PARALLEL COMPUTATION IN THE SYNTHESIS OF RANDOM WAVES BY DIGITAL FILTERING

by

Christos Solomonidis

A Dissertation submitted to the University of London
for the degree of Doctor of Philosophy (Ph.D)

University College London

October 1991

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

στους γονεις μου Δημητρη και Μαιρη

ABSTRACT

This thesis is concerned with the development of algorithmic tools for the simulation of multidirectional random wave kinematics. The digital filtering method is adopted since it affords the possibility of uninterrupted generation of random records of arbitrary duration. An implementation of the proposed method on parallel processors is presented and its performance discussed.

A review of the methods used for wave simulation is first presented. A distinction is drawn between general random field simulations and the simulation of random wave fields obeying linear wave mechanics in a dispersive medium. In the former case the appropriate ARMA model is three dimensional. In the case of the medium obeying wave mechanics and admitting a dispersion relation, like the sea waves, the use of the directional spreading function and a one dimensional propagation law makes it possible to limit the ARMA model to a univariate case.

In the filtering method proposed here, the surface elevation is generated at one gridpoint and transmitted to other gridpoints by further filtering operations. Recursive filters developed in previous work are adopted for the first part of the algorithm. For the transmission in the horizontal direction, recursive digital filters are designed here and proposed as more cost effective alternatives to the existing FIR filters. The errors committed by the two methods are compared. The results are also checked by examining the first to forth statistical moments.

As a further application to the method, the modification of the wave field around a large body is studied by the linear diffraction theory. Digital filters that incorporate the contribution of the diffracted waves in the total motion are designed.

The implementation of the proposed algorithm on a parallel computer is studied. Various configurations on an eight transputer board hosted in a standard PC are examined and aspects of optimizing the performance of the parallel program are presented. The resulting high speed continuous generation of waves at the nodes of a large spatial grid can be used for real time visual display. It may also be fed into a dynamical analysis of an offshore structure, with the ultimate purpose of estimating the temporal statistics of the response.

ACKNOWLEDGEMENTS

The author is deeply indebted to Dr. E. Yarimer for his advice, guidance and generous support. Without his valuable assistance and devotion of his time throughout the research, the completion of this work would have been very difficult indeed.

The author would also like to thank the research community of the Dept. of Civil Engineering of University College London for their moral support and stimulating discussions.

I would like to thank my parents for their continuous support and their belief in my effort.

And last but not least, I would like to thank Christina for her patience and persistence throughout these years and particularly for her precious help during the last months.

PARALLEL COMPUTATION IN THE SYNTHESIS OF RANDOM WAVES BY DIGITAL FILTERING

TABLE OF CONTENTS

Acronyms

AR	...	Autoregressive model
ARMA	...	Autoregressive - Moving Average model
DFT	...	Discrete Fourier Tranform
FFT	...	Fast Fourier Transform
IDFT	...	Inverse Discrete Fourier Transform
LP	...	Linear Programming
MA	...	Moving Average model
MDOF	...	Multi Degree of Freedom system
MIMD	...	Multiple Instruction stream - Multiple Data stream
MISD	...	Multiple Instruction stream - Single Data stream
PSD	...	Power Spectral Density function
SIMD	...	Single Instruction stream - Multiple Data stream
SISD	...	Single Instruction stream - Single Data stream
TRAM	...	Transputer Module

Symbols

Latin alphabet symbols are listed first, Greek alphabet in the end. Some symbols refer to particular chapters or pages only.

A_i, A_{ij}	...	Fourier coefficients (Amplitude of a component sine wave)
a_i	...	Coefficients of the numerator of a recursive digital filter
a_{ij}	...	Gaussian random variable (p. 16)
b_i	...	Coefficients of the denominator of a recursive digital filter
b_{ij}	...	Gaussian random variable (p. 16)
$[B]$	...	Matrix of cofficients (chapter 4)
$c, \overline{c}$	...	"damping" constants (p. 35)
$c_{\eta_1\eta_2}, c_{u_1u_2}$	...	Covariances between kinematic quantities at two grid points.
c_i	...	Coefficients of the numerator of a AR or ARMA model
$C(z)$	...	z-Transform of the numerator of an AR or ARMA model
d_i	...	Coefficients of the denominator of a AR or ARMA model
$D(z)$	...	z-Transform of the denominator of an AR or ARMA model
D	...	Diameter of a cylinder (Chapter 6)
$D(\omega,\theta)$, $D(\theta)$	...	Spreading function of directional waves

e	...	Output error measure (Chapter 4)
e^l	...	Equation error measure (Chapter 4)
$f(x)$	...	Spatial shape function of an initial disturbance (Appendix)
$F(\xi)$	...	Space-wise Fourie transform of $f(x)$
g	...	gravity acceleration
$g(k)$	...	Impulse response function af a digital filter
G	...	Constant (p. 35)
$G(z)$	...	z-Transform of a digital filter
$G(e^{i\omega T})$	...	PSD of the output of a digital filter
$h(t), h(k)$	...	Target Impulse response function
$H(\omega)$	...	Target Transfer function
$[H], [H_1], [H_2], [H_3]$	...	Matrix of target impulse response function coefficients, and its partitions (Chapter4)
$[H(a,b)]$	...	Hessian matrix in optimization
$H_m^{(1)}(x)$	...	Hankel function of the first kind, order m, argument x.
H	...	Significant wave height
I_1, I_2	...	Integral (p. 204)
ID	...	Identification of a processor in a pipeline (Chapter 8)
$J(a,b)$	...	Objective function in minimization algorithms.
$J_m(x)$	...	Bessel function, order m, argument x.
k, k_x, k_y	...	Wave number, and wave number projected to x or y axis
$k, \overline{k}$	...	"Spring" constants (p. 35)
L	...	Wave length
m_3^*, m_4^*	...	Third and fourth order statistical moments
M	...	Order of the denominator of a digital filter (Chapter 4)
N	...	Order of the numerator of a digital filter (Chapter 4)
$R(\theta)$	...	Wave runup on a vertical cylinder at angle θ
R	...	Maximum wave runup on a vertical cylinder
R_{yy}	...	Auto-correlation function of the output of a filter
$[R]$	...	Diagonal matrix of weights (Chapter 4)
$S(\omega)$	...	Power spectral density of a random process
T	...	Time step or digitization interval
T_{mux}, T_{prod}	...	Timings of activities (Chapter 8)
$[T(a,b)]$	...	Matrix of first order derivatives (Chapter 4)
u_x, u_y, u_z	...	Components of water particle velocity

$\dot{u}_x, \dot{u}_y, \dot{u}_z$	...	Components of water particle acceleration.
$U[0,1]$	...	Random number uniformly distributed between 0 and 1
v	...	wave velocity (p. 38)
$w(t)$, w_i	...	White noise (continuous or discrete time)
$w(t,x)$	...	Two-dimensional white noise
x, X	...	Transmission distance
y, Y	...	Output time series of a digital filter
α	...	Parameter (in Chapter 4)
β_m	...	Complex multiplier for the m-th order term (Chapter 6)
δ_i	...	Unit impulse
ε	...	Error measure
ε	...	Small number (p. 203-204)
$\eta(t), \eta(x,t), \eta(x,y,t)$	...	Wave elevation
θ	...	Wind direction
θ	...	Integration variable (p. 206)
θ_0, θ_{max}	...	Principal wind direction
λ	...	Scaling factor (Chapter 4)
μ	...	Mean value of a random process
ξ	...	Space-wise Fourier transform of x (Appendix A)
ρ_i	...	i-th pole of a digital filter
σ^2	...	Variance of a random process
τ	...	Time lag
ϕ_i, ϕ_{ij}	...	Random phase of component sine wave
ϕ, ϕ_w, ϕ_s	...	Total wave velocity potential, incident and scattered waves' potential.
$\omega, \omega_i, \omega_{ij}$	...	Frequency, frequency of component sine waves
ω_c	...	Cut-off frequency, Central frequency (p. 105)
ω_b	...	Frequency bandwidth

CORRIGENDA

Page	Line	Corrections
2	18	For ... "forth" read ... "fourth"
17	3	For ... "into" read ... "in"
21	15	For ... "Eq.(2.17)" read ... "Eq.(2.15)"
22	9	For ... "modeling" read ... "modelling"
22	16	For ... "but and" read ... "but also"
22	21	For ... "Eq.(2.12)" read ... "Eq.(2.10)"
23	11	For ... "Eq.(2.13)" read ... "Eq.(2.11)"
24	16	For ... "Eq.(2.24)" read ... "Eq.(2.21)"
40	18	For ... "A infinite" read ... "An infinite"
62	1	For ... "M-1" read ... "N-1"
62	1	For ... "N-1" read ... "M-1"
70	11	For ... "conditining" read ... "conditioning"
70	18	For ... "the the" read ... "the"
72	4	For ... "impovement" read ... "improvement"
79	11	For ... "minumum" read ... "minimum"
79	19	For ... "bellow" read ... "below"
104	15	For ... "succesfully" read ... "successfully"
113	Fig. 5.10	For ... "4098" read ... "4096"
114, 116	3, 1	For ... "CYLINDRES" read ... "CYLINDERS"
117	18	For ... " b_m" read ... " β_m"
120, 122	10, 11	For ... "Eg." read ... "e.g."
124	5	For ... "to" read ... "as"
124	14	For ... " $R(\theta)$" read ... " R"
127	Fig.6.1,2	For ... "Scaterred" read ... "Scattered"
137	10	For ... "tems" read ... "terms"
137	13	For ... "begining" read ... "beginning"
139	9	For ... "are taken" read ... "and are taken"
144	6	For ... "was" read ... "as"
160	11	For ... "in" read ... "at"
162	3	For ... "Figs.3" read ... "Figs.8.3"
163	1	For ... "Alternative" read ... "Alternatively"
166	18	For ... "were" read ... "where"
167	4	For ... "(Hart an" read ... "(Hart and"
171	6	For ... "2356986" read ... "23569856"
172	5	For ... "programme" read ... "program"
172	7	For ... "into" read ... "in"
188	7	For ... "Spanos" read ... "Spanos (1983)"
190	16	For ... "a Green's" read ... "Green's"
190	21	For ... "to" read ... "in"

Addition to the list of references

Flynn, M. J. (1966). Very high speed computing systems. Proceedings of the IEEE, **54** (12)

CHAPTER 1

INTRODUCTION

A. The research topic - Approach

A reliable analysis and design of structures subjected to adverse environmental conditions, such as wind loads, earthquakes or sea waves, would require an accurate description of the loading process. This would be possible if precise models for the generating mechanisms existed, which is not the case for most situations. A deterministic analysis therefore can be performed under idealized load patterns only. This is the approach adopted for example by most building codes for the case of earthquake and wind loads. A second approach is to derive the frequency domain statistical properties of the loading process from recorded data and then perform a random vibration analysis to calculate the response of the structure from input-output relations. However, this method would yield accurate results only for linear systems. When nonlinearities come into play, either in the description of the structural system, or in the loading process, as it may be required for limit design reliability assessments or fatigue analysis, certain, sometimes severe, approximations will have to be adopted, and nevertheless the problem becomes theoretically quite complicated. Monte Carlo simulations can then be used as a reliable but usually computationally expensive alternative. A family of time domain realizations of the process are applied to the structure and a deterministic non-linear analysis is used to derive the response. Response statistics on the ensemble can also be evaluated and compared with approximate theoretical solutions.

Time domain simulations are particularly important in the analysis of offshore structures, where significant non-linear effects in the loading and often in the response as well of a structural system must be accounted for. The basic aim of

response as well of a structural system must be accounted for. The basic aim of this thesis is the development of algorithmic tools for the generation of very long, uninterrupted, records of wave kinematics, from which the wave force time histories can be derived. These can then be used as excitation inputs simultaneously at different locations of an offshore structure.

The sea surface can be considered as a random field. The laws of hydrodynamics impose a certain functional form on the correlation structure of the random field. The majority of the simulation methods which are found in the literature represent the wave's random field using multidimensional or multivariate models which are based exclusively on matching the prescribed correlation structure. Little use is made of the propagation laws of the underlying medium.

A different approach is attempted in the present work. The aim is to derive the *transmittance* or *transfer* function of the medium from the knowledge of the wave motion at certain locations. The problem is similar to the initial value problems encountered for example in geophysics when using seismic exploration for petroleum prospecting. There, the aim is to deduce the earth's properties from measurements on the surface. Also in seismology the generating mechanism of earthquakes is frequently deduced from recorded seismograms on various locations. The basic principle is common in these applications and goes back to optics; the differential equation of the wave motion is used to derive, employing wave extrapolation methods, the transfer function of the medium. In some cases the solution is known and the required function can be easily acquired. For example in the case that is dealt in this thesis the incorporation of *Airy's* linear wave theory facilitates the extraction of a transfer function for the transmission of wave kinematics between two points. The corresponding impulse response function of the medium can be subsequently derived and used for convolutions in

the time domain. However, if the solution is not known, the initial value problem is solved by treating the differential equation either by approximations, or by finite difference methods. An attempt to derive the impulse response function by solving an initial value problem and using approximations to the hydrodynamical equations is presented in an Appendix. The outcome of this alternative approach is comparable with the one used in the main body of the thesis.

The existence of a large body in the sea, a structural element of an offshore structure with significant horizontal dimensions is an example, modifies the wave field around it. The principles used for the simulation of the undisturbed field still apply and it is possible to derive a transfer function that incorporates the effect of this disturbance in the generation of the kinematic quantities at various locations in the vicinity of the body. This application will be considered in the thesis as well.

Once the impulse response functions which are required for the time domain convolutions are obtained, the corresponding digital filters must be designed. Computational economy is a factor of major consideration in Monte Carlo simulations, and in particular in the present scheme where a substantial number of convolutions must be performed in order to reproduce the wave field. Thus the design of efficient and economical digital filters for the simulation of the multi-directional wave kinematics is a primary target of this thesis and will be examined thoroughly.

The rapid advance in the recent years of parallel computers have promulgated their use as an attractive option for implementing a simulation program such as the one proposed in this work. Furthermore the challenge of concurrent programming itself and as the analogies that can be drawn with a series of diverse applications and in particular with human mental processes present an irresistible fascination to the

programmer. The study of the performance of a parallel implementation of the simulation scheme will take a significant position in this thesis, and its optimization will be pursued.

B. Outline of the chapters

In the following the structure of the Thesis is presented in terms of the contents of its Chapters.

In Chapter 2 a critical review of the methodologies used for the simulation of random waves is presented. The sine wave superposition methods are presented, but particular attention is given to the digital filtering methods. The methods for the simulation of the multi-directional wave kinematics are examined separately. Analogies and differences are drawn with regard to the scheme that will be used in the thesis.

Chapter 3 starts with the presentation of the method used for the generation of wave motion at one point in the sea and then proceeds with the description of the method which uses propagation filters and convolutions to generate the whole directional wave field. Different impulse response functions are used for the generation of the various kinematic quantities and design aspects are considered. The need for the design of low order filters for the horizontal transmission of elevation is identified.

In Chapter 4 optimal recursive digital filters are designed in the time domain in order to replace the infinite length impulse response function used for the horizontal transmission of wave elevation. Special attention is given to this function because of its complicated functional form and the generally high order filters that it spawns. Optimization methods borrowed from digital signal processing are used in the design.

For completeness of the design aspects, Chapter 5 presents an attempt to design the recursive filters for the horizontal propagation in the frequency domain with the use of allpass filters. The properties of these filters are examined in some detail. Bandpass filtering is also introduced for some case.

In Chapter 6 the modification of the wave field around a large cylinder is examined and an impulse response function for the scattered waves is derived. Digital filters are created that together with the filters for the incident waves designed in previous chapters, can reproduce the wave field around the large body. The wave runups are examined and compared with theoretical values.

In Chapter 7 the sequential program written to implement the simulation scheme is presented and the quality of the generated time series is demonstrated in terms of their statistical properties. The unidirectional and the multidirectional cases are examined separately.

Chapter 8 presents the parallel implementation of the simulation program. Various topologies are tried. The performance of the distributed program is monitored using timing devices implanted in the actual code. As an independent check, a simulation of the performance of the program is accomplished, employing a proprietary package. A real time visual display is implemented as well.

Chapter 9 sums the results and conclusions of the present work, identifies certain limitations, and introduces some ideas for further research on the same lines.

Appendix A describes an alternative derivation of the impulse response function for the horizontal transmission of wave kinematics directly from the equations of motion. This function is compared with the one used in the thesis.

Finally in Appendix B the coefficients of the recursive filters designed in Chapter 4 are listed for possible use by prospective users of the proposed simulation scheme.

CHAPTER 2

SURVEY OF THE METHODS FOR THE SIMULATION OF RANDOM WAVES

CHAPTER TWO

CONTENTS

2.1 Introduction.

Although a variety of methods for the synthesis of random wave kinematics have been developed in the last three decades, they can nevertheless be divided into two broad categories: the summation of sine waves methods and the digital filtering methods. The basic features of these two groups will be examined in the present chapter. A more detailed presentation of the digital filtering methods is undertaken because the simulation scheme used in this thesis falls in this category. The method of Local Average Subdivision, proposed very recently (Fenton and Vanmarcke 1990) for the simulation of homogeneous random fields, is more suitable for finite element models and is not examined here. Furthermore, a method that combines elements from both main categories is included at the end of this chapter.

2.2 Superposition of sine waves.

The summation of sine waves method was proposed in the early sixties for the simulation of random processes and it has been widely used since then (Borgman 1969, 1979), (Shinozuka and Jan 1972), (Elgar, Guza, and Seymour 1985), (Wang 1986), (Miles and Funke 1987), (and in Schueller and Shinozuka 1987). In its simplest form, the single summation model, the elevation at a point in the sea is the superposition of a large number of single harmonics:

$$\eta(t) = \sum_{i=1}^{M} A_i \cos(-\omega_i t + \phi_i) \quad t = 0, T, 2T, 3T, \ldots NT \tag{2.1}$$

where $A_i = \sqrt{2S(\omega_i)\Delta\omega}$ and $\phi_i = 2\pi U[0,1]$. In this model the only random element is the phase, uniformly distributed between 0 and 2π. The amplitude is derived from a target spectrum, like Pierson-Moskowitz, Jonswap etc.

This model can be easily extended to the multidimensional case to include directional waves. A double summation over harmonics and directions gives the wave elevation as:

$$\eta(x,y,t) = \sum_{i=1}^{N}\sum_{j=1}^{M} A_{ij}\cos(\omega_i t - k_i(x\cos\theta_j + y\sin\theta_j) + \phi_{ij}) \tag{2.2}$$

$$t = 0, T, 2T, 3T, \ldots NT$$

In the above equation the coefficients A_{ij} and ϕ_{ij} can be determined in two ways. The phase only can be random, as in the single dimension model:

$$A_{ij} = \sqrt{2S(\omega_i,\theta_j)\Delta\omega\Delta\theta} \quad \text{and} \quad \phi_{ij} = 2\pi U[0,1] \tag{2.3}$$

or both phase and amplitude:

$$A_{ij} = \sqrt{a_{ij}^2 + b_{ij}^2} \quad \text{and} \quad \phi_{ij} = \tan^{-1}(b_{ij}/a_{ij}) \tag{2.4}$$

where a_{ij} and b_{ij} are Gaussian random variables with variance $S(\omega_i,\theta_j)\Delta\omega\Delta\theta$.

The directional spectrum from which these quantities are obtained is given by the relation:

$$S(\omega_i,\theta_j) = D(\omega_i,\theta_j)S(\omega_i) \tag{2.5}$$

where $D(\omega_i,\theta_j)$ is the spreading function.

Problems with the above models have been reported extensively. Jefferys (1987) points out the "phase locking" effect, "due to wave components travelling in different directions with identical frequencies". This results in wave fields that are neither ergodic nor homogeneous.

Alternative methods have been proposed to overcome the phase locking effect. The single direction per frequency model (Miles and Funke 1987):

$$\eta(x,y,t) = \sum_{i=1}^{N} \sum_{j=1}^{M} A_{ij} \cos(\omega_{ij} t - k_{ij}(x \cos\theta_j + y \sin\theta_j) + \phi_{ij}) \qquad (2.6)$$

$$t = 0, T, 2T, 3T, \ldots NT$$

where in every frequency interval all the component waves from different directions have different frequencies respectively, and the much simpler single summation model (Miles and Funke 1987), (Lando, Scarsi and Taramasso 1991) :

$$\eta(x,y,t) = \sum_{i=1}^{N} A_i \cos(\omega_i t - k_i(x \cos\theta_i + y \sin\theta_i) + \phi_i) \qquad (2.7)$$

$$t = 0, T, 2T, 3T, \ldots NT$$

which requires fewer components than the double summation and also avoids phase locking. M directions are considered in each frequency interval $\Delta\omega$, and N frequencies in the band of interest. In a modification of this model (Miles and Funke), the frequency and direction components are given as:

$$\omega_i = i(\Delta\omega/M) \quad \text{and} \quad \theta_i = ((i-1) \bmod M)\Delta\theta - \theta_0 \quad \text{where} \quad \Delta\theta = 2\theta_0/(M-1) \qquad (2.8)$$

Using this technique, each wave train (direction) contains N frequencies which are different from the ones included in any other wave train. Much shorter record lengths are sufficient to generate spatially homogeneous seas. Miles and Funke however suggest caution when using this method because it exhibits a variability which is inconsistent with real seas.

Any model that belongs to the summation of sine waves class can be interpreted as an inverse Fourier transform operation from frequency domain to the time domain; it is known that the computational effort involved is of the order of $N\log_2(N)$ with the best of available algorithms and that the storage requirement is proportional to N. On the other hand the digital filtering method involves a computational effort linear in N, and a very small amount of storage. While the comparison of the operation count does not show a pronounced advantage for the filtering method (putting aside the issue of numerical stability of the FFT method at very large N), the difference in storage requirements is decisively in favour of the filtering method.

2.3 Digital Filtering methods.

Digital filtering is already an accepted method in the synthesis of random waves in model tanks (Bryden and Greated 1984), (Bryden and Linfoot 1991), (Nunes 1987). However, until recently, the limitations of central computing facilities have militated against the use of this method in theoretical research, and the summation of sinewave components method has prevailed. The batch mode of job execution, common on large Mainframe computers, creates a preference towards multiple finite tasks as opposed to single continuous runs. Interactive time-shared Mainframe computing has not changed this position because indefinitely long runs are impractical in time-sharing environments. The advent of standalone office

computers, and more recently, the introduction of parallel computing facilities on such equipment, have made digital filtering an attractive proposition. Although proposed as early as 1972 (Burke and Tighe 1972), the method has not gained wide currency until nearly a decade later (Spanos and Hansen 1981), (Samii and Vandiver 1984), (Lin and Hartt 1984), (Mignolet and Spanos 1987).

The basic idea in this class of modelling is to construct a digital filter, or a "black box", to which the input is a sequence produced by a random generator, usually Gaussian white noise, while the output time series matches prescribed statistical characteristics. The aim of the design is to estimate the coefficients of these filters, using the lowest possible order.

We will distinguish between *univariate* digital filtering methods, where time series are generated at one point in the sea, such that their spectrum matches a "target", and *multivariate* or *multidimensional*, where the whole random field, in other words a whole area of the sea surface, is simulated. The first class will be first reviewed in some detail.

2.3.1 Univariate digital simulation.

Employing techniques used in time series' analysis, three main models of digital filters for random process simulations have been used. Autoregressive (or AR, or all-pole) models, moving average (MA or all-zero) models and the complete autoregressive-moving average model (ARMA). The univariate version of these models will be presented, as this will be used in the present work as a first step in the proposed method, although a generalization to the more general case will be included in a following section.

The transfer functions in the z-domain of these three types of filters are given below:

For the autoregressive model:

$$\hat{H}_{AR}(z) = \frac{c_0}{1 + \sum_{\xi=1}^{n} d_\xi z^{-\xi}} \tag{2.9}$$

for the moving average:

$$\hat{H}_{MA}(z) = \sum_{\xi=0}^{m} c_\xi z^{-\xi} \tag{2.10}$$

and for the ARMA model:

$$\hat{H}_{ARMA}(z) = \frac{C(z)}{D(z)} = \frac{\sum_{\xi=0}^{m} c_\xi z^{-\xi}}{1 + \sum_{\xi=1}^{n} d_\xi z^{-\xi}} \tag{2.11}$$

In the following the methodologies that have been used for the estimation of the coefficients of these models will be discussed.

i. Autoregressive models.

The difference equation that yields the value in the present time of the output $\hat{Y}$ of the AR filter (or all-pole filter) when the input is white noise is given by the relation:

$$\hat{Y}_l \equiv \hat{Y}(lT) = c_0 w_l - \sum_{\xi=1}^{n} d_\xi \hat{Y}_{l-\xi} \tag{2.12}$$

where T is the time step (or sampling period). Multiplying both sides of the equation by $\hat{Y}(l-k)$ and taking expectation one gets:

$$R_{\hat{y}\hat{y}}(l) + \sum_{\xi=1}^{n} d_\xi R_{\hat{y}\hat{y}}(\xi - l) = c_0 R_{\hat{y}w}(l) \qquad l = 0, \pm 1, \pm 2, \ldots \tag{2.13}$$

where $R_{\dot{y}\dot{y}}$ is the auto-correlation function of the AR process, and $R_{\dot{y}w}$ is the input-output cross-correlation function.

The coefficients of the AR model will be estimated so as to minimize a measure of the error between the target PSD and the spectrum of the generated time series (Spanos 1983):

$$\varepsilon_{AR} = \int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} \frac{S(\omega)}{G_{AR}(e^{i\omega T})} d\omega \tag{2.14}$$

where $G(e^{i\omega T})$ is the power spectrum of the output of the AR filter. Using Maximum Entropy methods (MEM), it is proved (Mignolet and Spanos 1987) that the minimum error is achieved when the coefficients d_i satisfy the following equations:

$$R_{yy}(l) + \sum_{\xi=1}^{n} d_\xi R_{yy}(\xi - l) = 0 \qquad l = 0, \pm 1, \pm 2, \ldots \tag{2.15}$$

which are the well-known *Yule-Walker* equations. In this set of equations R_{yy} is the auto-correlation function of the target process and is obtained as the inverse Fourier transform of the target PSD, discretized at the required time-lags. Once the d_i coefficients are determined from Eq. (2.17), coefficient c_0 can be obtained by equating the target and the AR auto-correlation functions at all the desired time-lags:

$$R_{\dot{y}\dot{y}}(l) = R_{YY}(l) \qquad l = 0, 1, 2, ..m \tag{2.16}$$

and then substituting, together with the estimated coefficients d_i, in Eq. (2.15) to yield the solution:

$$c_0^2 = \frac{T}{2\pi}\{R_{yy}(0) + \sum_{\xi=1}^{n} d_\xi R_{yy}(\xi)\} \qquad (2.17)$$

In various papers (Spanos 1983, Rivero 1987) directives can be found on how to proceed, step by step, to the methodology outlined above. Details such as how to estimate the order N of the AR model or how to avoid the possible ill-conditioning of the matrix of auto-correlation values (*Toeplitz* matrix) in the *Yule-Walker* equations can be found there.

Once the AR coefficients have been estimated, time ordinates of the process Y can be generated recursively using Eq. (2.14).

The AR modeling of a random process offers the advantage of its relative simplicity in the estimation of the coefficients and furthermore the solutions obtained from the methodology described here are guarantied to yield stable filters (Rivero 1987). However, the order of the filters obtained this way is usually high, making their use quite costly in computational terms. In fact Spanos (1983) has observed that in order to reproduce the peakedness of the target spectrum (Pierson-Moskowitz) the order of the AR filter must not be small (but also not too high). Another significant limitation is that they cannot accurately model processes whose PSD has a zero at the origin of the ω axis because the all-pole transfer function does not possess a zero.

ii. Moving average models.

The transfer function of the MA model, also called all-zero or FIR (Finite Impulse Response) model, was given in Eq. (2.12). The difference equation used to obtain the MA time series from white noise input is:

$$\hat{Y}_k \equiv \hat{Y}(kT) = \sum_{i=-\infty}^{\infty} c_i w_{k-i} \tag{2.18}$$

In this equation an infinite number of weighted white noise ordinates is used in the right hand side, but in practical applications a finite order of the MA coefficients, M, is sufficient.

As in the AR case the coefficients are estimated so as to minimize a measure of the error between the PSD of the generated time series and a target PSD. Since MA modelling was not used in the present work, details will not be given here, but can be found in various publications (Haykin 1979, 1983).

iii. Autoregressive - Moving average models.

In ARMA modelling both the poles and zeros of the transfer function in Eq. (2.13) must be estimated, thus making the design a rather complicated task, and moreover sometimes unstable. Nevertheless the computational economy attained by the use of the generally low-order ARMA filters render this approach very popular.

The corresponding difference equation is given by:

$$\hat{Y}_l \equiv \hat{Y}(lT) = \sum_{\xi=0}^{m} c_\xi w_{l-\xi} - \sum_{\xi=1}^{n} d_\xi \hat{Y}_{l-\xi} \tag{2.19}$$

Various methods have been proposed for estimating the coefficients of this model. The most straightforward is to extend the logic of the AR estimation, in other word minimise an error measure between the PSD of the generated time series and the target:

$$\varepsilon_{ARMA} = \int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} \frac{S(\omega)}{G_{ARMA}(e^{i\omega T})} d\omega \tag{2.20}$$

If the PSD's in the above equation are written in terms of the corresponding transfer functions $H(\omega)$ and $\hat{H}_{ARMA}(e^{i\omega T}) = C(e^{i\omega T})/D(e^{i\omega T})$, one obtains an equivalent error measure:

$$\varepsilon_{ARMA} = \int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} | H(\omega)D(e^{i\omega T}) - C(e^{i\omega T}) |^2 \, d\omega \tag{2.21}$$

In order to minimize ε from the above equation directly, a set of non-linear equations in c_i and d_i have to be solved. Most techniques developed to by-pass the nonlinearity involve a *two-stage* approach. An AR model is designed first according to the method described earlier. Subsequently this solution is used, in different ways according to the method followed, to estimate the ARMA coefficients.

According to Rivero (1987), an AR model is designed from the given second order information (target spectrum --> auto-correlation) using Maximum Entropy Methods. An impulse response function approximation to this AR filter is then obtained, which is used together with the given auto-correlation, in a second order minimization to estimate the coefficients of the ARMA filter. After some manipulation of Eq. (2.24), the error that will be minimized is given by:

$$\varepsilon = \sum_{i,j=0}^{n} d_i d_j R_{YY}(| i - j |) - \sum_{l=0}^{m} (\sum_{j=0}^{n} d_i h_{l-j})^2 \tag{2.22}$$

where h is the impulse response function of the AR model. This minimisation can be performed using iterative methods, or solving a system of equations, from which the coefficients d_i are determined. The coefficients c_i can thereafter be estimated by straightforward calculations:

$$c_i = \sum_{j=0}^{n} d_j h_{i-j} \quad \text{for} \quad i = 0,1,2,\ldots m \tag{2.23}$$

This method appears to be quite suitable and yielding satisfactory results for univariate ARMA modelling of sea waves [Rivero], but is not known to cope with the more general multivariate case. A quite different approach to the univariate ARMA filter design which uses an intermediate analog filter was propose by Spanos (1983), but will be presented in a subsequent chapter in detail, as it is the method used in this work for the generation of waves at one point in the sea.

2.3.2 Multivariate and multidimensional digital simulation.

It is important to distinguish between general random field simulations and the simulation of random wave fields obeying linear wave mechanics in a dispersive medium. In the former case the appropriate digital filter is three dimensional, two space and one time, or multivariate if the space points are absorbed into elements of a vector, (Mignolet 1987), (Samaras, Shinozuka, and Tsurui 1985), (Li and Kareem, 1990). When constructing these models the criterion usually is correlation matching. The general modelling techniques would still apply in the case of the medium obeying wave mechanics and admitting a dispersion relation; however for sea waves the use of the directional spreading function and a one dimensional propagation law makes it possible to limit the model to a univariate case over a single dimension. This reduction is sketched in Table 2.1. The methods that will be reviewed in the present section fall in the first two rows of this table, as they are 2D or 3D (or their corresponding multivariate) representations of the random wave field. The present work adopts the 1D representation given in the last row of Table 2.1, and furthermore

it utilises the medium's wave propagation law in the design of the filter.

Table 2.1

Space	From domain	---> Transformation --->	To domain
3D	ω,k_x,k_y	$\eta(t,x,y) = \int_{k_x}\int_{k_y}\int_{\omega} F(\omega,k_x,k_y)e^{-i(\omega t - xk_x - yk_y)}$ $d\omega dk_y dk_x$	t,x,y
2D at each θ	ω, $k\cos(\theta), k\sin(\theta)$	$\int_{\omega}\int_{k} \ldots e^{i(\omega t - kX)} d\omega dk$ where $X = r\cos(\theta - \psi)$ and $x = r\cos\psi \quad , \quad y = r\sin\psi$	t,x, at each θ
	$\int_{\omega}\int_{k}$	$\int_{\omega}\int_{k} \ldots \delta(k - \omega^2/g)\, d\omega dk$ (using dispersion law)	$\int_{\omega}$
1D at each θ		$\int_{\omega} \ldots e^{i\omega t} e^{-i\omega^2 X/g} d\omega$	t at fixed x, fixed θ

An example of two-dimensional linear filtering is the system used by Bryden and Greated (1984) for the generation of waves in tanks. The desired output is given by the convolution:

$$\eta(t,x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau,X)w(t-\tau,x-X)d\tau dX \tag{2.24}$$

where the input is a two-dimensional white noise $w(t,x)$. The impulse response function of the filter, $h(\tau,X)$, is determined from the inverse Fourier transform of the system's transfer function:

$$h(\tau,x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} H(\omega,K)e^{i(\omega\tau - Kx)}d\omega dK \tag{2.25}$$

This function is then either digitized and truncated in order to produce the required digital filter, or used directly in the convolution described by equation (2.9) to generate the desired output. As far as digitization is concerned, it should be noted that there exist more elaborate techniques (see chapter 3) than the ones used by the authors, which would improve the quality of the output.

As for the transfer function, it is obtained from the square root of the target directional spectrum.

$$H(\omega,K)=\sqrt{\frac{gS(\omega,\theta)}{W\omega^2\cos(\theta)}} \tag{2.26}$$

where use of the dispersion relation has been made, and W is the intensity of the input white noise spectrum.

As pointed out by the authors (Bryden and Greated 1984) spectra do not carry phase information, and thus an arbitrary choice has to be made with regard to the phase. Although the cross-correlations of the generated series appear to match theoretical values, there is no indication that the correct phase relation between separate points in the sea is retained. It should be noted that this criticism is valid for all the methods that are based on correlation matching, and will be reviewed next.

In the following we will examine the multivariate ARMA models, since the methodologies used for the multivariate AR or MA models are analogous to those for the corresponding univariate models examined earlier. In the ARMA models the similarities are still present, but the handling of the matrix and vector formulations is more delicate and needs some attention.

Autoregressive - Moving average models.

A multivariate ARMA process $\hat{Y}$ can be generated from previous values and multivariate white noise input W, according to the equation:

$$\hat{Y}_r = \sum_{l=0}^{m} C_l W_{r-l} - \sum_{k=1}^{n} D_k \hat{Y}_{r-k} \tag{2.27}$$

where C_l and D_k are the coefficient matrices, n x n for an n-variate process.

In order to determine the coefficient matrices of this ARMA model Mignolet and Spanos (1987), adopting a two-stage approach familiar from the univariate case, design initially an AR model and then use its transfer function to approximate the denominator of the ARMA model. The error to be minimized is an expression of the coefficient matrices C_l and D_l of the system, the auto-correlations of the output process and the cross-correlations R_{YW} of the AR system:

$$\varepsilon = \sum_{k=0}^{n}\sum_{i=0}^{n} D_k R_{YY}(k-i) D_i^t - \sum_{k=0}^{n}\sum_{l=0}^{m} D_k R_{YW}(k-l) C_l^t \tag{2.28}$$

$$- \sum_{k=0}^{n}\sum_{l=0}^{m} C_k R_{YW}(k-l) D_k^t + 2\pi/T \sum_{l=0}^{m} C_l C_l^t$$

where ^{t} denotes the transpose of a matrix or a vector, while T is the sampling period. Two methods are proposed for the minimization of this expression, a more straightforward auto cross-correlations matching procedure (ACM), which is an extension of the techniques used in AR modeling, and a power order matching procedure (POM), where the numerator of the frequency domain transfer function of the ARMA nodel, $C(e^{i\omega T})$, is presented as a Fourier approximation to the product $H(\omega)D(e^{i\omega T})$ (see Eq. 2.21), of order n+m. The C_l and D_l are estimated so as to match the Fourier coefficients of the two

expressions. The authors point out that these two methods are equivalent, but the latter is computationally more efficient. More details on this method can be found in (Mingolet and Spanos 1987). Samaras (1983) uses similar methods for determining the coefficient matrices of his multivariate ARMA model.

Although the previously described methodology is mathematically rigorous in its theoretical form, in practice the handling of large matrices for the determination of the coefficients of the multivariate ARMA filter may lead to computational difficulties. In order to simulate a extensive area of the sea surface, one would have to construct a filter that would incorporate a large number of locations (variables) thus leading to matrices of high orders. The authors present a quite extensive treatment of the stability of these filters but nevertheless restrict themselves to applications that require low order matrices. An additional reservation is that a simulation method which relies entirely on matching the statistical properties of the generated waves, neglects the physical properties of the underlying medium, that is the mechanics of wave propagation. This reservation will be elaborated in subsequent chapters.

2.4 A hybrid method.

Before closing this chapter a third, hybrid method of synthesizing random waves should be mentioned which combines Fourier transforms and linear filtering (Li and Kareem 1989), (Kareem and Li 1991). A large number of time series segments is generated using FFT techniques (summation of sine waves), and at second stage digital filtering is involved to synthesize these segments in to a final time series of long duration. This method is extended into the multivariate case using stochastic decomposition to construct the component random subprocesses' desired characteristics from the target directional spectrum in the first stage of the

simulation. The authors of this technique claim that they can combine the simplicity of FFT methods with the computational economy of linear filtering, since they can avoid the large storage requirements of the former.

CHAPTER 3

THE USE OF PROPAGATION METHODS FOR THE GENERATION OF WAVE KINEMATICS.

CHAPTER THREE

CONTENTS

THE USE OF PROPAGATION METHODS FOR THE GENERATION OF WAVE KINEMATICS.

3.1 Introduction.

The basic principle behind the simulation method described here is the acquisition of the transfer function for the propagation of the various wave kinematical quantities in an elastic medium from the knowledge of the wave motion characteristics at some points. The motion at any desired location can thereafter be obtained using convolution integrals. The method is similar to techniques used in optics, geophysics and seismology.

This chapter will start with an outline of the total simulation scheme. The first stages of the simulation, white noise generation and elevation at a "reference" node will be examined next. After that, the transfer function and the corresponding impulse response function for the horizontal transmission of wave elevation will be examined. Following is the presentation of the transfer functions for water particle velocity, acceleration, as well as vertical attenuation and the corresponding digital filters. Since the design of high precision digital filters for the horizontal transmission of wave elevation is central to this work, it will be detailed in following chapters. A generalization to the multidirectional case will be also included.

3.2 Organisation of the simulation scheme (directional waves)

The following flow chart assumes that wave kinematical quantities are required at points of a 3-dimensional grid covering an area of interest.

Do for successive time steps:
  Do for all wave directions:
    1. Produce one step of the input white noise process.
    2. Generate one surface elevation process value at the reference node.
  Complete over the directions.
  Do for all grid points:
    Do for all wave directions:
      1. Transfer by linear digital filtering to elevation, velocity vector components, and acceleration vector components at all other nodes
      2. Add directional contributions
    Complete over the directions.
    Do for all depths
      Transfer, using attenuation filters
    Complete over depths
  Complete over grid points
Complete all time steps.

In the following the steps of this scheme will be examined one by one. Although the final simulation package is multidirectional, the unidirectional case will be assumed first, which will be extended to the more general case in the end of this chapter.

3.3 Generating the surface elevation process at the reference node

Filters may be designed for a white noise input such that the output process has the desired autocorrelation function (or equivalently, autospectrum). Gersch and

co-workers have presented a two-stage procedure for digital filter design, both in the univariate and the multivariate case (Gersch and Liu 1976), (Gersch and Yonemoto 1977). Also see Samaras, Shinozuka and Tsurui (1985).

A filter design scheme specifically aimed at simulating wave motion by means of an autoregressive-moving average (ARMA) model has been developed by Spanos (1983). In the present study that scheme is adopted as the generating mechanism for the random process at the reference node. An ARMA model has the following power spectrum (Papoulis 1984):

$$S_{yy}(\omega) = \frac{\left| \sum_{\xi=0}^{m} c_\xi e^{-i\xi\omega T} \right|^2}{\left| 1 + \sum_{\xi=1}^{n} d_\xi e^{-i\xi\omega T} \right|^2} \tag{3.1}$$

The coefficients c_i and d_i must be determined in such a way as to make the power spectrum of the output process approximate the desired spectrum, e.g. Pierson-Moskowitz. Spanos accomplished this task by introducing an intermediate process whose output has the power spectrum:

$$\hat{S}(\omega) = \frac{G\omega^4}{[(\omega^2 - k)^2 + (c\omega)^2]\,[(\omega^2 - \bar{k})^2 + (\bar{c}\omega)^2]} \tag{3.2}$$

which represents the transfer function of a cascade of two linear systems in terms of their "spring and dashpot" constants.

