any more than a small number of processors in the farm due to the limited throughput of this topology. Thus, the original linear model of the processor farm has been extended by configuring the system as a tree topology.

B. Tree Topology (Depth 3)

This topology required a more sophisticated task allocation strategy. However, a reduction in communications overhead was expected, because, in this extended approach, there is a shorter path between the master and any worker. This extended model has the same conceptual architecture as the first model: a single controller processor with a number of workers, but in this case the workers are grouped in a tree structure with three branches.

Within this tree model, the worker process retained a similar software structure, as in the original linear array, for computing the tasks, but the routers were modified to allow the routing of data and results to the rest of the workers down the same branch. Two different task allocation strategies were implemented with this tree model. The first one is a processor farm scheme, where the computation of an element of the matrix is attributed dynamically to each worker. Fig. 6.11 shows the assignment of the elements of the matrix per processor, when a model order of 10 is considered.

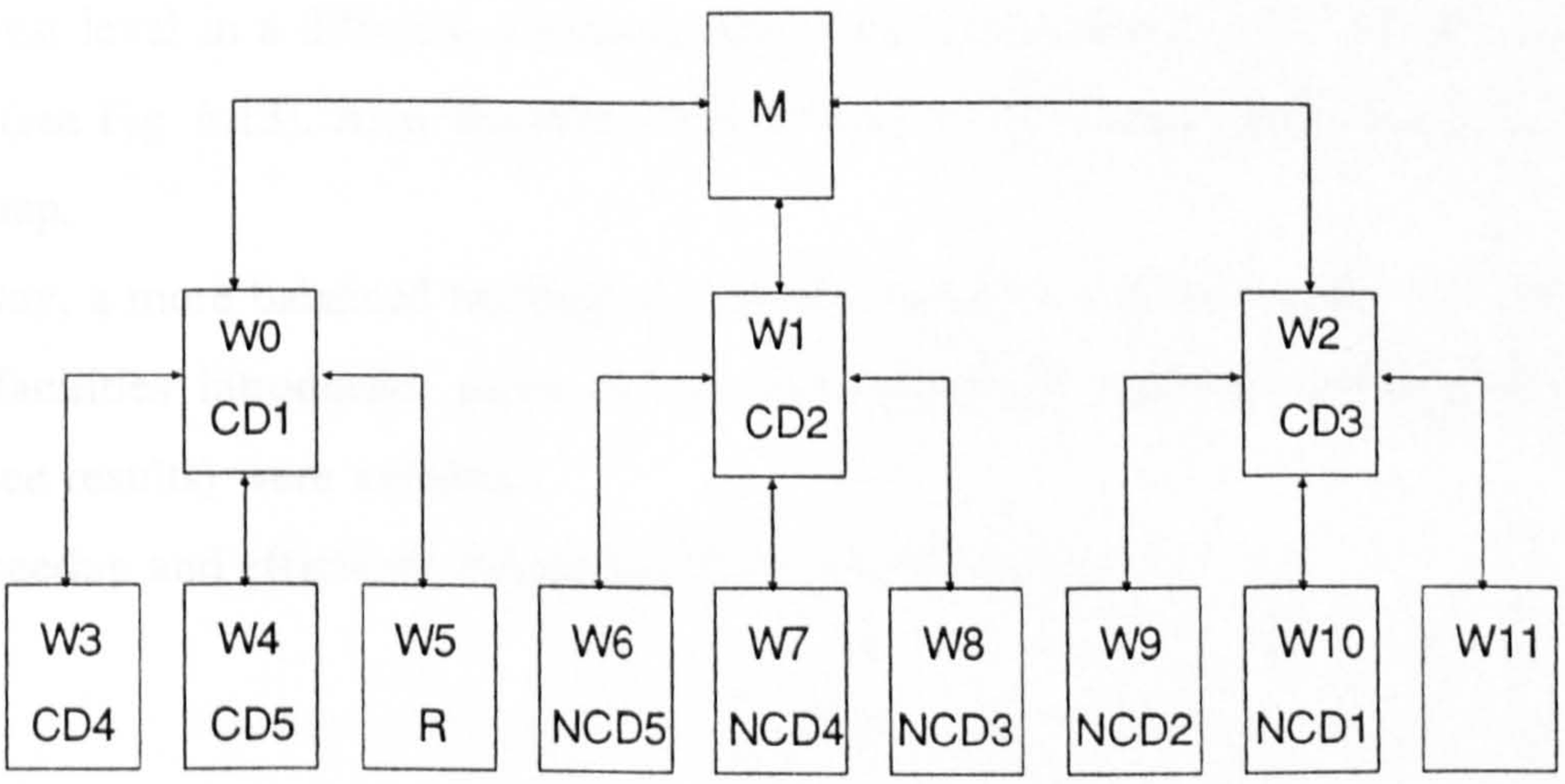
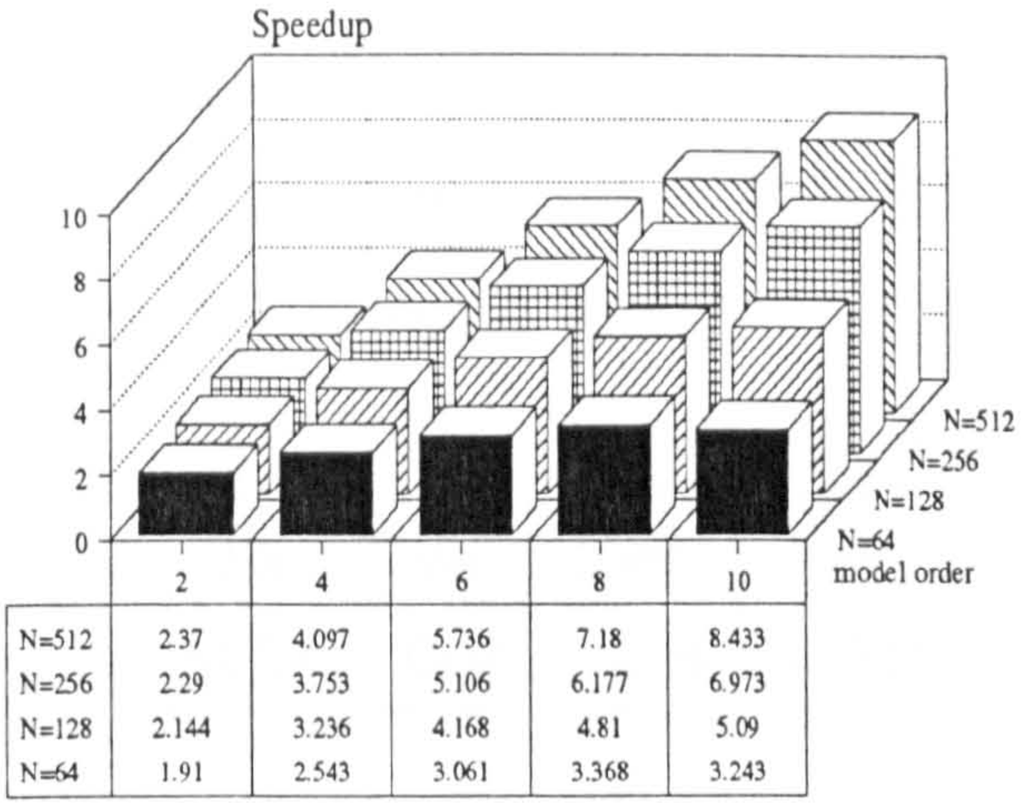


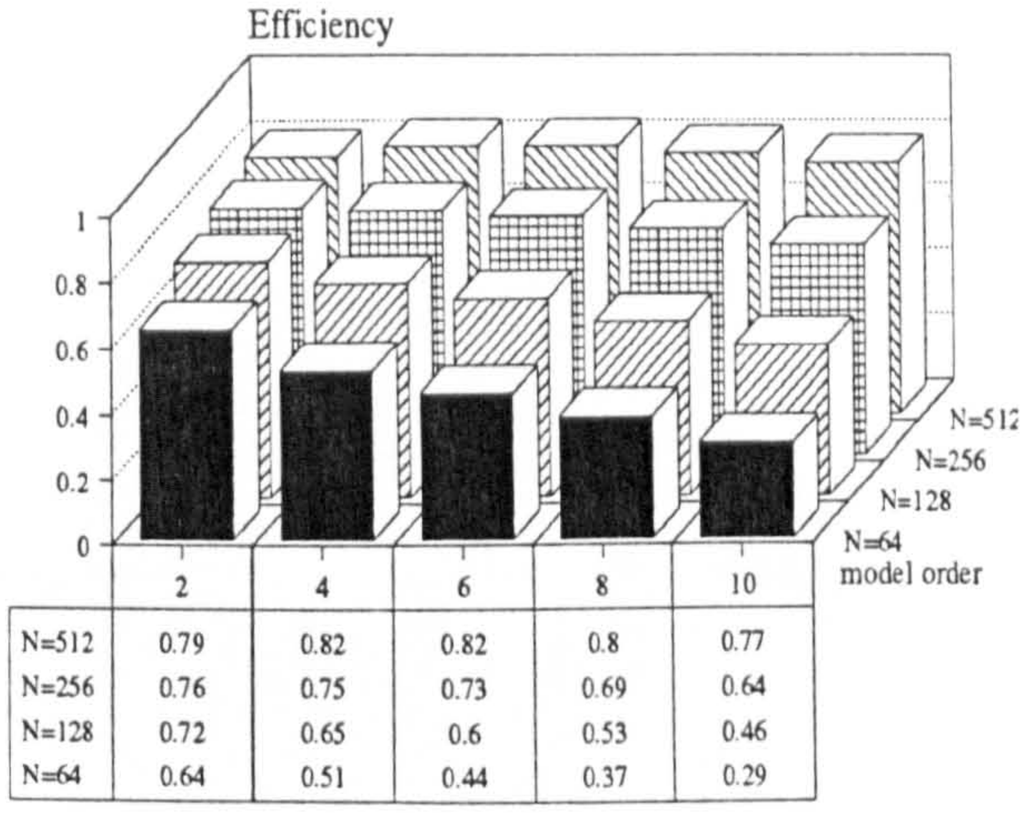
Fig. 6.11 : Tree topology (depth 3) - 1st approach and respective task allocation for a 10th order model (see Fig. 6.8 for abbreviations).

In this situation, the efficiency and speedup values obtained are depicted in Fig. 6.12.



	2	4	6	8	10
N=512	2.37	4.097	5.736	7.18	8.433
N=256	2.29	3.753	5.106	6.177	6.973
N=128	2.144	3.236	4.168	4.81	5.09
N=64	1.91	2.543	3.061	3.368	3.243

(a)



	2	4	6	8	10
N=512	0.79	0.82	0.82	0.8	0.77
N=256	0.76	0.75	0.73	0.69	0.64
N=128	0.72	0.65	0.6	0.53	0.46
N=64	0.64	0.51	0.44	0.37	0.29

(b)

Fig. 6.12: Tree topology (depth 3) - 1st approach **(a)** speedup and **(b)** efficiency.

The improvement obtained on the performance results when comparing with those obtained for the linear topology were smaller than what was expected. Therefore, an alternative allocation scheme was tried (see Fig. 6.13).

This second implementation approach using the tree topology (depth 3) numbered the workers of the lowest level in a different sequence, this being determined by the mother-worker of the first level (see Fig. 6.13). Also, the allocation of tasks to the workers became static, through the use of a map.

This way, a more balanced loading of each branch of the tree was achieved, and due to the mapping facilities introduced, some of the communication overheads (reflected on the last performance results) were avoided.

The speedup and efficiency results are illustrated in Fig. 6.14.

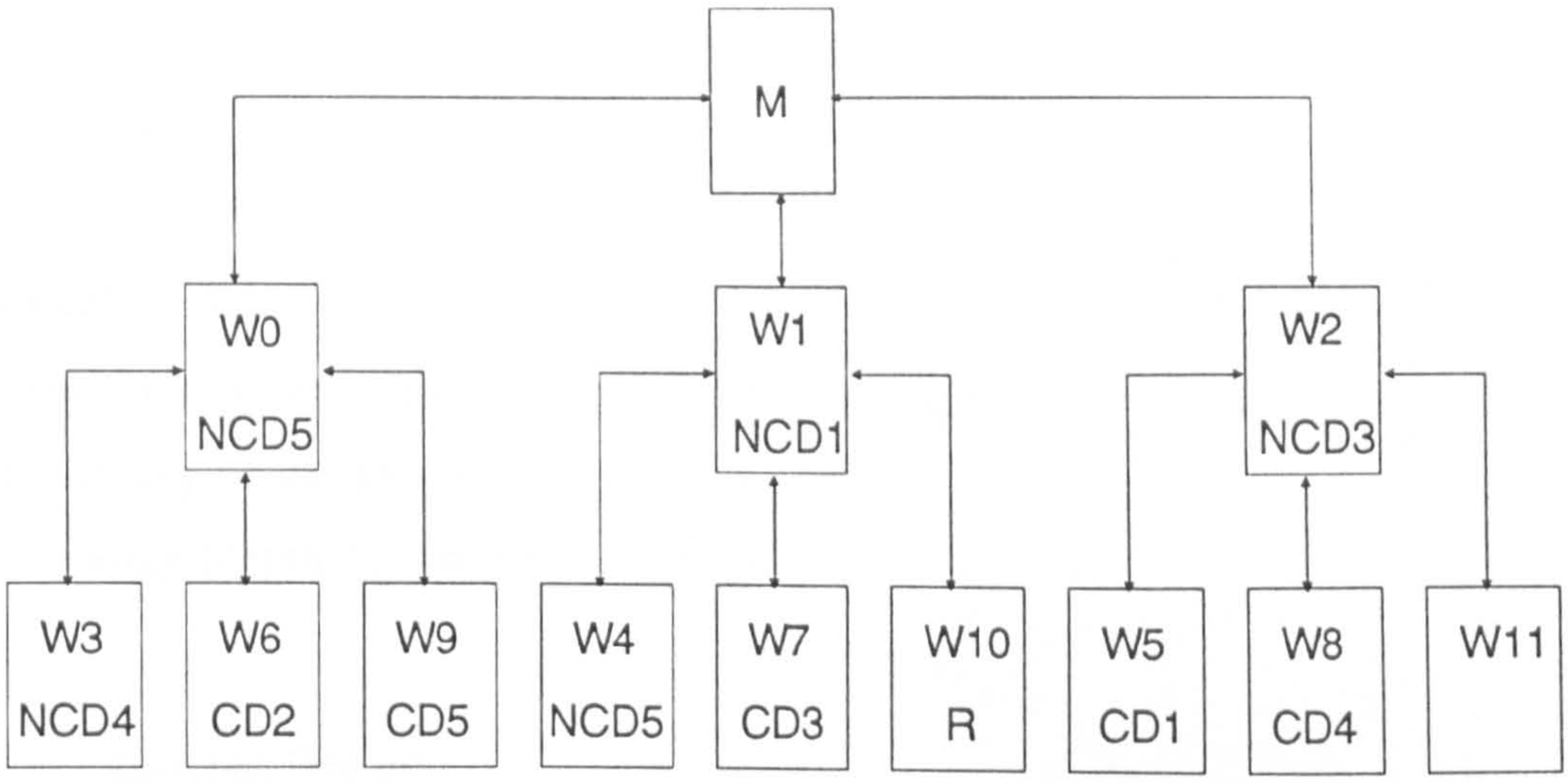
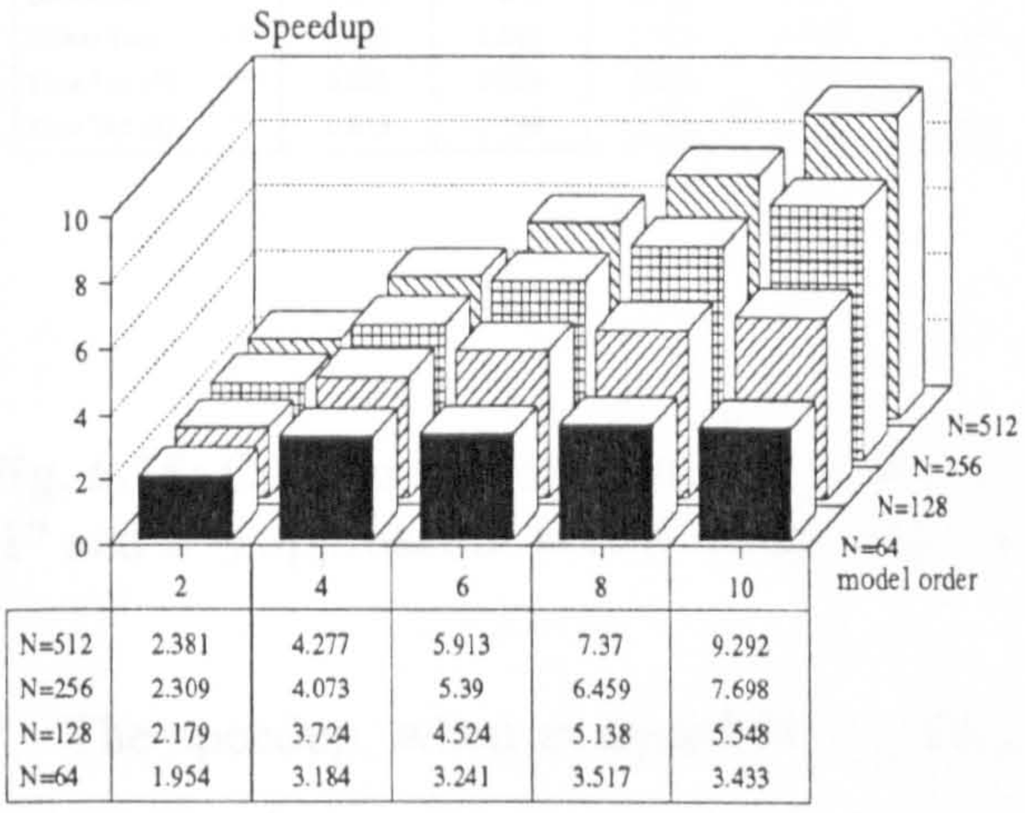
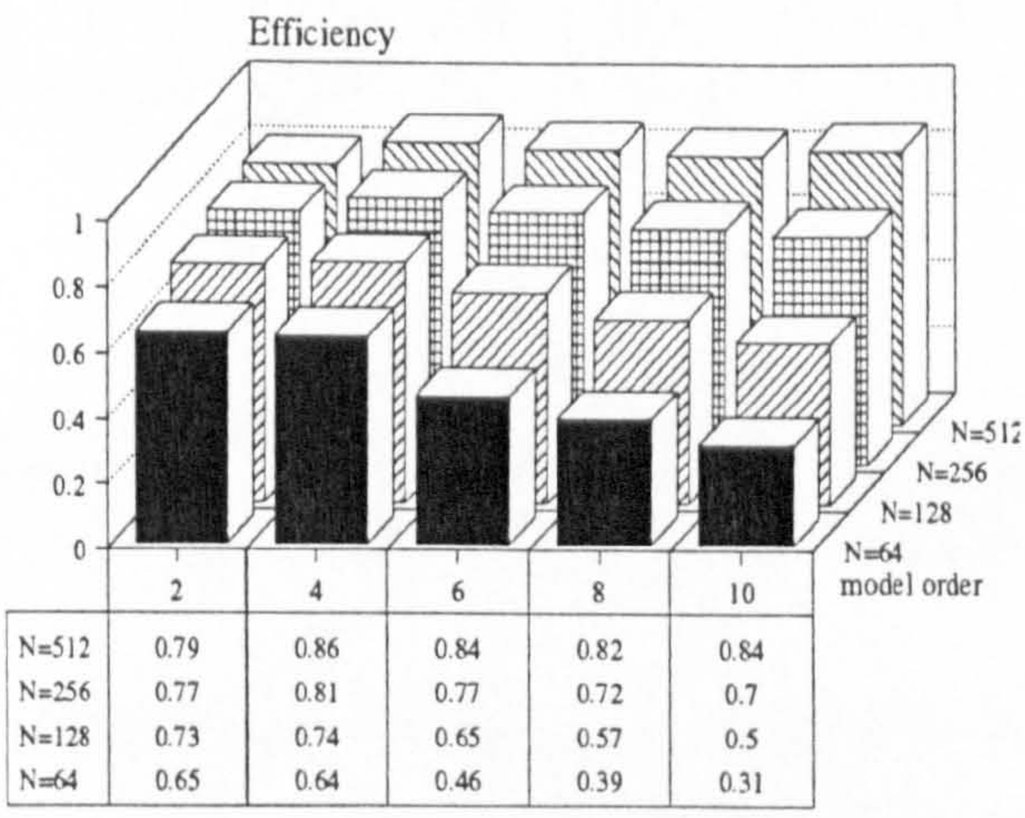


Fig. 6.13 : Tree topology (depth 3) - 2nd approach and respective task allocation for a 10th order model (see Fig. 6.8 for abbreviations).



	2	4	6	8	10
N=512	2.381	4.277	5.913	7.37	9.292
N=256	2.309	4.073	5.39	6.459	7.698
N=128	2.179	3.724	4.524	5.138	5.548
N=64	1.954	3.184	3.241	3.517	3.433

(a)



	2	4	6	8	10
N=512	0.79	0.86	0.84	0.82	0.84
N=256	0.77	0.81	0.77	0.72	0.7
N=128	0.73	0.74	0.65	0.57	0.5
N=64	0.65	0.64	0.46	0.39	0.31

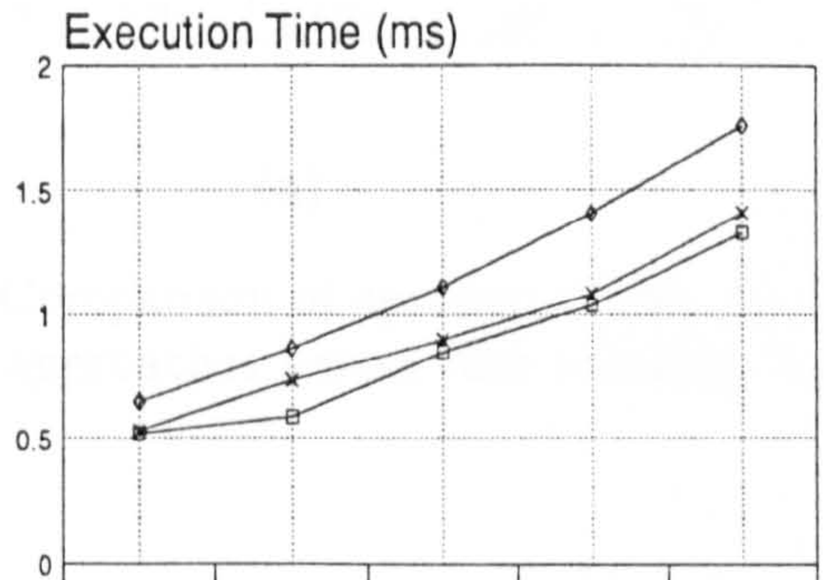
(b)

Fig. 6.14 : Tree topology (depth 3) - 2nd approach **(a)** speedup and **(b)** efficiency.

At this stage of the work, some graphs (Figs. 6.15 to 6.17) provide comparison of the three implemented schemes. This confirms that the tree topology (depth 3) using the second task allocation approach is the topology presenting best performance results.

Fig. 6.15 presents a comparison of the execution times taken by each scheme considered, for different model orders, and, the extreme cases of data sequence length, N=64 (Fig. 6.15 (a)) and N=512 (Fig. 6.15 (b)). These execution times refer to the implementation of the three approaches on the same transputer system. The tables underneath each graph include also, as a reference, the respective values of execution times obtained for the implementation of a sequential version of the covariance matrix and right-hand-side vector. From these reference values, the improvement obtained with any of the parallel schemes employed, and particularly with the static version of the tree topology (depth 3), becomes evident.

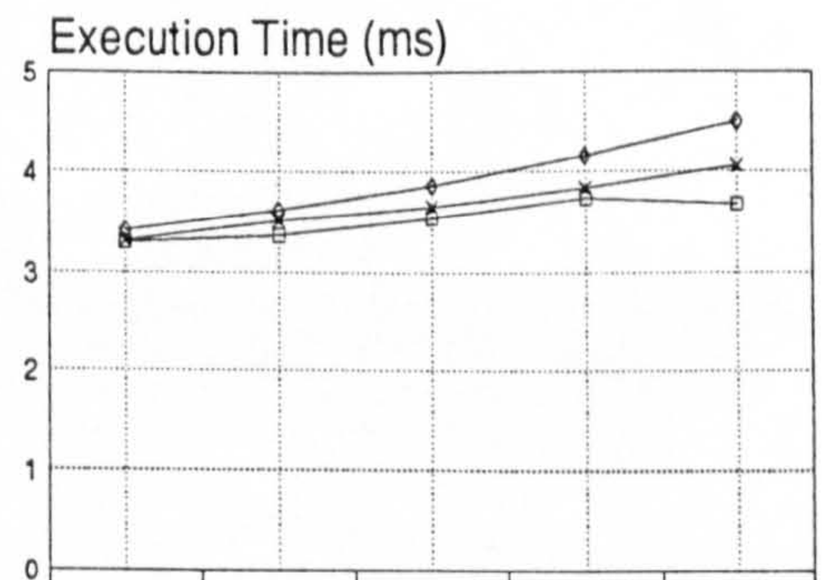
N=64



Model Order	p=2	p=4	p=6	p=8	p=10
Sequential	1.014	1.872	2.752	3.651	4.573
Linear Top.	0.648	0.864	1.114	1.409	1.758
Tree Top.(1)	0.531	0.736	0.899	1.084	1.41
Tree Top.(2)	0.519	0.588	0.849	1.038	1.332

(a)

N=512



Model Order	p=2	p=4	p=6	p=8	p=10
Sequential	7.855	14.409	20.984	27.579	34.196
Linear Top.	3.417	3.624	3.867	4.16	4.499
Tree Top.(1)	3.314	3.517	3.658	3.841	4.055
Tree Top.(2)	3.299	3.369	3.549	3.742	3.68

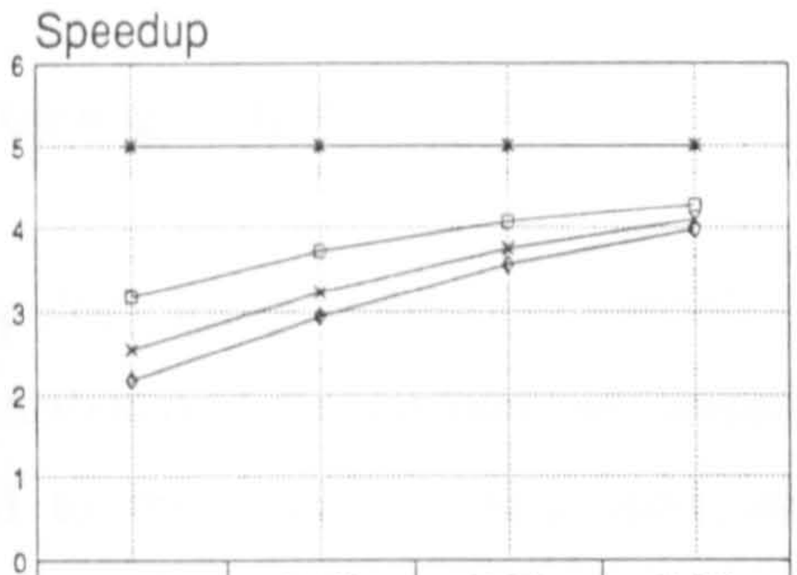
(b)

Fig. 6.15: Comparison of execution times obtained with the processor farm linear, tree (depth 3) (1^{st} and 2^{nd}) approaches versus model order, for data sequence lengths (N) of **(a)** 64 and **(b)** 512.

The speedup, when analyzed for a particular model order and N values ranging from 64 to 512, leads also to the conclusion that the static task allocation approach of the tree (depth 3) topology (depth 3) performs best.

Fig. 6.16 illustrates the speedup results obtained when a 4^{th} order model (Fig. 6.16 (a)), and a 10^{th} order model (Fig. 6.16 (b)) were implemented on a processor farm with 5 and 11 active workers, respectively.

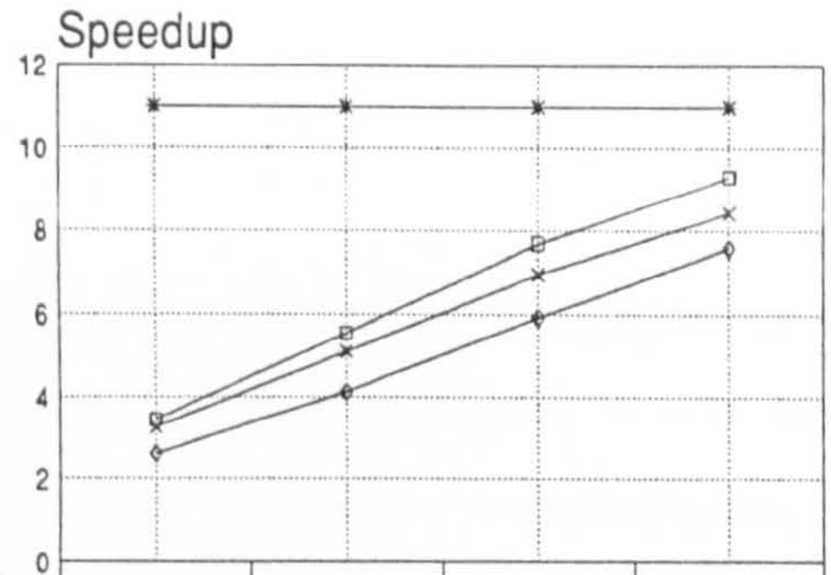
p=4



Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal	5	5	5	5
Linear Top.	2.166	2.945	3.569	3.98
Tree Top.(1)	2.543	3.236	3.753	4.097
Tree Top.(2)	3.184	3.724	4.073	4.277

(a)

p=10

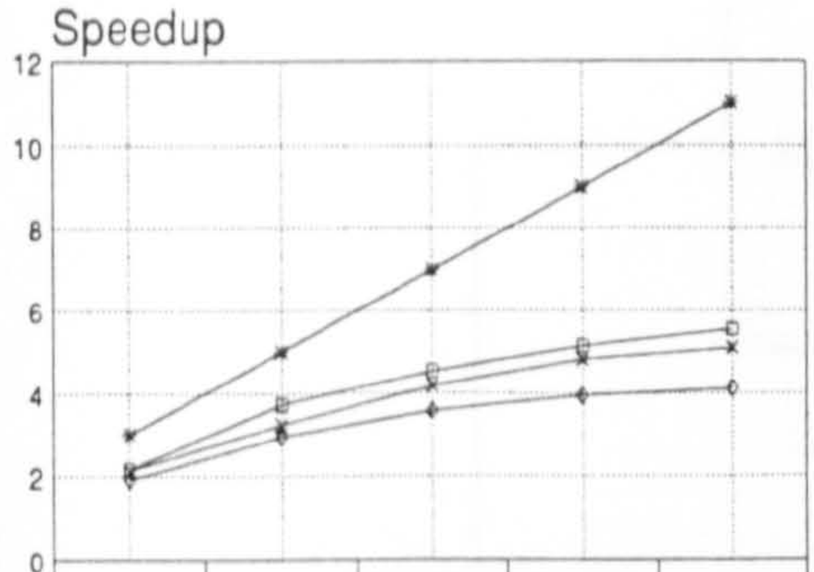


Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal	11	11	11	11
Linear Top.	2.601	4.116	5.931	7.6
Tree Top.(1)	3.243	5.09	6.973	8.433
Tree Top.(2)	3.433	5.548	7.698	9.292

(b)

Fig. 6.16: Comparison of speedup values obtained with the processor farm linear, tree (depth 3) 1st and 2nd approaches versus data sequence lengths, for model orders (p) of **(a)** 4 and **(b)** 10.

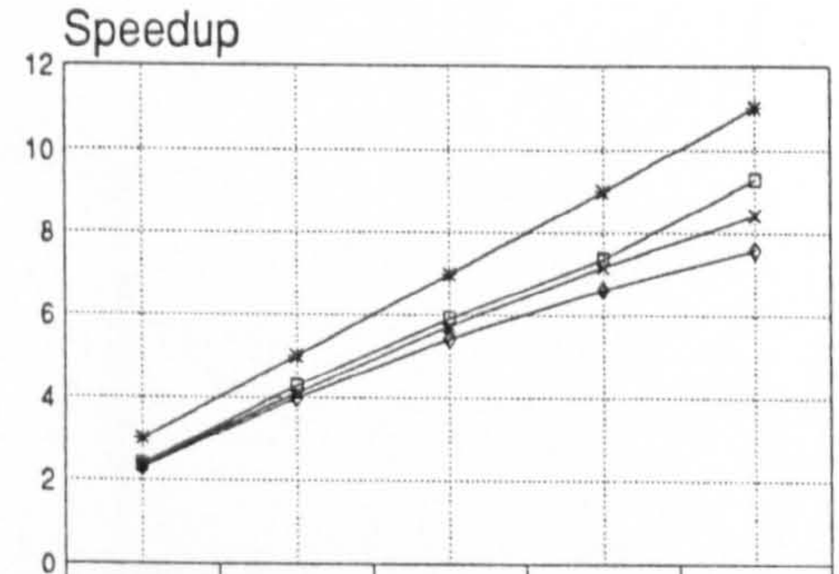
N=128



Model Order	p=2	p=4	p=6	p=8	p=10
Optimal	3	5	7	9	11
Linear Top.	1.922	2.945	3.57	3.936	4.116
Tree Top.(1)	2.144	3.236	4.168	4.81	5.09
Tree Top.(2)	2.179	3.724	4.524	5.138	5.548

(a)

N=512



Model Order	p=2	p=4	p=6	p=8	p=10
Optimal	3	5	7	9	11
Linear Top.	2.298	3.98	5.426	6.629	7.6
Tree Top.(1)	2.37	4.097	5.736	7.18	8.433
Tree Top.(2)	2.381	4.277	5.913	7.37	9.292

(b)

Fig. 6.17: Comparison of speedup values obtained with the processor farm linear, tree (depth 3) 1st and 2nd approaches versus model orders, for data sequence lengths (N) of **(a)** 128 and **(b)** 512.

If instead the analysis of speedup is carried out varying the order of the model for each data sequence length considered (see Fig. 6.17 for cases N=128 and N=512), again the tree topology

(depth 3) with the second task allocation approach is the one closest to the optimal speedup line. The values assumed by this optimal line are determined by the number of active workers.