The coefficients G, c, k etc. are determined by minimizing the residual error, in a least squares sense, between the target process and the power spectrum of Eq. 3.2. The time-domain differential equation which would yield this type of process from a white noise forcing function is easy to identify, and may in turn be used to find the Laplace Transform-transfer function involved. Finally using the approximate

technique of bilinear transformation, the s-transform is converted to a z-transform, which is the required filter of Eq. 3.1. In this way the following coefficients c_i and d_i for H(z) have been obtained from Spanos' expressions:

$$
\begin{aligned}
c_0 &= f & d_1 &= 2(-2p - r + v + 2q)/S \\
c_1 &= 0 & d_2 &= 2(3p - u + 3q)/S \\
c_2 &= -2f & d_3 &= 2(-2p + r - v + 2q)/S \\
c_3 &= 0 & d_4 &= (p - r + u - v + q)/S \\
c_4 &= f
\end{aligned}
$$

where: (3.3)

$$
\begin{aligned}
p &= 16 & \alpha &= c + \bar{c} & q &= \varepsilon T^4 \\
r &= 8\alpha T & \beta &= k + \bar{k} + c\bar{c} & S &= p + r + u + v + q \\
u &= 4\beta T^2 & \gamma &= c\bar{k} + \bar{c}k & f &= 4\sqrt{G}\,T^2/S \\
v &= 2\gamma T^3 & \varepsilon &= k\bar{k}
\end{aligned}
$$

Here T is the fundamental time-step of the digital filter. In general, if ω_b is the bandwidth of the spectrum to be matched, then T must be chosen less than π/ω_b. However Spanos recommends using a time step 4 times smaller than this limit in order to compensate for the error caused by the bilinear transformation.

Once the coefficients c_i and d_i have been determined, the recursive filter equation which will generate the output process from white noise input w is:

$$\hat{Y}_l \equiv \hat{Y}(lT) = \sum_{\xi=0}^{m} c_\xi w_{l-\xi} - \sum_{\xi=1}^{n} d_\xi \hat{Y}_{l-\xi} \tag{3.4}$$

It should be noted that Sorensen and Thoft-Christensen (1985) have developed an alternative algorithm for the simulation problem under consideration: this relies on

a transformation from a n-dimensional Markov vector random process to a final non-Markov process which is ARMA, of order n x m , and was reportedly proposed by Franklin. Sorensen and Thoft-Christensen have fitted the power spectrum of the output to both Pierson-Moskowitz and Jonswap spectra. The computational cost of this method is likely to be comparable to the former approach. In the present study it was decided to use Spanos' approach simply because more information was available on coefficient values fitting the model to various wave spectra.

In this study the Gaussian variates needed for the white noise input process have been generated by a procedure proposed by Ahrens and Dieter (in Bratley, Fox, and Shrage 1985) (algorithm L19) This is a composition method for generating standard normal variates. The total area under the standard normal probability-density curve is partitioned into a large trapezoidal subarea and four odd-shaped small subareas. Each subarea is treated as a density distribution. Random variates are chosen from the respective distributions, with probabilities proportional to the distributions' enclosed area.

The U(0,1) random variates required in this procedure are drawn from a uniform random number generator. The algorithm used in this report belongs to the class of multiplicative generators with prime modulus (algorithm L20 in Bratley, Fox and Shrage).

The procedures described above have been implemented as a testing Fortran program. The quality of the output was checked by calculating its power spectrum and comparing with the target spectrum. Fig. 3.1 shows such a comparison.

3.4 Convolution integrals for the transmission of wave kinematics.

In order to transfer the motion of the sea from the point where it was generated to another, the Duhamel convolution integral can be used. The generated time series at a point can be treated as a series of input displacement impulses, and the impulse response of the sea surface may be used to transmit this input to the second point. To accomplish this task, the transfer function of the sea surface must be sought first.

The transfer function can be obtained using methods generally described as *wave extrapolation*. Numerical or analytical procedures can be employed in wave extrapolation. It is worth presenting here in brief an optical application, described by Claerbout (1985). The problem there, is to project waves across a region of empty space. Using the wave equation in optics, the motion at (t,x,z) is described by the relation:

$$p(t,x,z) = P(\omega,k_x,z)e^{-i(\omega t - k_x x)}$$

where the amplitude of the motion $P(\omega,k_x,z)$ at z is related to the amplitude of a disturbance at z_0 via the equation:

$$P(\omega,k_x,z) = P(\omega,k_x,z_0)e^{i\left(\omega^2/v^2 - k_x^2\right)^{1/2}(z-z_0)}$$

where v is the wave velocity. From the above equation the filter transfer function can be easily deduced:

$$\exp\left\{i(\omega^2/v^2 - k_x^2)^{1/2}(z-z_0)\right\}$$

Inverse Fourier transforming this relation either in time or space domain or both, a corresponding impulse response function is obtained which can be used in convolutions.

If analytical solutions of the wave equation do not exist, for example if the velocity v is space variable, numerical wave extrapolation methods are used. Claerbout (1985) describes such a procedure in geophysics.

Coming to the present case of sea waves, linear Airy's theory is used to express the velocity potential of the surface waves for infinite depth water:

$$\phi = \frac{-igH}{2\omega} e^{i(kx-\omega t)}$$

where H is the significant wave height. From the above, using a boundary condition to the Laplace equation:

$$\eta = -\frac{1}{g}\left(\frac{\partial \phi}{\partial t}\right)_{z=0}$$

the equation which gives the instantaneous surface elevation, η, of a propagating single harmonic wave can be given as:

$$\eta(x,t) = A e^{i(\omega t - kx)} \qquad (3.5)$$

where ω is the angular frequency of the wave, k is the wave number, and $A = H / 2$ defines the amplitude. In practice only the real or the imaginary part of the equation is used.

If a constant amplitude harmonic input of frequency ω, x(t), is applied to a linear system, the corresponding output y(t) will be given by :

$$y(t) = H(\omega)x(t) \qquad (3.6)$$

where H(ω) is the system's complex frequency response function or transfer function evaluated at angular frequency ω. Taking account of the above equations, and using the dispersion relationship for deep water $k = \omega^2/g$, the transfer function becomes:

$$H_{\eta-\eta}(\omega,x) = e^{-i(\omega^2 x/g)} \tag{3.7}$$

It is noted that both the real and the imaginary parts of this function are even functions of ω. In general, the inverse Fourier transform of H(ω) is h(t). Under the present conditions h(t) would be a complex-valued function. It would not be valid to use the real part of such an h(t) function on its own, since this would not have the necessary frequency-domain properties (its transform would be a purely real-valued H function, giving zero phase). However if the function H(ω) were chosen conjugate-symmetric (even real part, odd imaginary part) its Fourier transform would be real-valued. Hence, the following is substituted for Eq.3.7:

$$\begin{aligned} H_{\eta-\eta}(\omega,x) &= e^{-i\, sgn(\omega)\omega^2 x/g} \\ &= \cos\left(\frac{\omega^2 x}{g}\right) - i\ sgn(\omega)\sin\left(\frac{\omega^2 x}{g}\right) \end{aligned} \tag{3.8}$$

This function is characterized by the two properties:

$$|H(\omega)| = 1 \quad \text{and} \quad arg\ [H(\omega)] = -sgn(\omega)\frac{\omega^2 x}{g} \tag{3.9}$$

A infinite length digital filter with these characteristics may be obtained by taking the inverse Fourier transform of the H(ω) function given above. This is the method used in the work of Vandiver and co-workers, see Samii and Vandiver (1984), Dommermuth (1986). The infinite length filters are usually truncated to a finite

length. In the present work they will be re-expressed in even more compact form as recursive filters. But first, the full impulse response function corresponding to Eq. 3.8 will be presented.

3.4.1 The impulse response function for the sea surface elevation

The Inverse Fourier transform of H(ω) as given in Eq. 3.8 is:

$$h(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-i\,sgn(\omega)\frac{\omega^2 x}{g}} e^{i\omega t} d\omega$$

Its explicit form may be obtained using the following integrals (Abramowitz and Stegun (eds.) 1970):

$$\int_0^{\infty} e^{-at}\cos(t^2)dt=\sqrt{\frac{\pi}{2}}\left\{\left[\frac{1}{2}-S\left(\frac{a}{2}\sqrt{\frac{2}{\pi}}\right)\right]\cos\left(\frac{a^2}{4}\right)-\left[\frac{1}{2}-C\left(\frac{a}{2}\sqrt{\frac{2}{\pi}}\right)\right]\sin\left(\frac{a^2}{4}\right)\right\} \quad (\Re a \geq 0)$$

and

$$\int_0^{\infty} e^{-at}\sin(t^2)dt=\sqrt{\frac{\pi}{2}}\left\{\left[\frac{1}{2}-C\left(\frac{a}{2}\sqrt{\frac{2}{\pi}}\right)\right]\cos\left(\frac{a^2}{4}\right)+\left[\frac{1}{2}-S\left(\frac{a}{2}\sqrt{\frac{2}{\pi}}\right)\right]\sin\left(\frac{a^2}{4}\right)\right\} \quad (\Re a \geq 0)$$

where C(.) and S(.) are the Fresnel cosine and sine integrals. Noting that:

$$\int_{-\infty}^{0} e^{-at}\cos(t^2)dt=\int_0^{\infty} e^{at}\cos(t^2)dt$$

and

$$\int_{-\infty}^{0} e^{-at}\sin(t^2)dt=\int_0^{\infty} e^{at}\sin(t^2)dt$$

The result is:

$$h(t)=\sqrt{\frac{g}{2x\pi}}\left\{\cos\left(\frac{t^2 g}{4x}\right)\left[\frac{1}{2}+C\left(t\sqrt{\frac{g}{2x\pi}}\right)\right]+\sin\left(\frac{t^2 g}{4x}\right)\left[\frac{1}{2}+S\left(t\sqrt{\frac{g}{2x\pi}}\right)\right]\right\} \quad (3.10)$$

This function is plotted in Fig. 3.2a. It is interesting that the foregoing function agrees with an impulse response function derived in hydrodynamical theory. This connection is described in Appendix A.

In practice h(t) of Eq. 3.10 will be truncated. This can be achieved either by a sharp cutoff, or the application of a window. If a sharp cutoff is used in the time domain, it results in a ripple effect in the frequency domain (Gibbs phenomenon). To illustrate this point, the magnitude of the H(ω) function has been obtained as a continuous Fourier transform for two different truncation lengths, 200 sec and 50 sec respectively (Figs. 3.3, 3.4). The phase property is affected mostly at low frequencies.

Alternatively, windowing can be performed in the frequency response function. If Eq. 3.8 is assumed zero for $|\omega| > \omega_c$, where ω_c is the cutoff frequency, Eq. 3.10 becomes (Dommermuth):

$$h(t) = \sqrt{\frac{g}{2x\pi}}\left\{\cos\left(\frac{t^2 g}{4x}\right)\left[C\left(\sqrt{\frac{2x}{\pi g}}\omega_c - \sqrt{\frac{g}{2x\pi}}t\right) + C\left(t\sqrt{\frac{g}{2x\pi}}\right)\right]\right. \tag{3.11}$$
$$\left. + \sin\left(\frac{t^2 g}{4x}\right)\left[S\left(\sqrt{\frac{2x}{\pi g}}\omega_c - \sqrt{\frac{g}{2x\pi}}t\right) + S\left(t\sqrt{\frac{g}{2x\pi}}\right)\right]\right\}$$

This function is shown in Fig. 3.2b

3.4.2 The generation of velocity and acceleration time histories.

The generation of velocity and acceleration time histories is necessary for the calculation of the wave forces on the members of an offshore structure, according to Morrison's formula. Initially forces can be generated at the grid points where elevation has been transferred and subsequently integrated to

yield the total member wave forces. Both components of particle velocity, horizontal u_x and vertical u_z, and particle acceleration $\dot{u}_x$ and $\dot{u}_z$ should be generated, since members can have varied orientations.

These time histories must be produced by differentiating the elevation time histories with respect to time, for the vertical particle velocity, and then the velocity time histories, to generate acceleration; horizontal particle velocity is similar to vertical, only shifted by 90^0 in the frequency domain. This technique must be applied with care because differentiation is known to amplify errors (noise) particularly in the high frequency region (Dommermuth). In this application where measured data is not involved, the noise is either due to the difference between the spectrum from the digital filter and the target spectrum (this depends on the quality of the digital filter design) , or it is due to roundoff errors in the computer (a remote possibility). If noise amplification is suspected, the remedy is to generate the elevation time-series with a shorter timestep (higher bandwidth), then apply differentiation and finally apply a low-pass filter in order to retain just the desired bandwidth. However in the present study noise amplification has not been found to be a problem, therefore no special measures are adopted in this respect. On the other hand a high performance filter has been used for the differentiation itself, in order to avoid the distortion inherent in the 2-point finite difference approximation.

A. Vertical water particle velocity.

Using Airy linear wave theory for deep water, the frequency response function for producing the vertical particle velocity from surface elevation is the Fourier transform of the differentiation operation (Dommermuth), (Isobe, Kondo and Horikawa 1984):

$$H_{u_2}(\omega) = i\,\omega \tag{3.12}$$

To design a digital filter from this function, its Fourier series representation is evaluated (Chen 1979):

$$h(nT) = \frac{T}{2\pi}\int_{-\pi T}^{\pi T} i\,\omega e^{i n \omega T} d\omega \tag{3.13}$$

where T is the digitization interval. Integration yields the following h(n) :

$$h(n) = 0 \qquad \text{for} \qquad n = 0 \tag{3.14}$$

$$h(n) = \frac{(-1)^n}{nT} \qquad \text{for} \qquad n \neq 0$$

The above sequence is antisymmetric about zero and tends asymptotically to zero as n tends to infinity, in the positive direction as well as the negative direction. The existence of ordinates for negative n indicates that, unless shifted, it is a non-causal filter. In practical terms this means that when used in a continuous convolution with an input series, some future time steps of the input must be produced and stored in advance. Shifting the filter about its central ordinate in order to render it causal is undesirable because it would distort the phase relation between input and output, which in the present application is important.

Since h(n) is of infinite length and decaying, it can be truncated when its value becomes smaller than a fraction of h(1). As usual, this results in ripples in the frequency response behaviour. To overcome this distortion problem the truncated sequence must be multiplied by a time domain window. The Hamming window was used for this particular function (Chen):

$$w(n) = 0.54 + 0.46\cos\left(\frac{\pi n}{\overline{N}}\right) \qquad \text{where} \qquad \overline{N} = \frac{N-1}{2} \tag{3.15}$$

N is the total length of the truncated h(n) sequence. The frequency response characteristics of the windowed filter for the generation of vertical velocity records from elevation inputs is shown in Fig 3.5 (for bandwidth π) and Fig. 3.6 (for bandwidth 2π).

It is worth noting here that the numerical differentiation could have been done with finite difference formulae from numerical analysis. The simplest such scheme is a convolution with the 2-point digital filter:

$$H(z) = \frac{1}{T}(1 - z^{-1}) \tag{3.16}$$

The frequency response function that corresponds to this filter is (Chen 1979):

$$H(\omega) = i\frac{e^{-i\omega T/2}}{T/2}\sin\left(\frac{\omega T}{2}\right) \tag{3.17}$$

which agrees with the correct function Eq.(3.9) only for low frequencies and thus results in large errors for higher frequency components of the input sequence.

B. Horizontal water particle velocity.

A separate digital filter is designed for the horizontal velocity, since its corresponding frequency response function differs from the vertical case in the phase term (Dommermuth), (Isobe, Kondo and Horikawa 1984):

$$H_{u_x}(\omega) = |\,\omega\,| \tag{3.18}$$

Again using the Fourier series:

$$h(nT)=\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T}|\omega|e^{in\omega T}d\omega=\frac{T}{2\pi}\left\{\int_{0}^{\pi/T}\omega e^{in\omega T}d\omega-\int_{-\pi/T}^{0}\omega e^{in\omega T}d\omega\right\} \tag{3.19}$$

one obtains the infinite sequence:

$$\begin{aligned} h(n)&=\frac{\pi}{2T} && \text{for} && n=0 \\ h(n)&=0 && \text{for} && n \text{ even} \\ h(n)&=\frac{-2}{\pi n^2 T} && \text{for} && n \text{ odd} \end{aligned} \tag{3.20}$$

This is a symmetric sequence of infinite length but decays rapidly due to the quadratic term in the denominator of each term. As in the case of the preceding differentiator, the filter is non-causal due to the symmetry about zero and consequently the same storage technique will be employed when this h(n) is used in the convolutions.

Normally a time domain window would be applied at this point in order to remove the ripples in the frequency domain caused by the truncation; but in this particular case the application of a window has been found to distort the natural kink of the H(ω) function at zero frequency. Only when the length of h(n) is large enough, which corresponds to a finer frequency resolution, this distortion will be insignificant, but this is not practical. Fortunately, the function under consideration has a rapid decay even without the use of a window, and the ripples in the frequency response function are of minor order (Figs 3.7 and 3.8). Therefore the h(n) sequence as given in Eq. 3.20 was used for this operation.

C. Acceleration records.

The horizontal water particle acceleration transfer function from elevation is $H_{\ddot{u}_x}(\omega)=-i\,\omega^2$ (for deep water) and the transfer function for the vertical

component of acceleration is $H_{\dot{u}_x}(\omega) = -\omega^2$ (Isobe, Kondo an Horikawa). Alternatively, if the input for the generation of the horizontal water particle acceleration is taken as the horizontal water particle velocity u_x, and the input for the vertical water particle acceleration is u_z, the transfer function is the same for both cases, namely the differentiator (Eq.3.12). Therefore it is more convenient in the simulation of the wave kinematics to generate the acceleration records from the corresponding velocity components, which are generated already, using the same transfer filter for both.

3.5 Vertical attenuation of wave motion.

The five components of wave motion that are initially generated at the surface grid points, namely elevation η, particle velocity, horizontal u_x and vertical u_z, and particle acceleration $\dot{u}_x$ and $\dot{u}_z$, should be transferred to the grid points in various depths below the free surface to calculate the wave forces on submerged structural members. The elevation record itself is needed at small distances below MWL where a member can be at times partly submerged, depending on the wave height.

For deep water, according to Airy linear wave theory, the transfer function for the vertical attenuation for all the wave motion components is (Dommermuth), (Isobe, Kondo and Horikawa):

$$H_{ver}(\omega) = e^{-\omega^2 dz/g} \tag{3.21}$$

where dz is the distance below the surface to which the wave motion is transferred. It can be seen that the attenuation of higher frequency components is more pronounced than that of low frequencies.

A digital filter can be designed by inverse Fourier transforming and digitising this function (Dommermuth):

$$h(n)=\frac{T}{2}\sqrt{\frac{g}{\pi dz}}\exp\left(\frac{-gn^2T^2}{4dz}\right) \tag{3.22}$$

where T is the digitization interval. This is a symmetric filter about zero and the discussion in an earlier Section for non-causal filters applies here as well. It should also be mentioned that this sequence decays rapidly (exponentially) and the filters it results in are in general of low order.

3.6 Multidirectional case.

The unidirectional scheme described above can be easily extended to the multidirectional case. Wave elevation series are generated at the "reference" point and then transmitted to various points for each direction considered, independently. The summation of the contributions from different directions is occurring at the points. The wave trains generated at the "reference" point for each direction are uncorrelated, but shaped according to known directional spectra.

The directional power spectrum adopted in the present work is given by the relation (Kree and Soize, 1983):

$$G_{\eta\eta}(\theta,\omega)=D(\theta,\theta_0)S_{\eta\eta}(\omega) \qquad \text{where} \qquad D(\theta,\theta_0)=A\cos^n(\theta-\theta_0) \tag{3.23}$$

and A is a normalizing constant such that $\int_\theta D(\theta,\theta_0)d\theta = 1$, θ_0 is the principal direction, and n=2.

An example grid used for the multidirectional generation of wave kinematics is shown in Fig. 7.2 of chapter 7.

3.6.1 Rounding errors.

The scheme presented in this chapter requires a set of pre-designed digital filters for each and every filtering operation involved. For the horizontal transmission of wave kinematics a series of recursive digital filters has been designed and is presented in following chapters. These filters cover propagation distances ranging from 1m to 50m in intervals of 1m. For intermediate distances, it is proposed to use the filter for the nearest integer distance. This rounding operation by a maximum of 0.5m has an insignificant effect on the cross-statistics of the wave kinematics between two points. Thus, the covariance of the elevation processes between two points differing by x (unidirectional case) is given, after some manipulation, by the integral:

$$c_{\eta_1\eta_2} = \int_0^{\infty} \cos\left(\frac{x\omega^2}{g}\right) S_{\eta\eta}(\omega) d\omega \tag{3.24}$$

where $S_{\eta\eta}(\omega)$ is the Pierson-Moskowitz spectrum for the surface elevation process. Similar relations can be written for the velocity and acceleration covariances between two points. In Fig. 3.9 the calculated covariances for distances x *(m)* and $x+0.5$ *(m)* are shown superimposed at abscissa x (continuous lines and markers respectively) in order to display the magnitude of the maximum error due to rounding. Two sets of covariance curves are presented for two extreme sea states, 20 Knots and 60 Knots windspeed. It is apparent that the error resulting from this rounding is in most cases negligible. The error is slightly more pronounced in the case of the acceleration process, for very small distances of propagation. In the multidirectional case, the error is expected to be even smaller, because rounding up and rounding down operations would tend to produce cancelling errors.

It should be noted that in contrary to the horizontal transmission filters, the depth attenuation filters can be designed at run time, and for the exact depth, therefore there will be no rounding error associated with this operation.

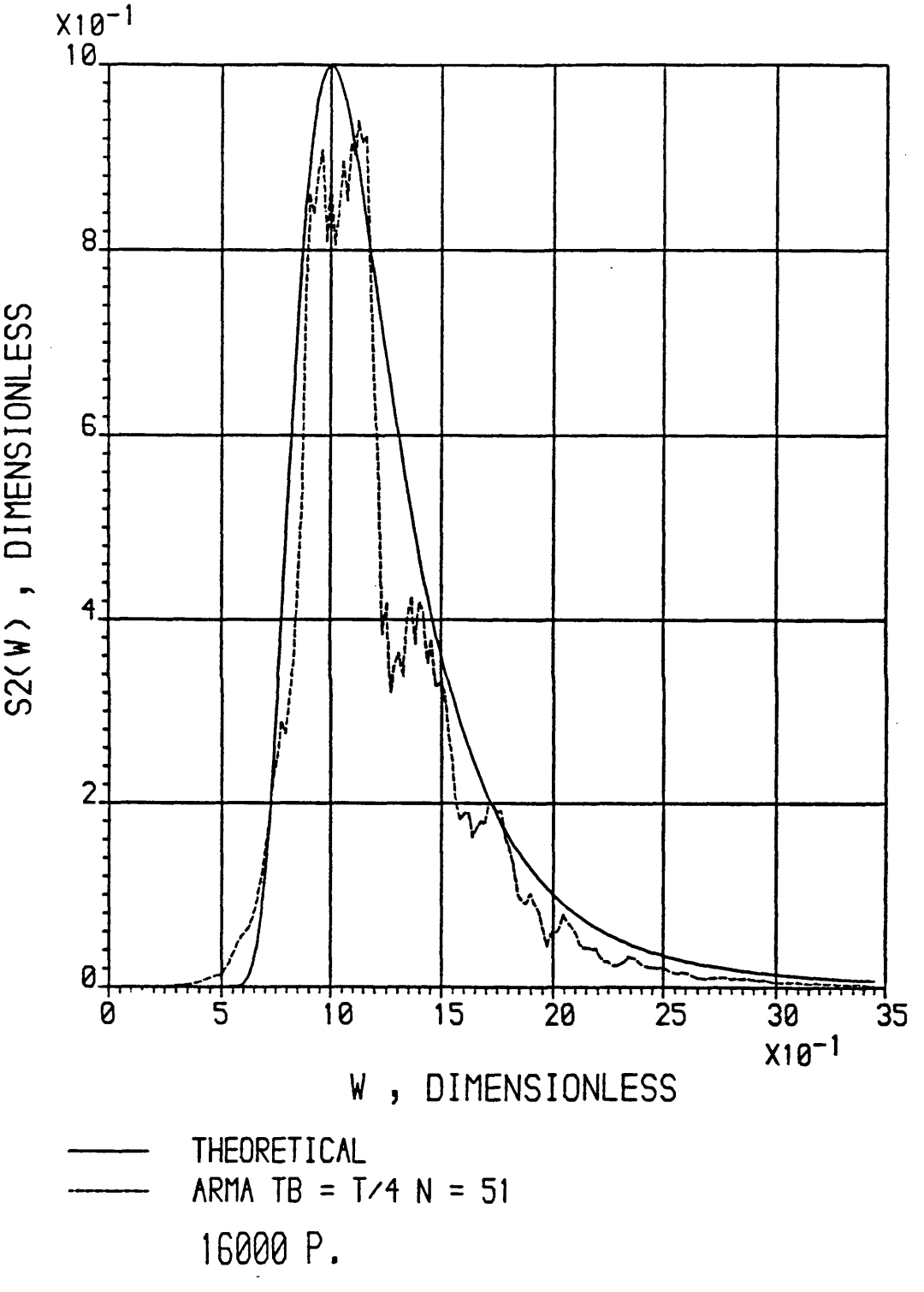


Fig. 3.1 Power spectrum of time series generated from an ARMA model compared with P.M. spectrum.

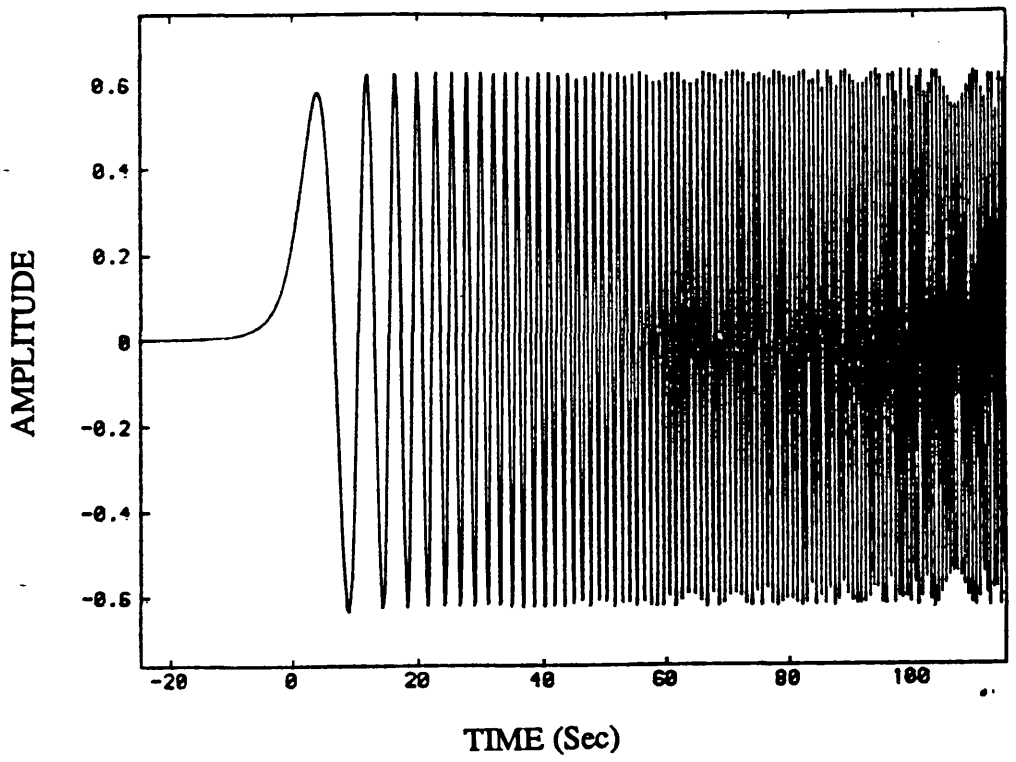


Fig.3.2.a h(t) function obtained from Fourier transform operation (Eq.3.10)

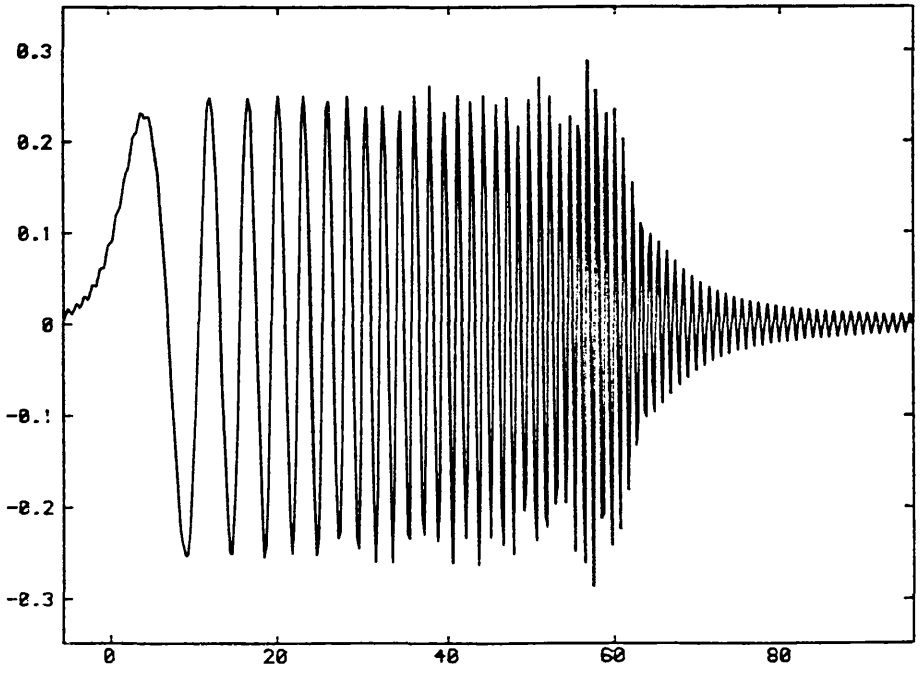


Fig.3.2.b h(t) function from transfer function truncated at $\omega_c = 2\pi$ (Eq.3.11)

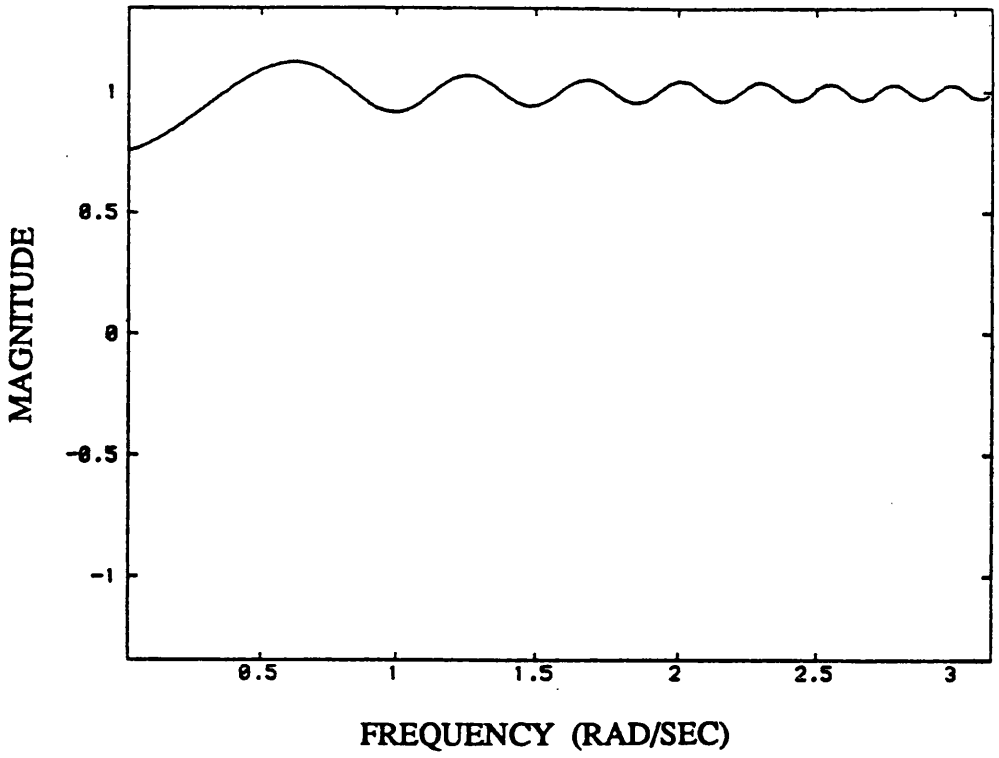


Fig. 3.3 Magnitude of transfer function for the impulse response in Fig. 3.2, truncated at 200 s.

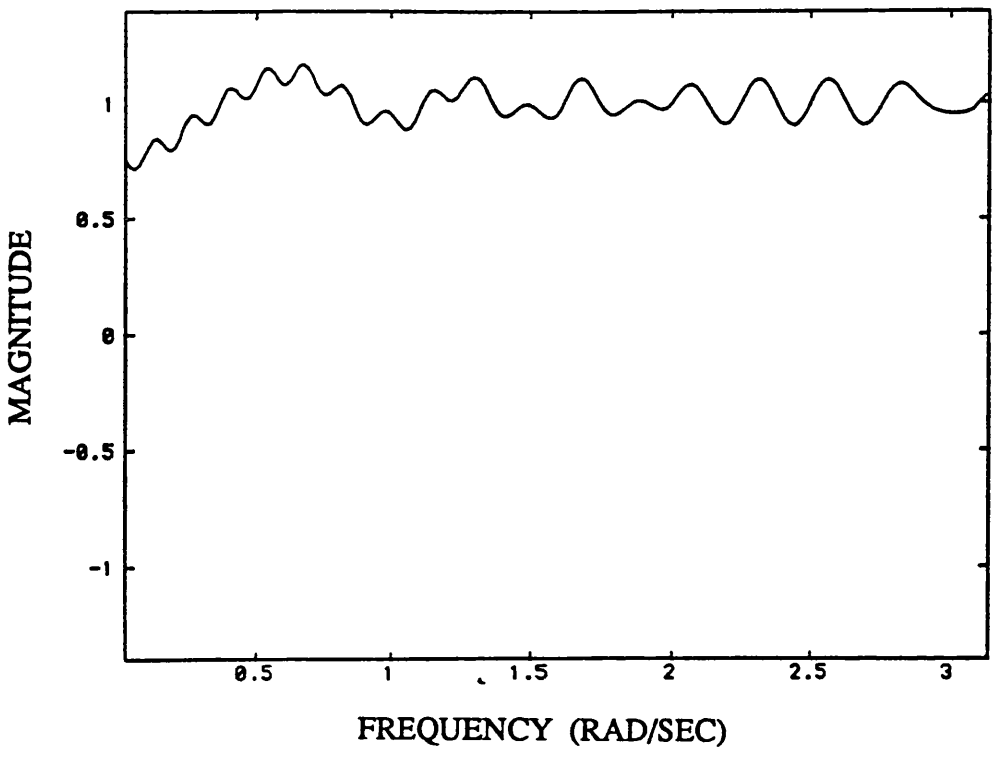


Fig. 3.4 As in Figure 3.3, but truncated at 50 s.