C. Tree Topology (Depth 2)

The last topology implemented was specifically concerned with improving the load balancing of the tasks attributed to the workers, and reducing the communications between them. It was also constrained to the usage of only three workers and a master processor, contemplating a commonly encountered practical situation where only a hardware board with four transputers is available.

This topology uses only a single-level of workers, whose tasks are statically attributed by a pre-defined map, which enables the allocation of the computation of the matrix elements in workers, according to the model order in use and satisfying the load balancing and communication requirements stated above. Fig. 6.18 illustrates this topology and the tasks attributed to each worker for the case of a model order of 10. (See Fig. 6.8 for abbreviations used).

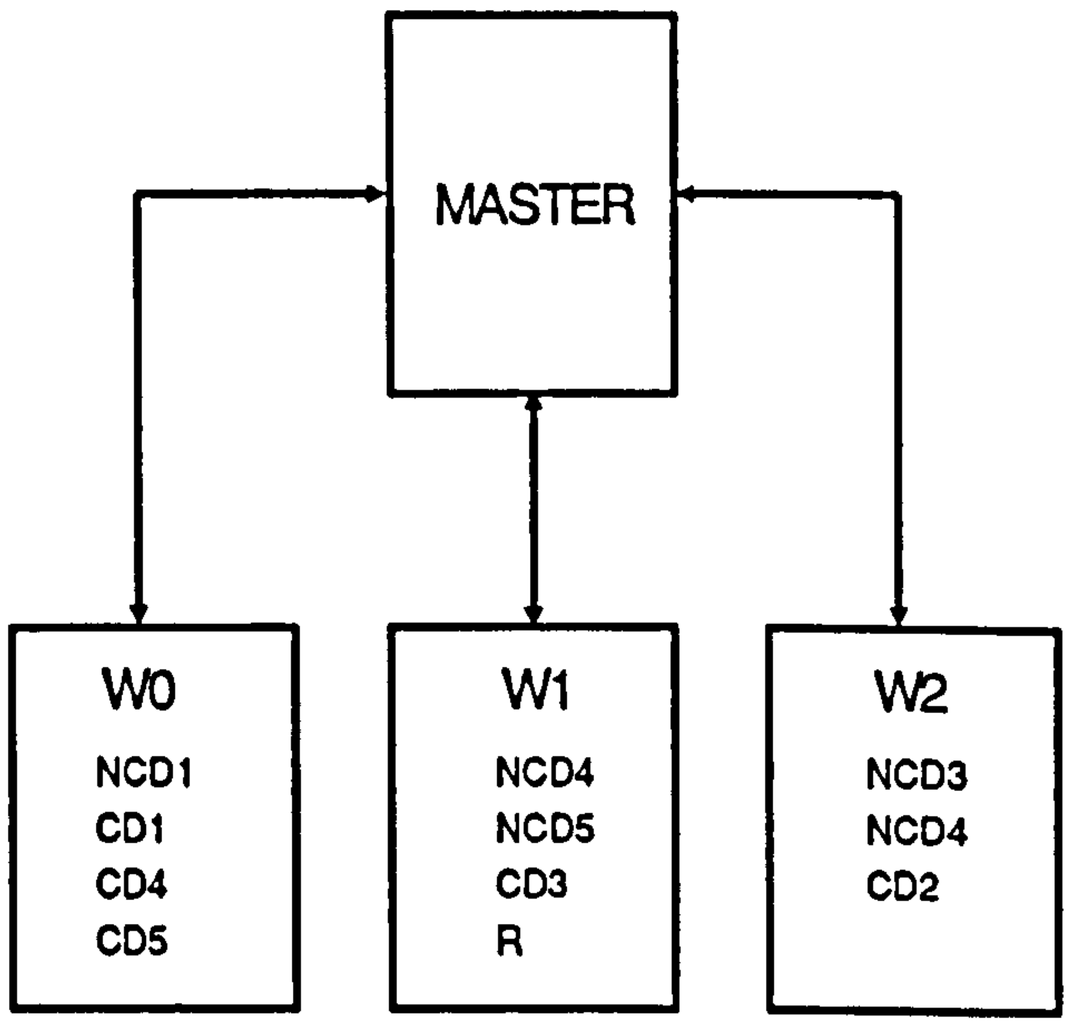
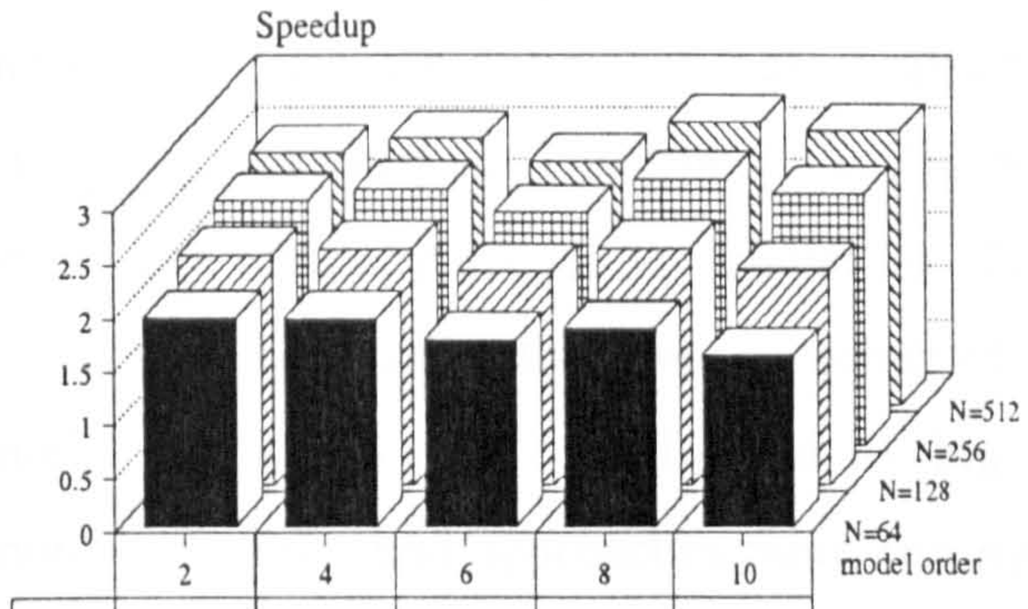


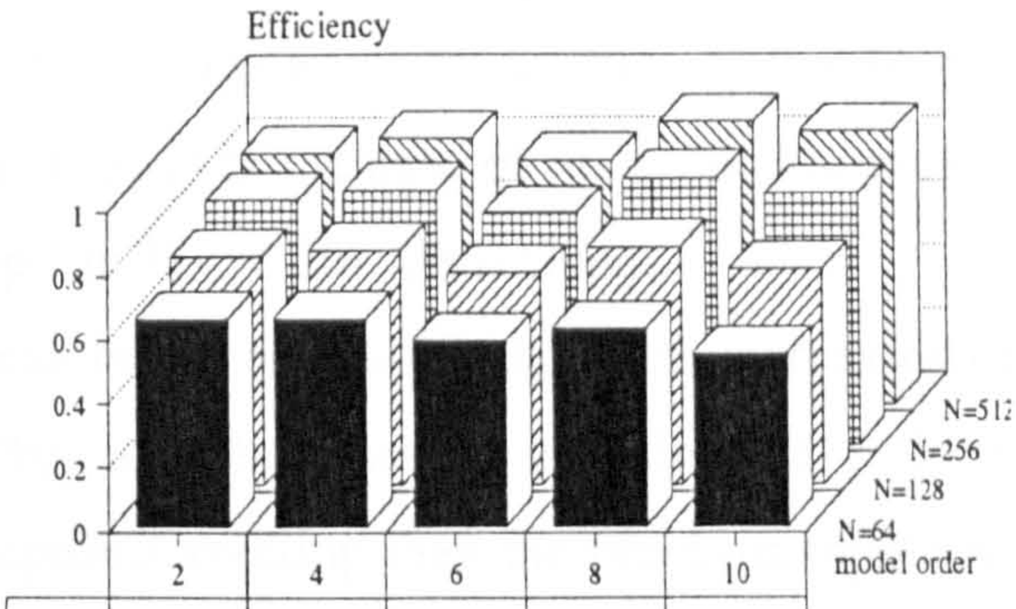
Fig. 6.18: Tree topology (depth 2) and respective task allocation for a 10^{th} order model (see Fig. 6.8 for abbreviations).

The performance results obtained with this topology are presented in Fig. 6.19.



	2	4	6	8	10
N=512	2.38	2.515	2.301	2.676	2.594
N=256	2.305	2.412	2.198	2.51	2.373
N=128	2.172	2.231	2.023	2.242	2.035
N=64	1.954	1.946	1.751	1.861	1.605

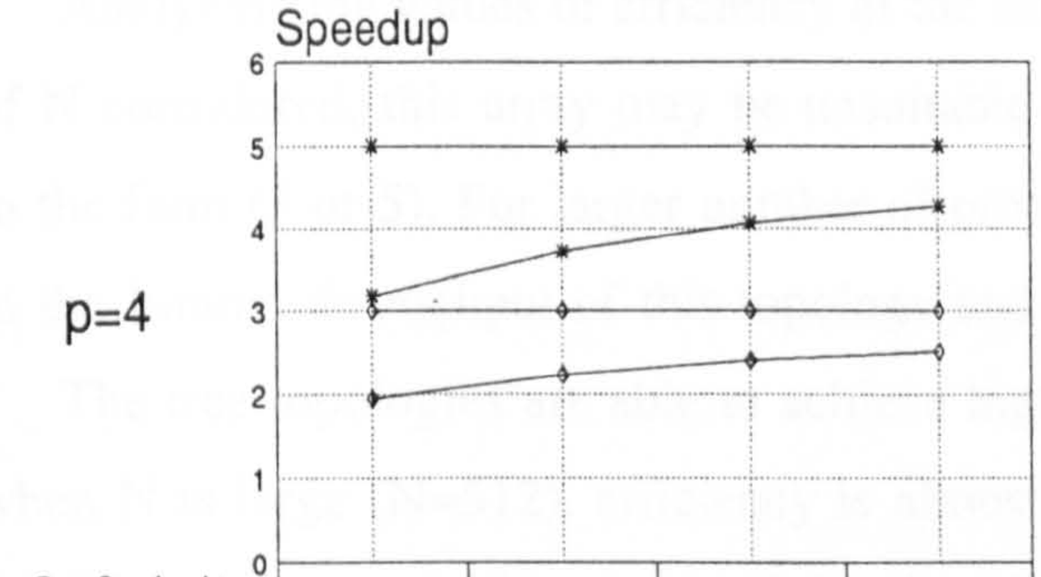
(a)



	2	4	6	8	10
N=512	0.79	0.84	0.77	0.89	0.86
N=256	0.77	0.8	0.73	0.84	0.79
N=128	0.72	0.74	0.67	0.75	0.68
N=64	0.65	0.65	0.58	0.62	0.54

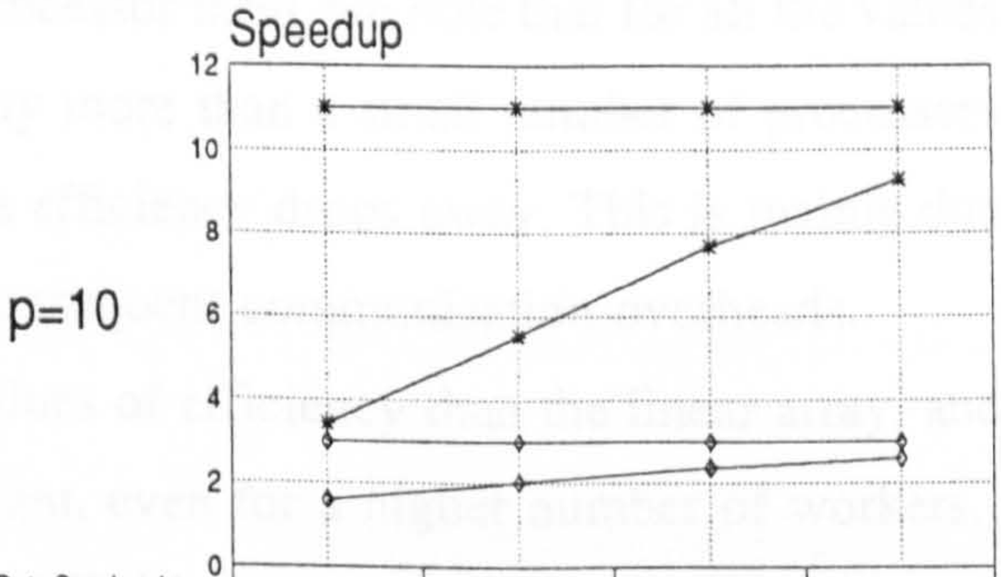
(b)

Fig. 6.19: Tree topology (depth 2) **(a)** speedup and **(b)** efficiency.



Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal (tt)	5	5	5	5
Optimal (tp)	3	3	3	3
Three proc.	1.946	2.231	2.412	2.515
Tree Top.(2)	3.184	3.724	4.073	4.277

(a)



Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal (tt)	11	11	11	11
Optimal (tp)	3	3	3	3
Three proc.	1.605	2.035	2.373	2.594
Tree Top.(2)	3.433	5.548	7.698	9.292

(b)

Fig. 6.20: Comparison of speedup values obtained with the tree (depth 3) 2nd approach ('Tree top.(2)') and tree (depth 2) ('Three Proc.') topologies versus data sequence lengths, for model orders of **(a)** 4 and **(b)** 10. The optimum values for 'Tree Top.(2)' and 'Three Proc.' are indicated by 'Optimal ((tt)' and 'Optimal (tp)' respectively.

Fig. 6.20 allows the comparison of speedup values obtained for this topology and the tree topology (depth 3) using a static task allocation. For better understanding of the behaviour of these topologies, the optimal values of speedup for each topology (3 in the case of tree (depth 2) topology (quoted as 'Three proc.' in Fig. 6.20), and (p+1) in the case of the second approach of the tree topology (depth 3)) (quoted as 'Tree Top.' in Fig. 6.20) are included.

In fact, comparing the speedup achieved by these two schemes for the particular cases of model orders of 4 (Fig. 6.20 (b)) and 10 (Fig. 6.20 (b)), it is clear that the tree (depth 2) follows more uniformly, and approaches more closely, the optimal speedup than the previous topology.

D. Performance Analysis

In the current realisation all parallel implementations achieved speedup. This was higher for the tree (depth 2) topology. In all cases the speedup was most significant when N was large, due to the coarser grain size. Speedup was also higher for higher values of model order, except for the tree (depth 2) topology, where the relationship between the model order (p) and the number of active workers (number of workers=p+1) is no longer true.

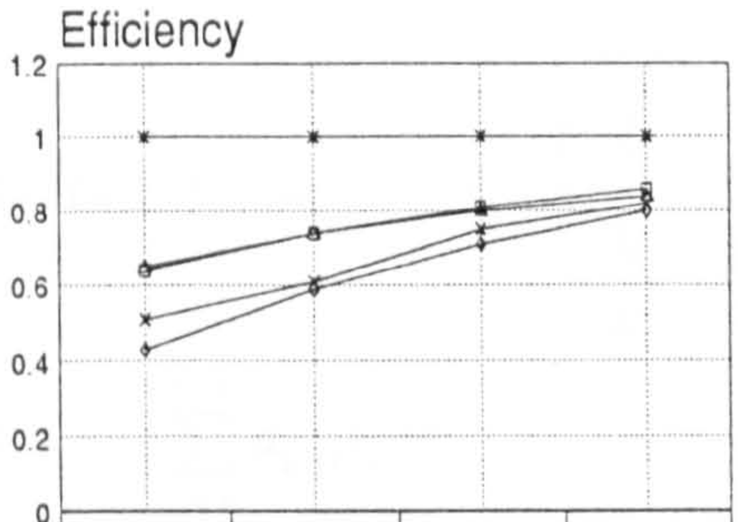
Analyzing the values of efficiency of the linear processor farm, we note that for all the values of N considered, this array may be unsuitable for any more than a small number of processors in the farm (3 or 5). For larger number of processors efficiency drops away. This is mainly due to the limited throughput of this topology and the consequent communication overheads.

The tree topologies are able to achieve higher values of efficiency than the linear array, and when N is large (N=512), efficiency is almost constant, even for a higher number of workers.

Of the tree topologies, the depth 2 offers the most efficient use of processors, as may be seen in Fig. 6.21 for different data sequence lengths and model order values of 4 and 8. Also, this topology is less sensitive to unbalanced loading of the workers, since the relative weight of a task to the overall weight of the tasks attributed to a particular worker, is less in the case of the depth 3 tree than in the other topologies considered.

Thus the depth 2 tree topology offers the best performance. It requires a more sophisticated task allocation strategy, however it offers a significant reduction in communication overheads as a result of shorter paths between the master and workers.

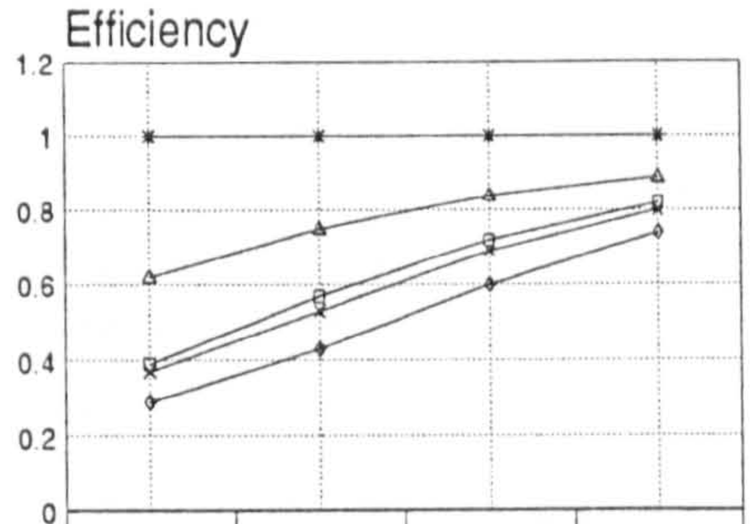
p=4



Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal	1	1	1	1
Linear Top.	0.43	0.59	0.71	0.8
Tree Top.(1)	0.51	0.61	0.75	0.82
Tree Top.(2)	0.64	0.74	0.81	0.86
Three Processors	0.65	0.74	0.8	0.84

(a)

p=8



Data Seq. lenght	N=64	N=128	N=256	N=512
Optimal	1	1	1	1
Linear Top.	0.29	0.43	0.6	0.74
Tree Top.(1)	0.37	0.53	0.69	0.8
Tree Top.(2)	0.39	0.57	0.72	0.82
Three Processors	0.62	0.75	0.84	0.89

(b)

Fig. 6.21: Comparison of efficiency values obtained with all the implemented topologies versus data sequence lengths, for model orders of **(a)** 4 and **(b)** 8.

6.4. SOLUTION OF THE LINEAR SYSTEM OF EQUATIONS

6.4.1. General Description

The solution of linear systems of equations is efficiently computed in sequential computers using Cholesky decomposition [17] [32] [37].

The solution of a system of equations $\mathbf{Ax=b}$, where $\mathbf{A}$ is a square matrix of dimension n, $\mathbf{x}$ and $\mathbf{b}$ are column vectors with size n, is performed in three steps:

Step 1- Decompose $\mathbf{A}$ into a product $\mathbf{LDL}^H$, where $\mathbf{L}$ (n×n) is a lower triangular matrix with ones as elements of its principal diagonal, $\mathbf{D}$ (n×n) is a diagonal matrix with real and positive elements on the principal diagonal , and, $\mathbf{L}^H$ is the Hermitian conjugate of the matrix $\mathbf{L}$. The $\mathbf{L}$ and $\mathbf{D}$ elements are computed by

$$d_1 = a_{11}$$

$$\text{for } i = 2,3,\ldots,n \qquad l_{ij} = \begin{cases} \dfrac{a_{i1}}{d_1} & \text{for } j = 1 \\ \dfrac{a_{ij}}{d_j} - \displaystyle\sum_{k=1}^{j-1} \frac{l_{ik}.d_k.l_{jk}^*}{d_j} & \text{for } j = 2,3,\ldots,i-1 \end{cases} \tag{6.8}$$

$$d_i = a_{ii} - \sum_{k=1}^{i-1} d_k.|l_{ik}|^2$$

Step 2- Set $\mathbf{y}=\mathbf{DL}^H\mathbf{x}$, and solve the system $\mathbf{Ly}=\mathbf{b}$ by back-substitution, thus

$$y_1 = b_1$$

$$y_k = b_k - \sum_{j=1}^{k-1} l_{kj}.y_j \qquad \text{for } k = 2,3,\ldots,n \tag{6.9}$$

Step 3- Compute the final solution, by determining $\mathbf{x}= (\mathbf{L}^H\mathbf{D}^{-1})^{-1}\mathbf{y}$, which again employs the backsubstitution recursion

$$x_n = \frac{y_n}{d_n}$$

$$x_k = \frac{y_k}{d_k} - \sum_{j=k+1}^{n} l_{jk}^*.x_j \qquad \text{for } k = n-1, n-2,\ldots,1 \tag{6.10}$$

Two parallel implementations of the Cholesky algorithm have been carried out following the above approach.

6.4.2. Parallel Implementations

Both parallel implementations used a transputer-based ring topology (see Fig. 2.4) with three processors. The highly recursive structure of the Cholesky algorithm did not enable a complete parallelization of the program. The partitioning of the algorithm may be described as follows:-

PAR

... *compute L*

... *compute D*

... *solve Ly=b*

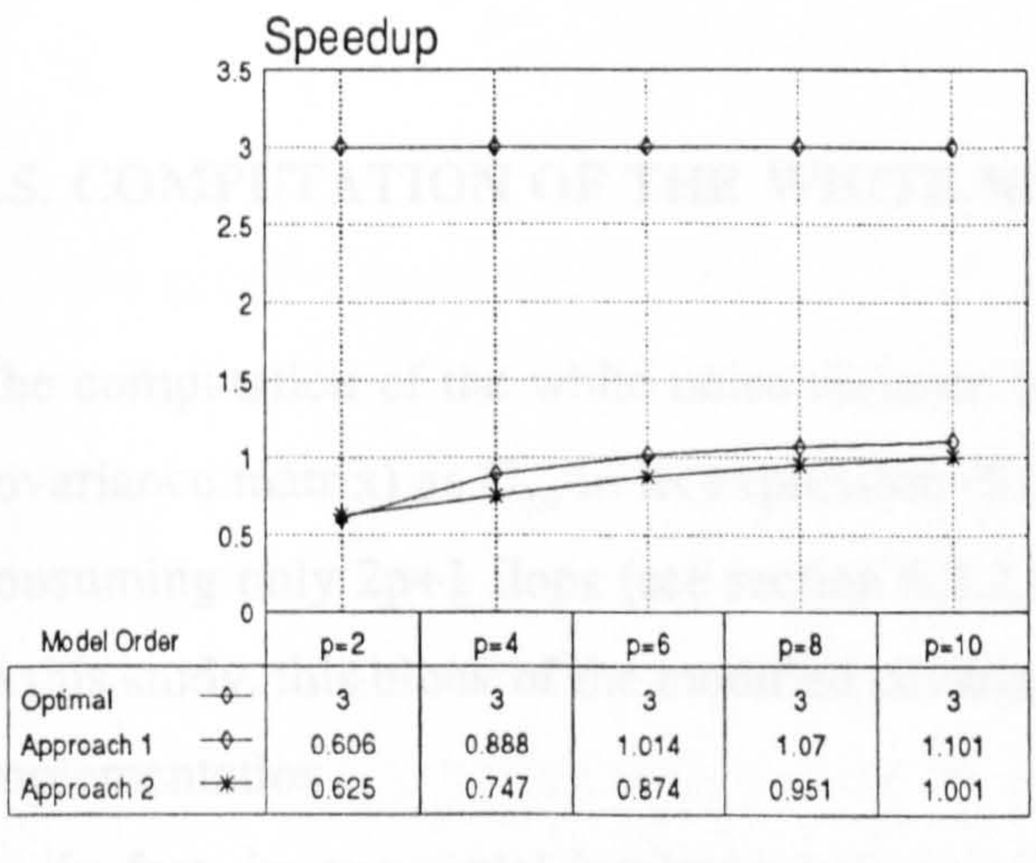
SEQ

... *compute final solution*

The main difference between the implementations of the above partitioning scheme is concerned with the way of establishing communications among the processors.

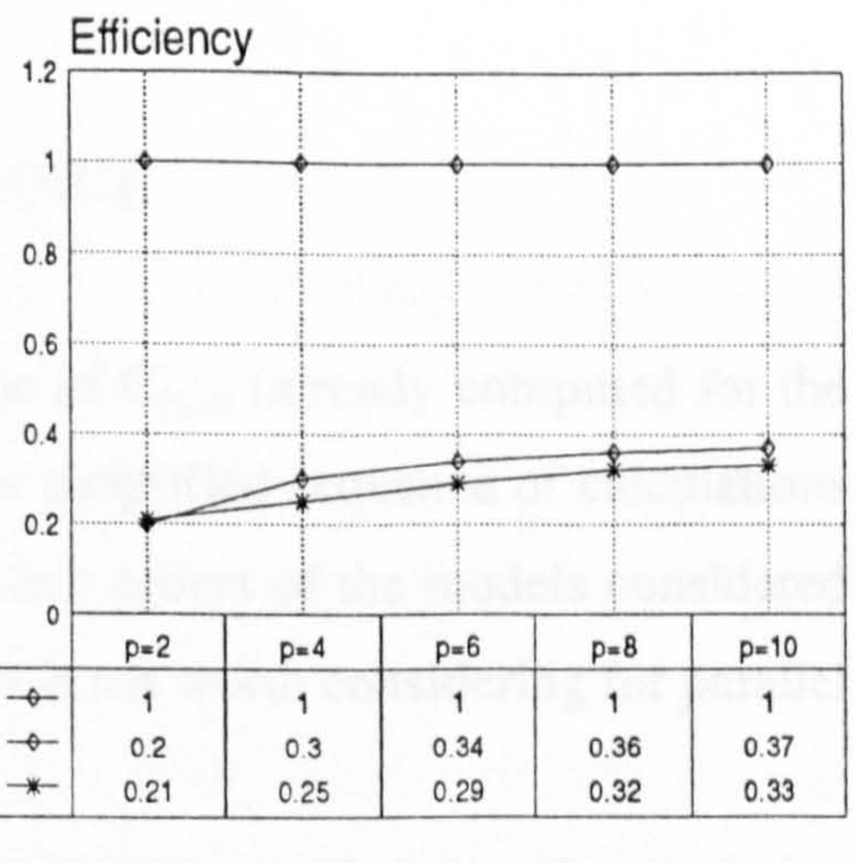
In the first approach, the processor in charge of the computation of the **L** elements, computed a whole column of the matrix and only afterwards communicated the values of the elements to the other processors, so that they could calculate the **D** elements and solve **y** (viewed as a sum of partial solutions).

In the second approach, as soon as the first element of each column of **L** is determined, it is sent to the other processors which immediately start the operations, and then, when the rest of the elements of that particular column are computed, are sent as a vector to the remaining processors.



Model Order	p=2	p=4	p=6	p=8	p=10
Optimal	3	3	3	3	3
Approach 1	0.606	0.888	1.014	1.07	1.101
Approach 2	0.625	0.747	0.874	0.951	1.001

(a)



Model Order	p=2	p=4	p=6	p=8	p=10
Optimal	1	1	1	1	1
Approach 1	0.2	0.3	0.34	0.36	0.37
Approach 2	0.21	0.25	0.29	0.32	0.33

(b)

Fig. 6.22: Results of (**a**) speedup and (**b**) efficiency obtained with two parallel implementations of the Cholesky algorithm.

Despite the effort, parallelization was unsatisfactory, as may be seen in Fig. 6.22. The execution times obtained for both implementations were approximately the same as if a completely sequential implementation of the algorithm was performed on the same transputer system (see Fig. 6.23). This result has been found elsewhere [92].

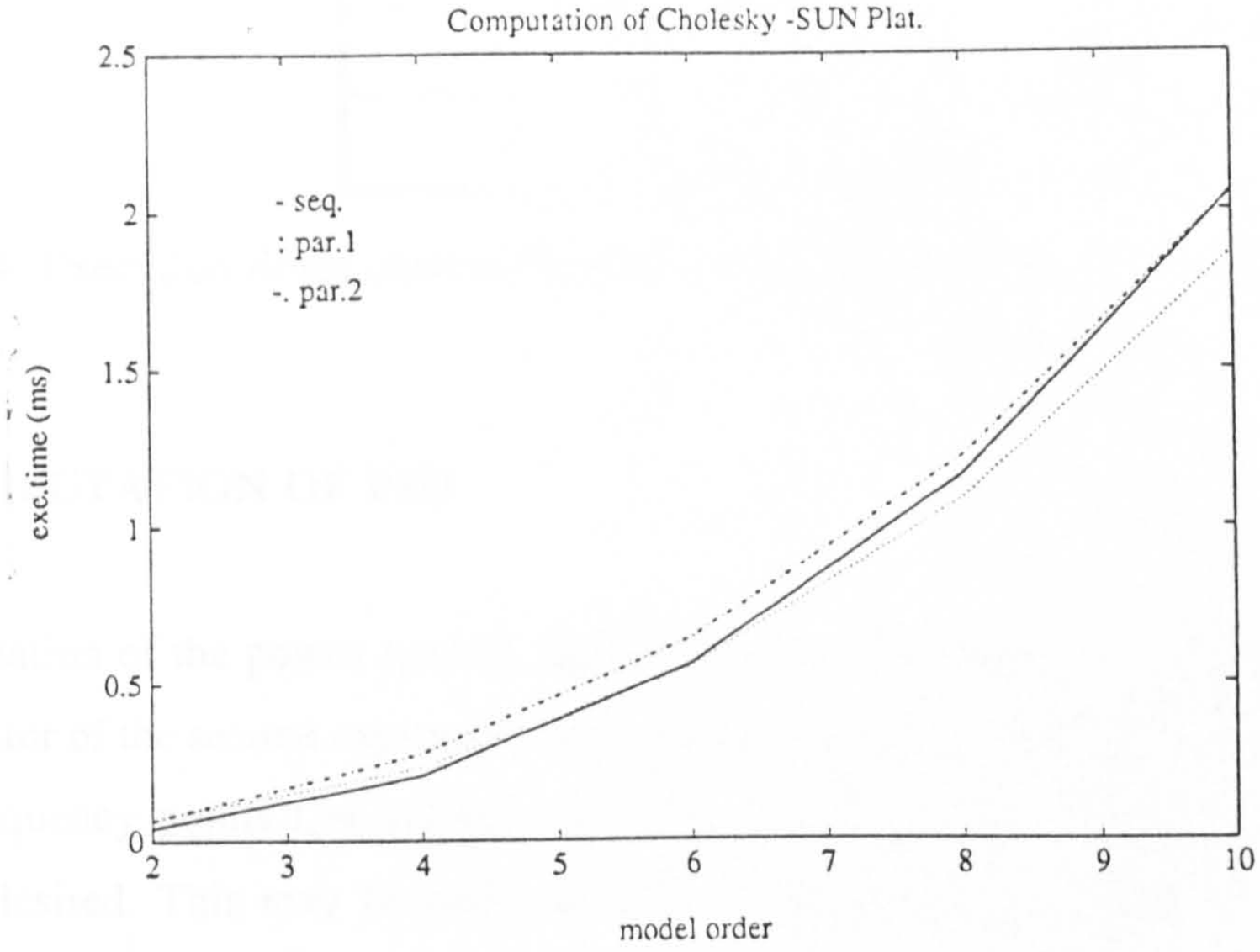


Fig. 6.23: Comparison of execution times observed for sequential and two different parallel implementations of Cholesky algorithm.

6.5. COMPUTATION OF THE WHITE NOISE VARIANCE

The computation of the white noise variance using the value of $C_{10,10}$ (already computed for the covariance matrix) as $C_{0,0}$ in its expression (6.3), results in a simplified sequence of calculations consuming only 2p+1 flops (see section 6.2.2.). Due to the low orders of the models considered in this study, this block of the modified covariance algorithm is not worth considering for parallel implementation.

In fact, its sequential implementation on one of the processors revealed that the execution times increased linearly with the model order, confirming the theoretical complexity of 2p+1 (section 6.2.2.).

Table 6.4 presents the execution times obtained for the sequential implementation on the transputer-based system considered, of the white noise variance computation, when model orders ranging from 2 to 10 (even values) were employed.

model order	execution times (ms)
2	0.011
4	0.019
6	0.026
8	0.034
10	0.043

Table 6.4: Execution times obtained for the computation of the white noise variance estimate.

6.6. COMPUTATION OF PSD

The estimation of the power spectral density ($P_{AR}(f)$) of the signal is obtained using (6.11). The denominator of the second expression is evaluated over the normalized frequency range [-1/2,1/2] at the frequency points $f_n=(-1/2)+(n-1)/N$, where n=1,...,N, and N is the number of frequency samples desired. This may be achieved in two ways, whether employing the DFT or, its fast implementation, the FFT.

$$P_{AR}(f_n) = \frac{\sigma^2}{|A(f_n)|^2} = \frac{\sigma^2}{\left| 1+\sum_{k=1}^{p} a[k].z^{-k} \Big|_{z=e^{j2\pi f_n}} \right|^2} \qquad (6.11)$$

The parallel implementation of the computation of the PSD was initially devised to include an available parallel implementation of the FFT [93]. However, it proved difficult to adapt this program to the SUN-hosted transputer system employed, and on which all other performance measurements were taken, and this lead to the development of a simplified parallel DFT, adapted to the type of signals used.

Current work is being carried out on the implementation of the FFT algorithm [94] on a PC-based transputer system [53] [95], in conjunction with a research colleague. This FFT algorithm is used on a project concerned with the development of alternative implementations of the modified covariance spectral estimation method [96].

The performance results of the FFT implementation may be evaluated as part of the overall performance of the modified covariance spectral estimation algorithm, reported in section 6.8.

The procedure used to compute and implement in parallel PSD calculations using the simplified DFT is now described.