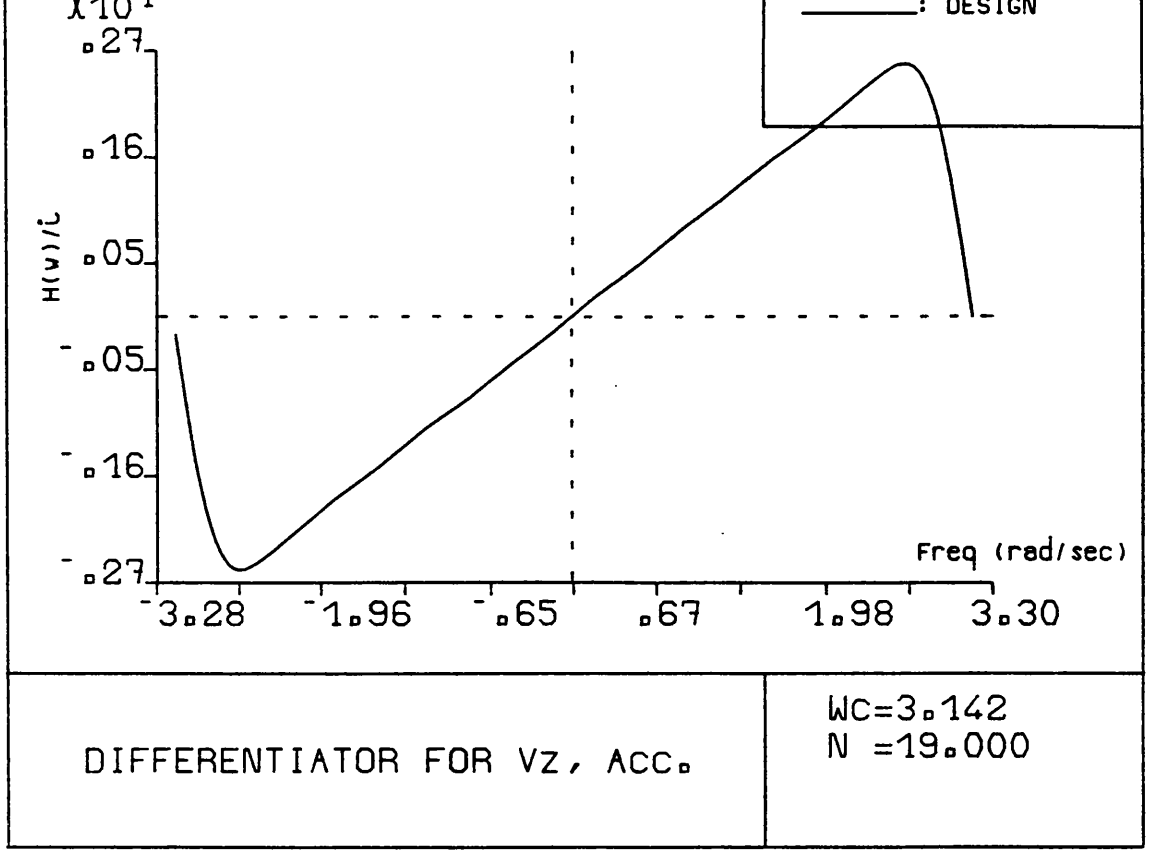


Fig. 3.5a

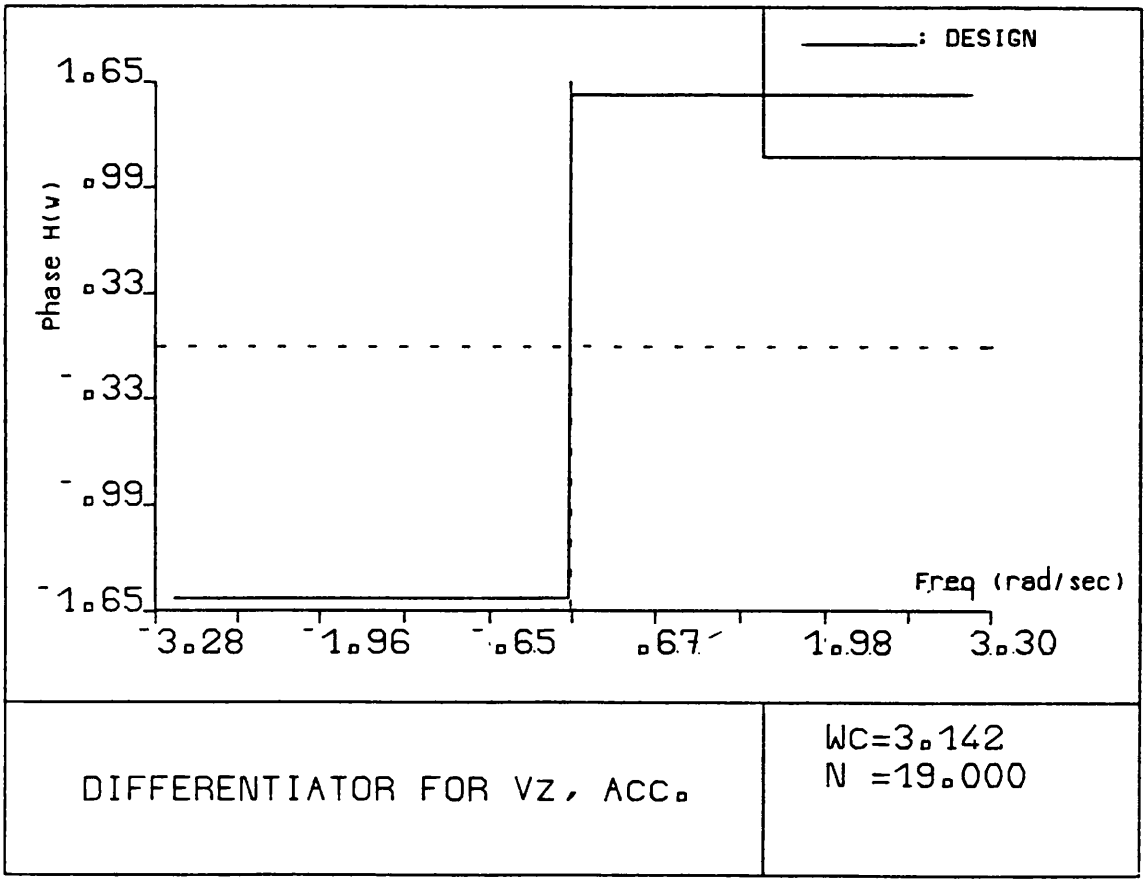


Fig. 3.5b

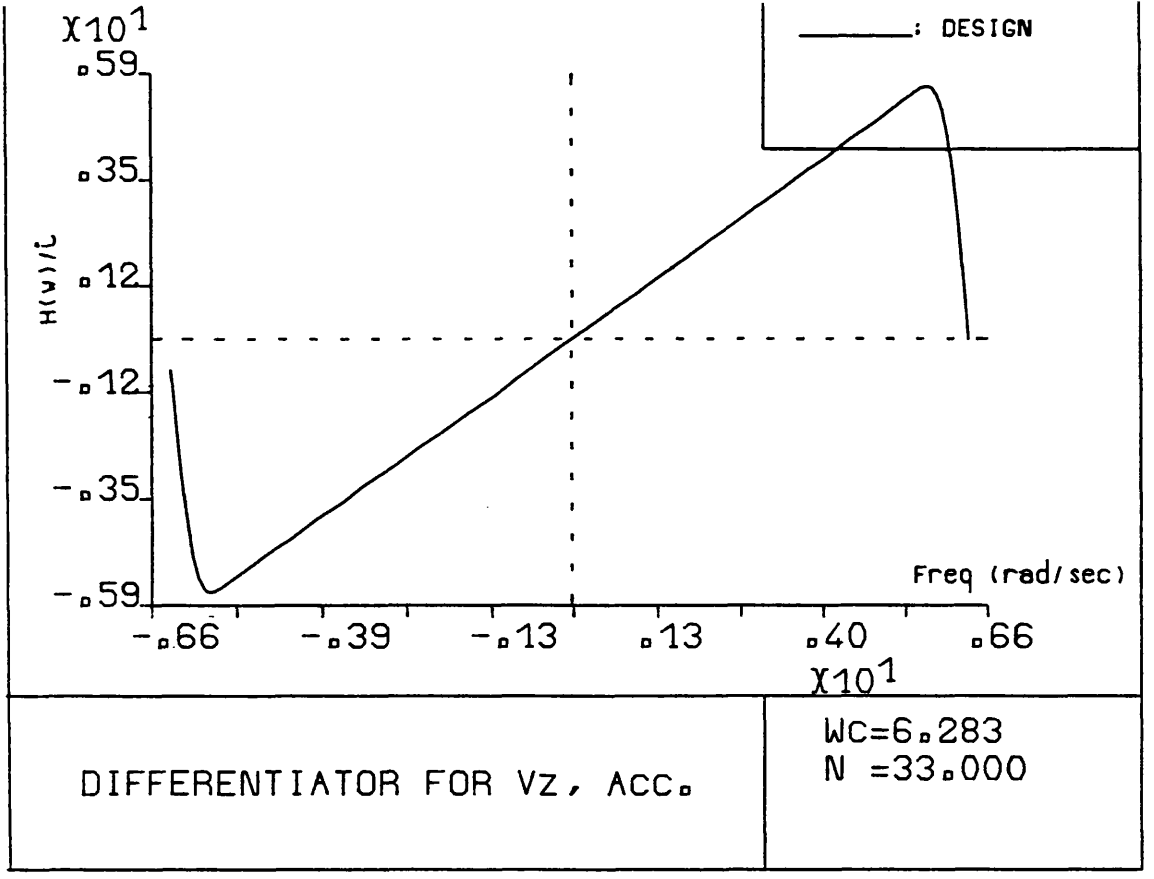


Fig. 3.6a

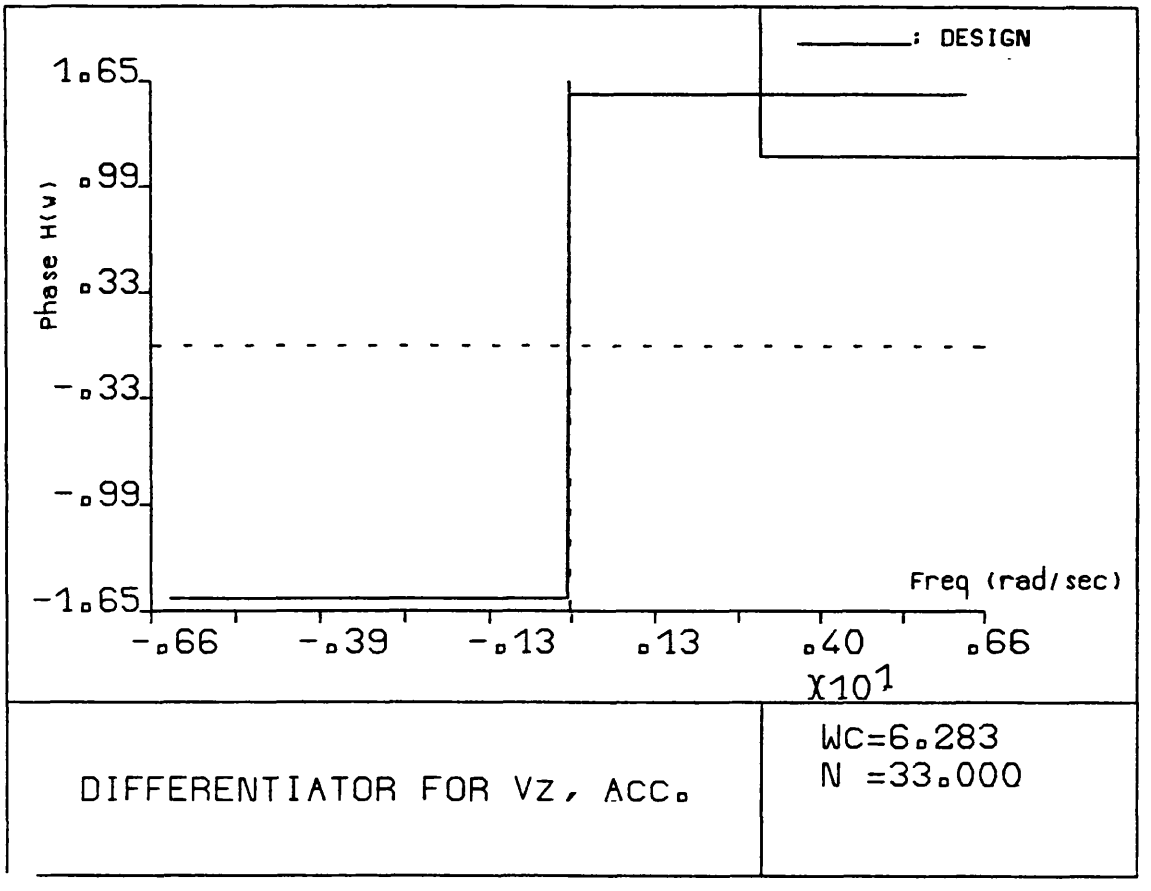


Fig. 3.6b

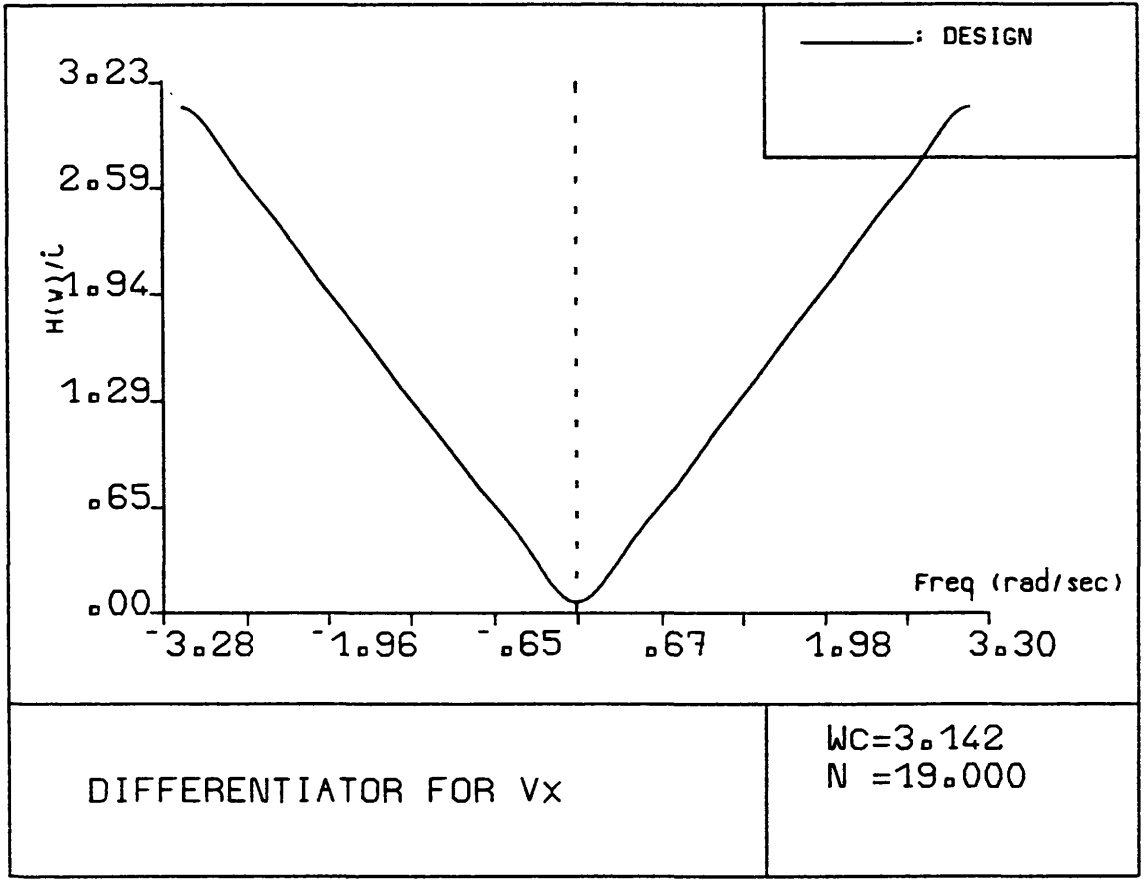


Fig. 3.7a

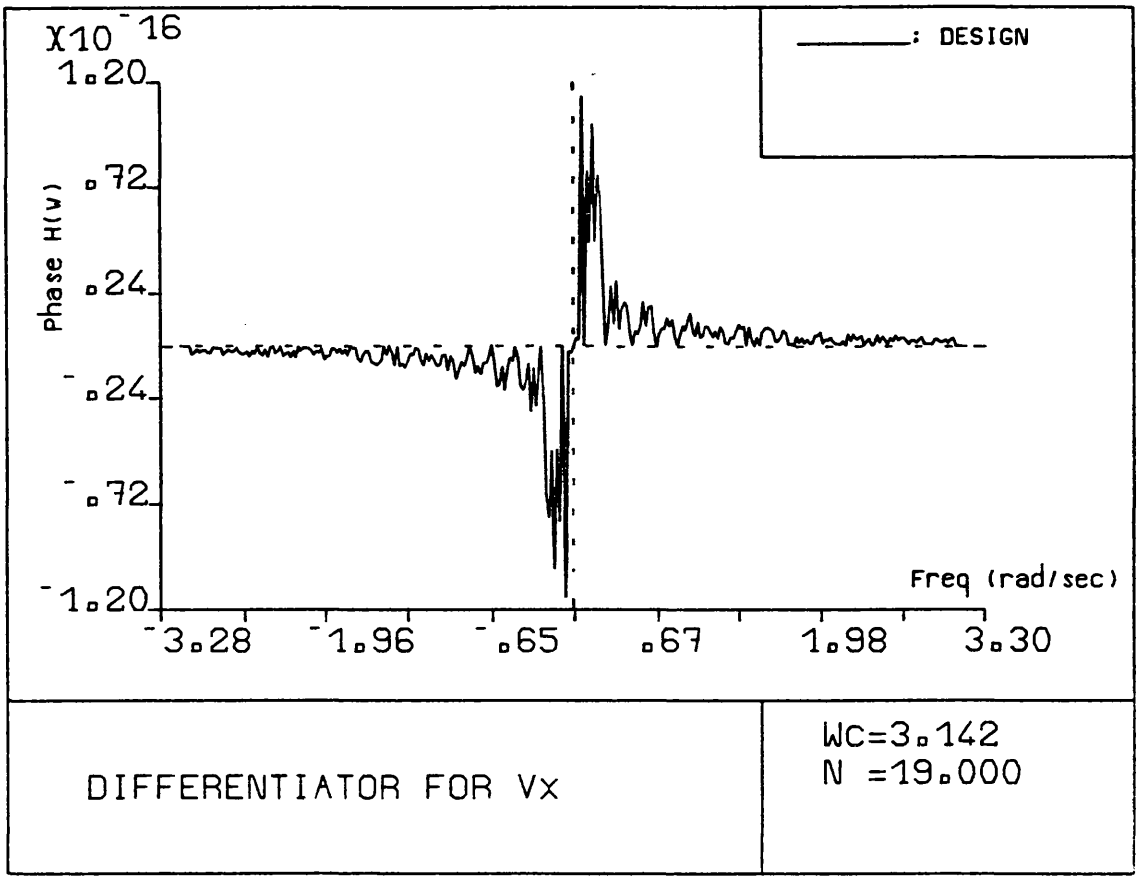


Fig. 3.7b

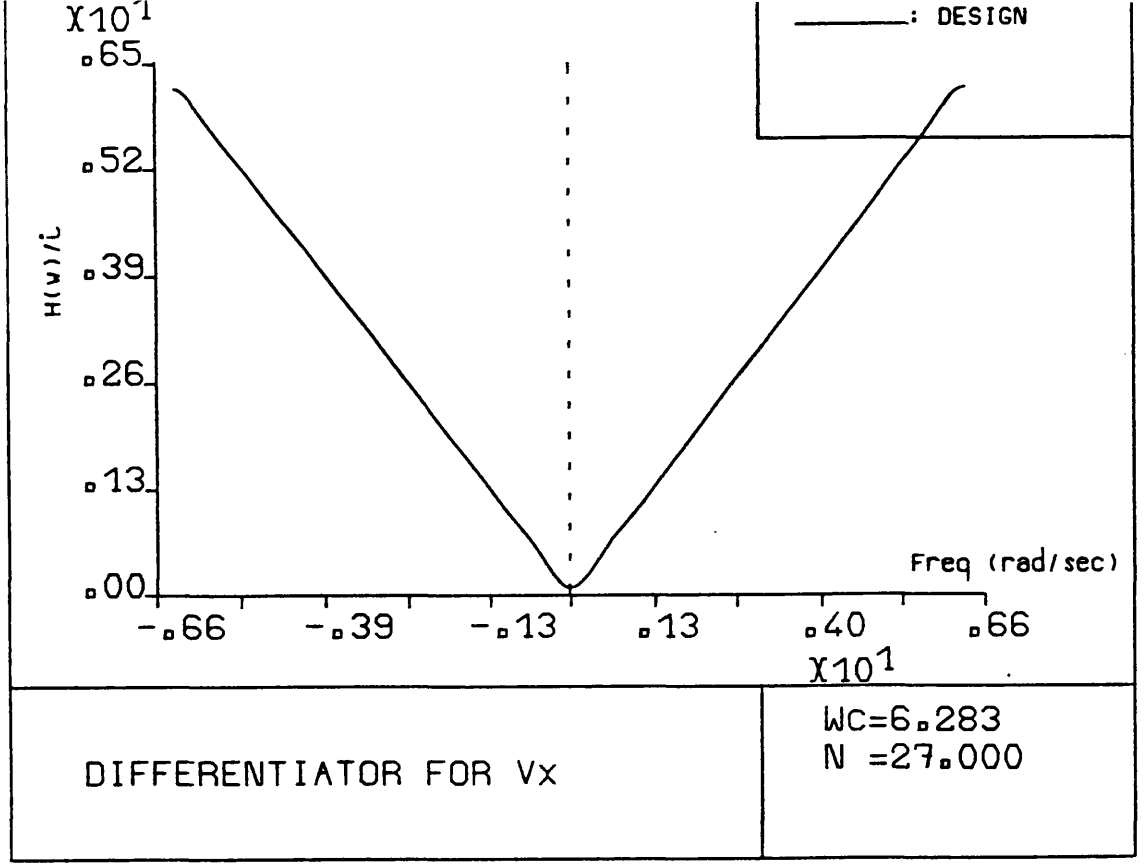


Fig. 3.8a

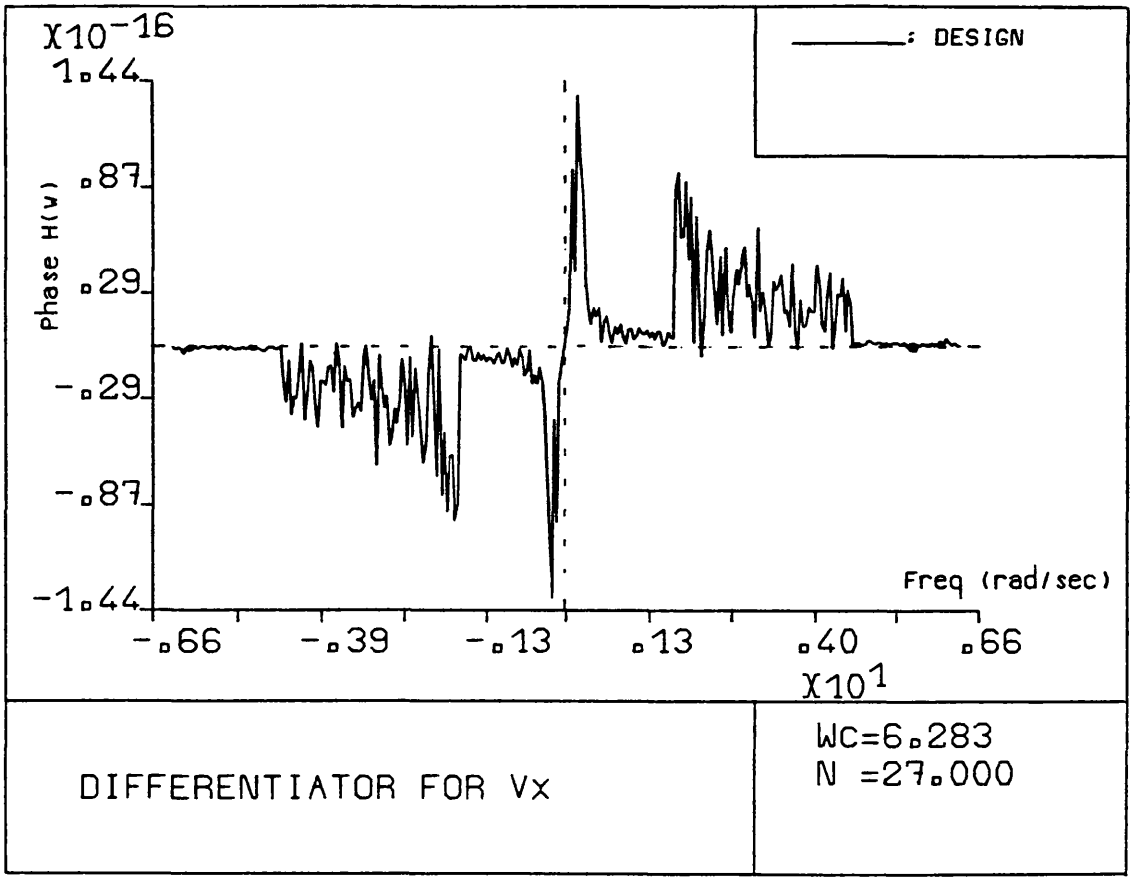


Fig. 3.8b

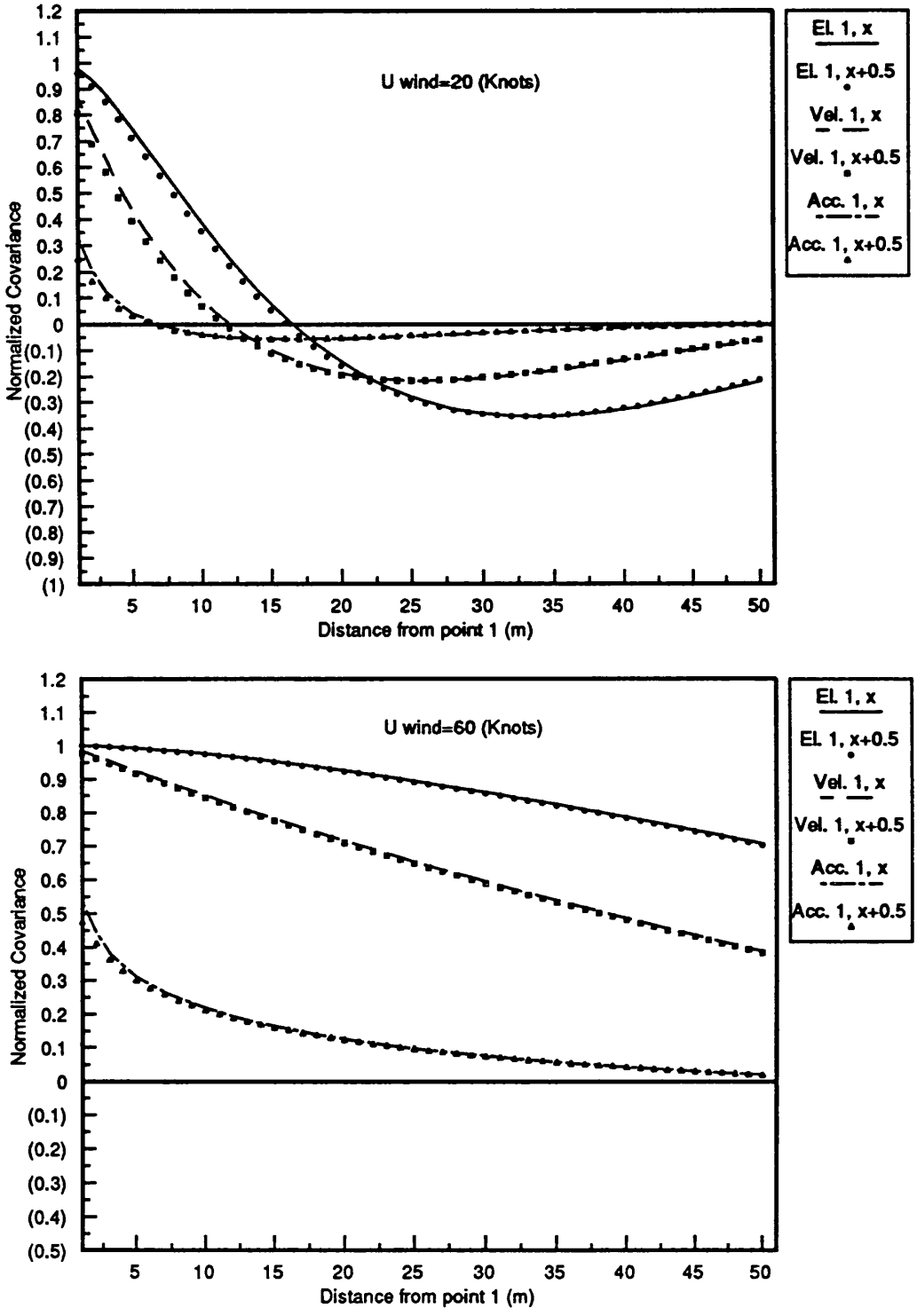


Fig. 3.9 Covariance Functions

CHAPTER 4

HORIZONTAL PROPAGATION FILTER:

TIME-DOMAIN DESIGN.

CHAPTER FOUR

CONTENTS

4.1 Introduction.

The design of a fixed order recursive digital filter to replace an infinite duration impulse response function can be carried out either in the frequency or in the time domain. In the first case the aim is to match the characteristics (magnitude and phase) of the corresponding frequency response function $H(\omega)$ of the target filter with those of a recursive filter over a densely spaced frequency range (Rabiner and Gold), (Steiglitz 1970), (Rabiner, Graham and Helms 1974), (Steiglitz 1977). In the second case, which is described in this chapter, the objective is to synthesize a recursive filter having an impulse response that approximates a finite portion of the given infinite length impulse response function (Burrus and Parks 1970), (Brophy and Salazar 1974), (Bertran 1975), (Cadzow 1976), (Mullis and Roberts 1976), (Hastings-James and Mehra 1977), (Yahagi 1981).

4.2 Description of the design method.

The z-transform representation of a recursive digital filter of order (N,M) is given by the relation:

$$G(z)=\frac{a_0+a_1z^{-1}+a_2z^{-2}+a_3z^{-3}+\ldots+a_{N-1}z^{-(N-1)}}{1+b_0z^{-1}+b_1z^{-1}+b_2z^{-2}+\ldots+b_{M-1}z^{-(M-1)}} \qquad (4.1)$$

The impulse response of this filter, g(k), is given by the MacLaurin series expansion of G(z) as:

$$G(z)=\sum_{i=0}^{\infty} g(i)z^{-i} \qquad (4.2)$$

g(k) can be obtained by a recursive calculation as:

$$g(k)=\sum_{i=0}^{M-1} a_i\delta_{k-i}-\sum_{i=1}^{N-1} b_i g(k-i) \qquad k=0,1,2,\ldots \tag{4.3}$$

where δ_i is a unit impulse sequence.

The time domain synthesis problem is to estimate the N+M-1 coefficients a_i and b_i such that the g(k) sequence can approximate a target h(k) over a range of k=0,1,2,...K-1.

4.2.1 Exact solution.

Following Burrus and Parks (1970) , if the order of the recursive filter is chosen so as to be M+N-1=K then there exists at least one exact solution for a_i and b_i. The function G(z) in this case is the Pade approximation of H(ω) of order (N,M) and g(k)=h(k) for k=0,1,2,...K-1. Correspondingly G(z)=H(z), where H(z) is the z-Transform of h(k), and the relation between the filter coefficients and the target impulse response function can be written as:

$$\sum_{n=0}^{M-1} b_n h_{i-n} = \begin{cases} a_i & i=0,1,\ldots N-1 \\ 0 & i \geq N \end{cases} \tag{4.4}$$

The above equation can be written in matrix form :

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ . \\ . \\ a_{N-1} \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix} = \begin{bmatrix} h_0 & 0 & 0 & . & . & . & 0 \\ h_1 & h_0 & 0 & & & & . \\ h_2 & h_1 & h_0 & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ h_{N-1} & & & & & & \\ h_N & & & & & & \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ h_{K-1} & h_{K-2} & h_{K-3} & . & . & . & h_{K-M} \end{bmatrix} \begin{bmatrix} 1 \\ b_1 \\ b_2 \\ . \\ . \\ . \\ b_{M-1} \end{bmatrix} \qquad (4.5)$$

or in simpler form, by partitioning:

$$\begin{bmatrix} \mathbf{a} \\ - \\ 0 \end{bmatrix} = \begin{bmatrix} H_1 \\ - \\ H_2 \end{bmatrix} [\mathbf{b}] \qquad (4.6)$$

If $\mathbf{h}^1$ is the first column of $[H_2]$ and $[H_3]$ the remaining M-1 by M-1 square matrix, then, if $[H_3]$ is non singular, there exists a unique solution for the b_i coefficients:

$$\mathbf{b}^* = -[H_3]^{-1}\mathbf{h}^1 \text{ and } b_0 = 1 \qquad (4.7)$$

where $\mathbf{b}^*$ is the **b** matrix without the element b_0. Once the b_i coefficients are determined, a_i can be obtained by direct multiplication:

$$\mathbf{a} = H_1\mathbf{b} \qquad (4.8)$$

4.2.2 Approximate methods.

As mentioned previously an exact solution exists only when M+N-1=K in which case:

$$g(k) = h(k) \quad \text{for} \quad k = 0,1,2,\ldots K-1$$

In cases where, in order to maintain adequate information in the truncated impulse response function, K has to be large, this clearly requires too high an order M and N. In designing lower order recursive filters the aim is to minimize some error measure, usually in the sum of the squares of the differences sense. In the "Output Error" formulation the error chosen is the difference between h(k) and g(k), **e**=**g**-**h**. Then the objective function is:

$$J(\mathbf{a},\mathbf{b}) = \mathbf{e}^T\mathbf{e} = \sum_{k=0}^{K-1}\{g(k)-h(k)\}^2 \qquad (4.9)$$

where the impulse response of the recursive filter, g(k), is given by Eq.4.3. It can be seen that minimizing the functional J(**a**,**b**) is a highly nonlinear optimisation problem. Appropriate methods will be described later.

Alternatively, it is possible to minimise the "Equation Error" : this compares the output of the convolution of the h(k) sequence with the **b** coefficients on the one hand, with the output of the convolution of a unit pulse with the **a** coefficients on the other hand. This is called the equation error because it represents the mismatch between the two sides of Eq.4.4. Application of this method will also be discussed below.

It has been observed (Cadzow 1976) that a relation exists between the time and the frequency domain minimisations. In the frequency problem the function being minimized is:

$$J_f(\mathbf{a},\mathbf{b}) = \frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} |\, G(e^{i\omega T}) - H(e^{i\omega T})\,|^2\, d\omega \qquad (4.10)$$

where H and G are the frequency responses of the target and the recursive systems respectively. Using Parseval's theorem one gets:

$$J_f(\mathbf{a},\mathbf{b}) = \frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} |\, G(e^{i\omega T}) - H(e^{i\omega T})\,|^2\, d\omega = \tag{4.11}$$

$$\sum_{k=0}^{K-1}\{g(k)-h(k)\}^2 + \sum_{k=K}^{\infty} g^2(k)$$

therefore the two functionals J and J_f differ by the second summation on the right hand side of this equation. When K is large and the recursive filter is stable, this term is negligible and the two functionals are equivalent. Cadzow has pointed out that in most cases the time domain formulation is more amenable to analysis.

4.2.2.1 Minimisation of the equation error.

The equations 4.5, 4.6, previously used for the special case M+N-1=K, can be extended to the more general case by the introduction of error terms e_i (Burrus and Parks 1970):

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \cdot \\ \cdot \\ a_{N-1} \\ - \\ e_N^1 \\ \cdot \\ \cdot \\ \cdot \\ e_{K-1}^1 \end{bmatrix} = \begin{bmatrix} h_0 & 0 & 0 & . & . & . & 0 \\ h_1 & h_0 & 0 & & & & . \\ h_2 & h_1 & h_0 & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ h_{N-1} & & & & & & \\ - & - & - & - & - & - & - \\ h_N & & & & & & \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ h_{K-1} & h_{K-2} & h_{K-3} & . & . & . & h_{K-M} \end{bmatrix} \begin{bmatrix} 1 \\ b_1 \\ b_2 \\ . \\ . \\ . \\ b_{M-1} \end{bmatrix} \tag{4.12}$$

or, in matrix notation:

$$\begin{bmatrix} \mathbf{a} \\ - \\ \mathbf{e}^1 \end{bmatrix} = \begin{bmatrix} H_1 \\ - \\ H_2 \end{bmatrix} [\mathbf{b}] \tag{4.13}$$

The vector to be minimized in this case is $\mathbf{e}^1$ instead of **e**, and compares the output of the convolution of the h(k) sequence with the **b** coefficients on the one hand, with the output of the convolution of a unit pulse with the **a** coefficients on the other hand (the latter convolution gives the **a** sequence itself). From Eq.4.13, using the notation of Eq.4.7, this error vector can be expressed as:

$$\mathbf{e}^1 = H_3\mathbf{b}^* + \mathbf{h}^1 \qquad (4.14)$$

To minimize $\mathbf{e}^1$ two different methods can be employed:

A. Linear Programming.

The objective (cost) function in this case is the largest absolute value residual in the lower partition of Eq.4.13:

$$J = \max_i |e_i^1| \qquad (4.15)$$

and the linear programming task is to minimize J with respect to b_i. In practice this minimax problem is converted to an LP by the introduction of a new unknown. The absolute values in the objective function are expanded into one pair of ordinary constraints each. Once the unknowns b_i have been determined, a_i are found from the upper partition of Eq.4.13.

B. Generalized inverse methods.

$[H_3]$ is a K-N by M-1 matrix and since K-N>M-1 the set of equations $[H_3]\mathbf{b}^* = -\mathbf{h}^1$ cannot have an exact solution. The residual vector is $\mathbf{e}^1$ and, if the rank of $[H_3]$ is M-1, the solution that minimizes the sum of the squares $J = \mathbf{e}^{1T}\mathbf{e}^1$ is given by:

$$\mathbf{b}^* = -[H_3^T H_3]^{-1} H_3^T \mathbf{h}^1 \qquad (4.16)$$

The matrix $H_3^* = [H_3^T H_3]^{-1} H_3^T$ is the generalized inverse of $[H_3]$ and it exists since $[H_3^T H_3]$ is a square M-1 by M-1 nonsingular matrix. However, in certain cases problems might arise if this matrix is ill-conditioned; treatment of such cases will be discussed below. Having calculated **b** from Eq.4.16, the coefficients **a** can be obtained by multiplication with the upper partition of matrix [H] Eq.4.12.

If the square matrix $[K]=[H_3^T H_3]$ is ill-conditioned and therefore cannot be inverted, the usual way to improve it is by scaling, that is, by multiplying the rows and columns by nonzero constants. In this particular application, [K] will be replaced by [S][K][S], where [S] is a diagonal matrix with elements $S_{ii} = 1/\sqrt{K_{ii}}$. Introducing a new vector of unknowns $\mathbf{b}^{*\prime}$, such that $\mathbf{b}^* = [S]\mathbf{b}^{*\prime}$, the solution is now given by:

$$\mathbf{b}^{*\prime} = -[S H_3^T H_3 S]^{-1} S H_3^T \mathbf{h}^1 \qquad (4.17)$$

It must be noted though, that scaling does not improve the matrix drastically in cases of severe ill-conditioning. An alternative method, which results in a better overall performance of the designed filter, as well as improving the condition number, is the introduction into Eq.4.16 of a free parameter λ (Hastings-James and Mehra 1977). The solution is now:

$$\mathbf{b}^* = -[H_3^T H_3 + \lambda I]^{-1} H_3^T \mathbf{h}^1 \qquad (4.18)$$

As a further improvement, the objective function $J=\mathbf{e}^{1T}\mathbf{e}^1$ can be redefined as a weighted sum of the squares of the residuals (Hasting-James and Mehra), $J'=[R]\mathbf{e}^{1T}\mathbf{e}^1$, where [R] is a diagonal matrix of weights:

$$[R] = diag\,[[R_0], [R_1], [R_2]]$$

Each $[R_i]$ is a diagonal submatrix:

$$R_0 = \text{diag}[r_0, r_1, ..., r_{N-1}] \tag{4.19}$$

$$R_1 = \text{diag}[r_N r_{N+1}, ..., r_{K-1}]$$

$$R_2 = \text{diag}[r_K r_{K+1}, ..., r_{K+M-1}]$$

If $[R_1]$ is chosen as [I] and $[R_2]$ is chosen as α[I] the solution that minimizes J′ is:

$$\mathbf{b}^* = -[H_3^T R_1 H_3 + H_4^T R_2 H_4]^{-1} H_3^T R_1 \mathbf{h}^1 \tag{4.20}$$

where $[H_4]$ is the M-1 by M-1 upper triangular matrix:

$$H_4 = \begin{bmatrix} h_{K-1} & h_{K-2} & \cdot & \cdot & \cdot & h_{K-M} \\ 0 & h_{K-1} & \cdot & \cdot & \cdot & h_{K-M+1} \\ & & \cdot & & & \cdot \\ & & & \cdot & & \cdot \\ & & & & \cdot & \\ 0 & & & & & h_{K-1} \end{bmatrix} \tag{4.21}$$

The minimum value of the objective function J′ given by this solution $\mathbf{b}^*$ is:

$$J'_{\min} = \mathbf{h}^{1T} R_1 \mathbf{h}^1 - \mathbf{b}^{*T} [H_3^T R_1 H_3 + H_4^T R_2 H_4] \mathbf{b}^* \tag{4.22}$$

This approach has been found to lead to more stable filters and numerically well conditioned matrices at the expense of a higher minimised error value.

Finally, it is noted that the equation error $\mathbf{e}^1$ used above pertains to the equations in the lower partition of Eq.4.12 only. The upper partition is assumed to be satisfied without error. However in the output error approach to be explained below, the errors are spread over both partitions. In that case the vector $\mathbf{e}^1$ may be extended to the whole set of rows. Although none of the

formulations reviewed here minimises the norm of this extended $\mathbf{e}^1$, it is nevertheless useful to define such an extended vector in order to relate the equation error approach to the output error approach.

4.2.2.2 Minimisation of the output error.

Another way to tackle the design of low order recursive filters, particularly in cases where the solution yielded by the previous methods is not satisfactory, is the minimisation of the norm of the error vector $\mathbf{e}=\mathbf{g}-\mathbf{h}$, called the output error. In some versions of the method this is quadratic in the unknowns and generalized inverses can be used again (Burrus and Parks 1970); in other versions the objective is fully nonlinear and steepest descent or similar iterative techniques must be used. In the latter case, results found by the previous methods can be used as starting point in the optimization task.

A. Non-iterative method.

In one version of the method generalized inverses are used again to minimise the norm of **e** as defined below:

$$\begin{bmatrix} \mathbf{a} \\ - \\ 0 \end{bmatrix} = B[\mathbf{h}+\mathbf{e}] \qquad (4.23)$$

where [B] is a K by K lower triangular matrix of the coefficients of the denominator of the recursive filter:

$$\begin{bmatrix} b_0 & 0 & 0 & . & . & . & 0 \\ b_1 & b_0 & 0 & & & & \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ b_{M-1} & & & & & & \\ 0 & & & & & & 0 \\ 0 & & & . & . & . & b_0 \end{bmatrix} \tag{4.24}$$

In the method described here **b** is found by minimising the norm of $\mathbf{e}^1$ according to Eq.4.16, and finding **a** follows by minimizing the norm of **e**. One can write from Eq.4.23:

$$\mathbf{h}+\mathbf{e}=B^{-1}\begin{bmatrix}\mathbf{a}\\ 0\end{bmatrix}=B_1^{inv}\mathbf{a}+B_2^{inv}\mathbf{0} \tag{4.25}$$

where $[B_1^{inv}]$ and $[B_2^{inv}]$ are the two partitions, K by N and K by (K-N), of matrix $[B]^{-1}$. Using generalized inverses to solve the normal equations resulting from the above equations, one gets the solution:

$$\mathbf{a}=\left[(B_1^{inv})^T B_1^{inv}\right]^{-1}(B_1^{inv})^T\mathbf{h} \tag{4.26}$$

It should be noted that all the problems mentioned previously concerning conditining of the matrices, stability and overall error measure of the resulting filter are even more likely to arise in the present case, because matrix [B] which has to be inverted is usually of high order. Linear programming could be used as an alternative, but this has not been attempted. The iterative method for minimizing output error, presented in the next section, leads to a more workable scheme.

At this stage a relation between $\mathbf{e}^1$ and **e=h-g** can be exhibited (Burrus and Parks 1970). An extension of the equation error measure $\mathbf{e}^1$ is introduced as described previously, and the matrix equations 4.13 become:

$$\begin{bmatrix} \mathbf{a} \\ - \\ 0 \end{bmatrix} + [\mathbf{e}^1] = \begin{bmatrix} H_1 \\ - \\ H_2 \end{bmatrix} [\mathbf{b}] \qquad (4.27)$$

Here $\mathbf{e}^1$ is a more general, K by 1 vector, in the sense that it allows for errors in the first N terms of the convolution $\mathbf{h}^*\mathbf{b}$ as well as in the terms for i>N. The above relation can be written:

$$\begin{bmatrix} \mathbf{a} \\ - \\ 0 \end{bmatrix} + [\mathbf{e}^1] = \begin{bmatrix} b_0 & 0 & 0 & . & . & . & 0 \\ b_1 & b_0 & 0 & & & & \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ b_{M-1} & & & & & & \\ 0 & & & & & & 0 \\ 0 & & & . & . & . & b_0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ . \\ . \\ . \\ h_{K-1} \end{bmatrix} = B\mathbf{h} \qquad (4.28)$$

from which, comparing with Eq.4.23 the relation between the two error measures is:

$$\mathbf{e}^1 = -B\mathbf{e} \qquad (4.29)$$

B. Iterative minimization of the output error.

Unlike the approach just described, this version minimises the functional:

$$J(\mathbf{a},\mathbf{b}) = \mathbf{e}^T\mathbf{e} = \sum_{k=0}^{K-1} \{g(k) - h(k)\}^2$$

with respect to both **a** and **b**. This is a nonlinear operation and does not possess an analytical solution; the optimum (**a**,**b**) can be sought by numerical optimisation instead. Among the most suitable schemes for such a task are the descent algorithms: If in the n-th iteration the solution vectors are ($\mathbf{a}^n$,$\mathbf{b}^n$), in the (n+1)-th they are updated to:

$$\begin{bmatrix} \mathbf{a}^{n+1} \\ \mathbf{b}^{n+1} \end{bmatrix} = \begin{bmatrix} \mathbf{a}^{n} \\ \mathbf{b}^{n} \end{bmatrix} + \lambda \begin{bmatrix} \Delta\mathbf{a} \\ \Delta\mathbf{b} \end{bmatrix} \tag{4.30}$$

where $\Delta\mathbf{a}$ and $\Delta\mathbf{b}$ are subvectors of the search direction, chosen so as to minimize the functional $J(\mathbf{a},\mathbf{b})$ locally and λ is a step size scalar. At each step the value of J and the impovement of the ratio:

$$\frac{J(\mathbf{a},\mathbf{b})}{\sum_{k=0}^{K-1} h(k)^2} \tag{4.31}$$

is observed. The optimization can be stopped according to one or a combination of various criteria: small improvement of the ratio of Eq.4.31 between successive iterations, negligible slope of the gradient vector or failure to converge after a number of iterations.

The choice of the search vectors is what differs in the various descent algorithms; the assignment of a step size for each new search direction can vary in sophistication from using a small positive constant in the simplest case, to more elaborate methods, where a strategy is employed to vary the step size adaptively. The optimization schemes that were found more appropriate for the present design case are detailed below.

B.1. Steepest descent.