6.6.1. Implementation of a Simplified DFT based PSD calculation

After the computation of the model parameter estimates and white noise variance estimate, the usual sequence of procedures [17] to obtain a PSD, may be summarized in the following steps:

Step 1 - Create a vector whose first element is unity (the first term of the summation in the denominator of (6.11)), the following p elements being the p model parameter estimates, and the rest of the elements, up to a total of N (number of points required in the spectrum) are made equal to zero;

Step 2 - Compute its Fourier transform, normally using the FFT algorithm. Here the simplified DFT will be used;

Step 3 - Evaluate the squared modulus of the complex vector obtained in Step 2, and, using the white noise estimate, compute the values of the PSD for N points;

Step 4 - Transpose halves of the spectrum and summarize the results into a vector of N/2 elements for graphical purposes.

Some simplifications have been carried out in the above procedure.

The first simplification regards the computation of the data vector (**d**) of Step 1, where only the first p+1 terms are now considered.

Another simplification is that the computation of the Fourier transform needs only to be performed for only half the number of frequency points (N/2), since the spectrum of real signals is symmetric and, for visual presentation of the spectrum only the last half of the spectrum is usually considered. Thus,

$$X[n-\frac{N}{2}] = \sum_{k=0}^{p} d[k].e^{-j2\pi n\frac{k}{N}} \qquad for\ n=\frac{N}{2},...,(N-1) \tag{6.12}$$

where the X[0],....,X[(N/2)-1] vector elements were used to evaluate the final PSD values,

$$P[i] = \frac{\sigma^2}{Re[X[i]]^2 + Im[X[i]]^2}. \qquad for\ i=0,1,...,(\frac{N}{2}-1) \tag{6.13}$$

This algorithm was implemented in a tree (depth 2) topology, using three processors. The range of frequencies for expression (6.12) and (6.13) was partitioned into three smaller ranges, each one being computed on a separate processor. A master processor received through its input channels the results calculated on the three worker processors, and, composed the final expression (6.13).

The OCCAM representation of the partitioning of the algorithm is:-

PAR

... compute the PSD for first range of frequencies

... compute the PSD for second range of frequencies

... compute the PSD for third range of frequencies

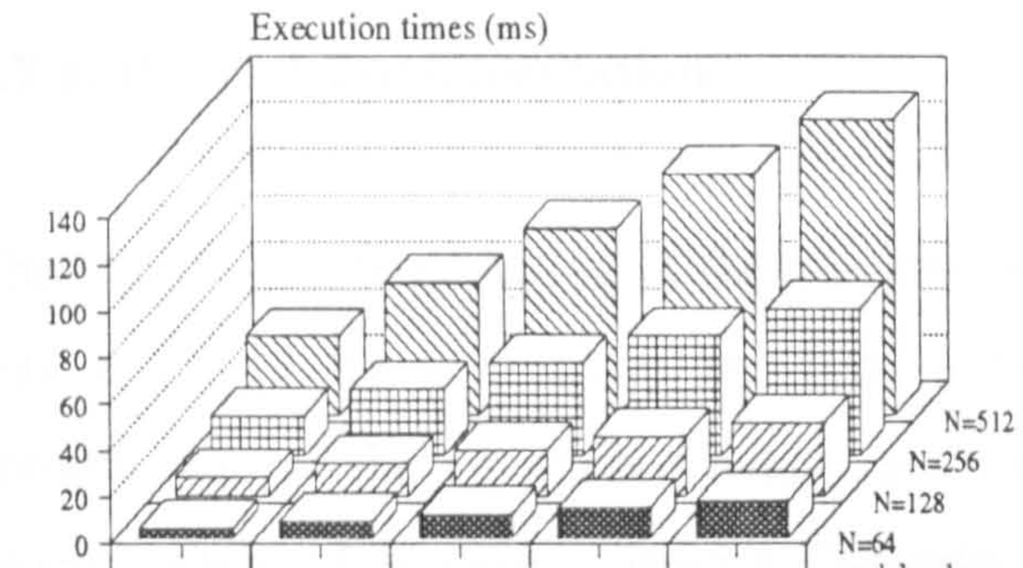
SEQ

... obtain the complete PSD

6.6.2. Performance Results

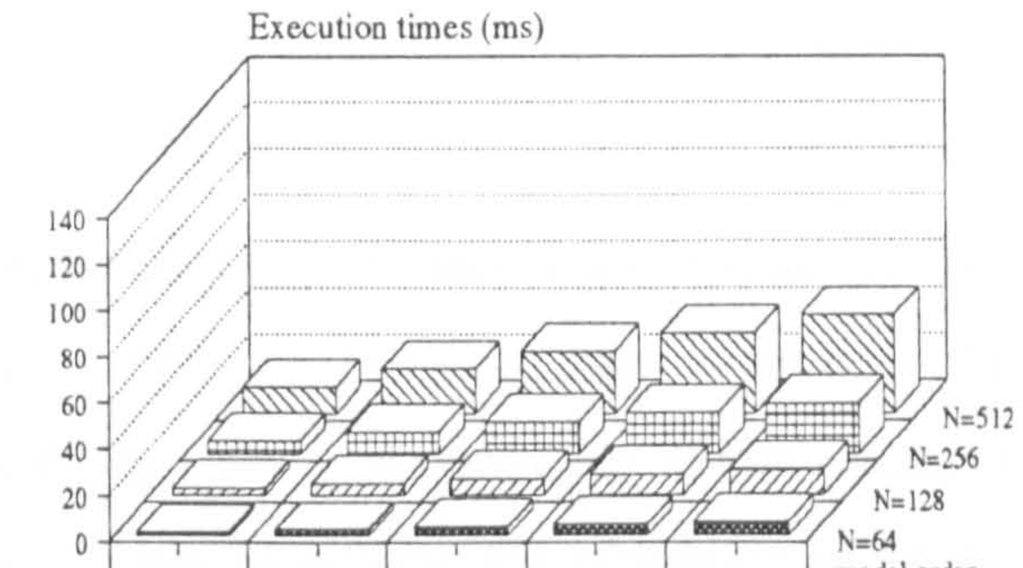
The sequential version of this simplified DFT was implemented on the SUN-hosted [91] transputer platform.

Fig. 6.24 illustrates the considerable decrease in the time consumed by the parallel implementation (Fig. 6.24 (b)) in comparison with the execution times obtained for the sequential implementation of the same algorithm (Fig. 6.24 (a)). The associated values of speedup and efficiency are depicted in Fig. 6.25. It should be noted that these are very close to the ideal case of linear speedup.



	2	4	6	8	10
N=512	34.326	57.553	80.836	104.056	127.314
N=256	17.155	28.745	40.373	51.939	63.554
N=128	8.57	14.34	20.14	25.88	31.674
N=64	4.277	7.138	10.026	12.85	15.736

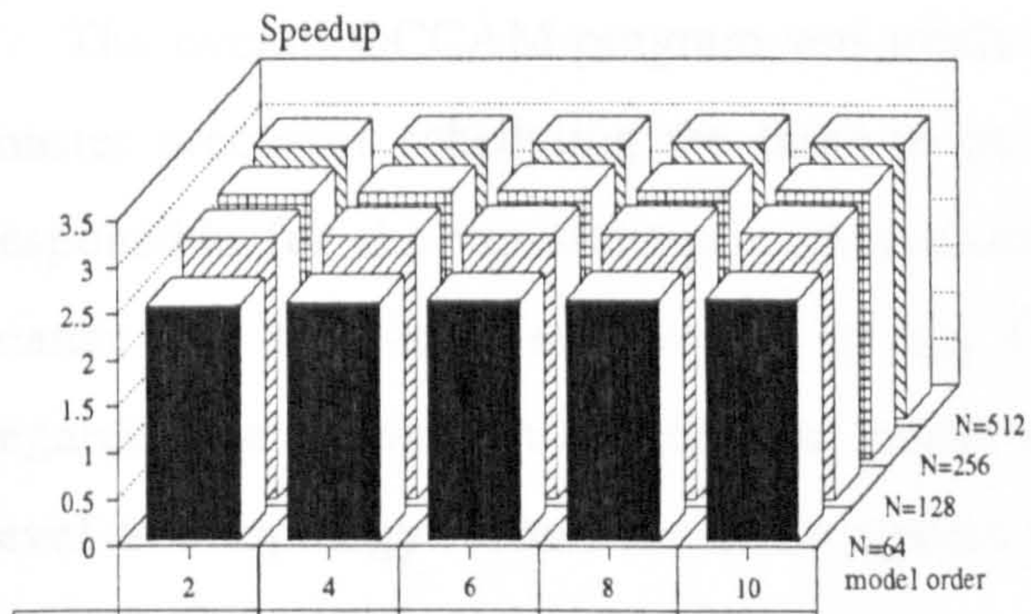
(a)



	2	4	6	8	10
N=512	11.739	19.544	27.391	35.208	43.029
N=256	6.031	10.03	14.048	18.042	22.052
N=128	3.045	5.048	7.063	9.056	11.066
N=64	1.69	2.78	3.883	4.961	6.066

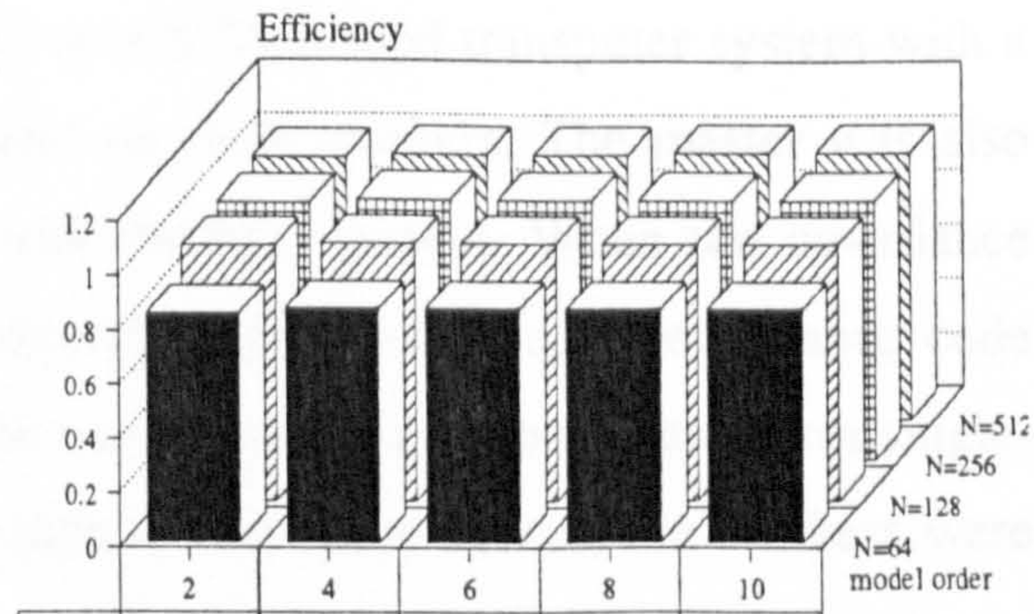
(b)

Fig. 6.24: Execution times for the (**a**) sequential and (**b**) parallel implementations of the simplified DFT algorithm.



	2	4	6	8	10
N=512	2.924	2.945	2.951	2.955	2.959
N=256	2.844	2.866	2.874	2.879	2.882
N=128	2.814	2.841	2.852	2.858	2.862
N=64	2.531	2.568	2.582	2.59	2.594

(a)



	2	4	6	8	10
N=512	0.97	0.98	0.98	0.99	0.99
N=256	0.95	0.96	0.96	0.96	0.96
N=128	0.94	0.95	0.95	0.95	0.95
N=64	0.84	0.86	0.86	0.86	0.86

(b)

Fig. 6.25: Simplified DFT algorithm (**a**) speedup and (**b**) efficiency.

6.7. FULL SPECTRAL ESTIMATION PROCEDURE

6.7.1. Parallel Implementation

The parallel implementation of the modified covariance spectral estimator algorithm as a whole was achieved by connecting the four computational blocks. These blocks comprised the computation of the covariance matrix and right-hand-side vector, the solution of the resultant linear system of equations using a Cholesky decomposition algorithm, the computation of the white noise estimate, and finally the calculation of the PSD.

For each of these blocks, with the exception of the computation of the white noise estimate (computed sequentially), a study of their possible parallel implementations and corresponding performance was carried out (see sections 6.3 to 6.6). Analysis of the performance results achieved by each implementation led to the identification of the topologies and task allocation schemes that performed best.

In this way, the OCCAM program developed to perform full spectral estimation, is the combination of the four partial programs associated with the best approaches. Some adjustments were made to enable the communication of data and results to all processors accordingly to their tasks and priority requirements.

The overall OCCAM program was implemented on a SUN-hosted transputer system with a master processor scheduling the tasks to be executed on three workers. The master was also responsible for the input/output communications with the host system. When the covariance matrix and right-hand-side vector, or the DFT algorithm were executed, the running code regarded the connection between the master and the workers as being a processor farm single-level tree topology. When the linear system of equations was being solved, the workers were viewed as connected in a ring topology.

6.7.2. Performance Analysis

The success of the overall implementation of the modified covariance method may be appreciated by analysis of Figures 6.26 and 6.27.

As the number of data points of the signal increased, the speedup also increased. The rate at which it increased depended on the order of the model considered, being greater for higher model

orders. Nevertheless, the speedup values achieved were very close to the maximum value, three (since three is the number of processors executing) for the larger N values.

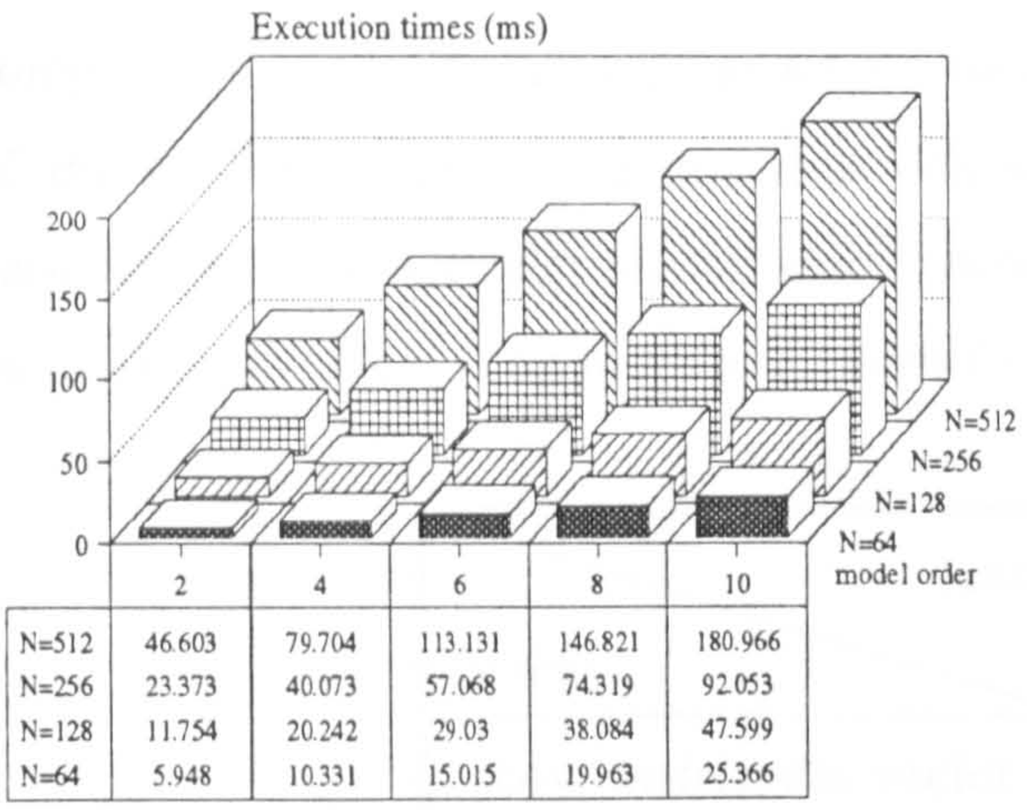
The same dependency on the data sequence length and order of the model is propagated to the efficiency metric. The lowest efficiency was obtained for the case where the highest model order and lowest data sequence length was considered.

In fact, the performance of the overall program was mainly dictated by the performance of the block of covariance matrix elements calculation, since the other major block, the PSD calculation, was almost insensitive to variations of both model order and data sequence lengths (see Fig. 6.25).

When higher model orders were considered, the main computation of each element of the covariance matrix (the summation) involved a smaller range of covariance values, contrasted by the fact that a greater number of elements need to be computed. If at the same time the data sequence length was smaller, the tradeoff between communications overhead and execution times became unbalanced, tending to be more influenced by the communications overhead.

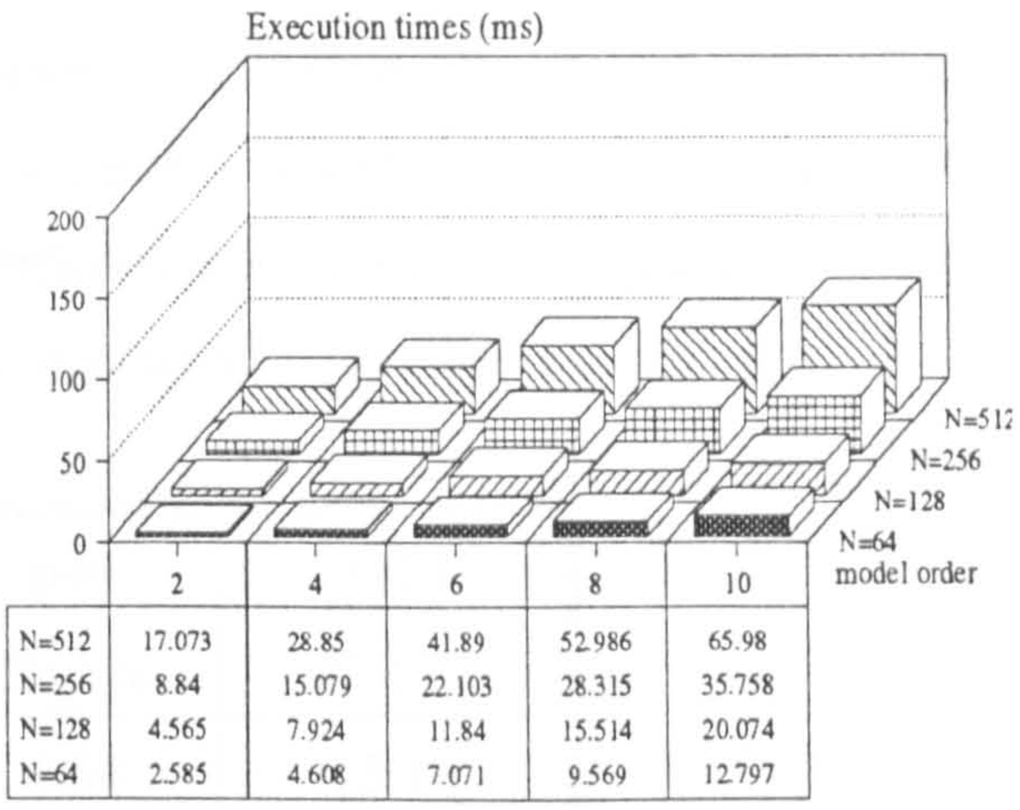
The analysis of the execution times is also relevant to evaluate the usefulness of this parallel implementation of the modified covariance spectral estimator algorithm.

The method is to be applied to a sequence of Doppler signal segments, each one with a number of data points determined by the product of the sampling frequency and the data sequence duration. In this research work the data segment time duration was fixed to 10ms, and, both sampling frequency and data sequence length were adjusted to satisfy the signals' constraints (see Chapter 2). In this situation, the analysis of Fig. 6.26 (b) show that only for signal simulation cases with up to 128 points length (considering the 4^{th} order model of interest, as indicated in Chapter 5), the algorithm executes within 10ms. However, in practice, sampling frequency and data segment duration may be different. In this case, the feasibility of this implementation should be assessed by comparing the actual data segment duration with the execution times indicated in Fig. 6.26 (b), for the data segment length closest to the actual length. It should be noticed that setting a particular sampling frequency implies appropriate scaling of spectral mean frequency and bandwidth values.



	2	4	6	8	10
N=512	46.603	79.704	113.131	146.821	180.966
N=256	23.373	40.073	57.068	74.319	92.053
N=128	11.754	20.242	29.03	38.084	47.599
N=64	5.948	10.331	15.015	19.963	25.366

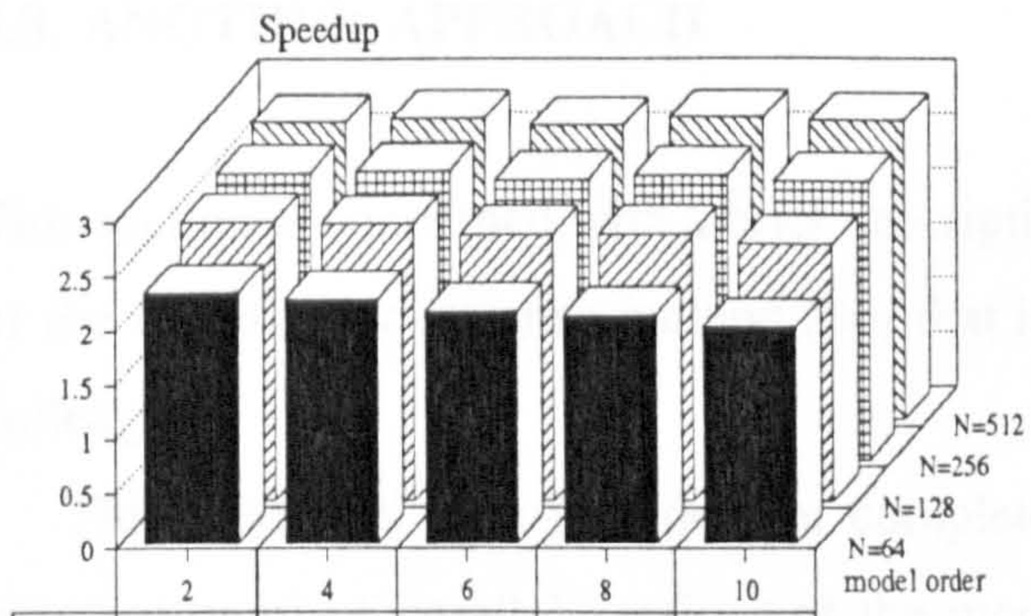
(a)



	2	4	6	8	10
N=512	17.073	28.85	41.89	52.986	65.98
N=256	8.84	15.079	22.103	28.315	35.758
N=128	4.565	7.924	11.84	15.514	20.074
N=64	2.585	4.608	7.071	9.569	12.797

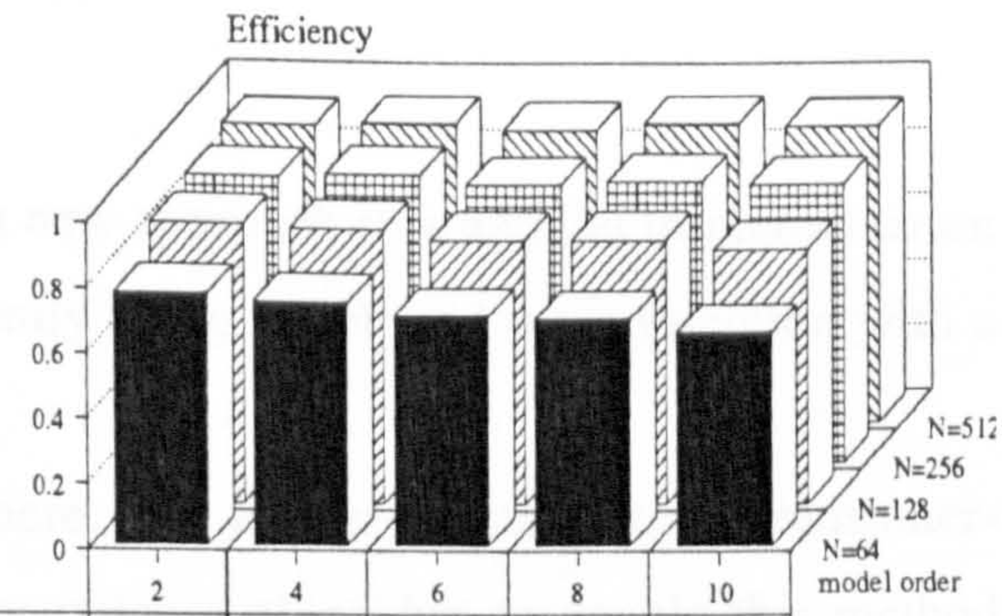
(b)

Fig. 6.26: Execution times of **(a)** sequential and **(b)** parallel implementations of the overall modified covariance spectral estimator algorithm. The parallel implementation is achieved using three active processors and a master scheduling their activities.



	2	4	6	8	10
N=512	2.73	2.763	2.701	2.771	2.743
N=256	2.644	2.658	2.582	2.625	2.574
N=128	2.575	2.555	2.452	2.454	2.371
N=64	2.301	2.242	2.123	2.086	1.982

(a)



	2	4	6	8	10
N=512	0.91	0.92	0.9	0.92	0.91
N=256	0.88	0.89	0.86	0.87	0.86
N=128	0.86	0.85	0.82	0.82	0.79
N=64	0.77	0.75	0.71	0.7	0.66

(b)

Fig. 6.27: Modified covariance spectral estimator algorithm performance: **(a)** speedup and **(b)** efficiency, when implemented on three active processors and a master scheduling their activities.

When a more detailed examination of the contributory execution times is carried out (see Table 6.5), we come to the conclusion that a significant part of the time is spent on the computation of the PSD rather than on the matrix elements calculation. Therefore the substitution of the DFT algorithm by a less time consuming algorithm would expand the range of the parameters of the Doppler signal which could be efficiently implemented in parallel. Another possibility involves using more processors for the computation of the DFT.

cases / blocks	p=4 N=128	p=10 N=512
cov. matrix, rhs vector	1.64	13.18
sol. of system (Cholesky)	0.23	1.86
white noise variance	0.02	0.04
computation of DFT	5.05	43.03

Table 6.5: Computational blocks and their respective contributory execution times for the overall algorithm (cases: p=4 and N=128, and, p=10 and N=512).

6.8. ANOTHER APPROACH

This section is concerned with a brief description of a new approach to a parallel implementation of the modified covariance method [96] that is currently being developed in conjunction with a colleague.

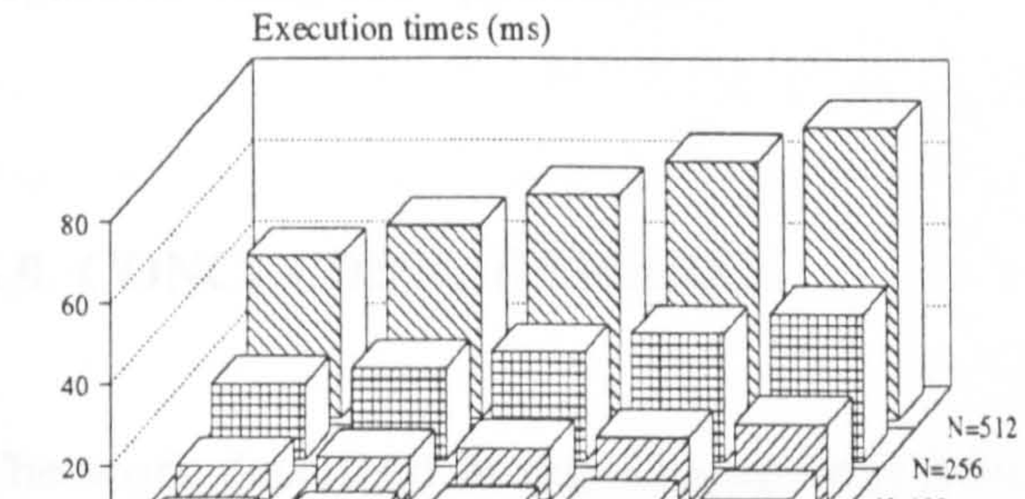
This work has been included for completeness here. The implementation (on a transputer-based system) of parallel versions of the modified covariance algorithm to enable this method of Doppler spectrum estimation to be achieved in real-time.

The strategy described in the previous section, regarded the spectral estimate of a single time segment of the Doppler signal (a 10ms segment is considered (see Chapter 3)), that is to say, it could only estimate the spectrum of the Doppler data segment corresponding to a single point in the cardiac cycle. However, each cardiac cycle (with an approximate duration of 1sec) contains several Doppler data segments. The parallel implementation approach presented in this section, enables the computation of a number of these Doppler data segments to be executed simultaneously.

These two approaches differ also in terms of the implementation strategy employed. In the previous approach, parallelism was achieved by means of partitioning and allocating the constituent processes of the modified covariance method among a number of processors (a master and three workers).

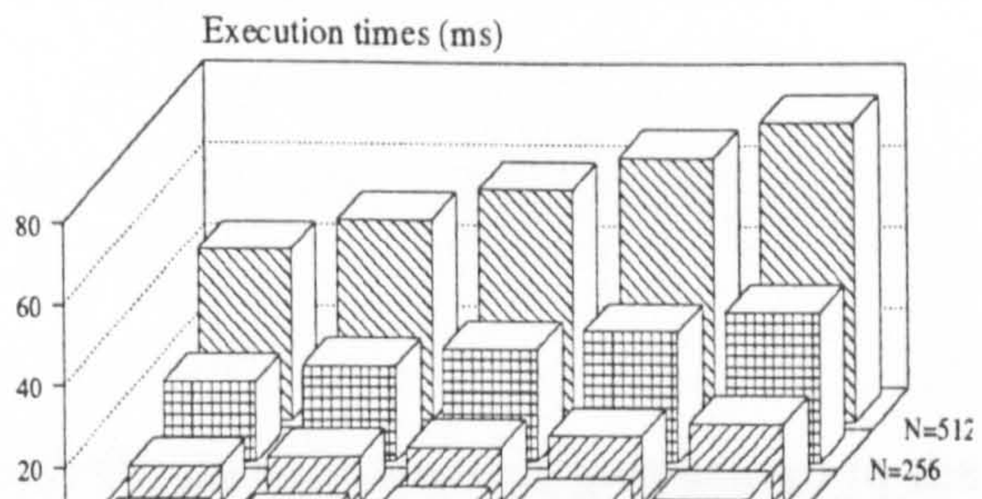
In the new approach, again a master and three workers are used, but the modified covariance method is now running sequentially on each processor, including the master. Each data segment in the block is executed in parallel. The number of segments processed concurrently depends on the number of processors used.

The new algorithm has been implemented on a tree (depth 2) topology, on a PC-based transputer network, populated with T800 INMOS transputers running at 20MHz. The execution times obtained for a sequential implementation are shown in Fig. 6.28(a). These times may be compared with those given in Fig. 6.26(a), remembering that the calculation of the PSD was performed using a simplified DFT in the case of Fig. 6.26(a) and an FFT algorithm (see section 6.6.1) in the case of Fig. 6.28(a). The times taken to execute the spectral estimation of four signal segments, using four transputers, are illustrated on Fig. 6.28(b). The execution times per data segment are one quarter of those shown.



	2	4	6	8	10
N=512	40.083	47.562	55.302	63.383	71.86
N=256	18.395	22.323	26.522	31.081	36.049
N=128	8.39	10.54	12.968	15.772	18.99
N=64	3.875	5.129	6.68	8.592	10.939

(a)



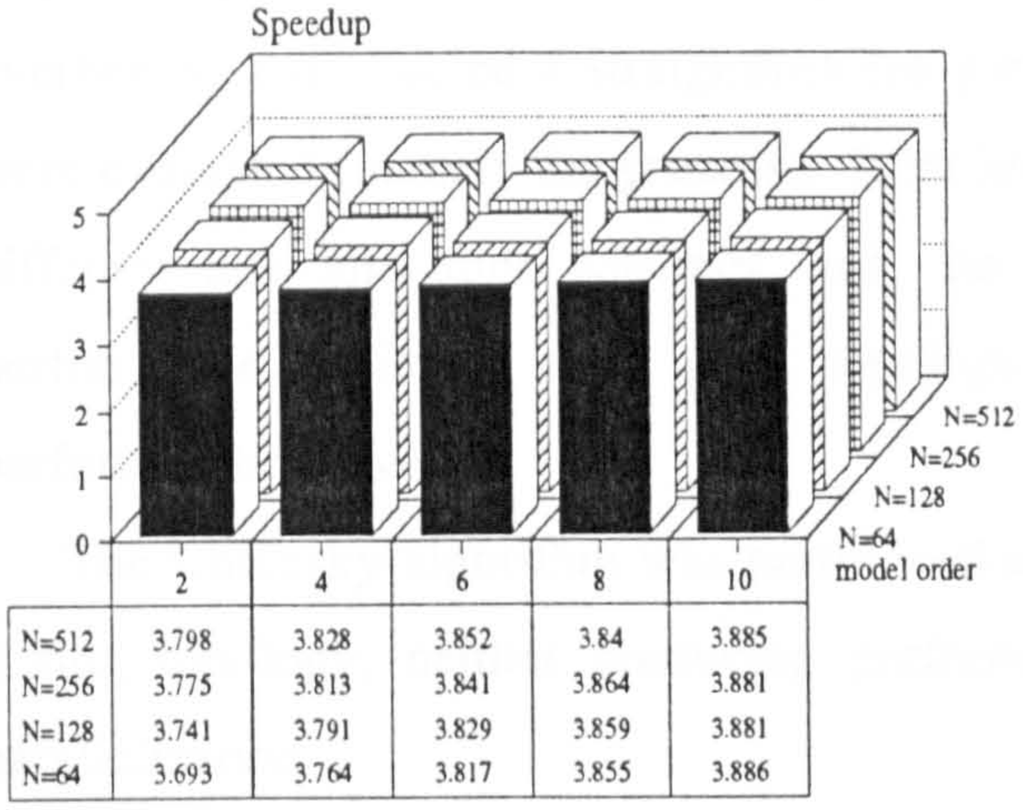
	2	4	6	8	10
N=512	42.22	49.697	57.432	65.512	73.988
N=256	19.492	23.416	27.617	32.178	37.148
N=128	8.97	11.12	13.544	16.345	19.572
N=64	4.197	5.49	6.999	8.916	11.259

(b)

Fig. 6.28: Execution times obtained with the 2nd approach implementation of the modified covariance spectral estimation of **(a)** a single data segment on a single processor, and, **(b)** four data segments on four processors.