Steepest descent is probably the commonest scheme for choosing search direction vectors. It is based on the idea that a function attains maximum local decrease in the direction of its negative gradient:

$$\begin{bmatrix} \Delta\mathbf{a} \\ \Delta\mathbf{b} \end{bmatrix} = -\begin{bmatrix} \nabla_a J(\mathbf{a}^n, \mathbf{b}^n) \\ \nabla_b J(\mathbf{a}^n, \mathbf{b}^n) \end{bmatrix} \tag{4.32}$$

where the gradient vectors are defined in the following relations:

$$\nabla_a J(\mathbf{a},\mathbf{b}) = \begin{bmatrix} \partial J(\mathbf{a},\mathbf{b})/\partial a_0 \\ \partial J(\mathbf{a},\mathbf{b})/\partial a_1 \\ \vdots \\ \partial J(\mathbf{a},\mathbf{b})/\partial a_{N-1} \end{bmatrix} \qquad (4.33)$$

for partial derivatives of the functional with respect to the unknown coefficients of the numerator of the recursive filter, and

$$\nabla_b J(\mathbf{a},\mathbf{b}) = \begin{bmatrix} \partial J(\mathbf{a},\mathbf{b})/\partial b_1 \\ \partial J(\mathbf{a},\mathbf{b})/\partial b_2 \\ \vdots \\ \partial J(\mathbf{a},\mathbf{b})/\partial b_{M-1} \end{bmatrix} \qquad (4.34)$$

for those of the denominator.

The partial derivatives that appear in the calculation of the gradient vectors can be evaluated numerically using finite differences. One first calculates **g** (Eq.4.3) at the current position $(\mathbf{a}^n,\mathbf{b}^n)$, the error vector $\mathbf{e}=\mathbf{h}-\mathbf{g}$ and the value of the functional at this position $J(\mathbf{a}^n,\mathbf{b}^n)=\text{norm}_2(\mathbf{e})$; then all the coefficients are perturbed in turn and the same calculations are repeated each time to estimate all the partial derivatives of J. For average and high order filters this procedure can be quite cumbersome. It has been shown (Cadzow) that the elements of the gradient vector can also be obtained analytically:

$$\partial J(\mathbf{a},\mathbf{b})/\partial a_i = 2\mathbf{e}^T\{\partial \mathbf{g}/\partial a_i\} \qquad i = 0,1,...,N-1 \qquad (4.35)$$

and similarly:

$$\partial J(\mathbf{a},\mathbf{b})/\partial b_i = 2\mathbf{e}^T\{\partial \mathbf{g}/\partial b_i\} \qquad i = 1,2,...,M-1 \qquad (4.36)$$

These equations imply that for each element of the gradient vector a new (K by 1) vector $\partial g/\partial a_i$ or $\partial g/\partial b_i$ has to be calculated. Fortunately there exists a recursive relation between this set of N+M-1 (K by 1 each) vectors that reduces the number of necessary vectors to just two:

$$\partial g(k)/\partial a_i = \partial g(k-i)/\partial a_0 \qquad i = 0,1,...N-1 \qquad (4.37)$$

and

$$\partial g(k)/\partial b_i = \partial g(k-i+1)/\partial b_1 \qquad i = 1,2,...M-1$$

The two fundamental vectors needed here can be generated as the response of the filter 1/B(z) to a unit pulse input, for the vector $\partial g/\partial a_0$, and the response of the filter $-z^{-1}/B(z)$ to the sequence **g** for the vector $\partial g/\partial b_1$; B(z) is the z-transform of the denominator of the recursive filter. These response sequences are:

$$\partial g(k)/\partial a_0 = \delta_{k-0} - \sum_{i=1}^{M-1} b_i \partial g(k-i)/\partial a_0 \qquad k = 0,1,...K-1 \qquad (4.38)$$

and:

$$\partial g(k)/\partial b_1 = -g(k-1) - \sum_{i=1}^{M-1} b_i \partial g(k-i)/\partial b_1 \qquad k = 0,1,...K-1 \qquad (4.39)$$

The initial conditions for the sequences $\partial g/\partial a_0$ and $\partial g/\partial b_1$ can be assumed equal to zero.

The steepest descent method is relatively simple to implement and guarantees convergence if the functional is convex. The major disadvantage is the linear

convergence which in most cases is too slow. An algorithm exhibiting quadratic convergence would be more desirable in this case. Such an algorithm is described next.

B.2. Quadratic approximation method.

If $J(\mathbf{a}^n,\mathbf{b}^n)$ is approximated by a quadratic functional about the position $\mathbf{a}^n$ and $\mathbf{b}^n$ the search direction vector can be taken as the one that minimizes this new quadratic form. This approach is the *Newton* method; the solution of the following set of (M+N-1) equations will give the direction vector at each position $(\mathbf{a}^n,\mathbf{b}^n)$:

$$H(\mathbf{a},\mathbf{b})\begin{bmatrix}\Delta\mathbf{a}\\ \Delta\mathbf{b}\end{bmatrix} = -\begin{bmatrix}\nabla_a J(\mathbf{a},\mathbf{b})\\ \nabla_b J(\mathbf{a},\mathbf{b})\end{bmatrix} \tag{4.40}$$

where [H(a,b)] is the *Hessian* matrix, containing the second partial derivatives of the functional $J(\mathbf{a}^n,\mathbf{b}^n)$ about the coefficient vectors at the current position $(\mathbf{a}^n,\mathbf{b}^n)$.

In applying this method for recursive digital filter design one may encounter difficulties in the calculation of the Hessian matrix (Cadzow). An alternative algorithm which avoids this difficulty is based on a modified *Gauss* method (Bertran 1975), (Cadzow 1976); the response vector **g** is expanded in a Taylor series about the current position $(\mathbf{a}^n,\mathbf{b}^n)$ and only the first order derivative terms are retained:

$$\begin{aligned}\mathbf{g}(\mathbf{a}^n+\Delta\mathbf{a},\mathbf{b}^n+\Delta\mathbf{b}) \approx \mathbf{g}(\mathbf{a}^n,\mathbf{b}^n) \\ +\sum_{i=0}^{N-1}\partial\mathbf{g}(\mathbf{a}^n,\mathbf{b}^n)/\partial a_i\Delta\mathbf{a} \\ +\sum_{i=1}^{M-1}\partial\mathbf{g}(\mathbf{a}^n,\mathbf{b}^n)/\partial b_i\Delta\mathbf{b}\end{aligned} \tag{4.41}$$

This relation is substituted into the definition of the objective functional $J(\mathbf{a}^n,\mathbf{b}^n)$, Eq.4.9,to yield an approximation quadratic in $[\Delta\mathbf{a},\Delta\mathbf{b}]^T$about the position $(\mathbf{a}^n,\mathbf{b}^n)$. The minimization of this new approximate functional can be accomplished by solving the following normal equations:

$$T(\mathbf{a}^n,\mathbf{b}^n)^T T(\mathbf{a}^n,\mathbf{b}^n)\begin{bmatrix}\Delta\mathbf{a}\\ \Delta\mathbf{b}\end{bmatrix} = -\begin{bmatrix}\nabla_a J(\mathbf{a},\mathbf{b})\\ \nabla_b J(\mathbf{a},\mathbf{b})\end{bmatrix} \qquad (4.42)$$

where $[T(\mathbf{a}^n,\mathbf{b}^n)]$ is a K by (N+M-1) matrix containing the first order derivative sequences of the functional J at the current position $(\mathbf{a}^n,\mathbf{b}^n)$. The same derivatives are contained in the gradient vector on the right hand side of this equation, as explained in the section for the steepest descent method. The calculation of these vectors is carried out in a recursive manner as in Eq.4.37. The solution of this set of (N+M-1) equations gives the search direction vector $[\Delta\mathbf{a},\Delta\mathbf{b}]^T$.

4.2.2.3 Stability considerations.

Although in the present design problem the target impulse sequence h(k) is stable, and the poles of a recursive approximation are expected to be inside the unit circle in the complex plane, in the course of an optimization program, iterative or not, poles and zeros might drift outside the unit circle. Not only is such a recursive filter undesirable, but the design process itself is hampered by the instability of the trial solutions. The possibility of appearance of a number of unstable poles during the design can be drastically reduced by taking precautions in the selection of the target impulse response sequence h(k): carefully choosing the digitization interval and the tapering of the sequence for large k ordinates; nevertheless poles outside the unit circle might still emerge. In the case of an iterative optimization routine there can be two ways to deal

with this problem. One is to let the system be unstable until the optimization converges and try to cure the instability later. This is not feasible in the algorithms that need the calculation of **g** or of its derivatives during the minimization, as these have to be evaluated as responses of an (unstable) system to some input. The alternative, and in some cases imperative, way is to cure the unstable poles whenever they appear.

If G(z) has a real[1] pole $z = \rho_i > 1$ and this is replaced by $z = 1/\rho_i < 1$, this operation is equivalent to multiplying G(z) by the function:

$$F(z) = \frac{z - \rho_i}{z - 1/\rho_i} \tag{4.43}$$

The frequency characteristics of F(z) are:

$$|F(\omega)| = |\rho_i| \tag{4.44}$$

and

$$\tan[arg(F(\omega))] = \frac{(\rho_i^2 - 1)\sin(\omega)}{2\rho_i - (\rho_i^2 + 1)\cos(\omega)}$$

which implies that if the pole is just outside the unit circle, as in this case, its inversion will leave the shape of the magnitude function unchanged but multiplied by a scalar value very near to 1, and the phase will undergo insignificant shifting on the frequency axis. The optimization routine will be normally expected to recover from these minor changes.

1 Real is an assumption that can be justified; since the polynomial of the denominator of the recursive filter is real, its roots will be either real or pairs of complex conjugate numbers. In the second case their product is real so they amount to a real pole.

4.3 Implementation and results.

The impulse response function for the horizontal propagation of the wave motion, Eq. (3.10), (Dommermuth 1986) is the target time domain sequence used in this recursive filter design. As mentioned previously, a finite portion of this infinite length sequence is used. A direct truncation of h(t) at some length t_{tot}=KT, where T is the digitization interval, would create ripples in the magnitude characteristics of the designed filter. In addition to that it was found that a function with sharp decay is more likely to create stability problems when approximated by a recursive digital filter. Windowing the function h(t) in the time domain would avoid the instability, but, due to the special frequency content distribution of h(t) along the time axis (Fig. 3.2), this procedure would suppress the higher frequencies present in h(t) in an uncontrolled manner; therefore the preferred solution is to window in the frequency domain in order to include all frequency components of interest and then inverse Fourier transform and digitize in the time domain. The analytical form of this function was presented in Eq. 3.11. The digitization interval T and the maximum (Nyquist) frequency 1/2T Hz it implies have to agree with the cutoff frequency used in the frequency domain windowing.

The filter designs contained in this study were obtained with the output error method (Section 4.2.2.2). An initial design was carried out by minimizing the equation error of the recursive filter under consideration using the non-iterative methods described in Section 4.2.2.1. The equation error obtained from this method is accompanied by an output error (see Eq.4.29), usually higher in value. If this output error was acceptable, the design process could be stopped; otherwise it would proceed to a second phase in order to further minimize the output error.

Both linear programming and generalized inverses were used in the preliminary task. In general linear programming would yield the smaller final equation error but would very often lead to an unstable solution, particularly for higher orders of N, M. The generalized inverse method found a rather less accurate but usually stable solution. Sometimes ill-conditioning was encountered. In mild cases the measures discussed in Section 4.2.2.1 overcame the problem, but in some cases the method had to be abandoned and the LP method was resorted to.

The non-iterative approach to minimize the final output error, discussed in Section 4.2.2.2 was found impractical due to conditioning problems of the generally large matrices [B]. Steepest descent would converge at a very slow rate whereas the modified Gauss method was found to lead the solution to the minumum after fewer iterations. However this method is not appropriate in certain cases because the solution of the normal equations for the determination of a search direction is meaningless if the matrix $[T^T T]$ in Eq.4.42 is severely ill-conditioned. In these cases steepest descent was the only option left. Instability was encountered during minimization and treating it by inverting the unstable poles was not found to disturb the process. The iterations would stop when both the value of the objective function J(**a**,**b**), and the slope of the gradient vector $[\nabla_a J(\mathbf{a},\mathbf{b}), \nabla_b J(\mathbf{a},\mathbf{b})]^T$ fell bellow a certain threshold, in this case 10^{-3}.

Choosing the order of the recursive filter was seen to be very much dependent on the particular problem considered. An initial choice N=M=K/4 was usually sufficient, but in some cases lower order filters could be designed whereas in others the necessary order could be as high as N=M=K/2.

Considering Eq. 3.10, the instantaneous frequency of the impulse response function h(t) is gt/4x, showing that higher frequencies occur at farther points

along the t axis. Therefore, if it is desired to capture frequencies up to ω_c, the larger the distance x it is desired to transfer wave motion, the larger is the time portion of h(t) to be retained. A practical implication of this observation is that higher frequencies cannot be easily propagated to large distances. These difficulties are encountered in the design of the corresponding recursive filter, where an inconvenient case would result in a higher order filter. When the cutoff frequency was chosen 2π recursive filters of reasonable order were designed for distances ranging from 3 to 20 m; reducing the frequency requirements to π it was possible to design filters for distances up to 50m. The designed recursive filters are listed in Appendix B, Table B.1 for $\omega_c=\pi$ and selected examples for the case $\omega_c=2\pi$ are listed in Table B.2.

The quality of the filters is demonstrated for certain cases in Figures 4.1-4.10. The almost perfect agreement with the magnitude and phase characteristics of the ideal filter is apparent. The error in the phase angle and in the magnitude between the designed and the ideal filters has been calculated as follows:

$$E_{mag} = \sum_i (|H_{ideal}(\omega)| - |H_{designed}(\omega)|)^2 \qquad (4.45)$$

$$E_{phase} = \sum_i [arg(H_{ideal}(\omega)) - arg(H_{designed}(\omega))]^2 \qquad (4.46)$$

In Figures 4.11-4.12 these are shown for both the recursive and the corresponding FIR filter (Eq. 3.11) of the same length (K=M+N), for all the designed cases. It can be seen that the error in the digital filter is in most cases more that an order of magnitude smaller than that of the corresponding FIR having the same length; the latter would have to be of a much higher length than N+M to achieve the same error levels.

4.4 Time series generation software

The horizontal transmission filters listed in Tables B.1 and B.2 have been incorporated into a general simulation program. This program, which is discussed in more detail in Chapter 7, generates records of elevation, x, y and z velocities, x, y and z accelerations at at specified grid points. Facilities for verifying the statistical moments of the records are included.