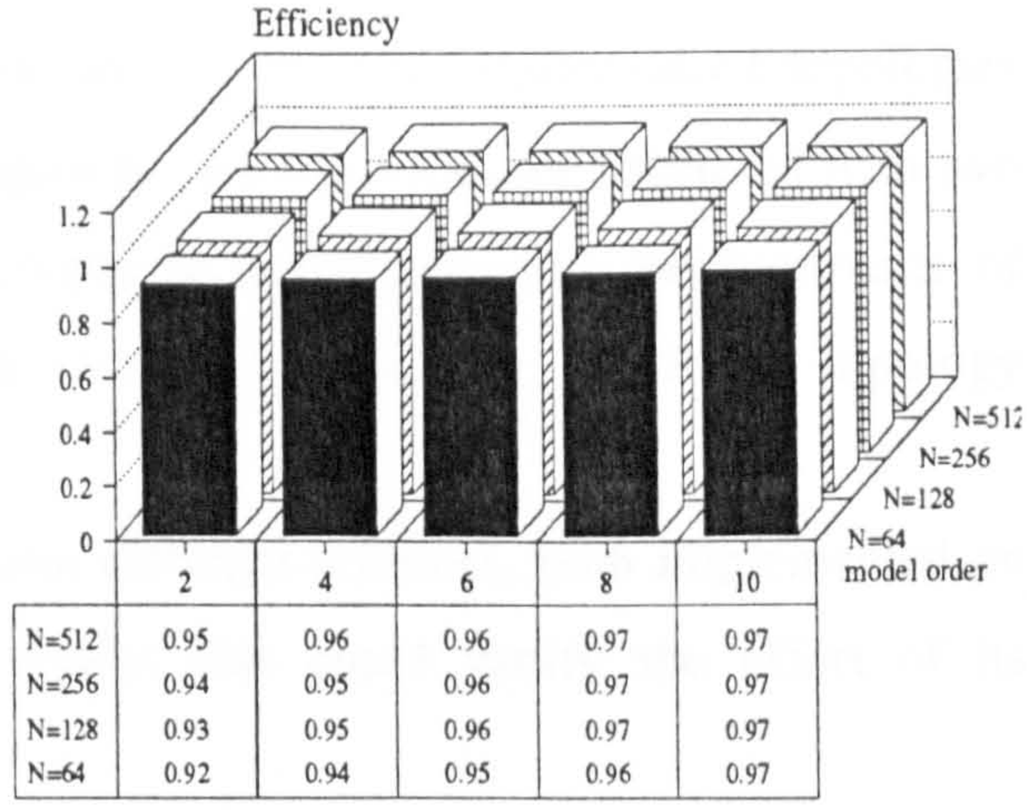
Values of speedup and efficiency obtained with this new approach are presented in Fig. 6.29.

In this approach (dealing with blocks of segments), a higher level of speedup and efficiency is achieved when compared with the single segment approach, mainly due to the lower communication requirements.



	2	4	6	8	10
N=512	3.798	3.828	3.852	3.84	3.885
N=256	3.775	3.813	3.841	3.864	3.881
N=128	3.741	3.791	3.829	3.859	3.881
N=64	3.693	3.764	3.817	3.855	3.886

(a)



	2	4	6	8	10
N=512	0.95	0.96	0.96	0.97	0.97
N=256	0.94	0.95	0.96	0.97	0.97
N=128	0.93	0.95	0.96	0.97	0.97
N=64	0.92	0.94	0.95	0.96	0.97

(b)

Fig. 6.29: Performance measurements of: (**a**) speedup and (**b**) efficiency obtained with the implementation of the 2nd approach of the modified covariance spectral estimation of four data segments using four processors.

6.9. CONCLUDING REMARKS

The work described in this Chapter was concerned with the identification of parallelism in the AR modified covariance spectral estimation algorithm, the partitioning of the problem into tasks that could be executed concurrently on different processing elements, and, the allocation of those tasks onto different number of processors of a transputer-based system.

The parallelization of the modified covariance algorithm was primarily developed considering the spectral estimation of an individual Doppler data segment. In this approach, each algorithmic block (the computation of the covariance matrix and right-hand-side vector elements, the solution of the linear system of equations using a Cholesky decomposition algorithm, the calculation of the white noise estimate, and, the computation of the PSD) were separately studied.

A new strategy to compute the covariance matrix and right-hand-side vector elements was presented, and its computational complexity (measured in terms of flops) was compared with those obtained with the published versions of the modified covariance method (versions due to Kay and Marple). This new strategy proved to be much more efficient than Kay's algorithm and similar to Marple's version. The simplified covariance algorithm was then partitioned in two ways, using fine-grain and medium-grain strategies. The latter presented lower communication overheads, and enabled a straightforward parallelization. Different transputer-based topologies were evaluated (namely the processor farm linear topology, the tree topology (depth 3) with two different task allocation schemes, and, the tree (depth 2) topology), and measurements of performance achieved using each topology were compared. The tree (depth 2) topology performed best overall.

The Cholesky algorithm was partitioned using two different schemes, both implemented on a ring topology, neither producing performance results that could justify the effort of its parallelization.

The white noise estimate, due to the simplifications introduced in the modified covariance version proposed in section 6.2.2, was computed sequentially, on a single processor.

An adaptation of the computation of the DFT to the specific signal of interest was developed in order to obtain the power spectral estimation of the signal. The simplified DFT algorithm was implemented on a depth 2 tree, where almost optimum speedup and efficiency values were attained. However, the parallel execution times obtained for the DFT did not satisfy the real-time requirement of executing in less than 10ms for some of the signal cases considered. The substitution of the DFT algorithm by an FFT algorithm is proposed.

The parallel approaches which performed best, for each computational block of the new modified covariance algorithm, were combined in an OCCAM program, to enable a complete spectral estimate of a Doppler signal segment to be obtained. Its performance results showed that the technique employed to parallelize the modified covariance method was efficient.

Another approach involves considering the existence of a consecutive sequence of Doppler signal segments, each one with an approximate duration of 10ms. Adapting this approach, the modified covariance algorithm is more efficiently used when implemented on each one of an array of transputers, each of them processing an individual data segment. This approach is being currently developed in collaboration with a research group colleague, and good levels of speedup

and efficiency have been achieved. This is more noticeable, due to a coarser grain size, when large values of data segment and AR model orders are used.

CHAPTER 7

CHAPTER 7

Conclusion

7.1. GENERAL CONCLUSIONS

The research work presented in this thesis aimed to increase the sensitivity to blood vessel disease detection and investigation using spectral analysis of the Doppler shifted ultrasound scattered from blood.

The sensitivity of disease detection depends *inter alia*, on the accuracy of estimation of the Doppler signal spectrum mean frequency and width. Intrusive vessel lesions produce disturbed flow, resulting in an increase in the range of Doppler shift frequency (spectral broadening), and variations in the blood velocity waveform of each cardiac cycle, and consequently in the Doppler spectrum mean frequency.

Pulsed ultrasonic Doppler instrument signals were considered in this work. These type of signals are characterized by a random nature, having a Gaussian probability density function in undisturbed flow, and with a time-varying frequency spectrum determined by the time-varying-blood velocity field and size and position, within the velocity field, of the instrument's resolution cell (sample volume). Under steady heart rate conditions and stable physiological state, the Doppler signal is cyclo-stationary. When a small sample volume in undisturbed flow is considered (case under study), the Doppler spectrum is under most conditions relatively narrow and single peaked.

Accurate methods of estimation of Doppler spectral mean frequency and bandwidth which could reduce the effects of the measurement procedure on the estimation of blood velocity and detection of flow disturbance, were therefore required. These methods should also contemplate general Doppler signal requirements, such as, the stationarity of the consecutive segments of the cardiac cycle to be analyzed, during which only short data records are available, and the narrow and single peaked shape of the expected spectrum.

In this context, reliable analysis of the spectral content of the Doppler signal estimations obtained by the different spectral estimators required a complete knowledge of the Doppler signal. Therefore, this research work started by the implementation of a Doppler signal simulator.

Two algorithms of simulation were implemented, comprising two different Doppler spectral shapes, the rectangular and the Gaussian shapes. The rectangular shape contemplates the situation were uniform sensitivity over the vessel cross-section, parabolic velocity profile, and, negligible spectral broadening resultant from transit time effects and deviations from the plane-wave conditions. On the other hand, the Gaussian spectral shape represents the case where large vessels with undisturbed flow and small sample volumes are considered. Algorithm 1 (as it is referred in this thesis) is based on Mo and Cobbold's [63] approach, while the other algorithm, algorithm 2, is based on Leuewen, Hoeks and Reneman's [64] theory. A modification to algorithm 1 was introduced aiming to reduce its computational burden when simulating a Doppler signal with Gaussian spectral shape. The signals simulated by modified algorithm 1 and algorithm 2 gave similar Chi-squared values when the Gaussianity of their amplitude distribution was tested. Thus, the selection of algorithm 2 as the algorithm to use in this research work was based on considerations of computational efficiency. In 90% of the simulated signals using algorithm 2 the amplitude of the spectrum of the simulated signal followed a Gaussian distribution to within limits of the statistical test (a Chi-squared variable with 5% significance level was considered).

The random nature of the Doppler signal induces randomness of the Doppler spectrum, leading to the application of ensemble averaging techniques in order to reduce spectral variance. The question of how many signals should constitute the ensemble was addressed in this project. Statistical expressions for estimating spectral mean frequency and half bandwidth parameters were developed, and measurements of their statistical performance were established. A compromise between the computational complexity of a large ensemble and the requirement of a low residual variance of the statistical measures resulting from the finite size of the ensemble led to the selection of an ensemble size of 150.

After establishing the use of a particular signal simulator algorithm, and testing both in time and frequency domains the statistical characteristics of the simulated signals, our attention was centred on the estimation of the spectrum of Doppler signal simulations with Gaussian spectral shape. Mean frequencies varying from 1kHz to 8kHz in octaves, and half bandwidths of 5% (except for the 1kHz signal), 10% and 20% of the mean frequency value (each of these signals was designated in the text as signal simulation case) were considered.

A pre-selection of the spectral estimation methods that could be applied to signals with such characteristics was carried out. Among the parametric methods, the autoregressive Yule-Walker, Burg and modified covariance methods were implemented, as well as the autoregressive moving average methods of least squares modified Yule-Walker equations and Mayne-Firoozan. The non-parametric method of minimum variance was also considered. For completeness, the traditional methods of spectral estimation employing the Fast Fourier transform of Boxcar and Hanning windowed signals were also included in this study.

Measurement of the percentage relative bias and standard deviation of the spectral mean frequency and half bandwidth estimators using the above spectral estimators was carried out. The ARMA Mayne-Firoozan and minimum variance methods were excluded at an early stage of this study, due to, respectively, the non-minimum-phase filter estimates and high spectral half bandwidth variances presented.

The analysis of the Doppler simulated spectra was performed in a first stage comprising a qualitative comparison of the statistical results obtained for both spectral parameters. A subjective selection of the parametric model order(s) that more appropriately combined the uniformity of the results of the four statistics (bias and standard deviation for both mean frequency and half bandwidth parameters) for each signal simulation case, was carried out. Global analysis of the parametric spectral estimators' mean squared error of both spectral parameters, including all the simulation cases and considering the previously subjective chosen model order(s), was attempted. The difficulty encountered in selecting a particular spectral estimator from those considered, by means of qualitatively comparing the statistical results obtained, motivated the development of a novel criterion of parametric model order selection and spectral estimator selection (from parametric and non-parametric methods).

The new criterion, the cost/benefit criterion [87] [88], is based on the accuracy of estimation of specific spectral parameters (mean frequency and bandwidth), under the constraint of low

computational complexity. It also allows the possibility of weighting appropriate to estimator (mean frequency and bandwidth) and signal frequency importance in disease detection.

The cost/benefit results obtained when the spectral estimation methods referred above were employed to estimate (with equal levels of importance) both mean frequency and half bandwidth parameters of the Doppler spectrum, revealed that the fourth order modified covariance method was the spectral estimator that more appropriately satisfied the requirements. This method was also selected for the individual estimation of spectral half bandwidth parameter. If instead only spectral mean frequency is required, the improvement in statistical performance by the use of a parametric method would not compensate for the cost of applying the less computationally complex FFT algorithm.

The parametric model orders selected by the cost/benefit criterion were compared with those selected by accepted model order selectors (FPE, AIC and CAT), for the same signal simulation cases. This comparison shown that the model orders chosen by the new criterion were lower than the selected by conventional criteria.

More detailed analysis of the behaviour of the cost/benefit criterion when cost of the algorithmic implementation is or is not taken into consideration, demonstrated that computational complexity does influence the selection of model order within a single signal simulation case, but, the selection of a model order appropriate for estimation of several signal cases is mainly determined by the statistical accuracy of the spectral parameter estimates.

Finally the parallel implementation of the spectral estimation method selected (under general conditions) by the cost/benefit criterion was carried out.

The modified covariance algorithm was first parallelized considering the spectral estimation of a single Doppler data segment. To achieve that, the algorithm was partitioned into blocks. They were, the computation of the covariance matrix and right-hand-side vector, the solution of the linear system of equations using a Cholesky decomposition algorithm, the calculation of the white noise variance estimate, and the computation of the power spectral density. This approach was implemented on a SUN-hosted transputer system. The parallel programming language employed was OCCAM.

Since the calculation of the elements of the covariance matrix and right-hand-side vector was the algorithmic block involving more computational effort, this block was treated in detail [97]. A new strategy to compute these elements (when real data sequences are considered) was developed, and its sequential computational complexity (measured in flops) was compared with

those presented by Kay and Marple versions of the modified covariance algorithm. The new modified covariance algorithm demonstrated a better computational performance than Kay's version, and comparable behaviour to Marple's version. Two strategies to parallel partition this computational block were investigated, the fine-grain and the medium-grain [89]. The latter revealed more capacity to be efficiently parallel-implemented, presenting less communication overheads than the fine-grain approach. Different transputer-based topologies were implemented, namely the processor farm linear topology, the two-level tree topology and the single-level tree topology. The two-level tree topology included two task allocation schemes, a dynamic task allocation (where the two-level tree may be regarded as a processor farm tree topology), and a static task allocation scheme achieved by means of a map. The single-level tree topology considered also a static allocation of the tasks onto three processors connected to a master processor (responsible for input/output communications) in a tree structure. The parallel implementation performance was measured in terms of execution times, speedup and efficiency. All topologies achieved speedup, but the single-level tree topology presented the best efficiency values.

The parallel implementation of the Cholesky algorithm used three processors connected in a ring structure. Two different task allocation schemes were considered, none of them presenting significant improvements in the performance results achieved by a sequential implementation of the same algorithm.

Due to simplifications resulting from the application of the new modified covariance algorithm, the computational burden involved in the calculation of the white noise estimate did not justify its parallelization.

The computation of the power spectral density was achieved by means of a simplified DFT, targeted to the specific signal of interest. Its parallel implementation used three processors in a single-level tree topology, revealing close to optimum values of speedup and efficiency.

Afterwards, best solutions for each computational block were interconnected to constitute a single spectral estimation parallel program. Performance measurements were taken, confirming the efficiency of the implementation achieved. However, the feasibility (in terms of execution times) of a real-time implementation of the modified covariance method using this simplified DFT algorithm will depend on the relationship between the Doppler data sequence length and the sampling frequency considered. For the signal simulation cases given as examples, where a

fixed 10ms duration Doppler signal segment is considered, only signals with a number of points up to 128 (when a 4^{th} order model is used) can be estimated in less than 10ms.

Another approach to the parallel implementation of the modified covariance algorithm comprised the simultaneous estimation of a number of Doppler data segments on a PC-based transputer system. This approach, currently in development, executes (sequentially) the estimation of each Doppler data segment on each single processor. In this way, the number of processors in the network defines the number of Doppler signal segments that can be spectral estimated concurrently. It includes also an FFT algorithm for the calculation of the Doppler power spectral density estimation instead of the simplified DFT algorithm used before. The execution times obtained for the computation of each spectral estimate decreased, therefore enabling more flexible choice of the range of values of the sampling frequency and data sequence length [96].

Although constrained with some tradeoff considerations concerning sampling frequency, data sequence length and data sequence duration, the parallel implementation performance results showed that spectral mean frequency and bandwidth estimation using the modified covariance spectral estimator is achievable in real-time.

7.2 FUTURE WORK

The interdisciplinary content of this research work enables pointing out future work in specific areas or research.

Starting by the area of Doppler ultrasound, an extension to non-stationary analysis of the Doppler signal may be developed. In this context, the Mo and Cobbold's signal simulator [98] may be used. On the other hand, the effects of additive Gaussian noise introduced by the Doppler signal instruments should be considered. Different levels of noise should be added to the Doppler signal, and, the performance of the each spectral estimators considered should be analyzed under these conditions.

The signal processing area, due to its applicability to several other research areas, is in continuous development. Therefore, the spectral estimation methods considered in this research work covered only a limited range of the methods available, some of them resulting from modifications of commonly referred methods in order to satisfy requirements of certain applications. Parametric spectral estimation methods that can constantly adapt to the characteristics of the signal, such as, the recursive least squares autoregressive methods [32], may

be explored. Then, is also the possibility of following another approach of spectral mean frequency and bandwidth estimation, following the strategy of analysis of AR pole location in the z-plane, presented by Ahn and Park [20].

With reference to the cost/benefit selection of spectral estimation methods a deeper study of the benefit weighting, and cost functions should be carried out. The benefit weighting function should reflect (in a more objective way) the clinical requirements from the procedure of which spectral estimation is a part. As a start, the benefit weighting should be altered to truly reflect the equal importance of mean frequency and bandwidth estimators (when this is required) rather than simply equally weighting their errors, since this leads to the greater effective weight being given to bandwidth estimation because of the large associated error. The cost function should also be improved to take into account factors such as the computational burden of input/output procedures, the effective cost of employing a specific hardware system, and, the *opacity* of the algorithms used. Opacity is understood as the capacity of an algorithm to be clearly understood so that its application and/or modification can be reliably carried out [99].