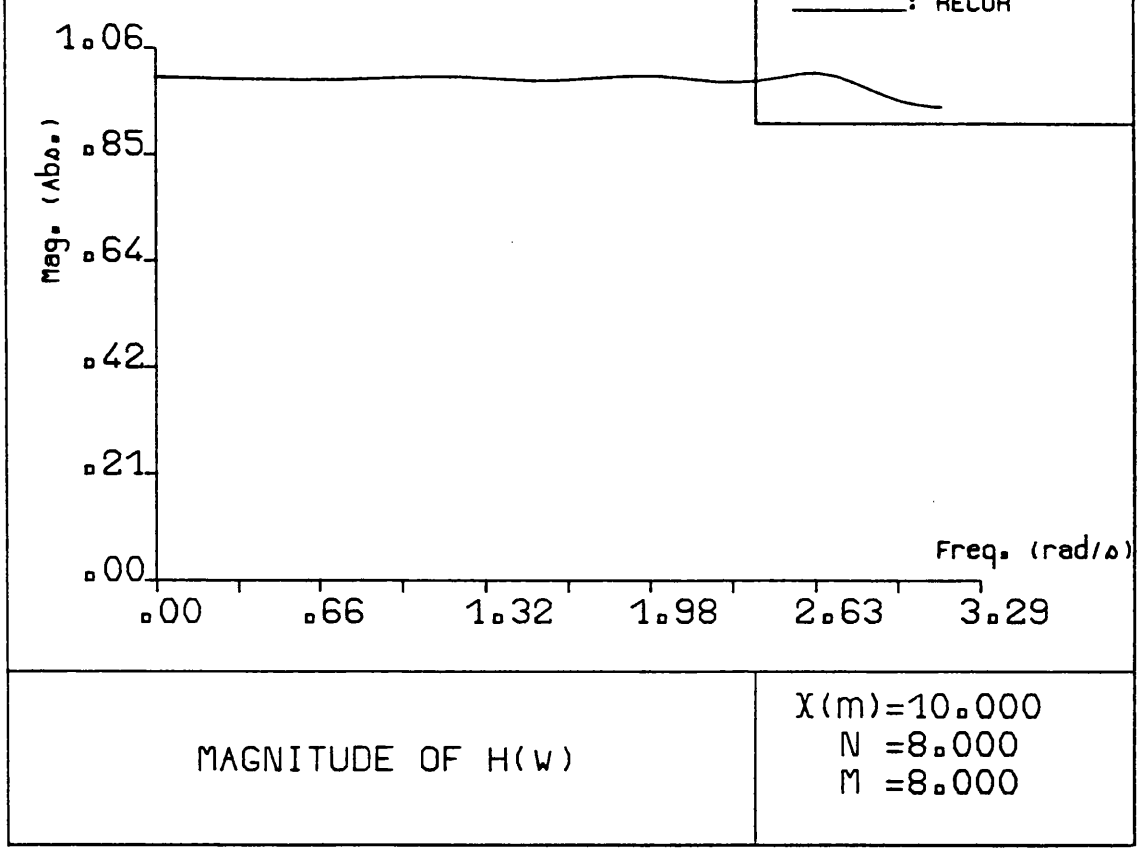


Fig. 4.1a

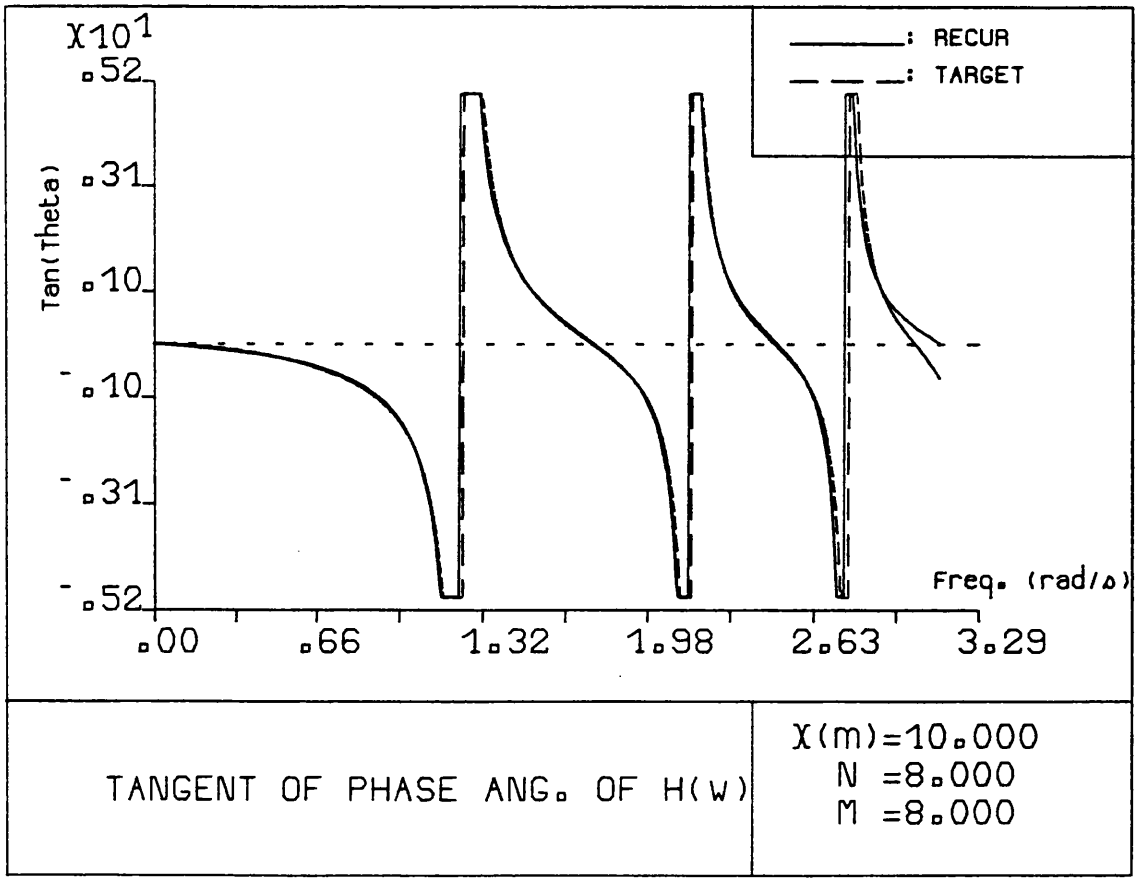


Fig. 4.1b

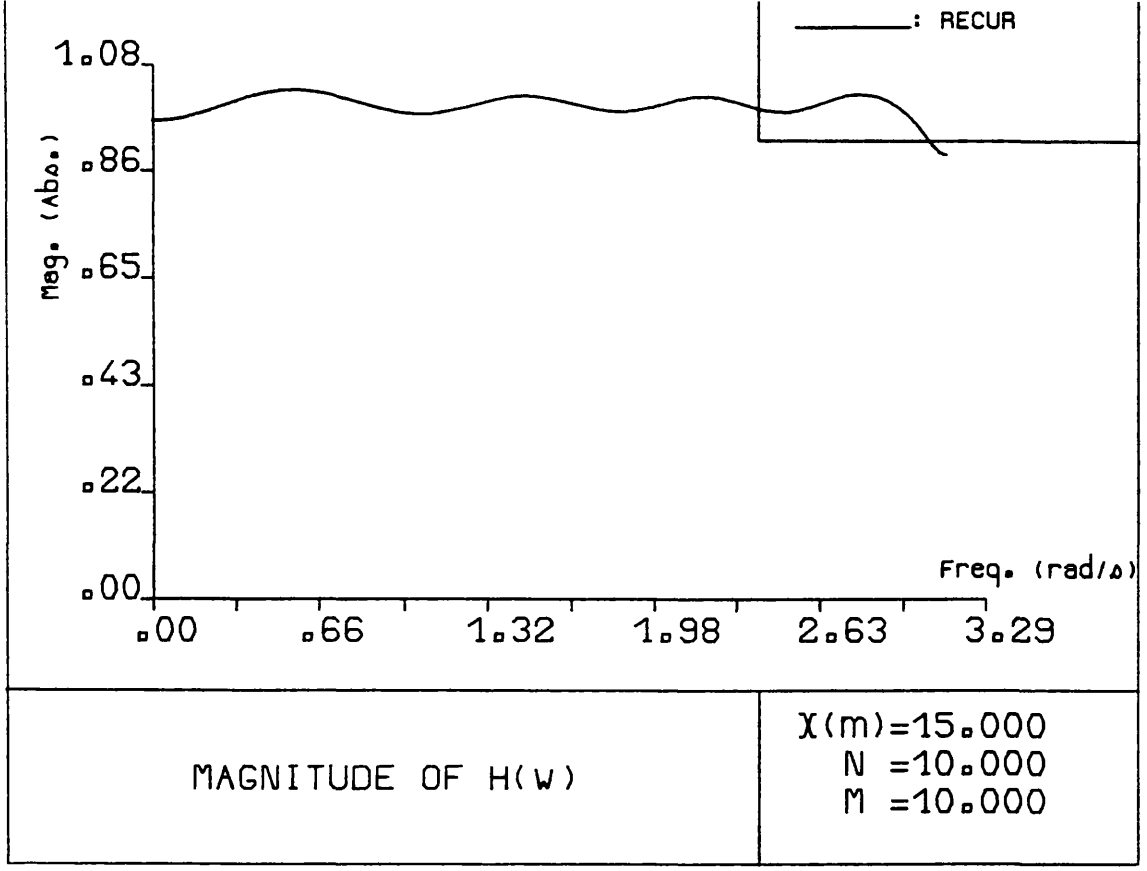


Fig. 4.2a

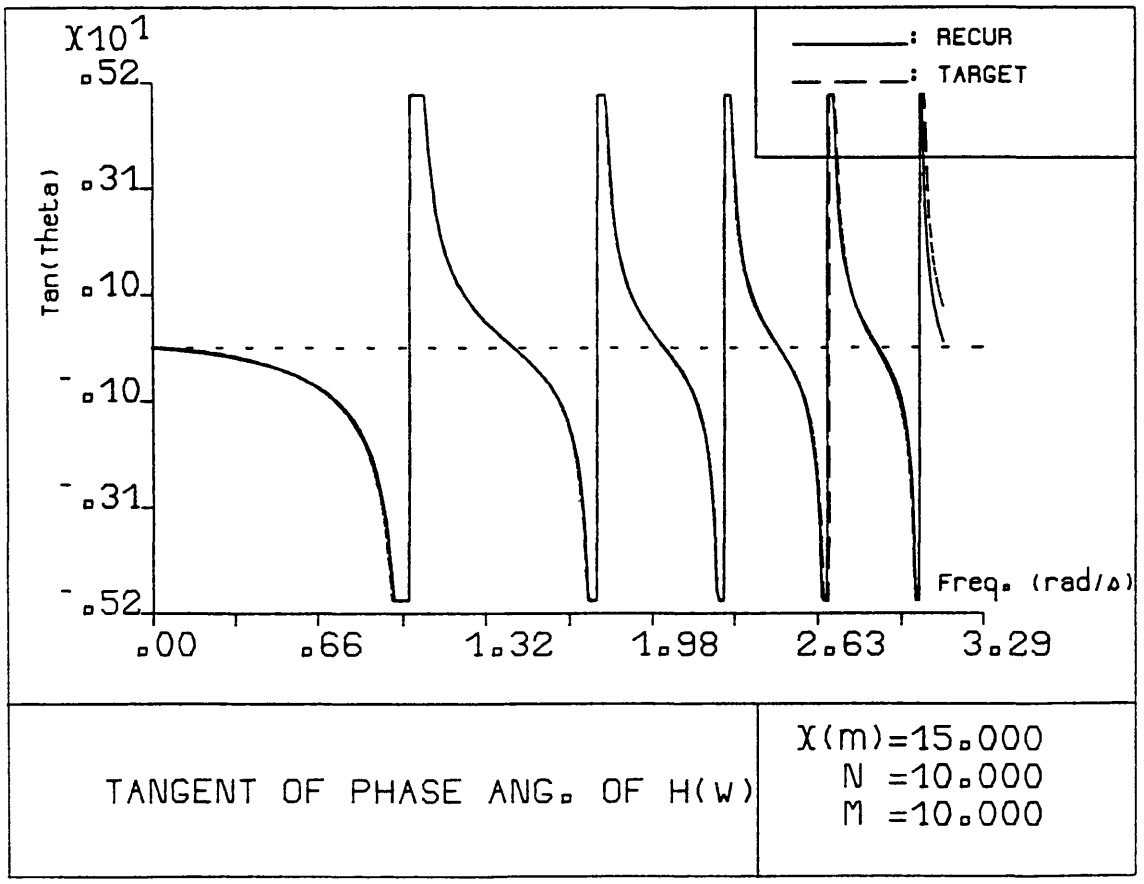


Fig. 4.2b

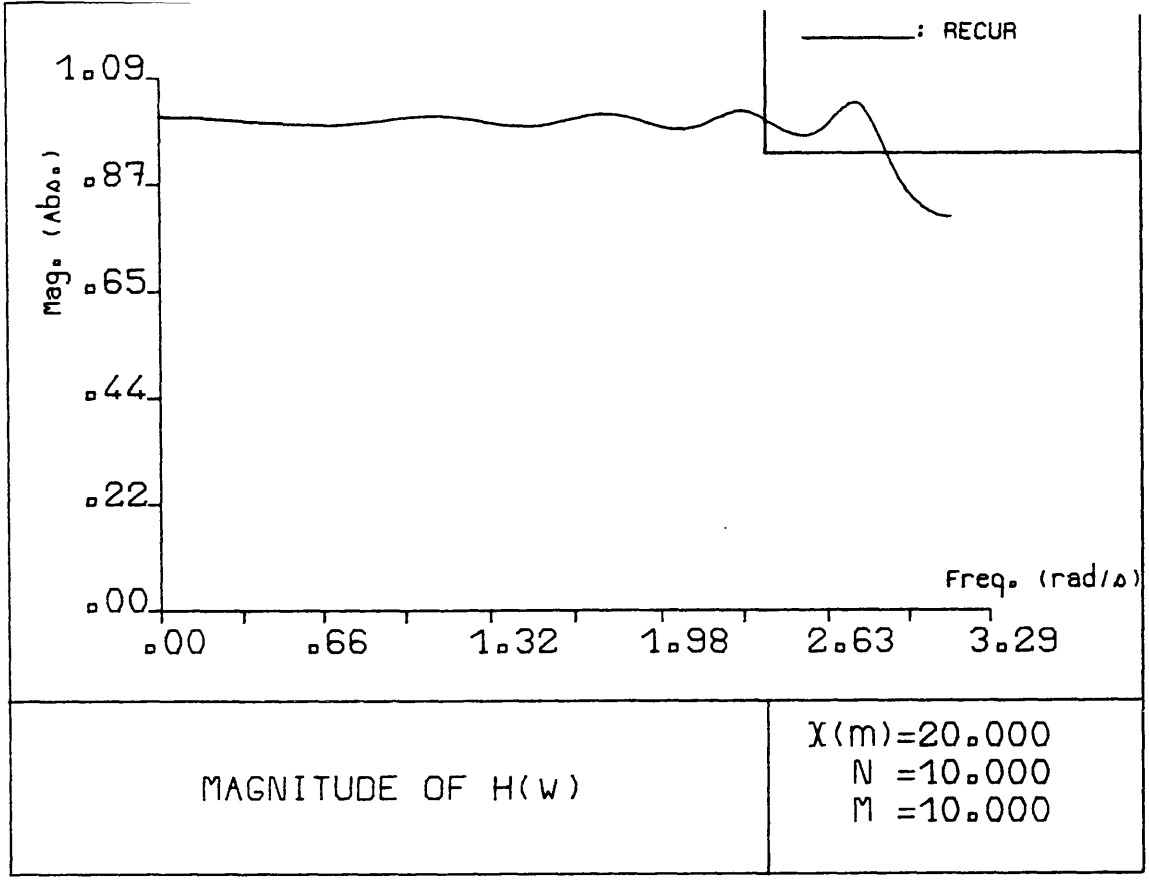


Fig. 4.3a

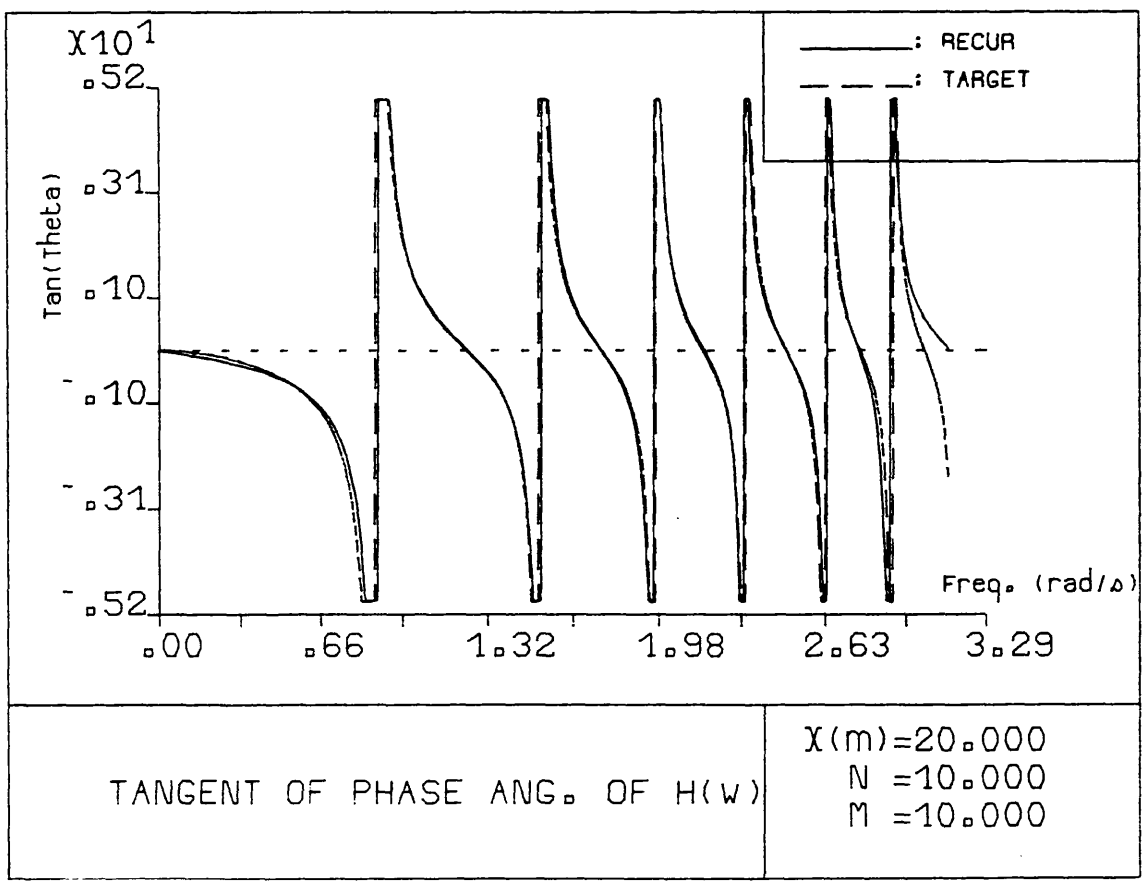


Fig. 4.3b

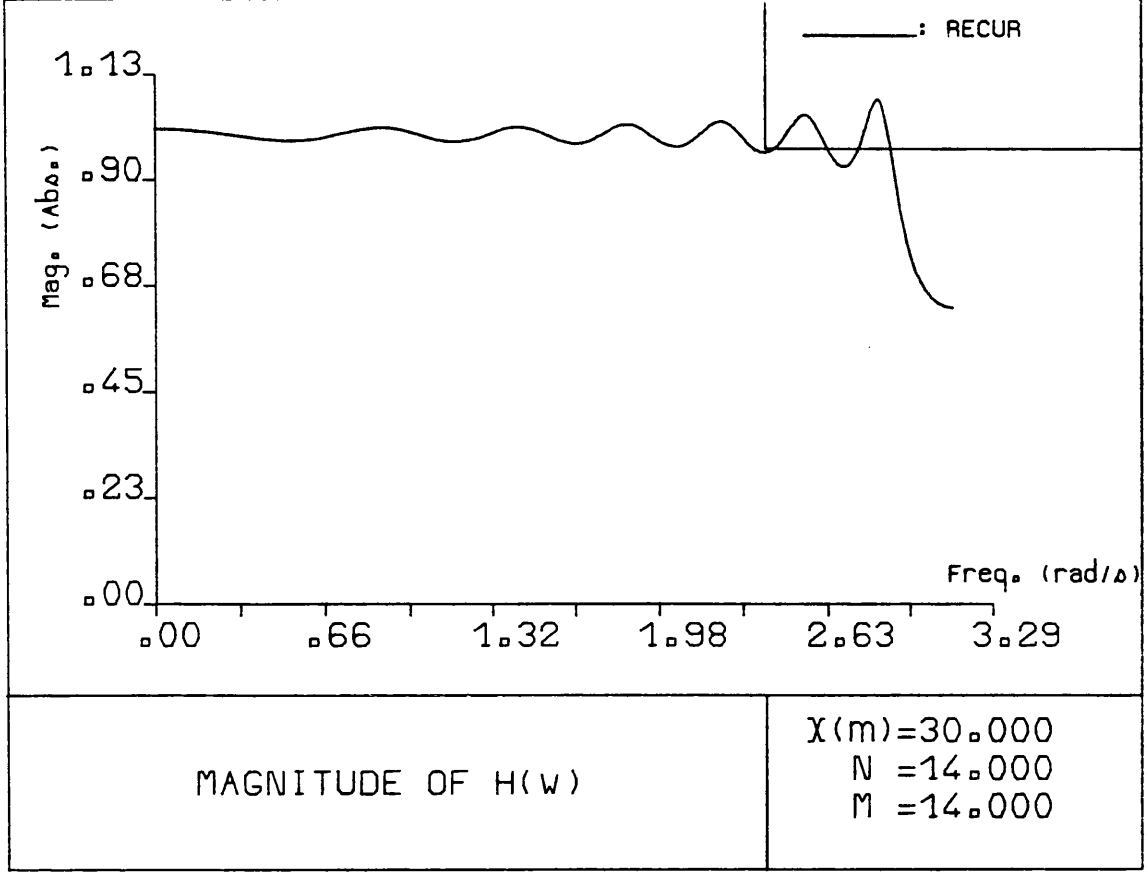


Fig. 4.4a

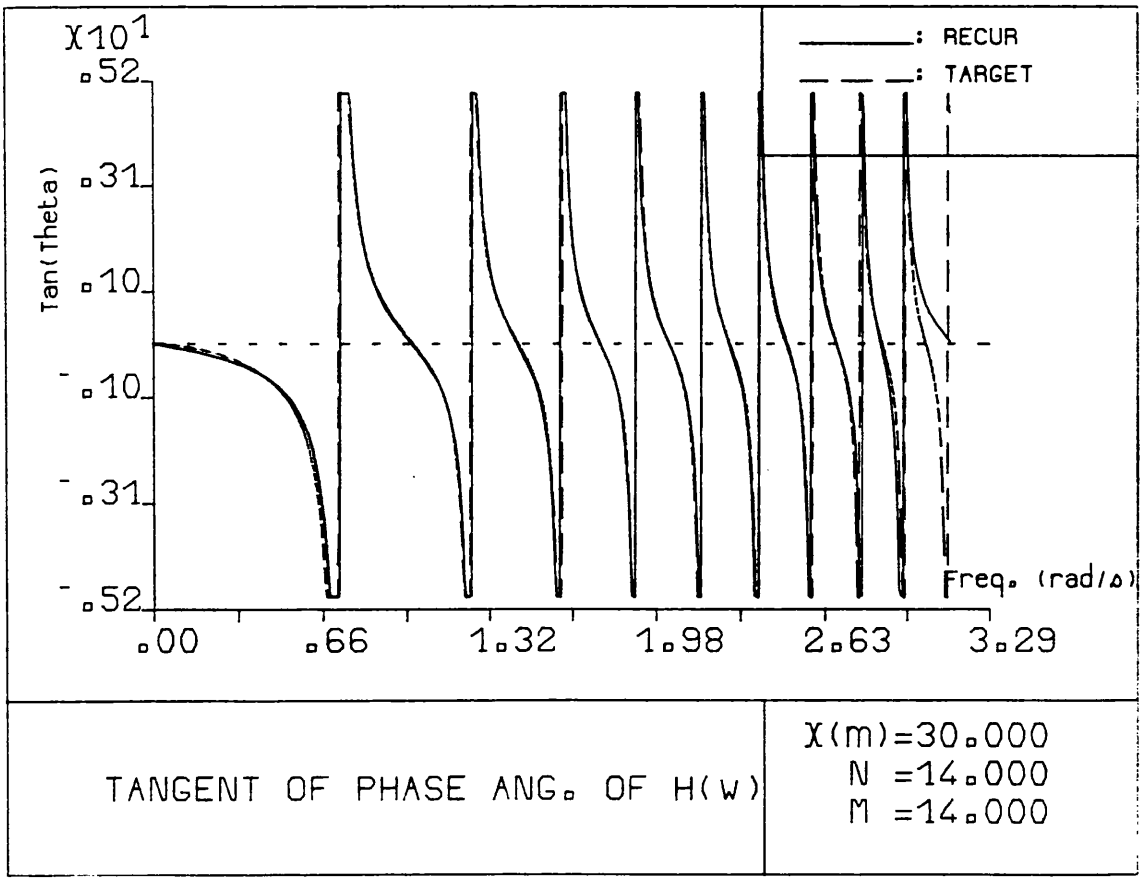


Fig. 4.4b

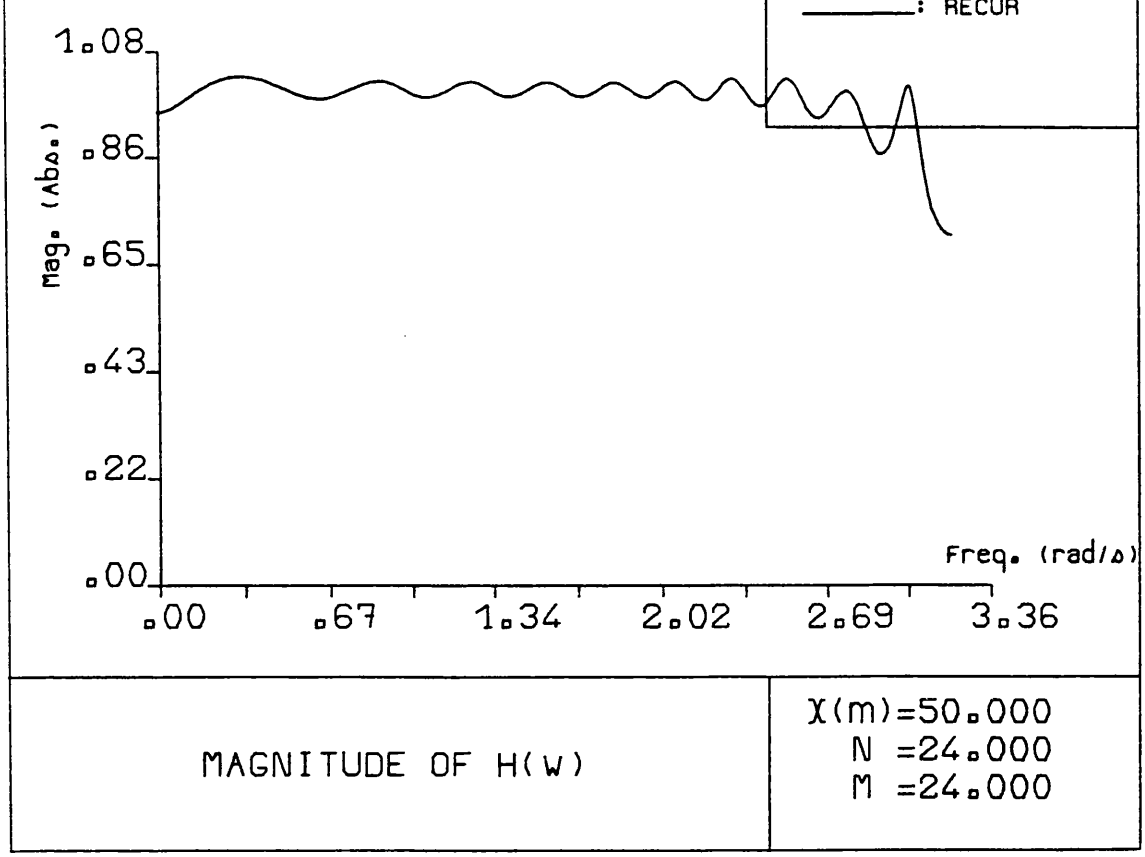


Fig. 4.5a

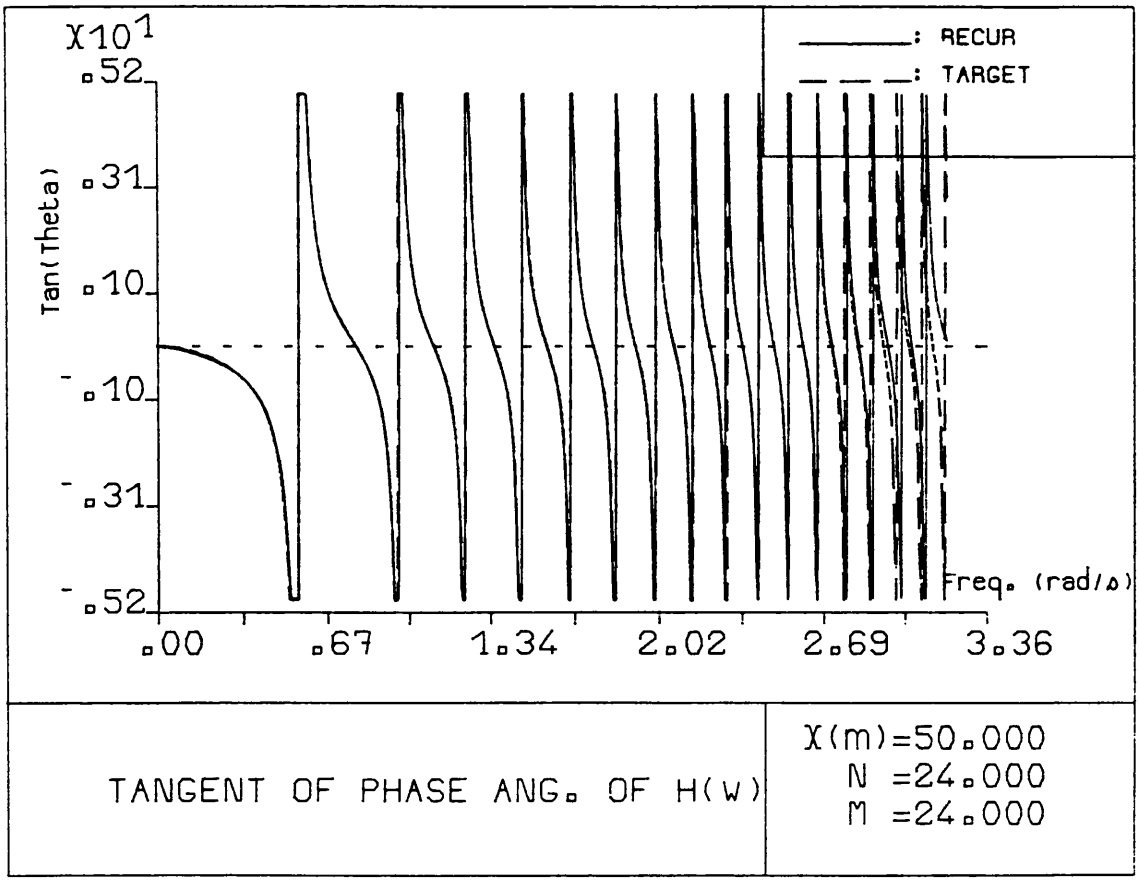


Fig. 4.5b

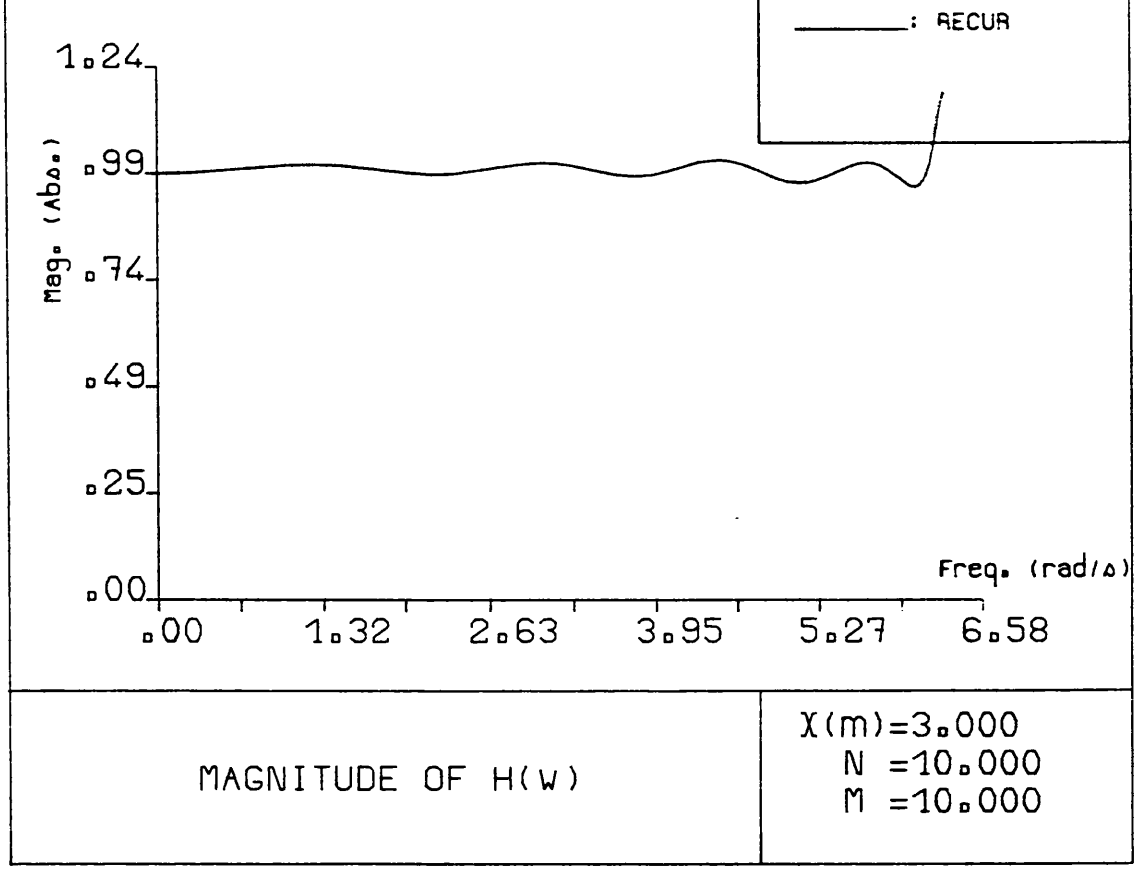


Fig. 4.6a

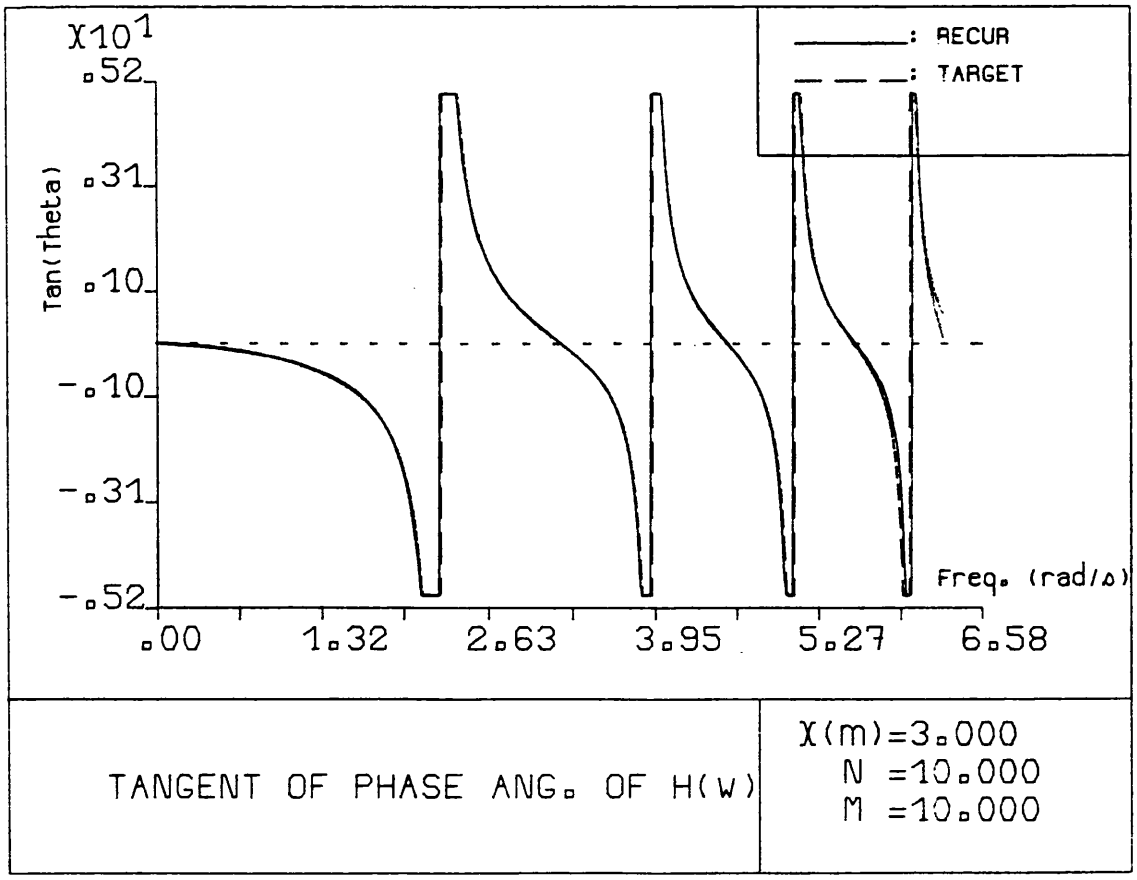


Fig. 4.6b

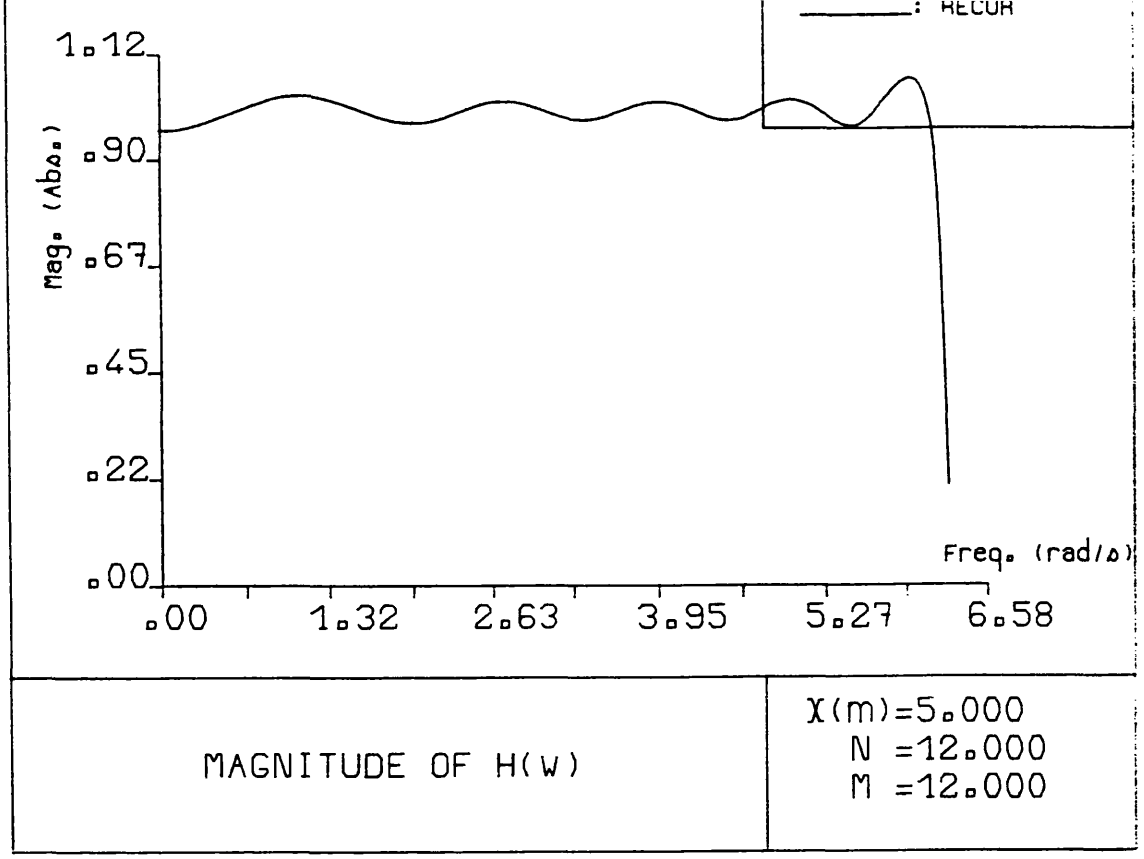


Fig. 4.7a

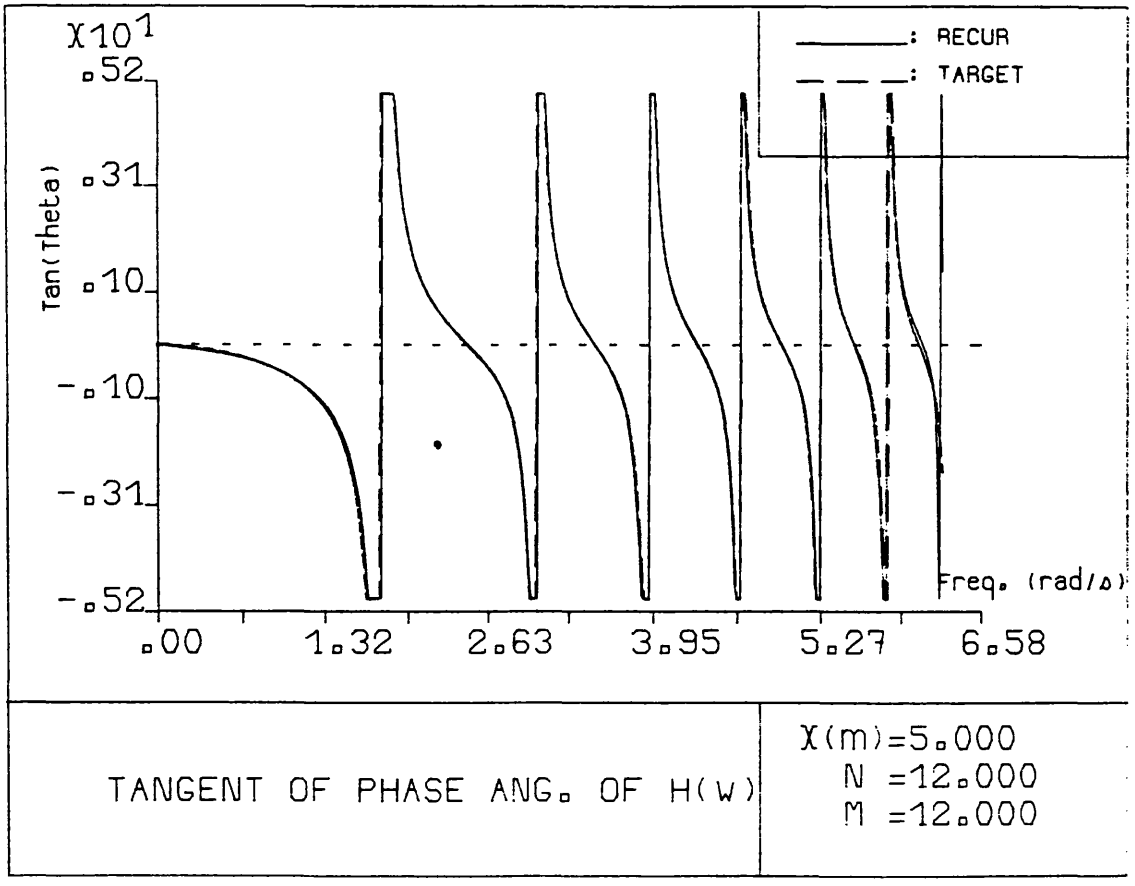


Fig. 4.7b

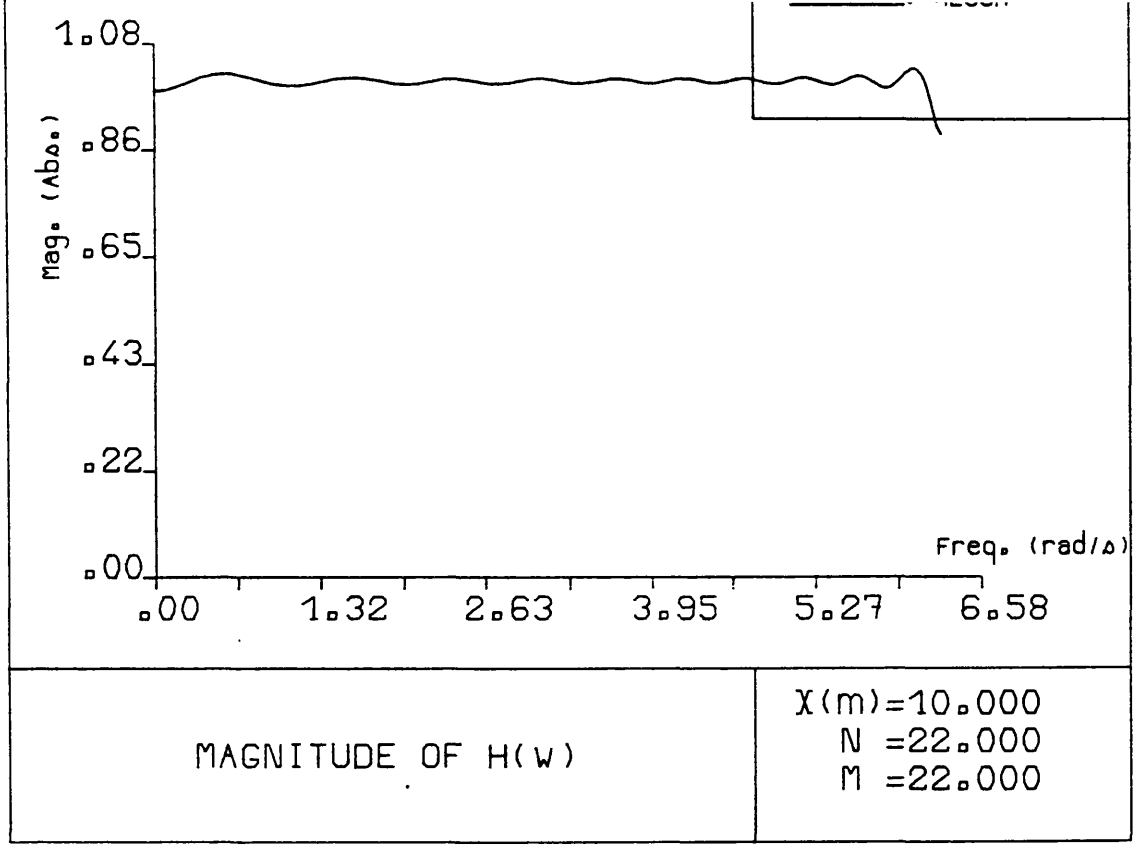


Fig. 4.8a

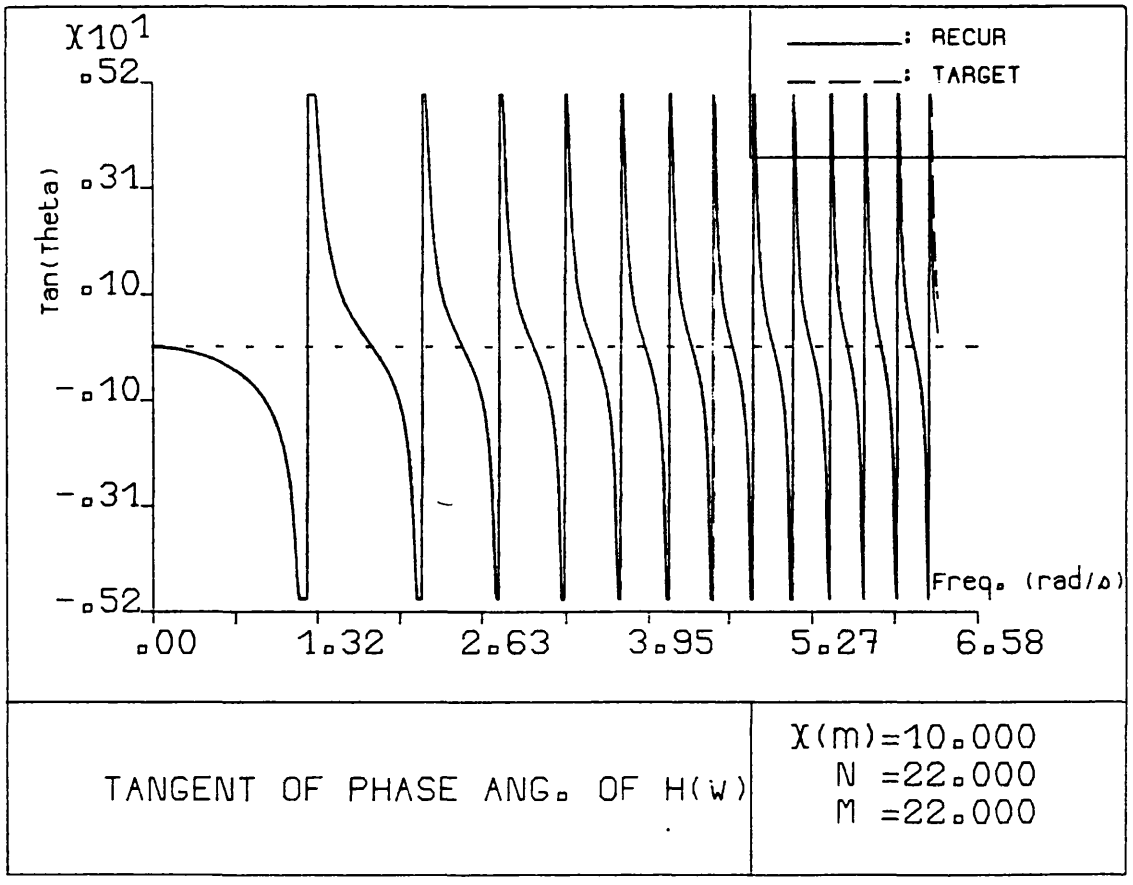


Fig. 4.8b

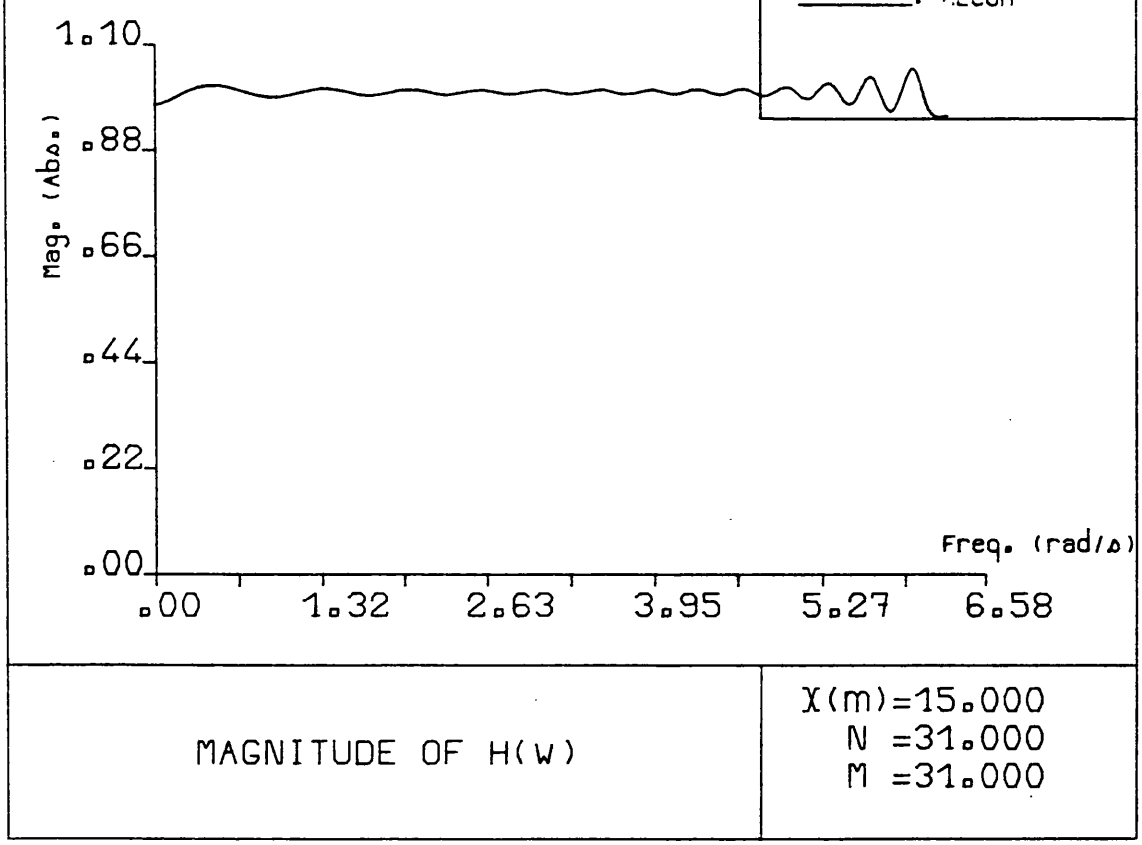


Fig. 4.9a

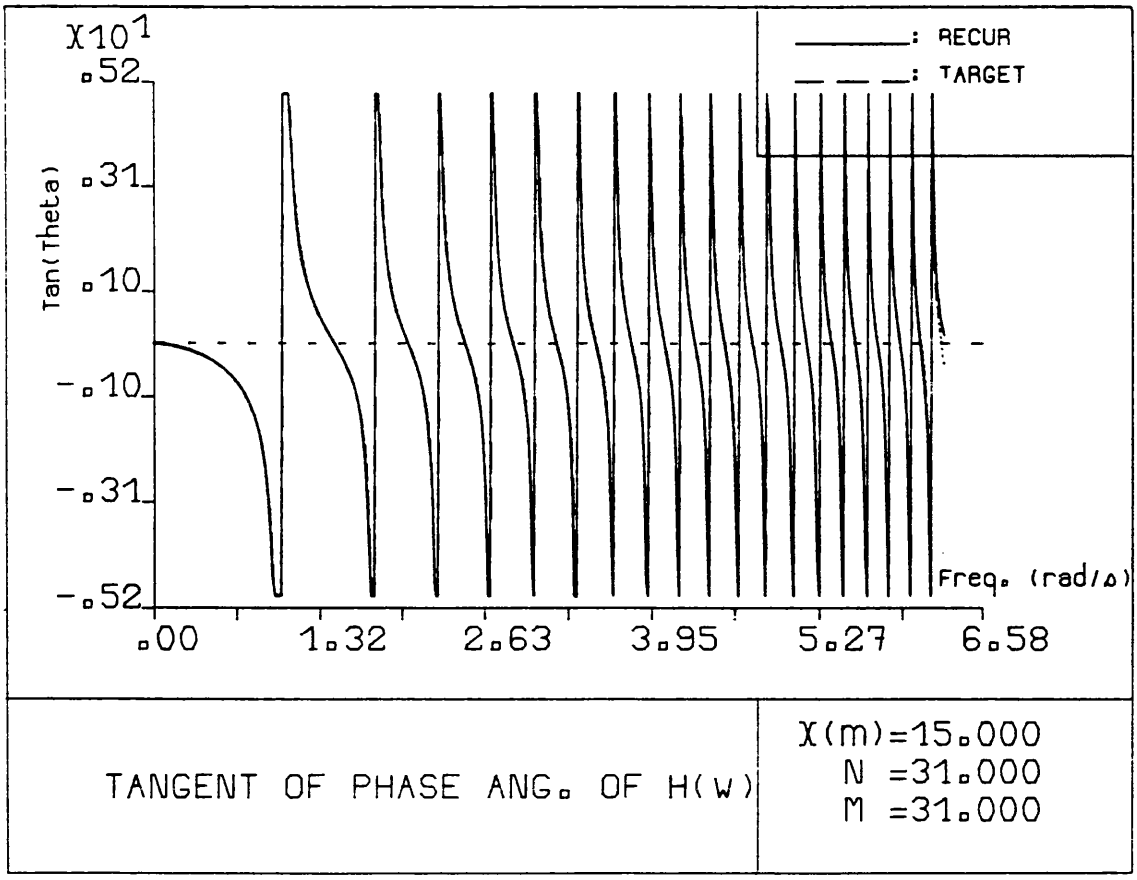


Fig. 4.9b

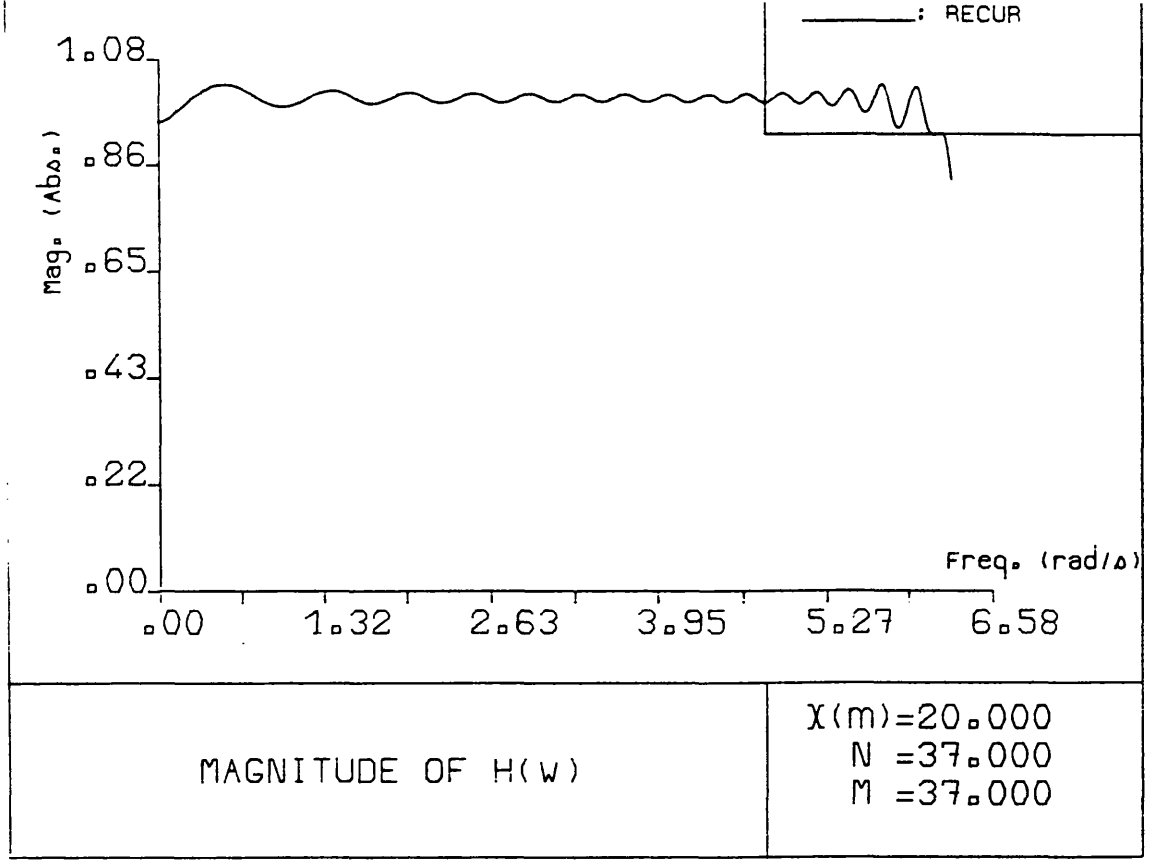


Fig. 4.10a

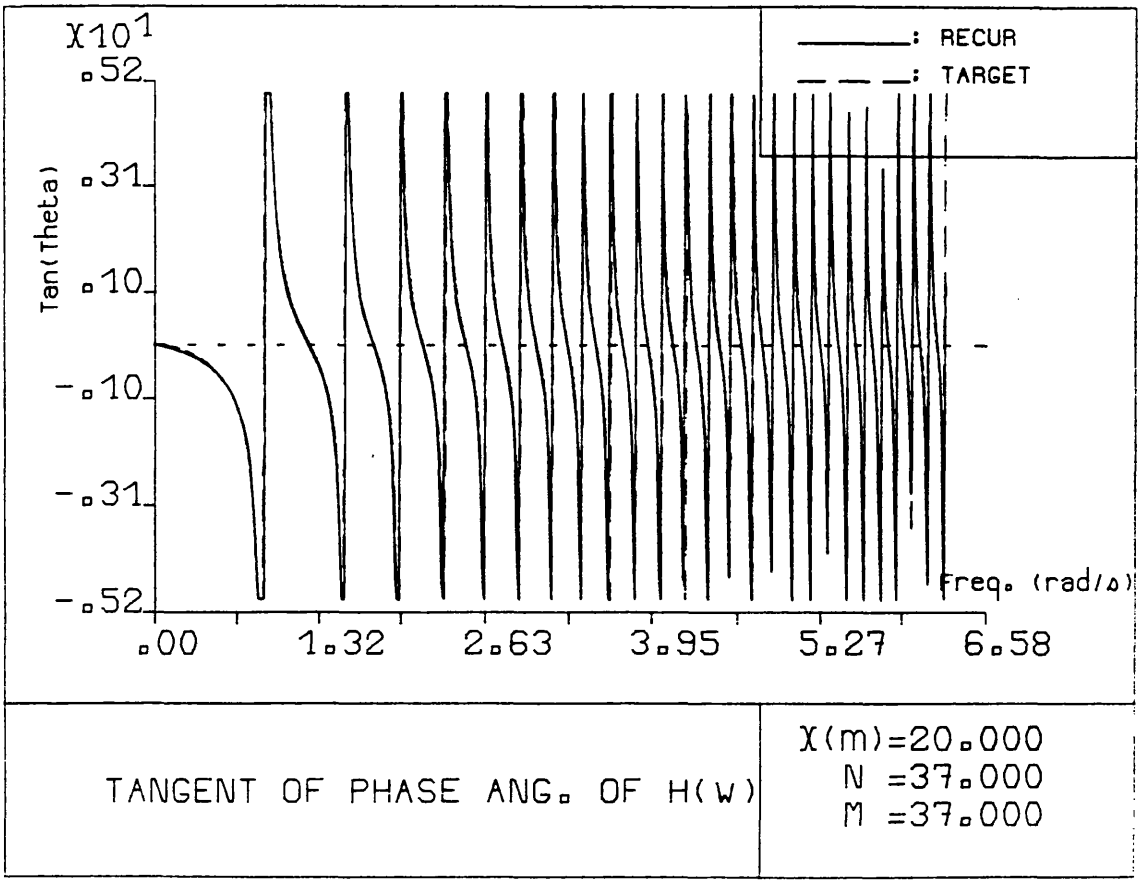


Fig. 4.10b

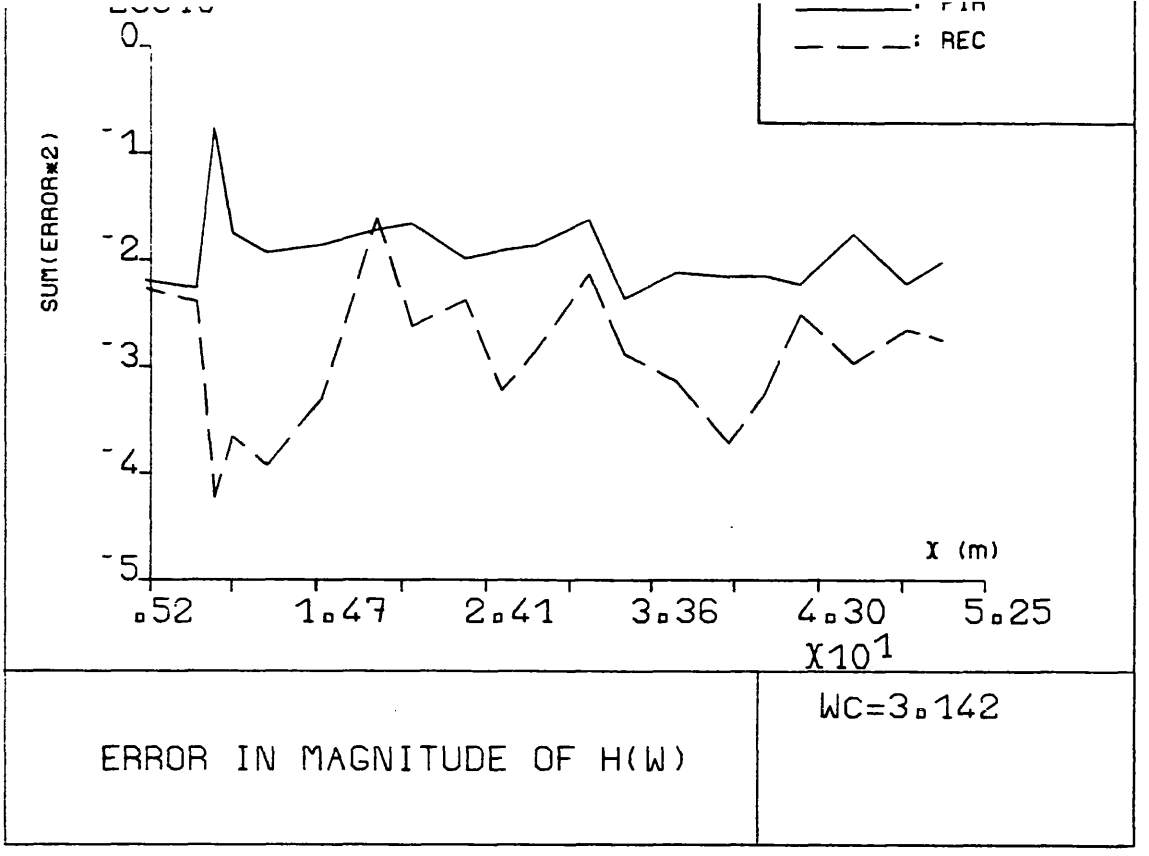


Fig. 4.11a

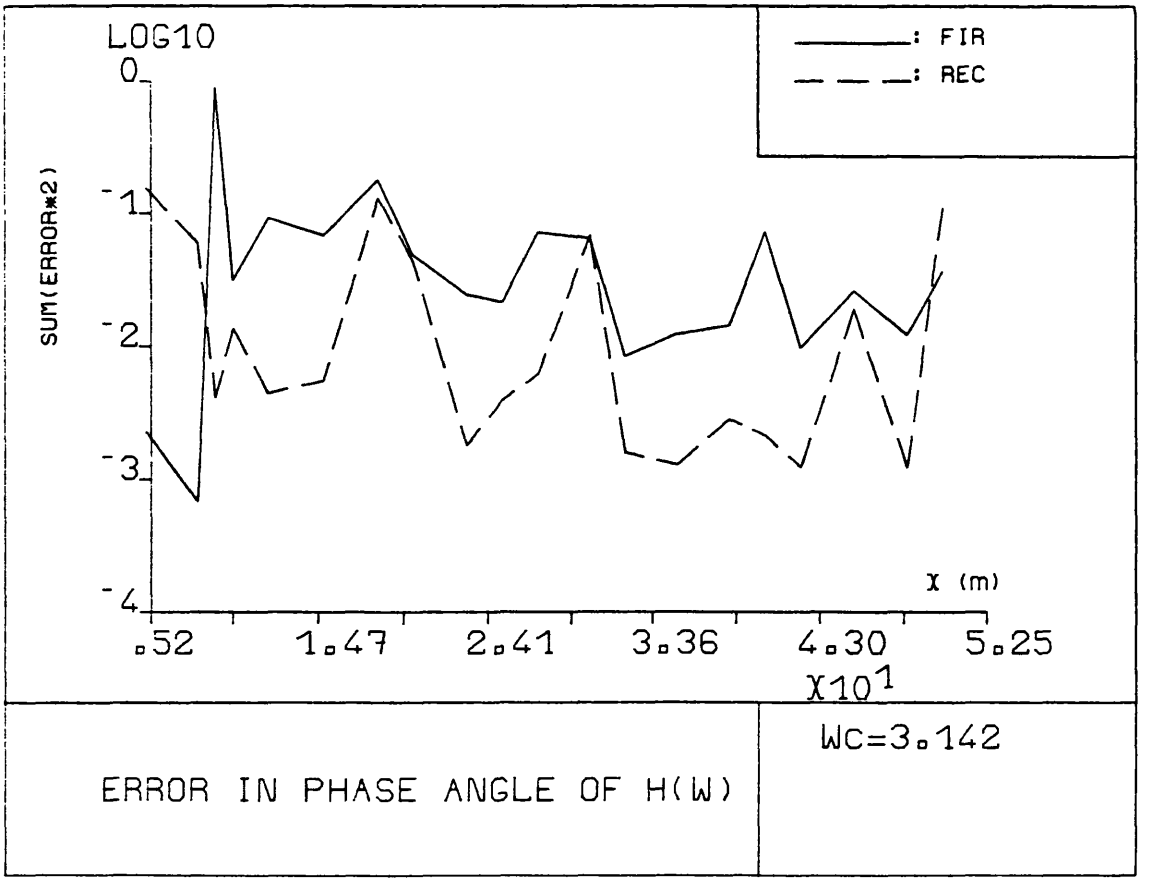


Fig. 4.11b

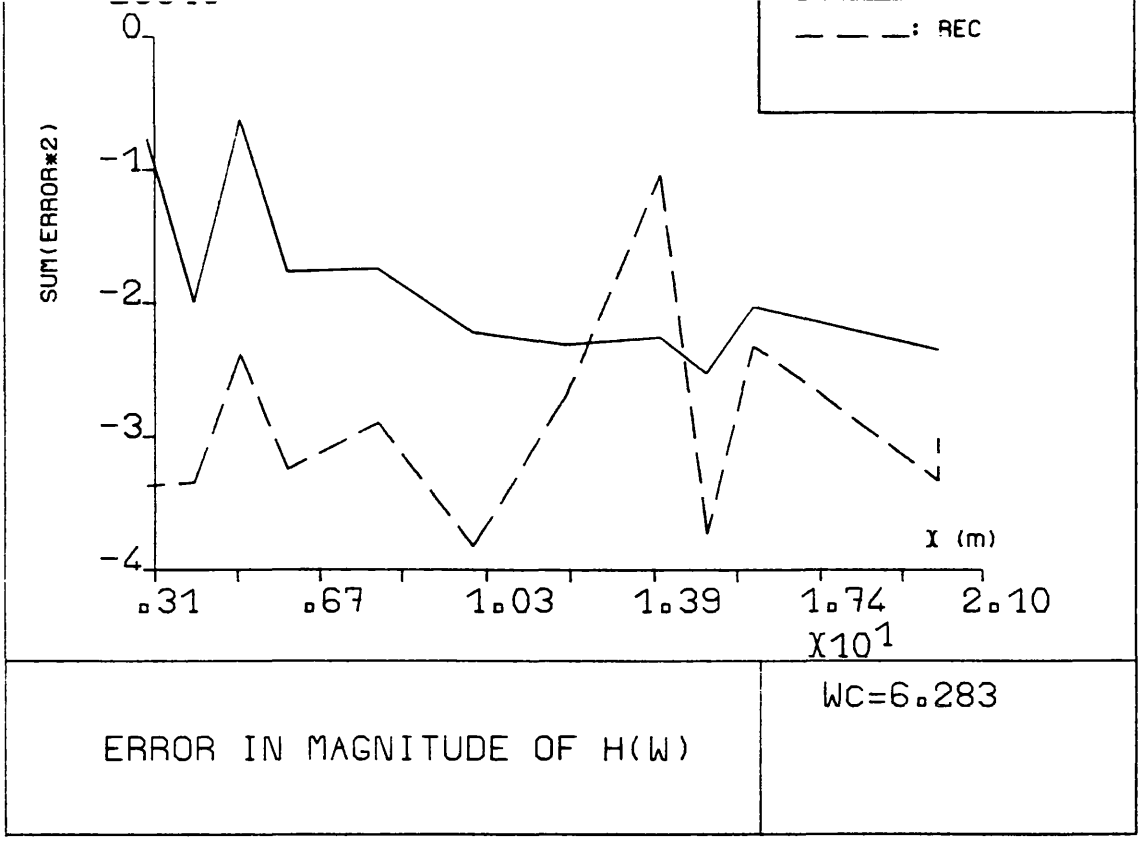


Fig. 4.12a

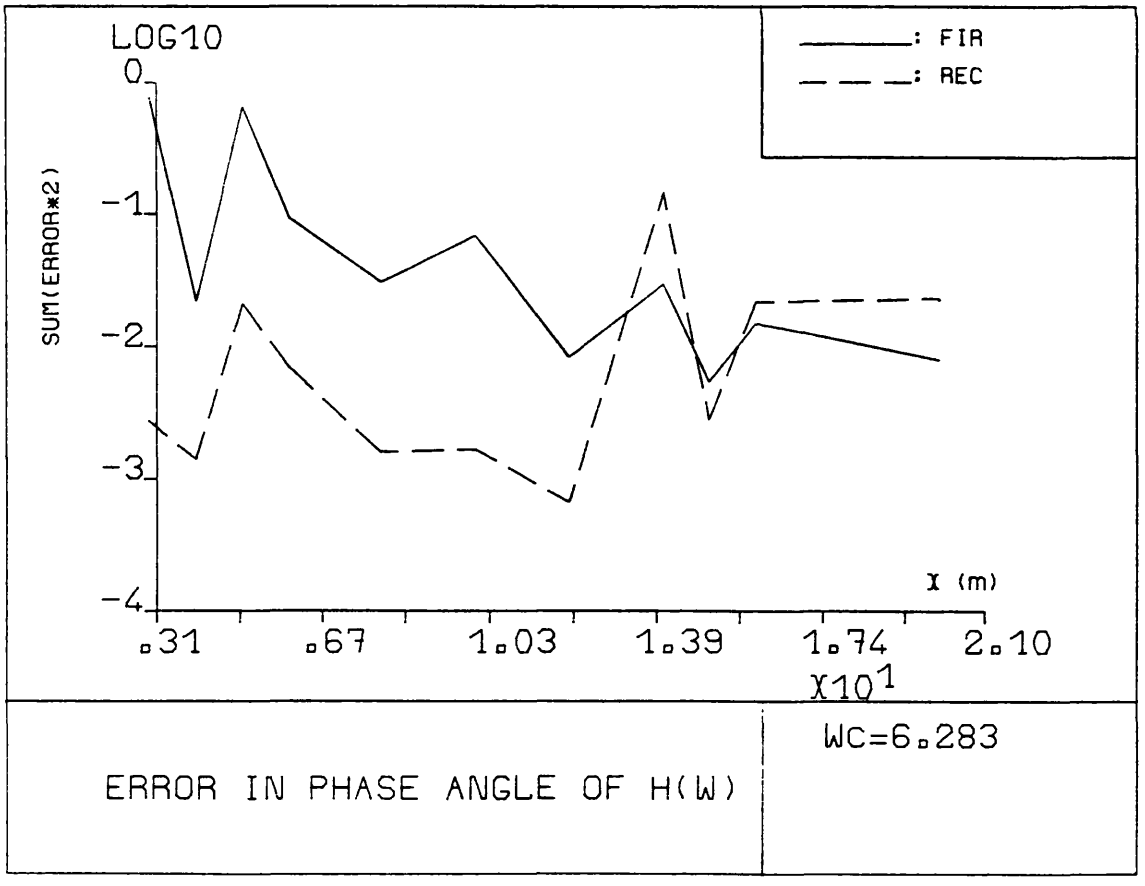


Fig. 4.12b

CHAPTER 5

FREQUENCY DOMAIN DESIGN OF RECURSIVE DIGITAL FILTERS FOR HORIZONTAL TRANSMISSION: THE USE OF ALLPASS FILTERS.

CHAPTER FIVE

CONTENTS

5.1 Introduction.

Again the aim is to replace the infinite length filters of Chapter 3 with a compact recursive digital filter so as to reduce the number of arithmetical operations during execution. In the frequency domain the filter needs to have a constant magnitude of unity and a phase angle varying quadratically with frequency. It is desired to find two polynomials in z^{-1}, such that the rational function (Eq. 4.1) built from them has these two properties.

Departing from tradition, an attempt was made to satisfy the magnitude property *exactly*, and the phase property as closely as optimisation would permit. An elementary recursive filter known as the allpass filter (Oppenheim and Schafer 1975) has the desired magnitude property. Unfortunately, its phase properties can only be matched with the target function over short intervals on the frequency axis. Therefore separate filters need to be designed for different parts of the frequency axis. The topic of allpass filters is reviewed in the following subsection. The overall design problem is considered in Section 5.5. It is emphasised that, compared to the time-domain design procedure reported above, this development is not fully worked out, and its relative merits still await evaluation.

5.2 Properties of a digital allpass filter

An elementary allpass filter is defined by the z-transform:

$$H(z) = \frac{z^{-1} - a^*}{1 - az^{-1}} \qquad (5.1)$$

The coefficient a may be complex, $a=a_1+ia_2$ or pure real. In frequency domain, the transfer function is:

$$H(\omega)=\frac{e^{-i\omega T}-a^*}{1-ae^{-i\omega T}} \tag{5.2}$$

where T is the digitisation time interval. The magnitude of $H(\omega)$ is unity by definition; let

$$H(\omega)=p(\omega)+iq(\omega)$$

where p and q represent the real and imaginary parts. We have:

$$p=\frac{(a_1^2-a_2^2+1)\cos(\omega T)+2a_1a_2\sin(\omega T)-2a_1}{1-2a_1\cos(\omega T)-2a_2\sin(\omega T)+a_1^2+a_2^2}$$

and (5.3)

$$q=\frac{(a_1^2-a_2^2-1)\sin(\omega T)-2a_1a_2\cos(\omega T)+2a_2}{1-2a_1\cos(\omega T)-2a_2\sin(\omega T)+a_1^2+a_2^2}$$

The phase angle function can be defined in terms of its tangent:

$$\tan(\theta)=-\frac{q}{p}=\frac{(a_1^2-a_2^2-1)\sin(\omega T)-2a_1a_2\cos(\omega T)+2a_2}{(a_1^2-a_2^2+1)\cos(\omega T)+2a_1a_2\sin(\omega T)-2a_1} \tag{5.4}$$

This is depicted in Fig. 5.1 for $a=a_1+ia_2=(0.6)+i(0.7)$. The expression in the denominator is periodic, with period $2\pi/T$. Depending on a_1 and a_2, it may have one or two zeros within the interval 0 to π, or none; however if $a_2=0$, there is one zero at most. In the last case, it may also be shown that there will be a zero if $a_1<1$

.

In order to match the desired phase function (Fig. 5.2) it will be necessary to cascade such elementary allpass filters in such a way that the infinities of the tangent occur at the same ω locations as in the desired function.

A cascade of allpass filters has the transform:

$$\prod_{i=1}^{i=n} \frac{z^{-1} - a_i^*}{1 - a_i z^{-1}} \qquad (5.5)$$

In the frequency domain the H(ω) function has magnitude unity. The phase angle function is the sum of the individual phase functions. The tangent may have 2n infinities at most. To verify this, consider the special case where all individual filters are identical, with phase angle function θ(ω). The H(ω) function for the cascade is of the form $(p+iq)^n$. If the real and imaginary parts of this function are P and Q respectively, then Q will be of the form:

$$Q = pq^n - p^2q^{n-2} - p^4q^{n-4}\ldots$$

By reference to Eqs. 5.3 above, each term of this expression will have products of sin(ω) and cos(ω) of total degree n , and the denominators will be common. Thus, in $\tan(n\theta) = Q/P$, the denominator will be a trigonometric expression having a maximum of 2n zeros.

An alternative way of demonstrating the above is to write H(ω) in polar form and to observe that in the Argand diagram $e^{in\theta(\omega)}$ varies more quickly than $e^{i\theta(\omega)}$ as ω varies.

In principle, a cascade of a large number of elementary allpass filters might be employed to produce the desired phase behaviour. However, in practice it has been found that when the coefficients of the individual filters are made equal, then a perfect matching of the phase functions all along the ω axis cannot be achieved.

On the other hand, if the coefficients are made unequal, then the filter design problem is numerically more taxing, and the order of the cascade has to be kept low in order to keep computation time finite. For this reason, the cascades used in the following are designed to match over limited regions, and bandpass filtering is employed to eliminate the phase effects over the parts of the ω axis where the fit is not controlled. Details of the actual bandpass filter used will be presented in Section 5.6.

5.3 Further properties of a cascade of elementary allpass filters

The digital filters under study are required to receive a real-valued input sequence and produce a real output. If the filter coefficients are complex-valued, the output sequence will be complex-valued even when the input sequence is real, but the real or imaginary parts of the output sequence may be utilised in isolation. Since the transform of the real part of the time series is the conjugate symmetric part of the Fourier transform, the phase property of interest is that between the transform of the input and the conjugate symmetric part of the transform of the output. These partial input-output relationships are:

$$Y_e(\omega) = H_e(\omega)X(\omega) \quad \text{and} \quad Y_o(\omega) = H_o(\omega)X(\omega) \tag{5.6}$$

where the subscripts e and o designate the conjugate symmetric and conjugate antisymmetric parts of the function, respectively. Therefore, in the design of the required filter, one must give $H_e(\omega)$ or $H_o(\omega)$ the desired characteristics.

The property invoked above, namely that the conjugate symmetric part of the Fourier transform is the transform of the real part of the time series, is usually presented in the context of continuous time series. However it applies also when the function in the time domain is digitised. In that case the appropriate transform

is the Fourier series coefficient for periodic functions (Oppenheim and Schafer 1974, Section 1.7) Thus, if $H(\omega)$ is conjugate symmetric the filter coefficient sequence h(n) will be real-valued[1]. Conversely, if the desired H(z) function is a ratio of polynomials in z^{-1} with real coefficients, then (by long division) the IIR filter coefficients are real, and the function $H(\omega)$ will be conjugate symmetric with respect to ω.

Another proof of this property will be given below in the context of a cascade of allpass filters.

A further important property of the cascaded filters considered here is the conjugate symmetry of the coefficients in the polynomials in z^{-1} between numerator and denominator. This is displayed in the following expression, and is a result of the special structure of the elementary allpass filter.

$$H(z) = \frac{b_0^* + b_1^* z^{-1} + b_2^* z^{-2} + b_3^* z^{-3} + \ldots + z^{-n}}{1 + b_{n-1} z^{-1} + b_{n-2} z^{-2} + b_{n-3} z^{-3} + \ldots + b_0 z^{-n}} \qquad (5.7)$$

This property holds whatever the constants a_i in the elementary filters. However, if these constants are pairwise conjugate in the following sense,

$$a_i = a_{n-i}^*$$

then it is found that the coefficients b_i in Eq. 5.7 will be real-valued. The property may be verified by multiplying the H(z) functions in pairs:

$$H_1(z)H_2(z) = \frac{z^{-1} - a^*}{1 - az^{-1}} \frac{z^{-1} - a}{1 - a^* z^{-1}} = \frac{aa^* - (a + a^*)z^{-1} + z^{-2}}{1 - (a + a^*)z^{-1} + aa^* z^{-2}} \qquad (5.8)$$

1 The transfer function in Eq. 3.5 does satisfy this property, in other words its imaginary part is an odd function.

It is obvious that the coefficients of z in the above expression are all real-valued quantities.

Finally, it will be shown that when the last-mentioned pairwise conjugacy holds, the function H(ω) is conjugate symmetric, or equivalently, that its conjugate antisymmetric part vanishes. Consider again Eq. 5.8 :