Referring to the modified covariance estimator's parallel implementation some points should be stressed. Firstly, the modified covariance algorithm should be developed to enable spectral estimation of complex signals since Doppler instruments normally use phase-quadrature demodulation in order to preserve flow direction information. Secondly, the solution of the linear system of equations could be more investigated.

The implementation of the modified covariance algorithm using other computer architectures is also desired. Its implementation on a *Digital Signal Processor (DSP)* is currently being considered by a research colleague. Comparative performance results obtained with the transputer and DSP implementations will be carried out.

Meanwhile, the development of the modified covariance algorithm is being also studied with regard to its implementation on an *Application Specific Integrated Circuit (ASIC)*, for incorporation into Doppler ultrasound instruments.

REFERENCES

REFERENCES

[1] Routh, H., Law, Y., Mo, L., Ojha, M., Vaitkus, P., Cobbold, R., Johnston, K., Bascom, P., 'Role of models in understanding and interpreting clinical Doppler ultrasound', Medical Progress through Technology, 15, 1989, pp. 155-169

[2] Strandness, D., 'Doppler ultrasonic techniques in vascular disease', in Bernstein, E. (ed), 'Noninvasive diagnostic techniques in vascular disease', The C. V. Mosby Company, 1985, pp. 13-18

[3] Fish, P., 'Nonstationarity broadening in pulsed Doppler spectrum measurements', Ultrasound in Medicine & Biology, Vol. 17 Nº. 2, 1991, pp. 147-155

[4] Angelsen, B., 'A theoretical study of the scattering of ultrasound from blood', IEEE Transactions on Biomedical Engineering, Vol. 27, Nº. 2, 1980, pp. 61-67

[5] Fish, P., 'Doppler methods', in Hill, C. (ed.), Physical principles of medical ultrasonics', Ellis Horwood Series in Applied Physics, 1986, pp. 338-376

[6] Beach, K., Strandness, D., 'Pulsed Doppler ultrasound for blood velocity measurements', in Bernstein, E. (ed), 'Noninvasive diagnostic techniques in vascular disease', The C. V. Mosby Company, 1985, pp. 25-32

[7] Johnston, K., Zuech, P., Kassam, M., 'Accuracy and potential pitfalls of carotid Doppler frequency analysis', in Bernstein, E. (ed), 'Noninvasive diagnostic techniques in vascular disease', The C. V. Mosby Company, 1985, pp. 69-76

[8] Giddens, D., Kitney, R., 'Blood flow disturbances and spectral analysis', in Bernstein, E. (ed), 'Noninvasive diagnostic techniques in vascular disease', The C. V. Mosby Company, 1985, pp. 58-68

[9] Kitney, R., Giddens, D., 'Extraction and characterisation of underlying velocity waveforms in poststenotic flow', IEE Proceedings, Pt. A, Vol. 129, Nº. 9, 1982, pp. 651-662

[10] Talhami, H., Kitney, R., 'Maximum likelihood frequency tracking of the audio pulsed Doppler ultrasound signal using a Kalman filter', Ultrasound in Medicine & Biology, Vol. 14, Nº. 7, 1988, pp. 599-609

[11] Vaitkus, P., Cobbold, R., 'A comparative study and assessment of Doppler ultrasound spectral estimation techniques. Part I: estimation methods', Ultrasound in Medicine & Biology, Vol. 14, Nº. 8, 1988, pp. 661-672

[12] Vaitkus, P., Cobbold, R., Johnston, W., 'A comparative study and assessment of Doppler ultrasound spectral estimation techniques. Part II: methods and results', Ultrasound in Medicine & Biology, Vol. 14, Nº. 8, 1988, pp. 673-688

[13] Kaluzynski, K., 'Analysis of application possibilities of autoregressive modelling to Doppler blood flow signal spectral analysis', Medical & Biological Engineering & Computing, 25, 1987, pp. 373-376

[14] Kaluzynski, K., 'Order selection in Doppler blood flow signal spectral analysis using autoregressive modelling', Medical & Biological Engineering & Computing, 27, 1989, pp. 89-92

[15] Schlindwein, F., Evans, D., 'Selection of the order of autoregressive models for spectral analysis of Doppler ultrasound signals', Ultrasound in Medicine & Biology, Vol. 16, Nº. 1, 1990, pp. 81-91

[16] Linkens, D., 'Short-time-series spectral analysis of biomedical data', IEE Proceedings, Pt. A, Vol. 129, Nº. 9, 1982, pp. 663-672

[17] Kay, S., 'Modern spectral estimation - theory & application', Prentice-Hall, 1988

[18] Kristoffersen, K., 'Time-domain estimation of the center frequency and spread of Doppler spectra in diagnostic ultrasound', IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, Vol. 35, Nº. 4, 1988, pp. 484-497

[19] Loupas, T., McDicken, W., 'Low-order complex AR models for mean and maximum frequency estimation in the context of Doppler color flow mapping', IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, Vol. 37, Nº. 6, 1990, pp. 590-601

[20] Ahn, Y., Park, S., 'Estimation of mean frequency and variance of ultrasonic Doppler signal by using second-order autoregressive model', IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, Vol. 38, Nº. 3, 1991, pp. 172-182

[21] Schlindwein, F., Evans, D., 'A real-time autoregressive spectrum analyzer for Doppler ultrasound signals', Ultrasound in Medicine & Biology, Vol. 15, Nº. 3, 1989, pp. 263-272

[22] Harp, J., Roberts, J., Ward, J., 'Signal processing with transputer arrays (TRAPS)', Computer Physics Communications, 37, 1985, pp. 77-86

[23] Krishna, H., Morgera, S., 'The Levinson recurrence and fast algorithms for solving Toeplitz systems of linear equations', IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 35, Nº. 6, 1987, pp. 839-848

[24] Kalouptsidis, N., Theodoridis, S., 'Parallel implementation of efficient LS algorithms for filtering and prediction', IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 35, Nº. 11, 1987, pp. 1565-1569

[25] Atkinson, P., Woodcock, J., 'Doppler ultrasound and its use in clinical measurement', Academic Press, 1982

[26] Barnes, R., 'Continuous-wave Doppler ultrasound', in Bernstein, E. (ed), 'Noninvasive diagnostic techniques in vascular disease', The C. V. Mosby Company, 1985, pp. 19-24

[27] Mo, L., Cobbold, R., 'A stochastic model of the backscattered Doppler ultrasound from blood', IEEE Transactions on Biomedical Engineering, Vol. 33, Nº. 1, 1986, pp. 20-27

[28] Burckhardt, C., 'Speckle in ultrasound b-mode scans', IEEE Transactions on Sonics and Ultrasonics, Vol. 25, Nº. 1, 1978, pp. 1-6

[29] Lindgren, B., 'Statistical theory', 3rd ed., MacMillan Publishing, 1976

[30] Evans, D., McDicken, W., Skidmore, R., Woodcock, J., 'Doppler ultrasound', John Wiley & Sons, 1989

[31] Papoulis, S., 'Signal Analysis', McGraw-Hill, 1984

[32] Marple, S., 'Digital spectral analysis with applications', Prentice-Hall, 1987

[33] David, J., Jones, S., Giddens, D., 'Modern spectral analysis techniques for blood flow velocity and spectral measurements with pulsed Doppler ultrasound', IEEE Transactions on Biomedical Engineering, Vol. 38, Nº. 6, 1991, pp. 589-596

[34] Fish, P., 'Physics and instrumentation of diagnostics medical ultrasound', John Wiley & Sons, 1990

[35] Marple, S., 'A tutorial overview of modern spectral estimation', Proceedings of IEEE ICASSP, 1989, pp. 2152-2157

[36] Challis, R., Kitney, R., 'Biomedical signal processing. Part I: time-domain methods', Medical & Biological Engineering & Computing, 28, 1990, pp. 509-524

[37] Proakis, J., Manolakis, D., 'Introduction to digital signal processing', MacMillan Publishing, 1988

[38] Challis, R., Kitney, R., 'Biomedical signal processing. Part II: the frequency transforms and their inter-relationships', Medical & Biological Engineering & Computing, 29, 1991, pp. 1-17

[39] Challis, R., Kitney, R., 'Biomedical signal processing. Part III: the power spectrum and coherence function', Medical & Biological Engineering & Computing, 29, 1991, pp. 225-241

[40] Kay, S., Marple, S., 'Spectrum analysis - a modern perspective', Proceedings of the IEEE, Vol. 69, Nº. 11, 1981, pp. 1380-1419

[41] Ulrych, T., Ooe, M., 'Autoregressive and mixed autoregressive-moving average models and spectra', in Haykin, S. (ed.), 'Topics in applied physics: nonlinear methods of spectral analysis', Vol. 34, 2nd ed. Springer Verlag, 1983, pp. 73-125

[42] Stone, H., 'High-performance computer architecture', Addison-Wesley Series in Electrical and Computer Engineering, 1987

[43] Dettmer, R., 'Occam and the transputer', IEE Electronics and Power, April 1985, pp. 283-287

[44] Jones, D., 'Parallel processing computer architectures', IEE Workshop on Parallel Processing and Control, U.C.N.W., Bangor, U.K., Sept. 1989

[45] Eager, D., Zahorjan, J., Lazowska, E., 'Speedup versus efficiency in parallel systems', IEEE Transactions on Computers, Vol. 38, Nº. 3, 1989, pp. 408-423

[46] Amdahl, G., 'Validity of the single processor approach to achieving large-scale computing capabilities', AFIPS Conf. Proc. Thompson Books, Washington, 1967, pp. 483-485

[47] Sarkar, V., 'Partitioning and scheduling parallel programs for multiprocessors', Pitman Publishing, 1989

[48] Flynn, M., 'Very high speed computer systems', Proceedings of the IEEE, Vol. 54, Nº. 12, 1966, pp. 1901-1909

[49] May, D., Shepherd, 'The transputer implementation of Occam', in Fleming, P. (ed.) 'Parallel processing in control: the transputer and other architectures', Peter Peregrinus Ltd., 1988, pp. 85-98

[50] Inmos Ltd., 'Transputer overview', The Transputer Databook, 2nd ed., 1989

[51] Inmos Ltd., 'Transputer architecture', The Transputer Databook, 2nd ed., 1989

[52] Jones, D., Fleming, P., 'Control applications of transputers', in Fleming, P. (ed.) 'Parallel processing in control: the transputer and other architectures', Peter Peregrinus Ltd., 1988, pp. 101-125

[53] Inmos Ltd., 'IMS B004 evaluation board', User Manual, 1985

[54] Inmos Ltd., 'Transputer development system', Inmos, 1988

[55] Hoare, C., 'Communicating sequential processes', Communications of A.C.M., Vol. 21, Nº 8, 1978, pp. 666-667

[56] Inmos Ltd., 'Occam 2 reference manual', Prentice Hall, Series in Computer Science, 1988

[57] Wilson, P., 'Occam architecture eases system design', Computer Design, November 1983, pp. 107-115

[58] Burns, A., 'Programming in Occam 2', The Instruction Set Ltd., 1988

[59] Inmos Ltd., 'Occam programming manual', Prentice Hall International, 1984

[60] Galletly, J., 'Occam 2', Pitman Publishing, 1990

[61] Parkinson, D., Liddel, H., 'The measurement of performance on a highly parallel system', IEEE Transactions on Computers, Vol. 32, Nº. 1, 1989, pp. 32-37

[62] Flat, H., Kennedy, K., 'Performance of parallel processors', Parallel Computing, Vol. 12, October 1989, pp. 1-20

[63] Mo, L., Cobbold, R., '"Speckle" in continuous wave Doppler ultrasound spectra: a simulation study', IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, Vol. 33, Nº. 6, 1986, pp. 747-752

[64] Leeuwen, G., Hoeks, A., Reneman, R., 'Simulation of real-time frequency estimators for pulsed Doppler systems', Ultrasonic Imaging, 8, 1986, pp. 252-271

[65] Cobbold, R., Veltink, P., Johnston, K., 'Influence of beam profile and degree of insonation on the CW Doppler ultrasound spectrum and mean velocity', IEEE Transactions on Sonics and Ultrasonics, Vol. 30, Nº. 6, 1983, pp. 364-370

[66] Gardner, W., 'Introduction to random processes with applications to signals and systems', MacMillan Publishing, 1986

[67] Angelsen, B., Kristoffersen, K., 'Discrete time estimation of the mean Doppler frequency in ultrasonic blood velocity measurements', IEEE Transactions on Biomedical Engineering, Vol. 30, Nº. 4, 1983, pp. 207-214

[68] Stoodley, K., 'Basic statistical techniques', Bradford University Press, 1974

[69] Munro, D., 'Fortran 77', Edward Arnold, 1982

[70] 'Nag Fortran manual', Numerical Algorithms Group Ltd., 1984

[71] 'Ginograf user manual', CADCentre Ltd., 1986

[72] Papoulis, A., 'Probability, random variables, and stochastic processes', 2nd ed., McGraw-Hill, 1984

[73] Murteira, B., Black, G., 'Estatistica descritiva', McGraw-Hill, 1983

[74] Meyer, P., 'Probabilidade, aplicacoes a estatistica', 2nd ed., Addison-Wesley, 1983

[75] Patel, J., Kapadia, C., Owen, D., 'Handbook of statistical distribution', Marcel Dekker, 1976

[76] Jenkins, G., Watts, D., 'Spectral analysis and its applications', Holden-Day, 1968

[77] Oppenheim, A., Schaffer, P., 'Digital signal processing', Englewood Cliffs, Prentice-Hall, 1975

[78] Moler, C., Little, J., Bargert, S., 'Pro-Matlab user's guide', The MathWorks Inc., South Natick, MA01760, USA, 1987

[79] Papoulis, A., 'Maximum entropy and spectral estimation: a review', IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 29, 1981, pp. 1176-1186

[80] Marple, L., 'High resolution autoregressive spectrum analysis using noise power cancellation', Proceedings of IEEE ICASSP, 1978, pp. 345-348

[81] Lim, J., Oppenheim, A. (ed.), 'Advanced topics in signal processing', Prentice-Hall, 1988

[82] Prado, G., Moroney, P., 'The accuracy of center frequency estimators using linear predictive methods', Prado, G., Moroney, P., Proceedings of IEEE ICASSP, 1978, pp. 361-364

[83] Wax, M., 'Order selection for AR models by predictive least squares', IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 36, Nº. 4, 1988, pp. 581-588

[84] Landers, T., Lacoss, R., 'Some geophysical applications of autoregressive spectral estimators', IEEE Transactions on Geosci. Electron., Vol. 15, 1977, pp. 26-32

[85] Aho, A., Hopcroft, J., Ullman, J., 'Data structures and algorithms', Addison-Wesley, 1983

[86] Kuc, R., Li, H., 'Reduced-order autoregressive modelling for center-frequency estimation', Ultrasonic Imaging, 7, 1985, pp. 244-251

[87] Ruano, M., Fish, P., 'Cost/benefit selection of spectral estimators for use with ultrasonic Doppler blood flow instruments', Proceedings of IEEE ICASSP, 1992, (in press)

[88] Ruano, M., Fish, P., 'Cost/benefit criterion for selection of pulsed Doppler ultrasound spectral mean frequency and bandwidth estimators', submitted to IEEE Transactions on Biomedical Engineering

[89] Ruano, M., Nocetti, D., Fish, P., Fleming, P., 'A spectral estimator using parallel processing for use in a Doppler blood flow instrument', Proceedings of the European Workshop on Parallel Computing, Barcelona, Spain, March 1992 (in press)

[90] Modi, J., 'Parallel algorithms and matrix computation', Clarendon Press, 1988

[91] 'MCP 1000 Technical reference manual', Transtech Devices Ltd., 1990

[92] Lawrie, D., 'Algorithms and architectures for the implementation of a parallel Kalman filter', PhD. thesis, University of Wales, U.K., 1990

[93] 'Liverpool parallel Occam library', CMSR, University of Liverpool and N.A. Software Ltd., 1988

[94] Gudvangen, S., Holt, A., 'Evaluation of fast Fourier transform and Hartley transforms on a loosely coupled multiprocessor based upon the Inmos transputer', IEE Colloquium on the Transputer and Signal Processing, London, 1990

[95] Inmos Ltd., 'IMS B003 evaluation board', User Manual, 1986

[96] Ruano, M., Nocetti, D., Fish, P., Fleming, P., 'Parallel implementation of an AR estimator for real-time ultrasonic blood flow instrumentation', submitted to the International Conference on Parallel Computing and Transputer Applications, Barcelona, Spain, 21-25 Sept. 1992

[97] Ruano, M., Nocetti, D., Fish, P., Fleming, P., 'Alternative parallel implementations of an AR-modified-covariance spectral estimator for diagnostic ultrasonic blood flow studies', submitted to Medical & Biological Engineering & Computing

[98] Mo, L., Cobbold, R., 'A nonstationary signal simulation model for continuous wave and pulsed Doppler ultrasound', IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, Vol. 36, №. 5, 1989, pp. 522-530

[99] Ruano, M., Fish, P., 'Complexity error product minimization for order selection in autoregressive estimators', to be submitted to Electronic Letters