$$H(\omega) = H_1(\omega)H_2(\omega) = \frac{e^{-i\omega T} - a^*}{1 - ae^{-i\omega T}} \frac{e^{-i\omega T} - a}{1 - a^* e^{-i\omega T}} = \frac{A}{B}$$

Using the fact that the modulus of each part is unity it may be shown that the complex conjugate of the above is:

$$H^*(\omega) = \frac{B}{A} = \frac{1 - ae^{-i\omega T}}{e^{-i\omega T} - a^*} \frac{1 - a^* e^{-i\omega T}}{e^{-i\omega T} - a}$$

$$\text{and} \quad H^*(-\omega) = \frac{1 - ae^{i\omega T}}{e^{i\omega T} - a^*} \frac{1 - a^* e^{i\omega T}}{e^{i\omega T} - a}$$

And finally,

$$2\, H_{odd}(\omega) = H(\omega) - H^*(-\omega) = 0$$

Since the product of several conjugate symmetric functions is even, the final H(ω) function derived from pairwise conjugate constants a_i must be conjugate symmetric.

5.4 Stability

For stability, the poles of the H(z) function should be inside the unit circle.The cascaded filter will be stable if each elementary allpass filter is stable. This requires that $|a_i| < 1$

The filter designs presented in the sequel satisfy this requirement.

5.5 Design of component filters

In this section the problem of matching partial cascades of elementary allpass filters to the target phase function will be examined. There are two basic schemes: in the first, a (possibly large) number of identical elementary filters are juxtaposed; in the second, the elementary filters of the cascade have distinct coefficients. For the latter, the computational burden in the design process limits the order of the cascade to about six at the present time.

It is worth emphasising that the numerical determination of the coefficients is specific to one choice of the parameters X, ΔT (distance and digitisation timestep). The process must be repeated for a set of choices.

5.5.1 A product of identical elementary allpass filters

Only the case of real a_i is considered here. The $H(\omega)$ function for this filter was briefly described in Section 5.3 above. When the number of cascaded identical elementary filters is large, the tangent of the phase angle function appears as in Fig. 5.3. The solid line designates the constructed filter, and the symbols designates the target phase angle function. The similarity between these curves is apparent. If it were possible to match the values of ω at which the tangent has infinities with those of the target function, the design would be highly succesful. Unfortunately this matching appears to be possible only over limited ranges on the ω axis. This is explained by the fact that the infinities occur whenever the phase angle function crosses $k\pi/2$ for k=1,3,5... The intervals between ω points at which these crossings occur are governed locally by the gradient of the phase angle with respect to ω. Moreover, the second derivative

governs the rate at which the intervals shorten along the axis. Therefore, it is desirable to match the gradient and the curvature, as well as the value of the phase angle.

With only two parameters to vary, a and N , that is the (real) constant in the elementary allpass filter and the number of such filters in the cascade, it is clear that it will not be possible to match these three values at more than one point on the ω axis. By the device of bandpass filtering, the effect of the filter away from this point can be suppressed. In the designs of Figs. 5.3 -5.5 only the slope and the curvature were matched (in an exact sense). It may be seen that the fit is locally very good, although the order of the designed cascade had to be as high as 57 in one case. This method holds some promise, especially at large ω where other methods have difficulty; but further work is needed to include the third matching criterion into the design algorithm.

5.5.2 Coefficients for a heterogenous cascade

It was seen in the previous section that the cascade of identical allpass filters may give a good fit in the region of large ω. For values of ω near the origin it is necessary to construct a cascade of allpass filters with unequal coefficients. For reasons explained previously, complex-valued coefficients are used forming symmetric pairs, yielding a real-valued h(n).

If the number of the unknown coefficients is chosen to be the same as the number of the ω-locations where the functions are matched, the problem is equivalent to the solution of a number of simultaneous nonlinear equations. When the problem involves more positions on the ω-axis than the number of

the unknown a_i coefficients, the design will try to minimize the sum of the squares of certain residuals at chosen locations. Numerical algorithms for both these strategies were implemented in this work.

Since the phase angle $\tan^{-1}(q/p)$ of a single allpass filter is a multiple valued function, see Eq. 5.4, it is not practical to match the quantity

$$\tan^{-1}\left(\frac{q_1}{p_1}\right)+\tan^{-1}\left(\frac{q_2}{p_2}\right)+..$$

for the overall filter. Instead, the routines match the *tangent* of the phase of the cascade with recursively updated p and q.

For one filter only, one can always find *Re*(a) and *Im*(a) that would give $\tan(\theta)$ the same value as $\tan(-\omega^2 x/g)$ at two arbitrary locations. It was found more convenient to align the locations of the zero and of the singularity in $\tan(\theta)$ where $\sin(\omega^2 x/g)$ or $\cos(\omega^2 x/g)$ is zero.

One elementary filter covers at most two branches of the $\tan(\theta)$ function. When trying to match more branches it was found that a maximum of six branches could be succesfully matched with one cascade of allpass filters, thus covering the range 0-1.84 rad/sec. Other, similarly constructed cascades may be used to cover the rest of the ω-axis. The effect of each cascade on the phase behaviour outside the band of interest is eliminated by bandpass filtering.

In one particular case (x=50m, T=1s), the error minimisation exercise resulted in the digital filter presented in Table 5.1. The quality of the matching achieved by this filter can be seen in Fig. 5.6.

Table 5.1

Coefficients b_i for x=50m, T=1s	
0	0.4288517
1	-0.9145078
2	1.9277369
3	-2.1324127
4	2.4873923
5	-1.3609384
6	1.0000000

The filter in Table 5.1 has been performance-tested by inputting white noise and analysing the cross-spectrum between input and output. The results, shown in Figs. 5.7 and 5.8, indicate that the performance is adequate.

5.6 Bandpass filtering

The need for a bandpass filter attached to each cascade of allpass filters has been mentioned in previous sections. An ideal bandpass filter should possess a transfer function $H(\omega)$ with unity magnitude over the frequency band for which it is designed and zero outside the band. In addition, to avoid interference with the phase characteristics of the main filtering process, it needs to be a zero-phase filter.

The classical bandpass filter of central frequency ω_c and width ω_b:

$$h(n) = \frac{2\cos(nT\omega_c)\sin(nT\omega_b/2)}{nT\pi} \qquad (5.9)$$

is of infinite duration, and is not zero-phase. It is modified as follows: first the length is truncated by the application of the Kaiser window (Rabiner and Gold). Secondly the phase is corrected. Due to the symmetry of the time domain function about its midpoint, the phase is originally linear in ω. This may be cancelled out by appending the filter H(ω)=exp(-ibω) where b is the constant slope. In the time domain, this corresponds to shifting the convolution function until it becomes symmetric with respect to the origin (and hence noncausal). The noncausality can be easily overcome in practical applications by storing some of the future input process values in memory (a lookahead technique).

The coefficients of a bandpass filter of length 127 are plotted in Fig. 5.9. Its performance may be judged from Figs. 5.10 and 5.11 which shows the magnitude and phase angle functions as inferred from the cross-spectrum between a white noise input and the output. The phase angle results require some interpretation: within the passband, the angle is suitably small. Outside the passband, the digital filter produces a nonzero phase whereas the corresponding ideal passband filter would yield zero phase. This effect is attributable to the differences between the frequency domain characteristics of the two filters. It should be observed that the unwanted phase behaviour outside the passband will not appear in the final compounded result since the contributions to process ordinates at a particular frequency will be dominated by one filter only, whose passband contains this frequency.

From a comparison of Figs. 5.10, 5.11 with Figs. 5.7, 5.8 it may be seen that a filter length of 127 gives more accuracy than required in this application, in view of the lower accuracy accepted elsewhere. In fact, Rabiner and Gold present an example with N=46 which yields sufficient accuracy (attenuation of -40.8 dB in the stopband).

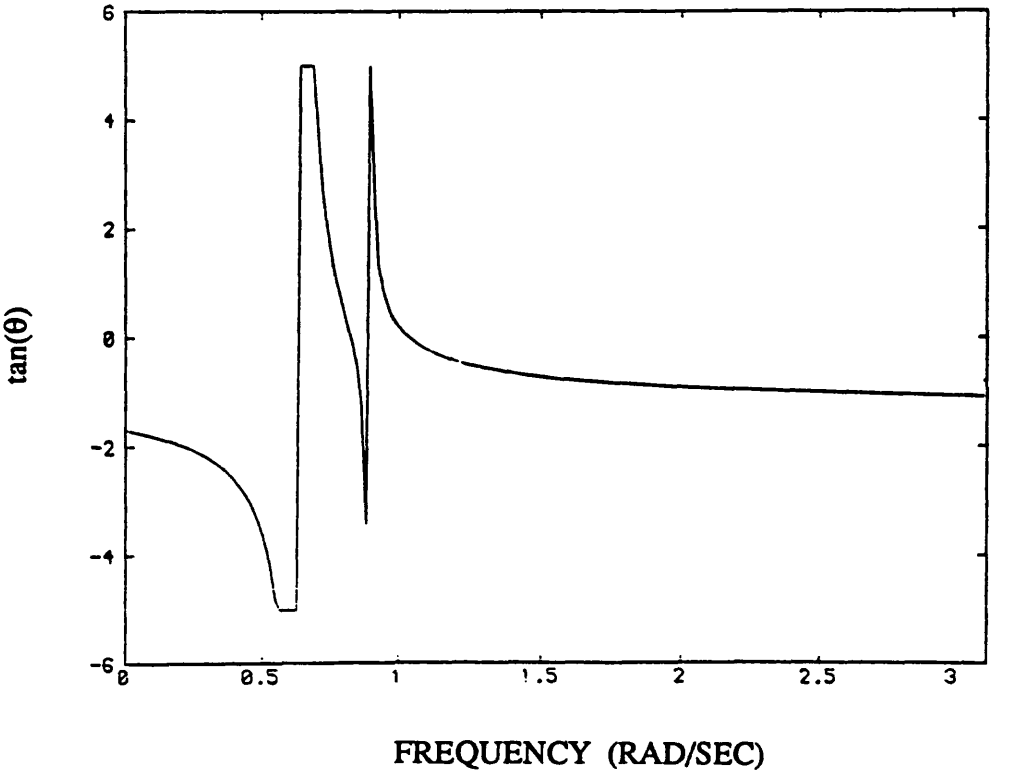


Fig. 5.1 tan(θ) for a single allpass filter with a=(0.6)+i(0.7)

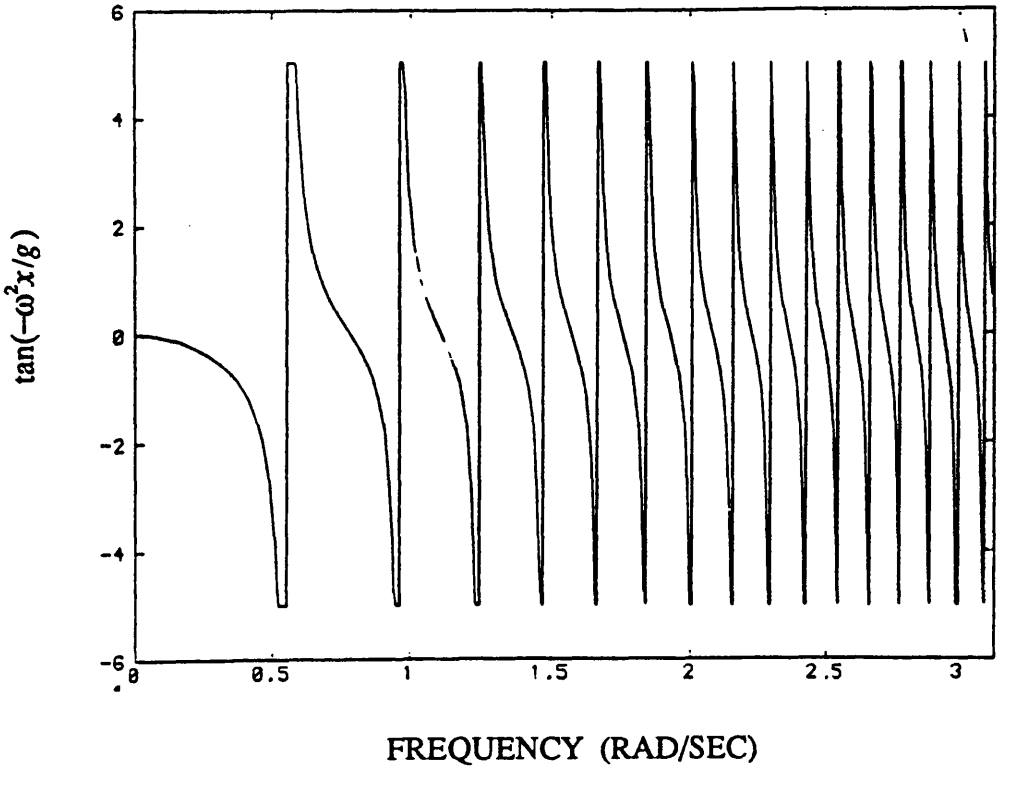


Fig. 5.2 The phase angle characteristic of the target transfer function.

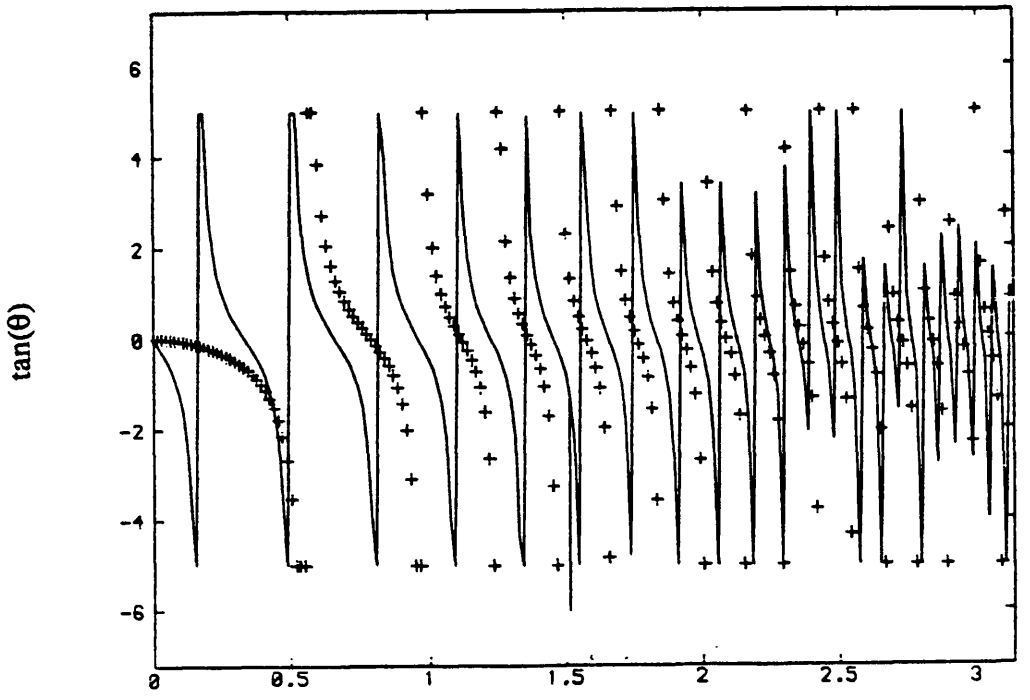


Fig. 5.3 Cascade of identical filters matching at ω=1.5 rad/sec

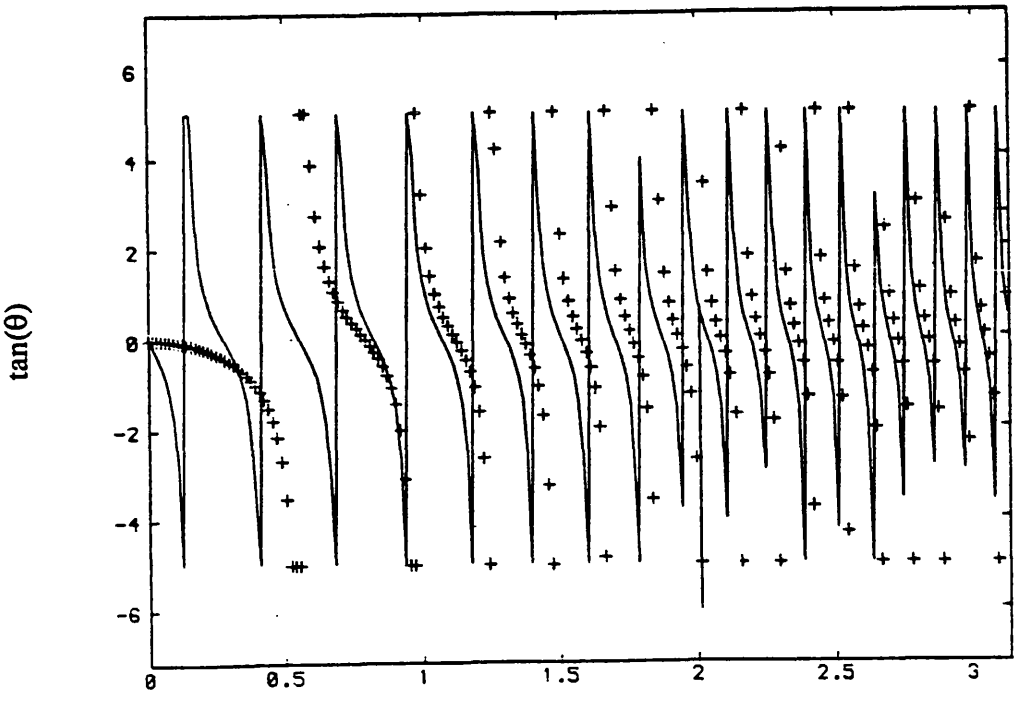


Fig. 5.4 Cascade of identical filters matching at ω=2.0 rad/sec

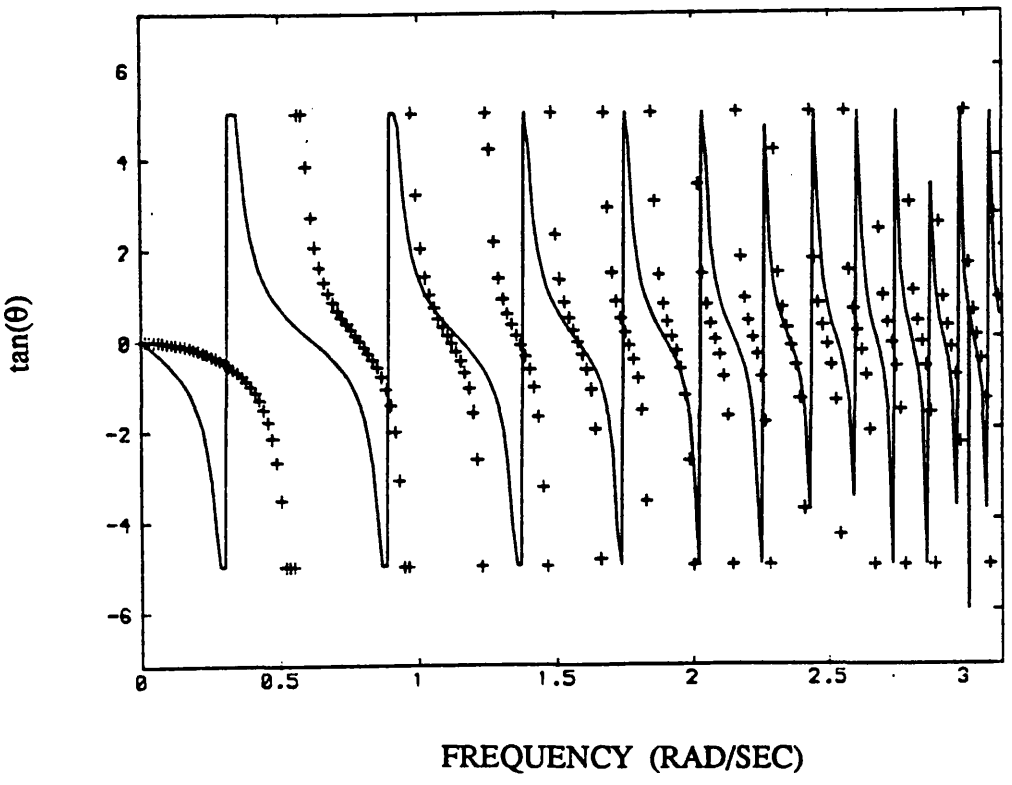


Fig. 5.5 Cascade of identical filters matching at ω=3.0 rad/sec

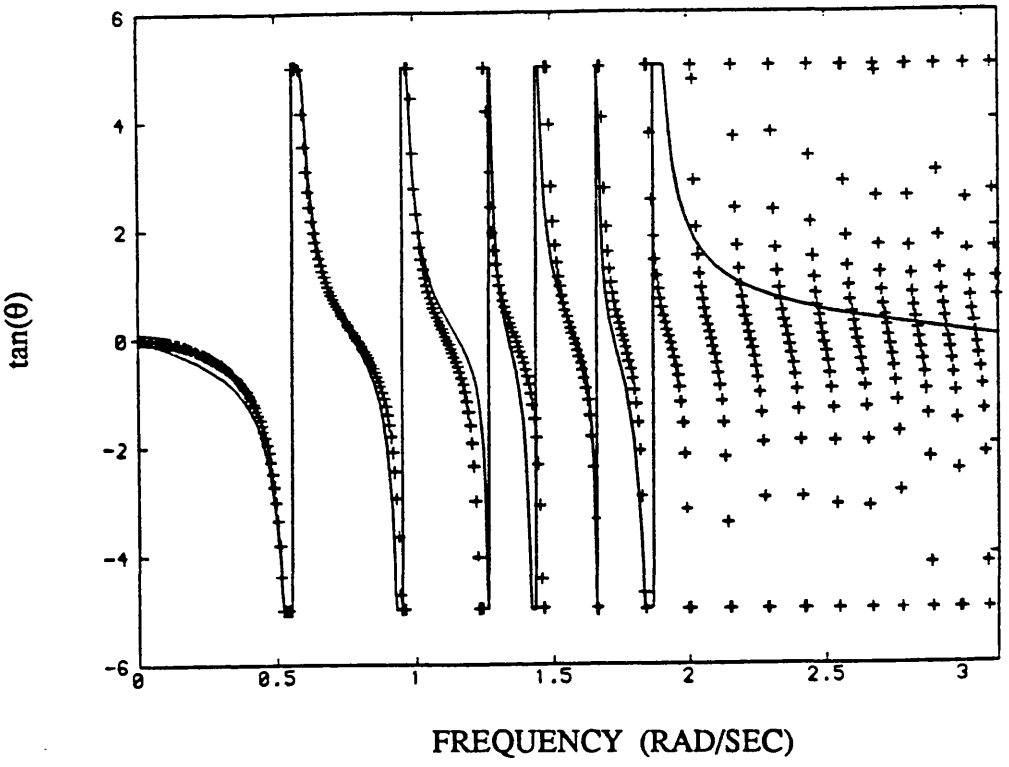


Fig. 5.6 Cascade of 6 non-identical filters matching at
$\omega = 0$ to $\omega = 1.84$ rad/sec

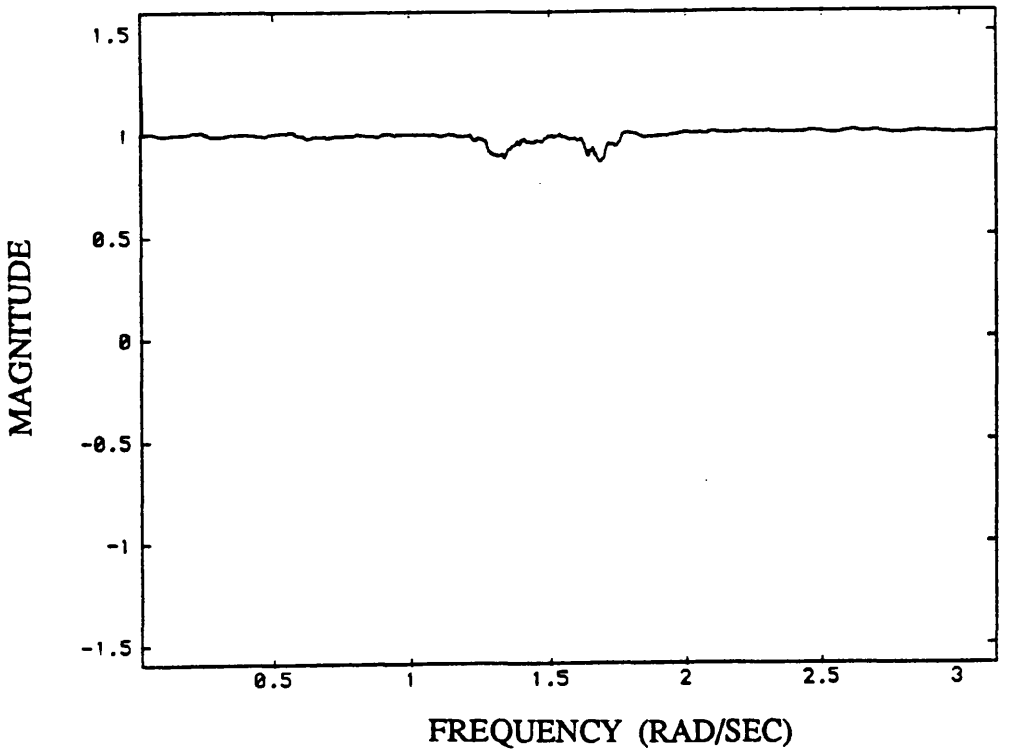


Fig. 5.7 Magnitude of the cross spectrum of the output of the filter in Fig. 5.6 with white noise input.

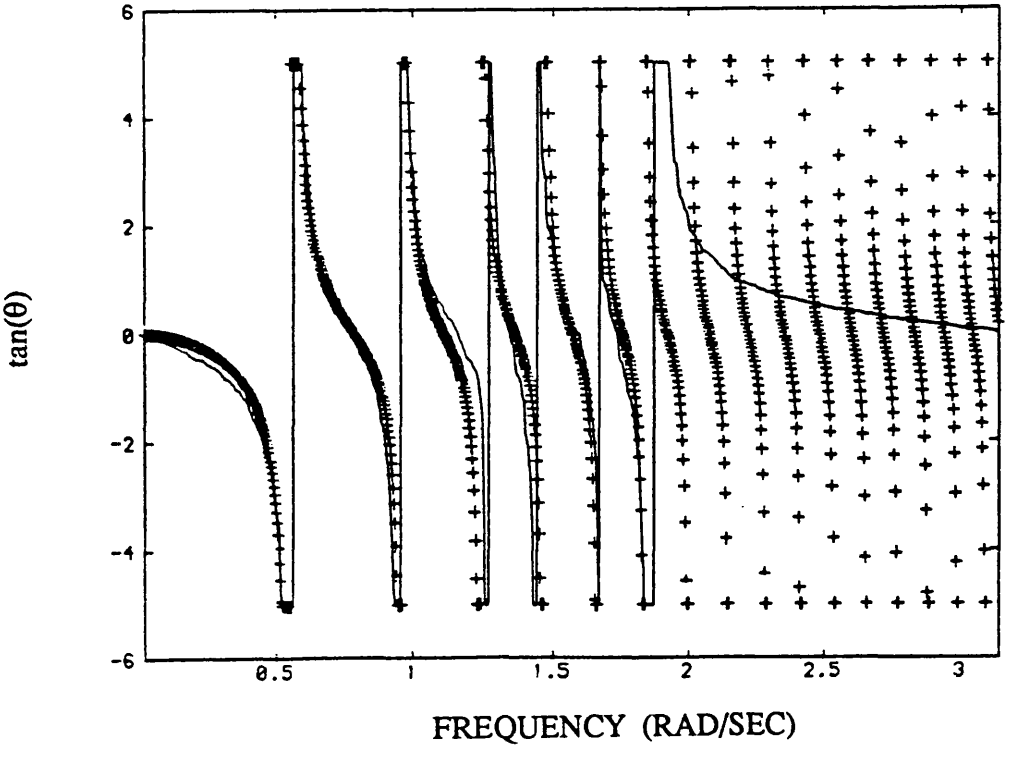


Fig. 5.8 Phase characteristics corresponding to the cross spectrum of Fig. 5.7 compared with the target.

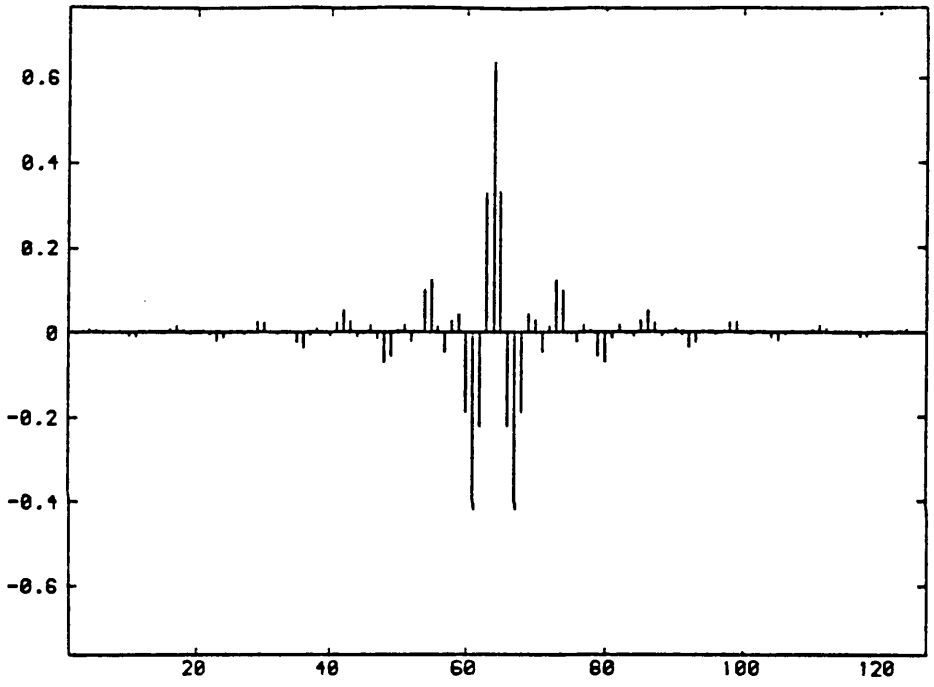


Fig. 5.9 127 coefficients of a bandpass filter, ω_b=2.0, ω_c=2.0, Kaiser window, b=2.120

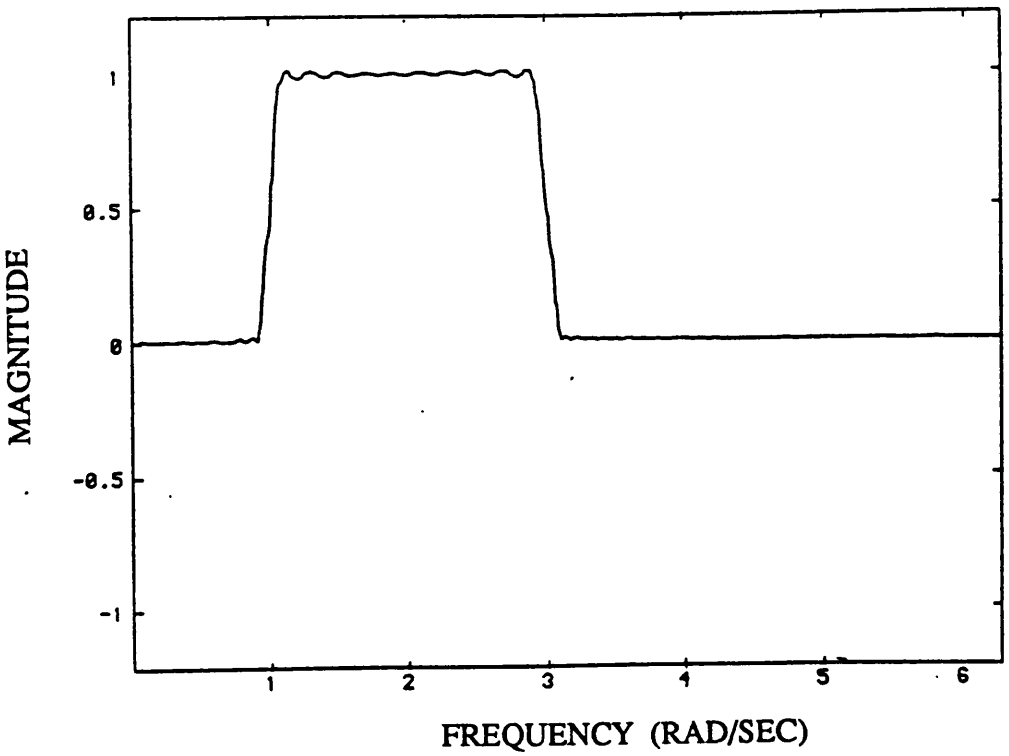


Fig. 5.10 Magnitude of the cross spectrum of the output of the filter in Fig. 5.9 with white noise input. 4098 pts FFT, 30 pts averaging.

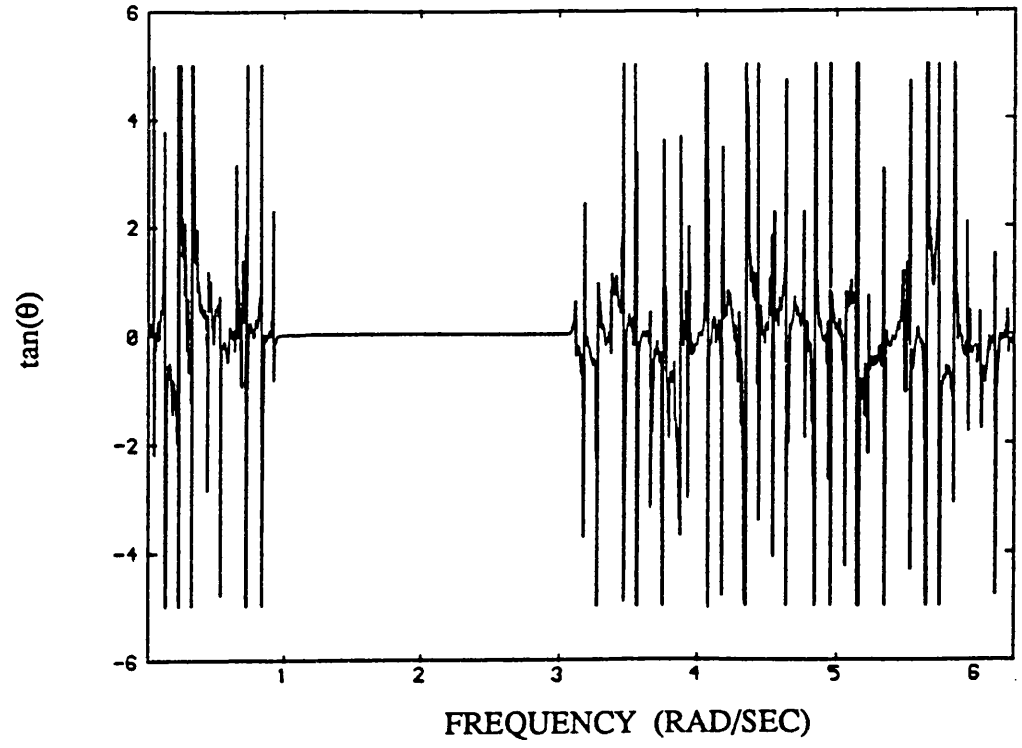


Fig. 5.11 Phase characteristics corresponding to the cross spectrum of Fig. 5.10

CHAPTER 6

DIGITAL FILTERS FOR WAVE DIFFRACTION AROUND LARGE CYLINDRES.

CHAPTER SIX

CONTENTS

6.1 Introduction.

In the design of digital filters for the simulation of wave kinematics the effect of the structure on the wave field has been so far neglected. The flow in the region near the structure was considered undisturbed. This assumption is justified in cases where the horizontal dimensions of the elements of the structure are small compared with the predominant wave characteristics. For a cylindrical element this measure can be quantified in the ratio D/L, where D is the diameter of the cylinder and L the predominant wave length; if this ratio is small, say $D/L < 0.2$, the incident waves only can sufficiently represent the wave field (Sarpkaya and Isaacson 1981). Many modern offshore structures though incorporate elements with large horizontal dimensions. In that case, the effect of the scattered waves on the modulation of the wave field in the vicinity of the structure can no longer be ignored. The aim of this chapter is to introduce an extension of the methodologies used so far, more specifically the use of digital filters, in order to include the effects of the scattered waves. This chapter should not be considered as a complete study of the subject, but rather as a demonstration of the applicability of the principles underlying this work in a similar situation.

6.2 Elements of linear wave diffraction theory.

Linear wave diffraction theory has been employed by various researchers in the past (MacCamy and Fuchs 1954), (Hogben and Standing 1974), (Isaacson 1977), (Isaacson 1978a), (Sarpkaya and Isaacson 1981) to obtain the elevation of the sea waves around large bodies. Although from the 80's onwards attention has been drawn to non-linear, usually second order diffraction theories (Eatock-Taylor and

Hung 1987), (Wu and Eatock-Taylor 1989) etc., linear theory can still be used in cases where the Keulegan-Carpenter number K is small compared to the ratio D/L (Isaacson 1978b). Due to its simplicity and the fact that for the case of circular cylinders of large diameter, vertical to the free surface, there exist analytic solutions, linear theory was adopted in the present work. An attempt to design digital filters that can incorporate the effects of the scattered waves in the simulation of the wave field will be presented.

The wave velocity potential ϕ can be considered as consisting of two elements, the incident wave potential ϕ_w and the scattered wave potential ϕ_s.

$$\phi = \phi_w + \phi_s \tag{6.1}$$

The incident wave potential has already been presented in Chapter 3, but it will be expressed here in cylindrical coordinate system as this is more convenient for the present case. For deep water the incident wave potential at a point on the sea surface, expanding the expression as an infinite series of Bessel functions, is given by the expression (Sarpkaya and Isaacson 1981):

$$\phi_w = A\left[\sum_{m=0}^{\infty} \beta_m J_m(kr)\cos(m\theta)\right]e^{-i\omega t} \tag{6.2}$$

where

$$b_m = \begin{cases} 1 & \text{for } m = 0 \\ 2i^m & \text{for } m \geq 1 \end{cases} \tag{6.3}$$

and k in the argument of the Bessel functions is the wave number and r and θ are the cylindrical coordinates of the point. The constant A, which may be considered

as the amplitude of the velocity potential, can be expressed, using linear wave theory derivations, as a function of the significant wave height H (Dean and Dalrymple 1984), (Sarpkaya and Isaacson 1981):

$$A = \frac{-igH}{2\omega} \tag{6.4}$$

A similar series expansion as in Eq. 6.2, involving Hankel functions of the first kind, can be written for the scattered waves (Sarpkaya and Isaacson):

$$\phi_s = A\left[\sum_{m=0}^{\infty} \beta_m B_m H_m^{(1)}(kr)\cos(m\theta)\right]e^{-i\omega t}$$

where B_m is a function of the radius of the cylinder:

$$B_m = \frac{-J_m'(ka)}{H_m^{(1)\prime}(ka)}$$

Combining the two potentials, the total velocity potential, for deep water, can be written as:

$$\phi = A\left[\sum_{m=0}^{\infty} \beta_m \left(J_m(kr) - \frac{J'_m(ka)}{H_m^{(1)\prime}(ka)} H_m^{(1)}(kr)\right)\cos(m\theta)\right]e^{-i\omega t} \tag{6.5}$$

where prime denotes derivative with respect to the argument, and $a = D/2$ is the radius of the cylinder.

The wave elevation η is extracted from the expressions for the velocity potential as a boundary condition for the Laplace equation it must satisfy:

$$\eta = -\frac{1}{g}\left(\frac{\partial \phi}{\partial t}\right)_{z=0} \tag{6.6}$$

Substituting Eqs. (6.5) and (6.4) into Eq. (6.6), the total elevation, due to incident and scattered waves, at a distance r from the centre of a cylinder vertical the sea surface with radius a is given as:

$$\eta = \frac{H}{2}\left[\sum_{m=0}^{\infty} \beta_m \left(J_m(kr) - \frac{J'_m(ka)}{H_m^{(1)\prime}(ka)} H_m^{(1)}(kr) \right) \cos(m\theta) \right] e^{i\omega t} \tag{6.7}$$

The above equation will be used as the basis for the derivation of a transfer function for the scattered waves in the following section.

6.3 The transfer function for the scattered waves.

In keeping with the logic of Chapter 3 the Transfer function for the wave elevation can be derived from the above relation: If a wave of significant height H is "generated" at the centre of a large cylinder of radius a, the elevation at a point with cylindrical coordinates (r, θ) can be found as the response of a linear system with transfer function:

$$H(\omega) = \left[\sum_{m=0}^{\infty} \beta_m \left(J_m\left(\frac{\omega^2 r}{g}\right) - \frac{J'_m(\omega^2 a/g)}{H_m^{(1)\prime}(\omega^2 a/g)} H_m^{(1)}\left(\frac{\omega^2 r}{g}\right) \right) \cos(m\theta) \right] \tag{6.8}$$

where use of the dispersion relationship $k = \omega^2/g$ has been made. It can easily be seen that the above transfer function consists of two parts; an expression that represents the incident waves and a counterpart for the scattered waves:

$$H(\omega) = H_w(\omega) + H_s(\omega) \tag{6.9}$$

Separating the transfer function for the scattered waves only, yields the series expression:

$$H_s(\omega) = -\left[\sum_{m=0}^{\infty} \beta_m \frac{J'_m(\omega^2 a/g)}{H_m^{(1)\prime}(\omega^2 a/g)} H_m^{(1)}\left(\frac{\omega^2 r}{g}\right) \cos(m\theta) \right] \tag{6.10}$$

which will be the "target" function in the design of digital filters in the next section.

The separation of the transfer function into two parts is advantageous in the present study, where filters for the incident waves have already been designed. The elevation at a point (r, θ) can be initially generated using the existing horizontal propagation filters; the contribution of the scattered waves can be superimposed consecutively.

The presence of a large vertical cylinder in an example sea-grid (see Chapter 7) is depicted in Fig. 6.0.; r and θ_i are the polar coordinates of the point where the elevation of the scattered waves is obtained (Eg. point 3 in the drawing), a is the radius of the cylinder. The angle θ is measured in reference to a particular wave propagation direction i or j. According to this scheme the total elevation at a grid point, due to incident and scattered waves, from all directions, is given as:

$$\eta(x,y,t) = \sum_i \{\eta_{w_i}(x,y,t) + \eta_{s_i}(r,\theta_i,t)\}$$

where (x, y) are the coordinates of the grid point measured from the "reference" point and (r, θ_i) its polar coordinates in reference to the cylinder and the wave direction i.

The expression in Eq. (6.10) can be further simplified; the derivatives with respect to the argument of the Bessel and Hankel functions can be obtained from the expressions (Abramowitz and Stegun 1970):

$$J_n'(x) = J_{n-1}(x) - \frac{n}{x} J_n(x) \qquad (6.11)$$

$$H_n'(x) = H_{n-1}(x) - \frac{n}{x} H_n(x)$$

The existence of recursive relations for the Bessel and Hankel functions themselves allows the generation of higher orders from only the zero-th and first order functions. In practice though, a few terms of the series expansion need be used. It can be seen that the Bessel functions of high orders have significant values for large arguments only as follows from their asymptotic expressions for large orders (Abramowitz and Stegun 1970):

$$J_n(x) \approx \frac{1}{\sqrt{2\pi n}}\left(\frac{ex}{2n}\right)^n \quad \text{and} \quad Y_n(x) \approx -\sqrt{\frac{2}{\pi n}}\left(\frac{ex}{2n}\right)^{-n} \tag{6.12}$$

Consequently if

$$x \ll 2n/e \quad \Rightarrow \quad J_n \sim 0 \quad \text{and} \quad 1/Y_n \sim 0$$

In the present study where the frequency band of interest extends to π, and in some cases to 2π, and for cylinders of diameter D up to two meters, it was found that the first five orders of the series are adequate to express the transfer function. In Figs (6.1, 6.2) the frequency domain characteristics of the transfer function of Eq. (6.10) are shown for five and ten terms, for two different cases. It is seen that in both graphs the differences between five and ten term series are insignificant for frequencies up to 2π.

6.4 Design of digital filters for the superposition of the scattered waves.

Two methods can be considered to extract a digital filter from the aforementioned transfer function. One is to derive the analytical, infinite duration, impulse response function (as an inverse Fourier transform of Eq. (6.10)), truncate it and

consequently digitize it. This was the method followed in previous chapters. The alternative is to sample the Transfer function and use numerical inverse Fourier Transforms to obtain the time domain representation.

In the present case the theoretical frequency domain expression is given as a series expansion, and only a few terms from this expansion are retained. The result is an expression which is accurate only at small values of ω as shown in the previous section, acceptable though for the purposes of this design. This fact itself imposes a windowing in the frequency domain. Thus it was thought appropriate, and convenient, to sample directly the function in the frequency domain and following that, use IDFT to obtain the digitized impulse response function (FIR). The resulting series can itself be truncated for values less than a percentage (Eg. 1/100th) of the maximum ordinate. In Figs (6.3, 6.4) two such FIR filters are shown, that correspond to cases shown in Figs (6.1, 6.2). It is seen that the orders of the filters are quite low (18 and 30 respectively).

The frequency domain qualities of these filters can be examined using the same methods as in chapters 4 an 5: the magnitude and phase that correspond to these FIR functions are compared with the "ideal" values, for the frequency band of interest. Of course in this case the ideal functions are given by the full series expansion of Eq. (6.10). In Figs.(6.5 and 6.6) these comparisons are shown, for frequencies up to 2π

The design of filters for the scattered waves over the whole area around a large cylinder would be very tedious. For the unidirectional case for each and every location (r, θ) around the cylinder a separate filter would have to be implemented. The problem is even more pronounced in the multidirectional case where the angle

varies with the wave direction as well (see Fig. 6.0). A significant reduction of the required effort can be achieved by the following simplification: Eq. (6.10) can be written as:

$$H_s(\omega) = H_0(\omega) + H_1(\omega)\cos(\theta) + H_2(\omega)\cos(2\theta) + .. \qquad (6.13)$$

where $H_i(\omega)$ is the i-th order term of the series expansion. From the properties of Fourier Transforms, a corresponding relation in the time-domain can be easily written:

$$h_s(nT) = h_0(nT) + h_1(nT)\cos(\theta) + h_2(nT)\cos(2\theta) + .. \qquad (6.14)$$

where $H_i(\omega) \Leftrightarrow h_i(nT)$ are Fourier transform pairs. The FIR filters $h_i(nT)$ $i = 1..5$ for a particular radial distance r from the centre of a cylinder are produced and stored once, and then for each angle of orientation θ the full FIR filter is constructed according to Eq. (6.14). Truncation of the insignificant ordinates occurs at this latest stage (instead of the individual $h_i(nT)$ $i = 1..5$). In Figs. (6.7, 6.8) the quality of two filters constructed according to this procedure is shown.

The design of the filters for the scattered waves was left at this stage (FIR filters). No effort to reduce further their order by constructing their recursive counterparts was made, since the filters are generally acceptably short, and since the techniques for accomplishing this task were demonstrated earlier in this thesis.

6.5 Wave runup around large cylinders using the digital filters.

The validity of the digital filtering methods discussed here can be further demonstrated by evaluating the total wave runup around a large vertical cylinder that would be produced if these filters were used. Theoretical expressions for the wave runup can be found in the literature. and comparisons can be made.

The wave runup $R(\theta)$ is defined as the sea surface elevation on the periphery of a cylinder, usually normalized by the significant wave height H. Isaacson (1978) and Sarpkaya and Isaacson (1981) give the expression:

$$\frac{R(\theta)}{H} = \left| \sum_{m=0}^{\infty} \frac{\beta_m \cos(m\theta)}{2\pi k a H_m^{(1)\gamma}(ka)} \right| \tag{6.15}$$

where the notation is the same to the one used in previous equations in the same chapter and θ is the angle with respect to the wave direction. This relation can be easily derived from Eq. 6.7, after some manipulation. The maximum runup occurs in the back of the cylinder $\theta = 180^0$ according to this relation.

In addition to this series expansion, Isaacson gives an approximate expression, which comes from the previous formula retaining only up to second order terms:

$$\frac{R(\theta)}{H} = \frac{1}{2}\left\{1 + (ka)^2\left[2\cos^2(\theta) + \ln(ka/2) + \frac{2\gamma - 1}{2}\right]\right\}^{1/2} \tag{6.16}$$

where $\gamma = 0.5772..$ (Euler's constant). From this the maximum runup is easily shown as (Isaacson 1987):

$$\frac{|R|}{H} = \frac{1}{2}\left\{1 + (ka)^2\left[\ln(ka/2) + \frac{2\gamma + 3}{2}\right]\right\}^{1/2} \tag{6.17}$$

It should be noted that this expression is only valid for very small ka as will become apparent from the figures presented below.

From the previous relations it is obvious that the wave runup is closely associated with the transfer function for the total wave elevation on the periphery of a cylinder (Eq. 6.7). In fact the same conclusion would have been drawn from the relation:

$$y(t) = H(\omega)x(t)$$

which was presented in chapter 3. The transfer function that corresponds to the digital filters for the scattered waves, which were designed in this chapter, and the same for the incident waves, can be easily evaluated. Thus the runup profile can be obtained without actually generating time series of the elevation. This profile can then be compared with the theoretical values presented earlier. In Figs 6.9a and 6.9b the runup obtained from the digital filters is shown as a function of the angle of orientation θ and compared with the theoretical curve and Isaacson's approximation. The runup which is resulting from fifty digital filters (one for each angle considered), with lengths varying from 6 to 40 coefficients, is presented in these Figures, for two different diffraction parameters ka. In Fig. 6.10 the maximum runup profile, at $\theta = 180^0$, is shown as a function of ka for filters corresponding to three different cylinders.

Although the case chosen in this chapter to demonstrate the applicability of the transmission filters is quite simple, the very good agreement of the results with known solutions shown in the previous figures indicates that this method could be extended to more complicated cases, such as large bodies of arbitrary shape.

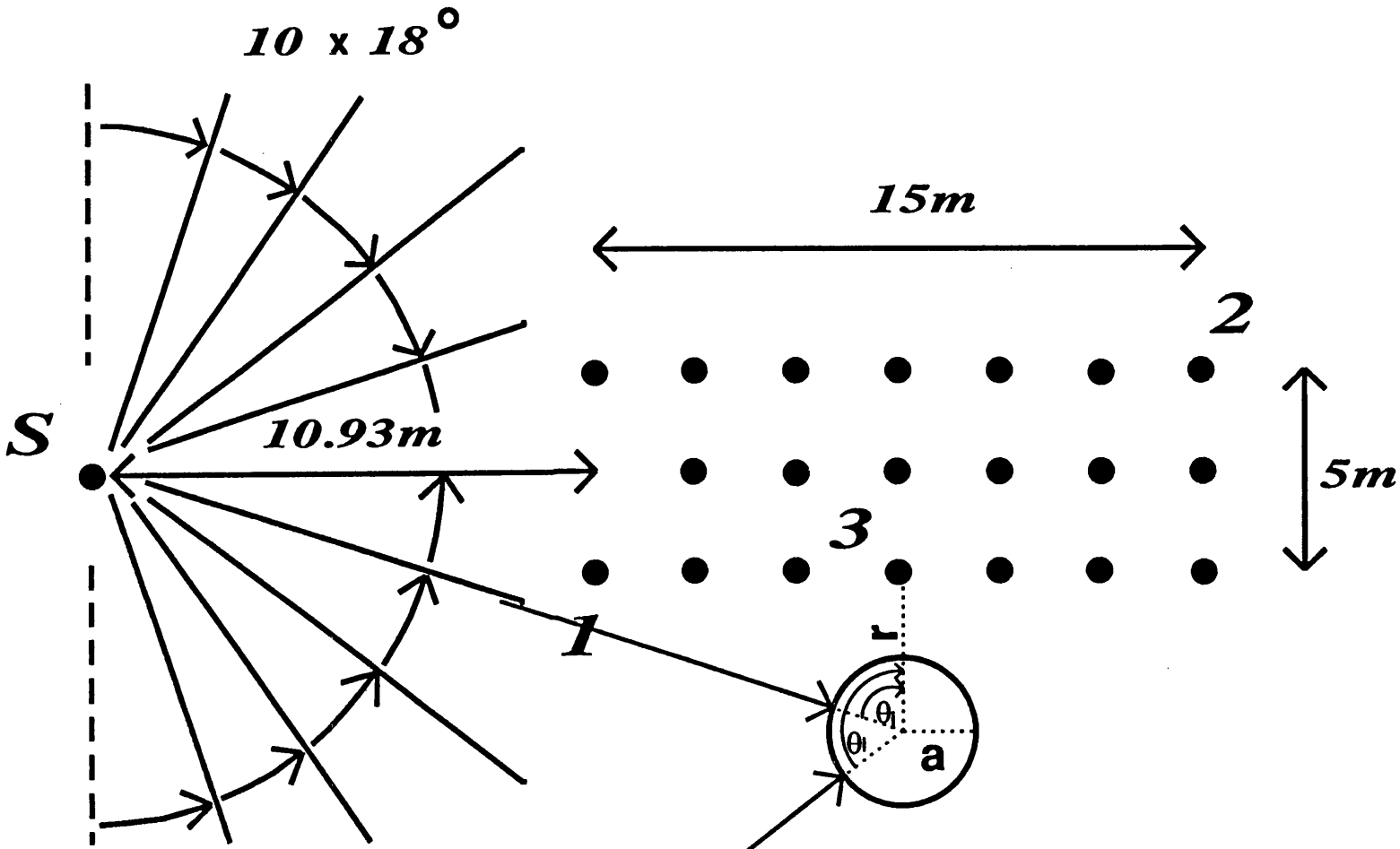


Fig. 6.0 Sea grid and a large vertical cylinder

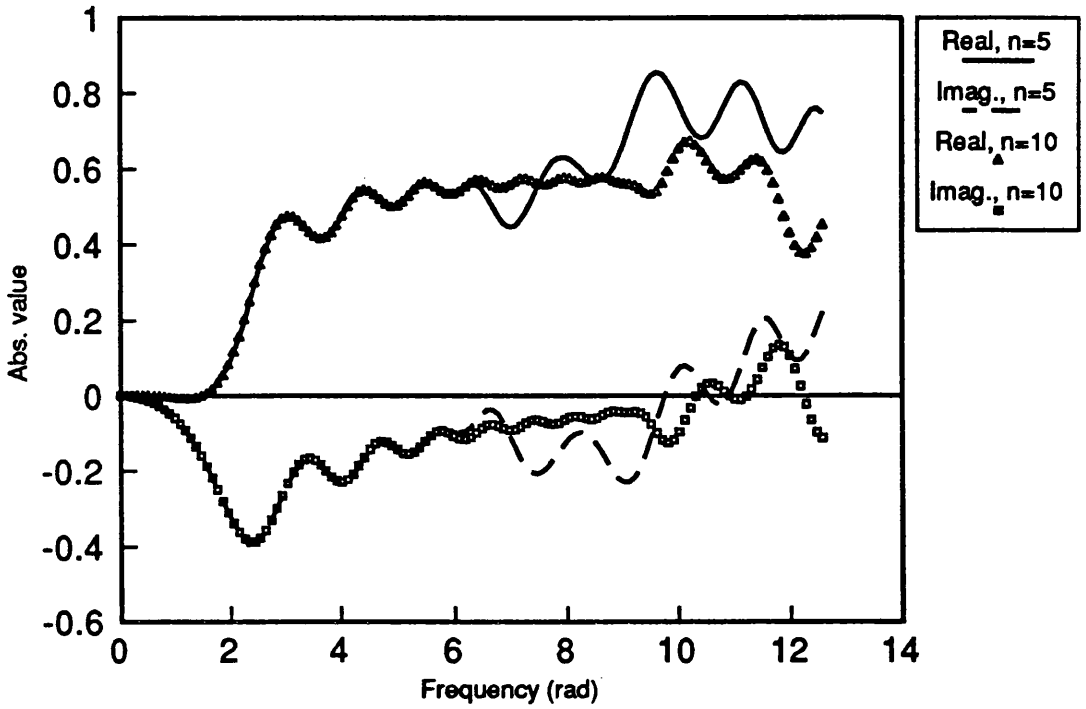


Fig. 6.1 Real and Imag. parts of H(w) (Scaterred)

a = 1(m), r = 2(m), $\theta=\pi$(rad)

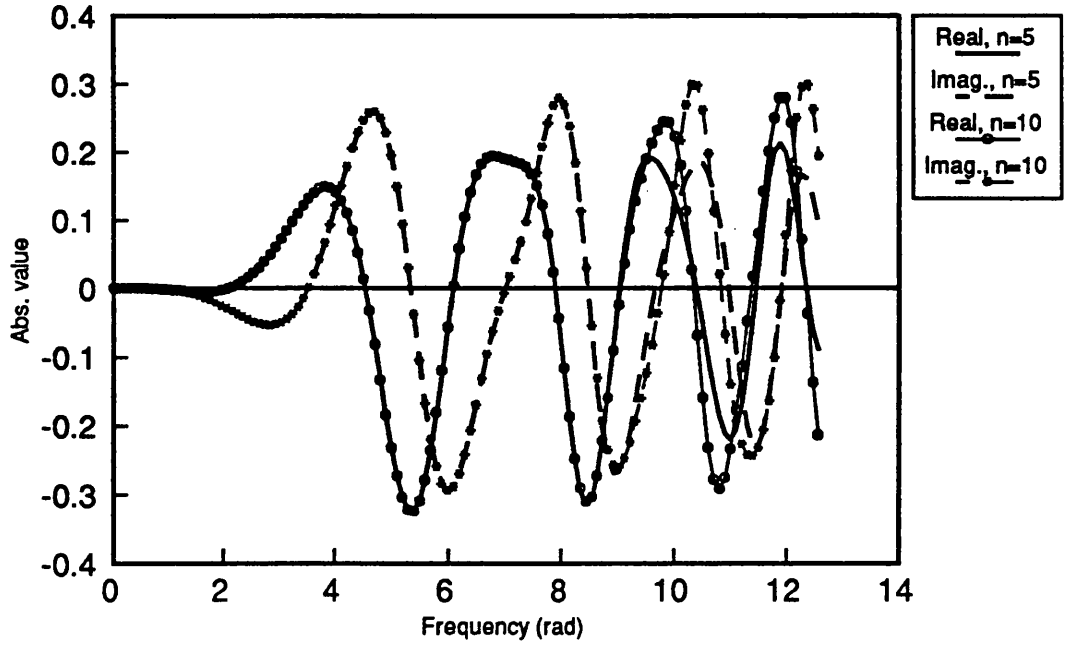


Fig. 6.2 Real and Imag. parts of H(w) (Scaterred)

a = 0.5(m), r = 2(m), $\theta=\pi/2$ (rad)

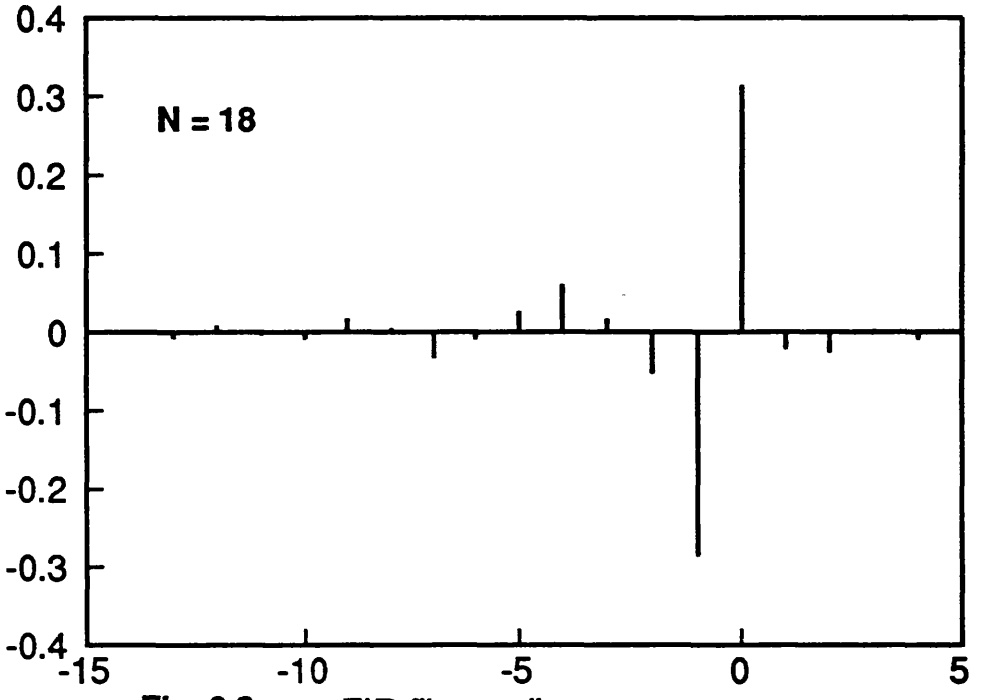


Fig. 6.3 FIR filter ordinates

$a = 1(m), r = 2\ (m), \theta = \pi\ (rad)$

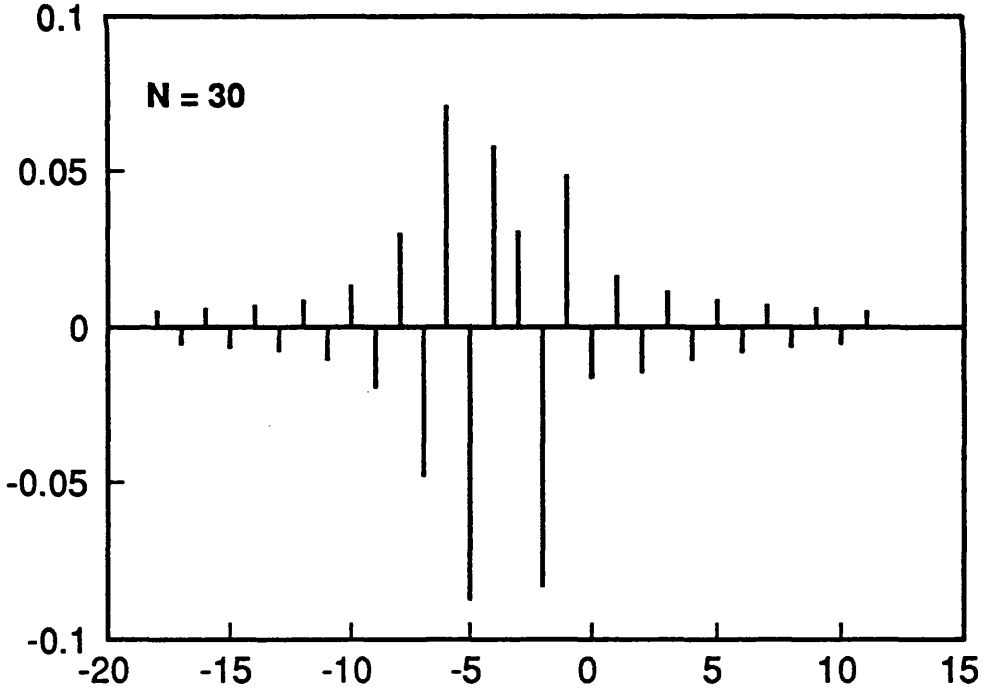


Fig. 6.4 FIR filter ordinates

$a = 0.5(m), r = 2\ (m), \theta = \pi/2\ (rad)$

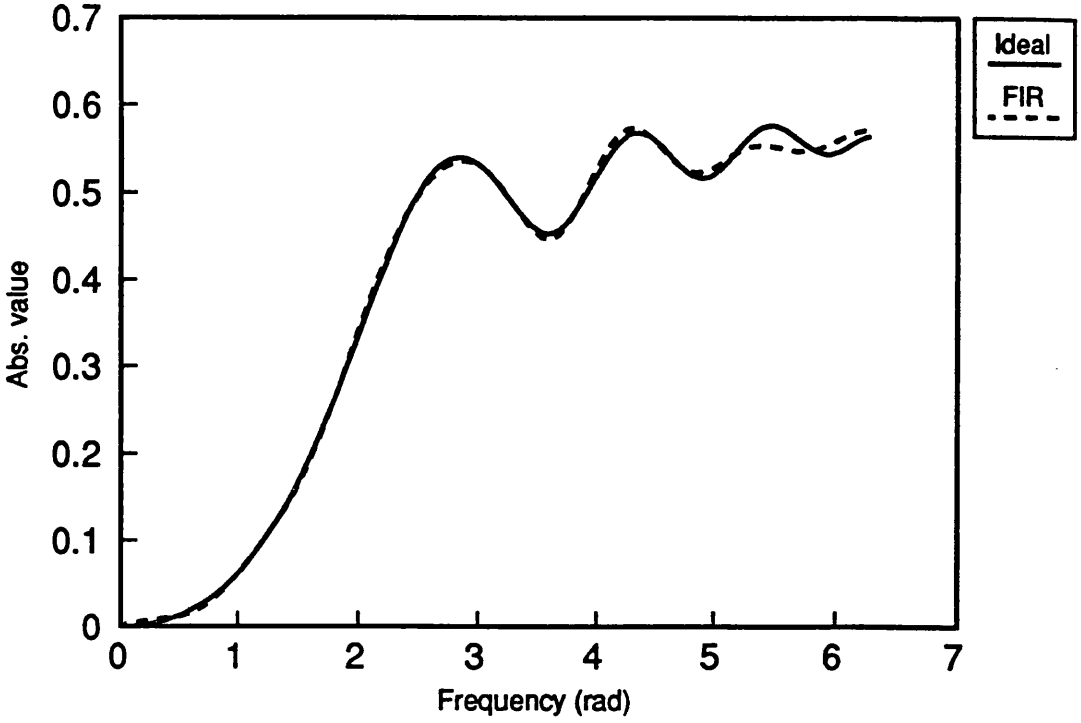


Fig. 6.5a Magnitude of H(w) for FIR of fig 6.3

a = 1(m), r = 2(m), θ=π(rad)

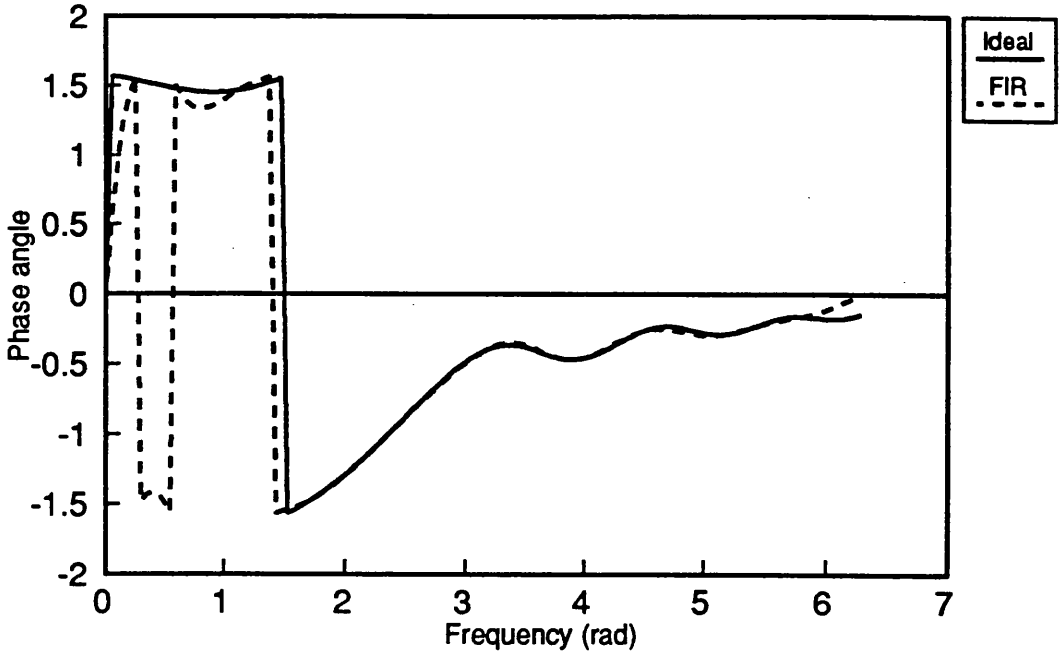


Fig. 6.5b Phase angle of H(w) for FIR of fig 6.3

a = 1(m), r = 2(m), θ=π (rad)

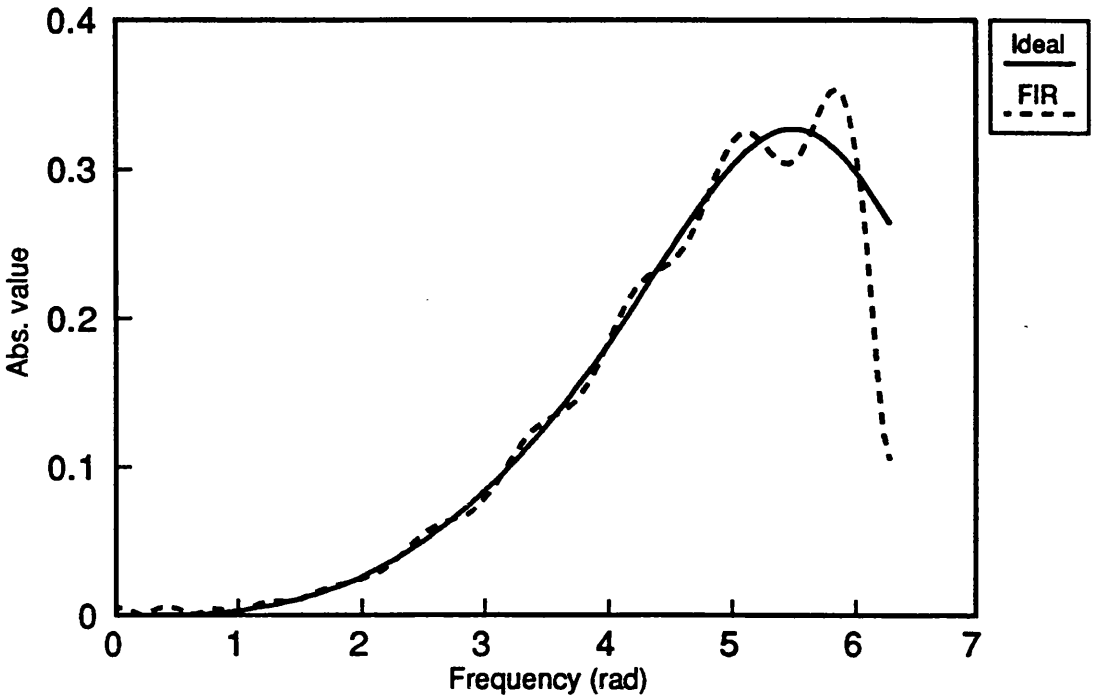


Fig. 6.6a Magnitude of H(w) for FIR of fig 6.4

a = 0.5(m), r = 2(m), $\theta = \pi/2$ (rad)

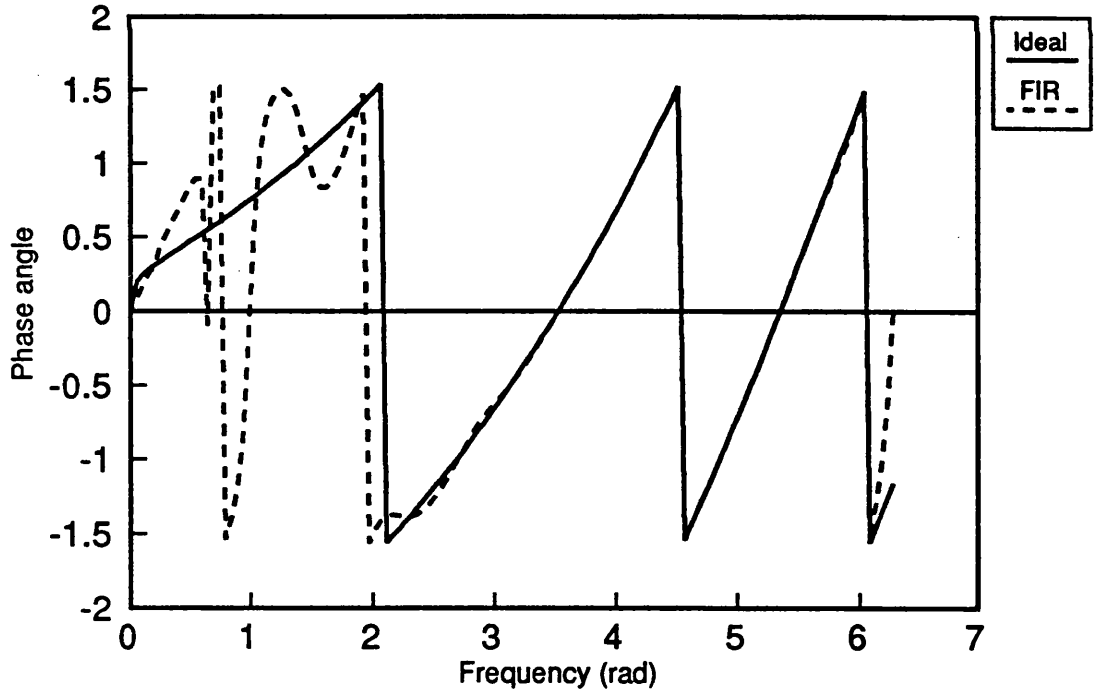


Fig. 6.6b Phase angle of H(w) for FIR of fig 6.4

a = 0.5(m), r = 2(m), $\theta = \pi/2$ (rad)

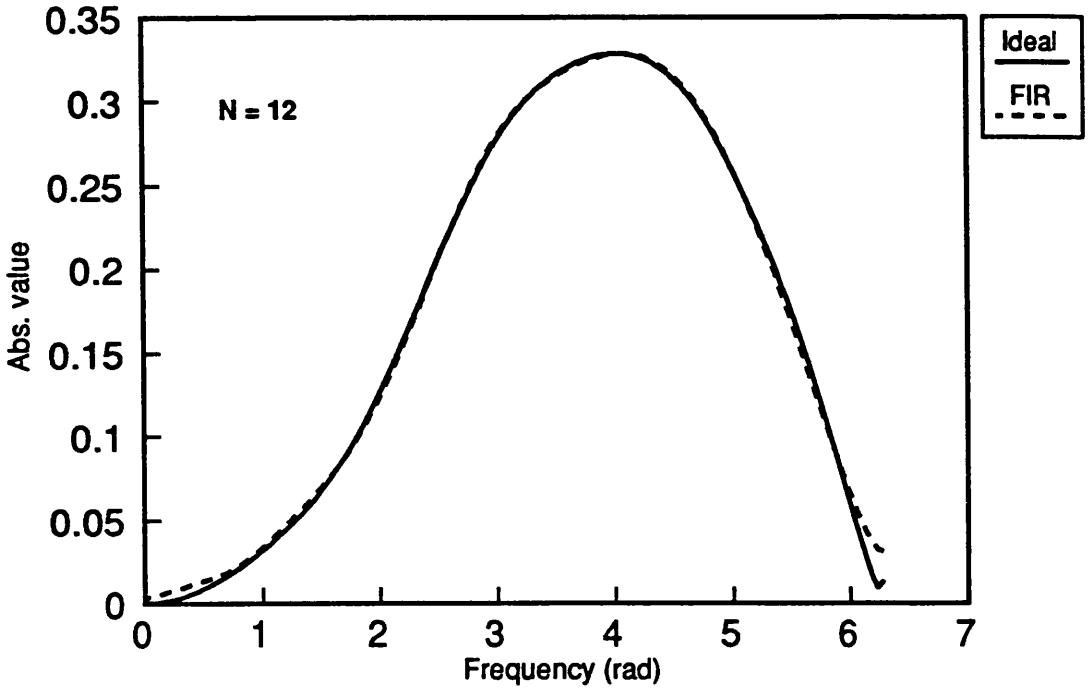


Fig. 6.7a Magnitude of H(w)
a = 1(m), r = 2(m), $\theta = \pi/4$ (rad)

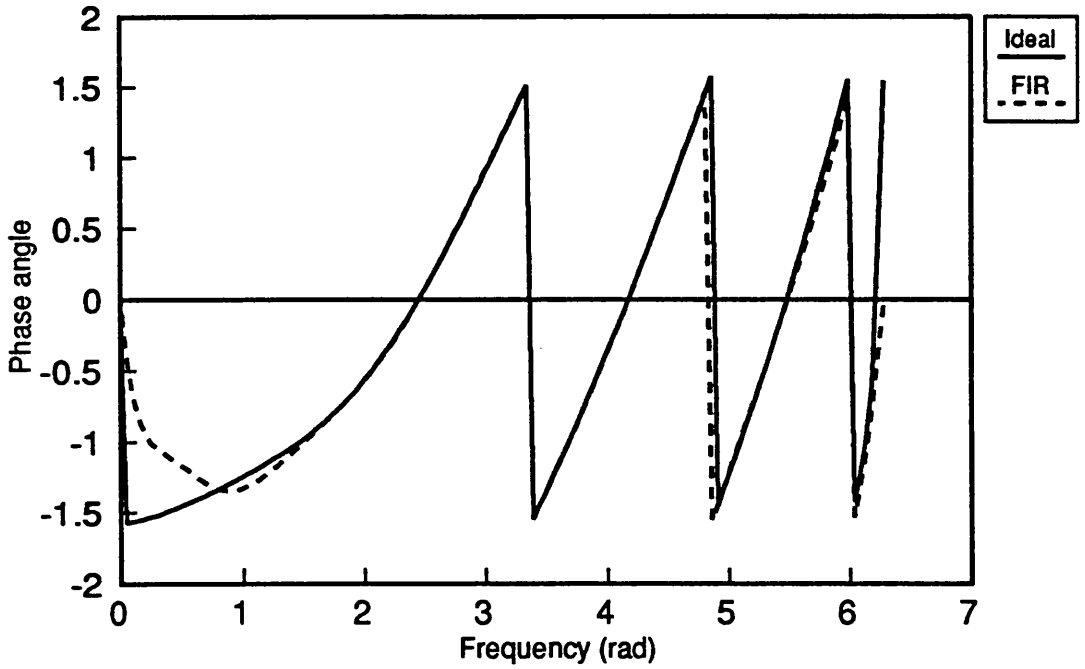


Fig. 6.7b Phase angle of H(w)
a = 1(m), r = 2(m), $\theta = \pi/4$ (rad)

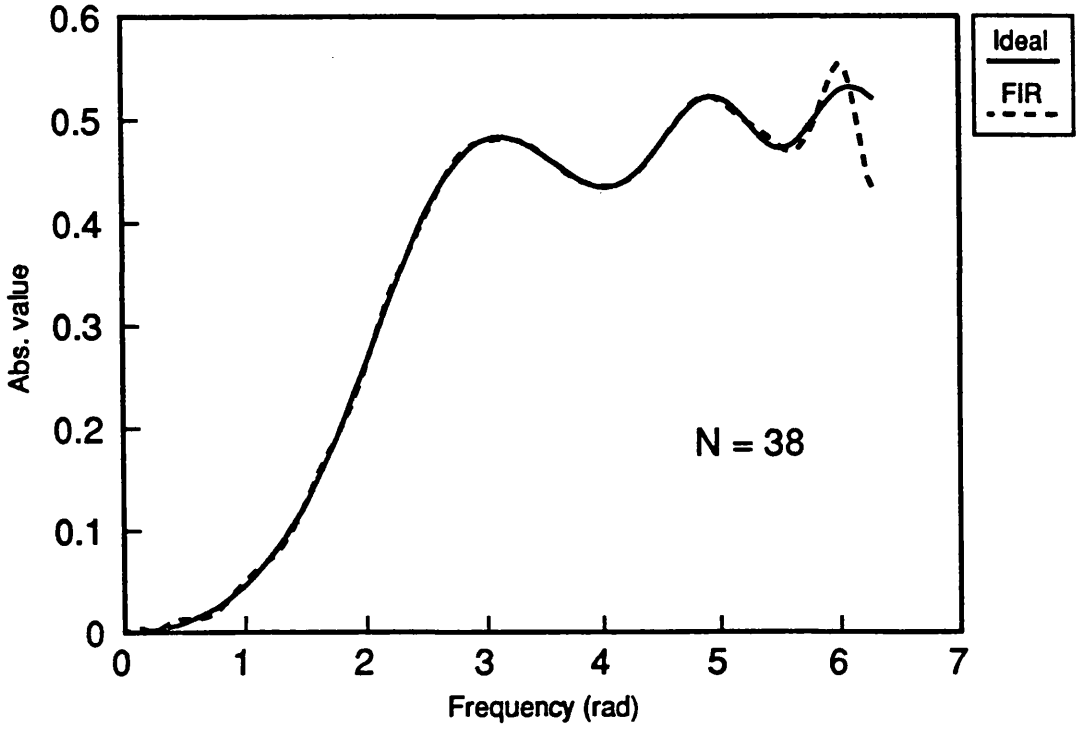


Fig. 6.8a Magnitude of H(w)

a = 1(m), r = 2(m), $\theta = 3\pi/4$ (rad)

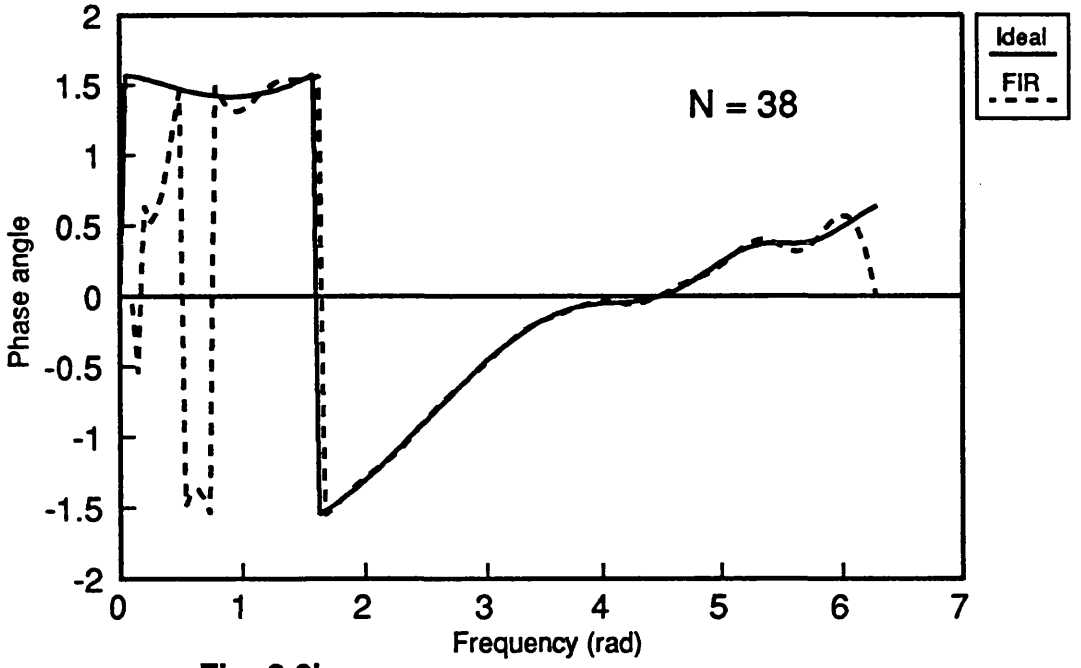


Fig. 6.8b Phase angle of H(w)

a = 1(m), r = 2(m), $\theta = 3\pi/4$ (rad)

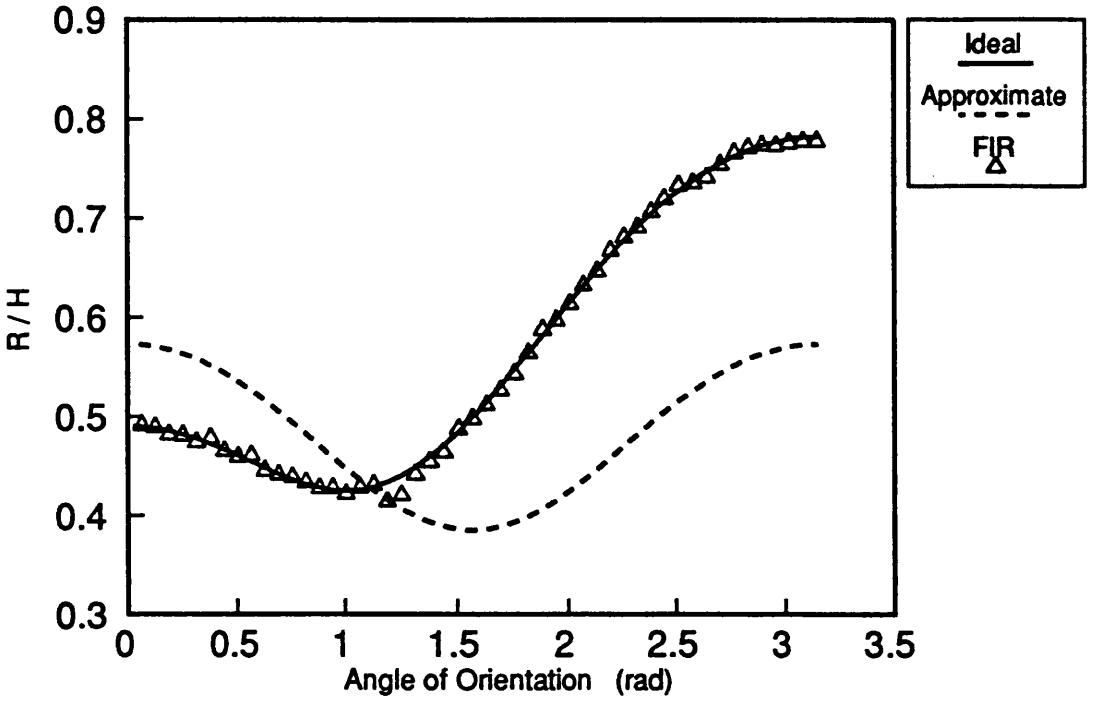


Fig. 6.9a Runup Profiles R(θ)/H
ka = 0.6

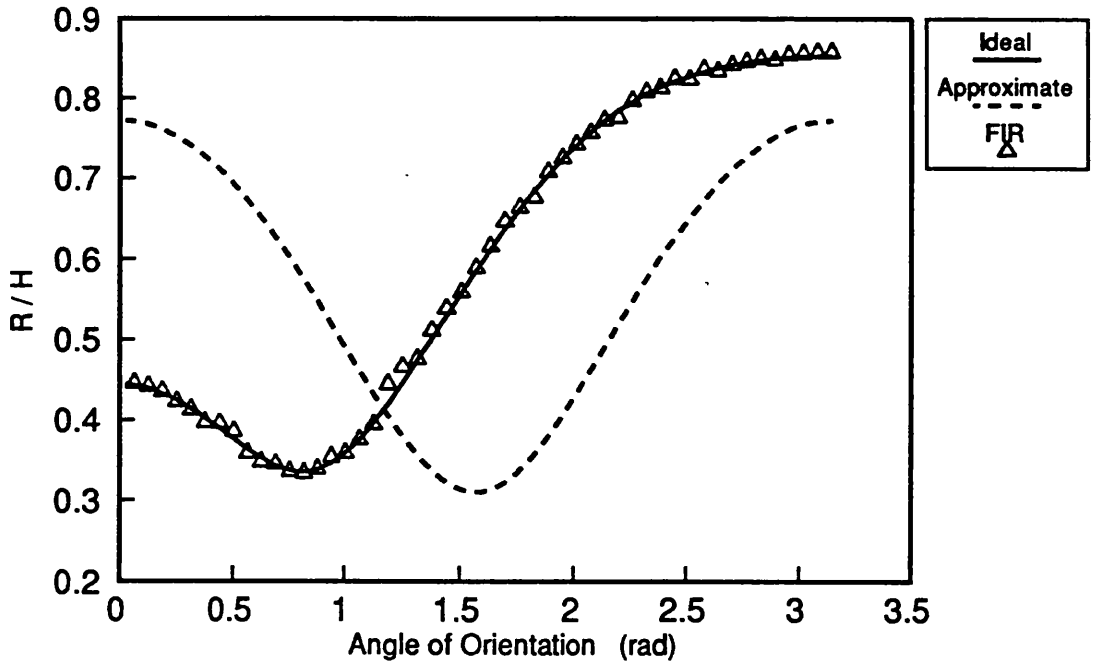


Fig. 6.9b Runup Profiles R(θ)/H
ka = 1.0

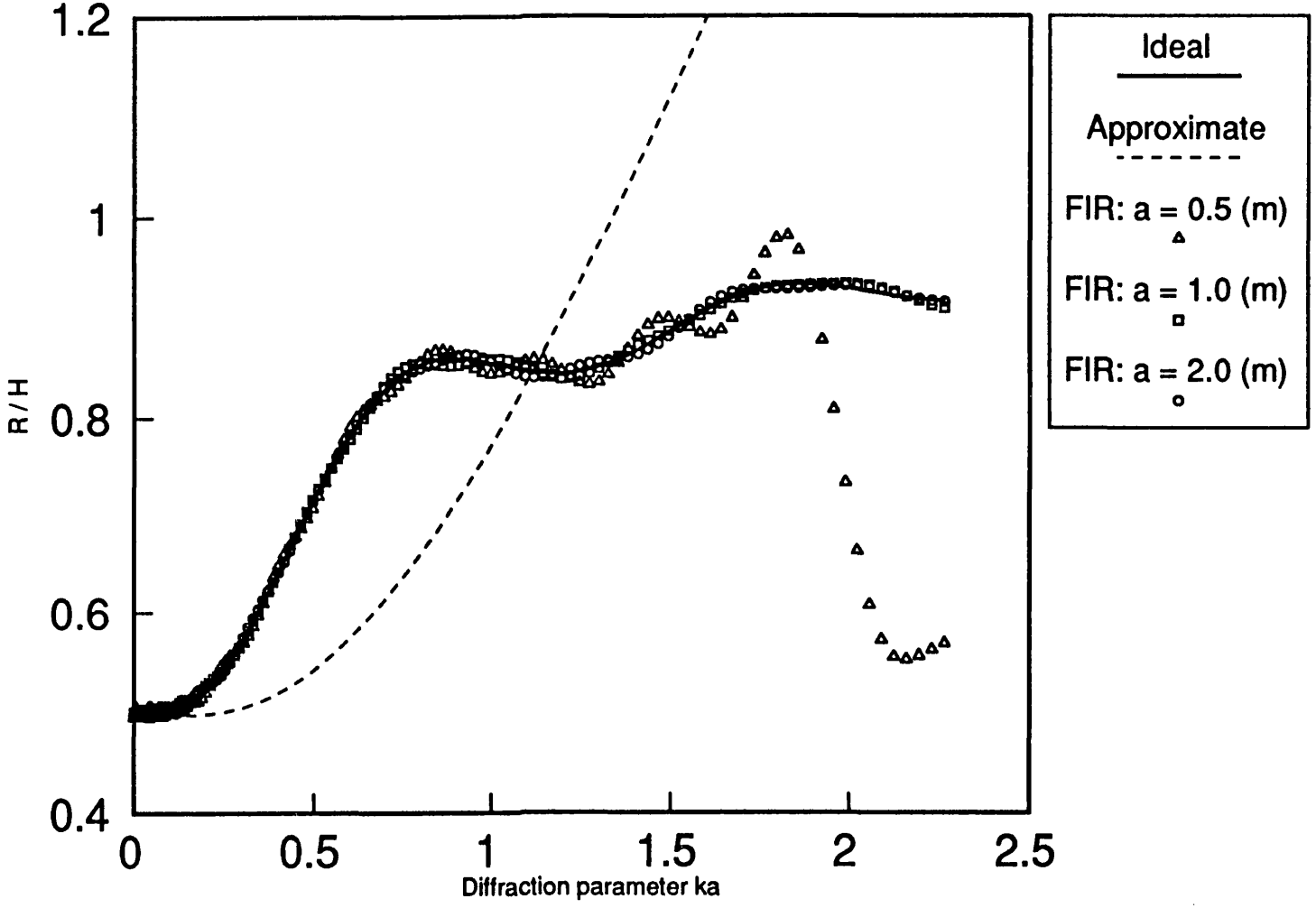


Fig. 6.10 Maximum Runup Profiles R/H

CHAPTER 7

PROGRAM IMPLEMENTATION AND QUALITY VERIFICATION.

CHAPTER SEVEN

CONTENTS

7.1 Introduction.

The aim of this chapter is to show the quality of the simulated time series in terms of their second order statistical properties. The unidirectional and the multidirectional cases will be examined separately. For the unidirectional case the higher order moments will be also evaluated and compared with their theoretical values. For the multidirectional case cross-statistics between kinematic quantities at different locations will be given.

A brief documentation of the (sequential) program used will be first presented in tems of its main subroutines, as well as its flow chart. The parallel implementation and studies on its performance will be examined in a subsequent chapter.

7.2 Brief Program Documentation.

In the begining of this chapter a guide to the principal components of the time series generation program is given. A flow chart of this program is shown in Fig. 7.1.

7.2.1 Control routine.

Subroutine CHOICE is the routine that provides the user interface. All the control parameters, like seastate, distances from the source point, depth for vertical attenuation etc., are input through this routine. It should be noted that the distances from the source are restricted to a finite set.

7.2.2 Routines for the preparation of filters.

A series of subroutines prepare the filter coefficients that will be used in the generation process.

Subroutine READEL will open the file containing the coefficients of the recursive filters designed for the horizontal propagation of wave motion. The filters are contained in two files, and in the thesis are given in Appendix B, one for each of the two frequency bands considered (π or 2π). The user specifies which file will be used. An error message will be produced if one of the distances of the horizontal propagation as input by the user does not correspond to any of the filters contained in the file.

Subroutine ARMCO will prepare the coefficients of the ARMA filter that is used to generate free surface elevation at the source point. The sea-state (wind speed in knots) is the input to this routine. The design follows Spanos (1983)

Subroutine DIFFER prepares the coefficients of the two differentiators that will be used to generate velocity from elevation records and acceleration from velocity.

Subroutine VERTI finally prepares the filter used for depth attenuation. The distance below the surface z is the input to this routine.

7.2.3 Generation routines.

Functions TRPNRM and UNIF generate a white noise ordinate with Normal p.d.f and unit variance, by the method of Ahrens and Dieter (Section 3.1). The calling argument to UNIF is an integer seed which in this application is provided by an intrinsic function which reads the time from the computer's clock.

Subroutine ELEV when called generates elevation at the source point, plus elevation records at all the grid points on the surface and at depth z. The records are stored in output files in the hard disk, for possible statistical post-processing. Because the generation is continuous, at each time step the corresponding ordinate of all time histories under consideration is produced and written on consecutive lines before proceeding to the next time instant. Because non-causal filters are used, the filter output is not synchronous with the input time series, but lags it by the width of the anticipatory part of the filter. These lags are written on the first lines of the results file are taken into account in any subroutines using the time series. Such a subroutine is DTANGL which produces statistical moments (see below).

Subroutine UWUDWD generates the complete water kinematics, that is elevation, horizontal and vertical water particle velocity and horizontal and vertical particle acceleration time series at all the grid points considered. The generation is continuous, therefore the same organization of the output files as in ELEV is followed. Each line in the file contains the ordinates of the five time histories at one instance, at one grid point; the corresponding ordinates for all grid points on one vertical axis are generated and written on consecutive lines before continuing with the next time step. The appropriate time lags are written in the beginning of the file.

It should be noted that the output files produced by UWUDWD may be very large. As a yardstick, 5000 time steps at two grid points will produce two files in excess of 800K each. This problem is remedied in the parallel version by buffering large amounts of the generated data at run time and processing parts of it in parallel (see Chapter 8).

7.2.4 Routine for calculating the statistics of the output.

Subroutine DTANGL, if activated by the user, will open the output files in order to read the generated time series, perform simple statistics on them and compare them with the corresponding theoretical values. The results are reported in output files

7.3 Quality verification - Unidirectional case

As a verification of the quality of the simulation certain statistics of the generated time series have been compared with their target values.

The statistics which have been evaluated are the first four standardized moments of the generated time series; for n=1,2,3,4 that is the mean, variance, third and fourth moments:

$$\mu = \frac{\sum_{k=1}^{K_{total}} x}{K_{total}} \tag{7.1}$$

$$\sigma^2 = \frac{\sum_{k=1}^{K_{total}} x^2}{K_{total}} - \mu^2$$

$$m_3^* = \frac{\sum_{k=1}^{K_{total}} x^3}{K_{total}} / \sigma^3$$

$$m_4^* = \frac{\sum_{k=1}^{K_{total}} x^4}{K_{total}} / \sigma^4$$

The variance of the elevation records at the surface grid points is compared with the zero-th moment of the target power spectral density function (Pierson-Moskowitz):

$$\sigma^2 = \frac{A}{4B} \tag{7.1}$$

where:

$$A = 8.1 \times 10^{-3} 2\pi g^2$$

$$B = 0.74\left(\frac{g}{U_{wind}}\right)^4$$

where U_{wind} is the wind speed in m sec^{-1}. Additionally, the zero-th moment of the P.S.D of the output of the ARMA filter at the source point to white noise is calculated from the filter coefficients. These three measures of the variance (Pierson-Moskowitz, ARMA model, and from the generated time series) are tabulated in Tables 7.1 and 7.3 as σ^2_{PM} , σ^2_{ARMA} , σ^2_{SIM} respectively. The variance of the other time series (velocity and acceleration records at all points as well as elevation at points below the surface) are compared with the zero-th moments of the corresponding P.S.D.s. The latter have been calculated by numerical integration from the theoretical spectrum of the ARMA model for elevation, multiplied by the squared magnitude of the corresponding transfer function. In Tables 7.1, 7.3 these variances are shown as

$\sigma^2_{V,ARMA}$, $\sigma^2_{V,SIM}$, $\sigma^2_{A,ARMA}$, $\sigma^2_{A,SIM}$

Since the generated time series are expected to be zero-mean stationary random processes with Gaussian p.d.f, the correct values for the remaining statistics are: mean=0, third moment=0 and fourth moment=3. Tables 7.2 and 7.4 list the values actually attained.

The figures given in Tables 7.1-7.4 were obtained from a simulation run of length 5000 points, using $\omega_c = \pi$, L_1=10 m , L_2= 20 m , depth z=15 m and wind speed U_{wind}=40 knots.

Table 7.1 COMPARISON OF VARIANCES AT GRIDPOINT 1

TARGET VARIANCES

	Elev.	Veloc.	Accel.
P.M., surface	10.2231		
ARMA, surface	9.5022	2.6150	1.1891
ARMA, depth 2	4.6408	9.6777E-001	2.5718E-001

VARIANCE OF THE TIME SERIES

	Elev.	x-Veloc.	z-Veloc.	x-Accel.	z-Accel.
surface	9.8007	2.6404	2.7200	1.1995	1.2152
depth 2	4.7733	9.7261E-01	1.0146	2.5510E-01	2.6431E-01

Table 7.2 VERIFICATION OF 1st, 3rd, AND 4th MOMENTS AT GRIDPOINT 1

	Elev.	x-Veloc.	z-Veloc.	x-Accel.	z-Accel.
MEAN OF THE TIME SERIES					
surface	3.2603E-04	-7.2309E-04	1.2719E-03	-3.2272E-05	-1.4636E-04
depth 2	-1.7481E-03	1.9221E-04	1.1508E-05	1.7558E-04	3.9459E-05
SKEWNESS OF THE TIME SERIES					
surface	-6.0640E-02	-3.5534E-02	-9.5401E-02	9.6478E-02	-5.2310E-02
depth 2	-3.4865E-02	-3.6582E-04	-4.4289E-02	6.6154E-02	-1.9695E-02
KURTOSIS OF THE TIME SERIES					
surface	3.2374	3.1419	3.1731	3.0828	3.0607
depth 2	3.2183	3.2177	3.2221	3.1767	3.1709

Table 7.3 COMPARISON OF VARIANCES AT GRIDPOINT 2

TARGET VARIANCES

	Elev.	Veloc.	Accel.
P.M., surface	10.2231		
ARMA, surface	9.5022	2.6150	1.1891
ARMA, depth 2	4.6408	9.6777E-001	2.5718E-001

VARIANCE OF THE TIME SERIES

	Elev.	x-Veloc.	z-Veloc.	x-Accel.	z-Accel.
surface	9.7291	2.6193	2.6972	1.1918	1.2075
depth 2	4.7432	9.6445E-01	1.0060	2.5274E-01	2.6180E-01

Table 7.4 VERIFICATION OF 1st, 3rd, AND 4th MOMENTS AT GRIDPOINT 2

	Elev.	x-Veloc.	z-Veloc.	x-Accel.	z-Accel.
MEAN OF THE TIME SERIES					
surface	1.0882E-03	-1.0199E-03	1.1065E-03	-8.8348E-05	-1.5237E-04
depth 2	-2.2429E-03	2.1862E-04	5.4033E-05	1.5868E-04	6.3899E-05
SKEWNESS OF THE TIME SERIES					
surface	-6.1489E-02	2.1248E-02	-1.0495E-01	1.0536E-01	2.5446E-02
depth 2	-3.4557E-02	1.7607E-02	-4.5957E-02	7.2808E-02	1.2595E-02
KURTOSIS OF THE TIME SERIES					
surface	3.2125	3.2047	3.1300	3.0369	3.1335
depth 2	3.2153	3.2270	3.2172	3.1703	3.1840

7.4 Quality verification - Directional case.

Second order statistics for the generated time series in the multidirectional scheme have also been evaluated and compared with the target values. The cross statistics between different locations on the grid are of particular interest in this case.

The directional power spectrum adopted in the present work is given by the relation (Kree and Soize, 1983) was presented in Chapter 3:

$$G_{\eta\eta}(\theta,\omega) = D(\theta,\theta_0)S_{\eta\eta}(\omega) \qquad \text{where} \qquad D(\theta,\theta_0) = A\cos^n(\theta-\theta_0)$$

An example grid used for the multidirectional generation of wave kinematics is shown in Figure 7.2.

All the target statistical values are evaluated as a function of the PSD of the elevation time series $S_{\eta\eta}(\omega)$, generated at the reference point using the ARMA digital filter mentioned earlier. So for example the covariance of the elevation time series between two grid points **1** and **2** with coordinates x_1, y_1 and x_2, y_2 respectively is given by:

$$c_{\eta_1\eta_2} = \frac{A}{\omega_c}\int_\theta\int_\omega \cos\left(\frac{dx(\theta)\omega^2}{g}\right)\cos^n\theta S_{\eta\eta}(\omega)d\theta d\omega \tag{7.2}$$

where $dx(\theta) = (x_2-x_1)\cos\theta + (y_2-y_1)\sin\theta$. Similar relations can be written for the second order cross-statistics between velocity and acceleration time series generated at different grid points for the x,y,z directions. The covariance between velocity at **1** and velocity at **2** in the x direction is:

$$c_{u_1u_2} = \frac{A}{\omega_c}\int_\theta\int_\omega \omega^2\cos\left(\frac{dx(\theta)\omega^2}{g}\right)\cos^n\theta\cos^2\theta S_{\eta\eta}(\omega)d\theta d\omega \tag{7.3}$$

and the covariance between velocity at **1** and acceleration at **2** in the y direction is:

$$c_{u_1\dot{u}_2} = \frac{A}{\omega_c}\int_{\theta}\int_{\omega}\omega^3 \sin\left(\frac{dx(\theta)\omega^2}{g}\right)\cos^n\theta\sin^2\theta S_{\eta\eta}(\omega)d\theta d\omega \tag{7.4}$$

whereas $c_{\dot{u}_1 u_2} = -c_{u_1\dot{u}_2}$ in all directions. Finally the covariance between acceleration in **1** and acceleration in **2** in the z direction is:

$$c_{\dot{u}_1\dot{u}_2} = \frac{A}{\omega_c}\int_{\theta}\int_{\omega}\omega^4 \cos\left(\frac{dx(\theta)\omega^2}{g}\right)\cos^n\theta S_{\eta\eta}(\omega)d\theta d\omega \tag{7.5}$$

All the integrations are carried out numerically.

Cross-statistics are evaluated for points 1 and 2 of the above grid and are shown in Table 7.5, in conjunction with the corresponding target values:

Table 7.5 Target vs Calculated covariances

Wind Speed=40.00 (knots), Nr of directions=9, Ntotal=5000 pts.

	Target Values	Calculated Values	Direction
$\eta_1\eta_2$	8.6181	8.2103	
u_1u_2	1.4964	1.4356	
$u_1\dot{u}_2$	-0.6008	-0.5569	X
$\dot{u}_1u_2$	0.6008	0.5574	
$\dot{u}_1\dot{u}_2$	0.3954	0.3557	
u_1u_2	0.5452	0.5049	
$u_1\dot{u}_2$	-0.1655	-0.1615	Y
$\dot{u}_1u_2$	0.1655	0.1615	
$\dot{u}_1\dot{u}_2$	0.1713	0.1553	
u_1u_2	2.0416	1.8584	
$u_1\dot{u}_2$	-0.7663	-0.6986	
$\dot{u}_1u_2$	0.7663	0.6991	Z
$\dot{u}_1\dot{u}_2$	0.5666	0.4891	

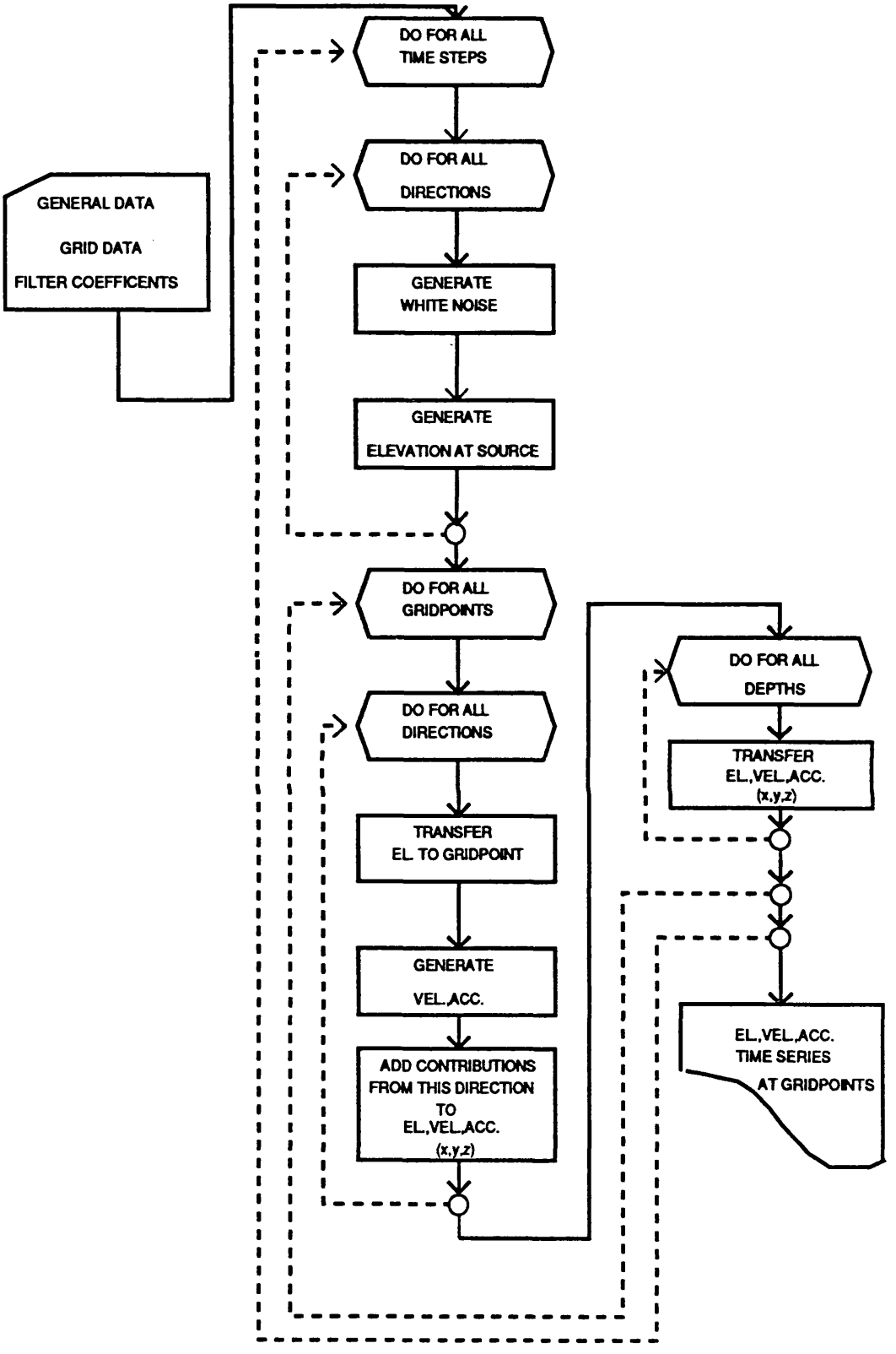


Fig. 7.1 Flow chart of the simulation program

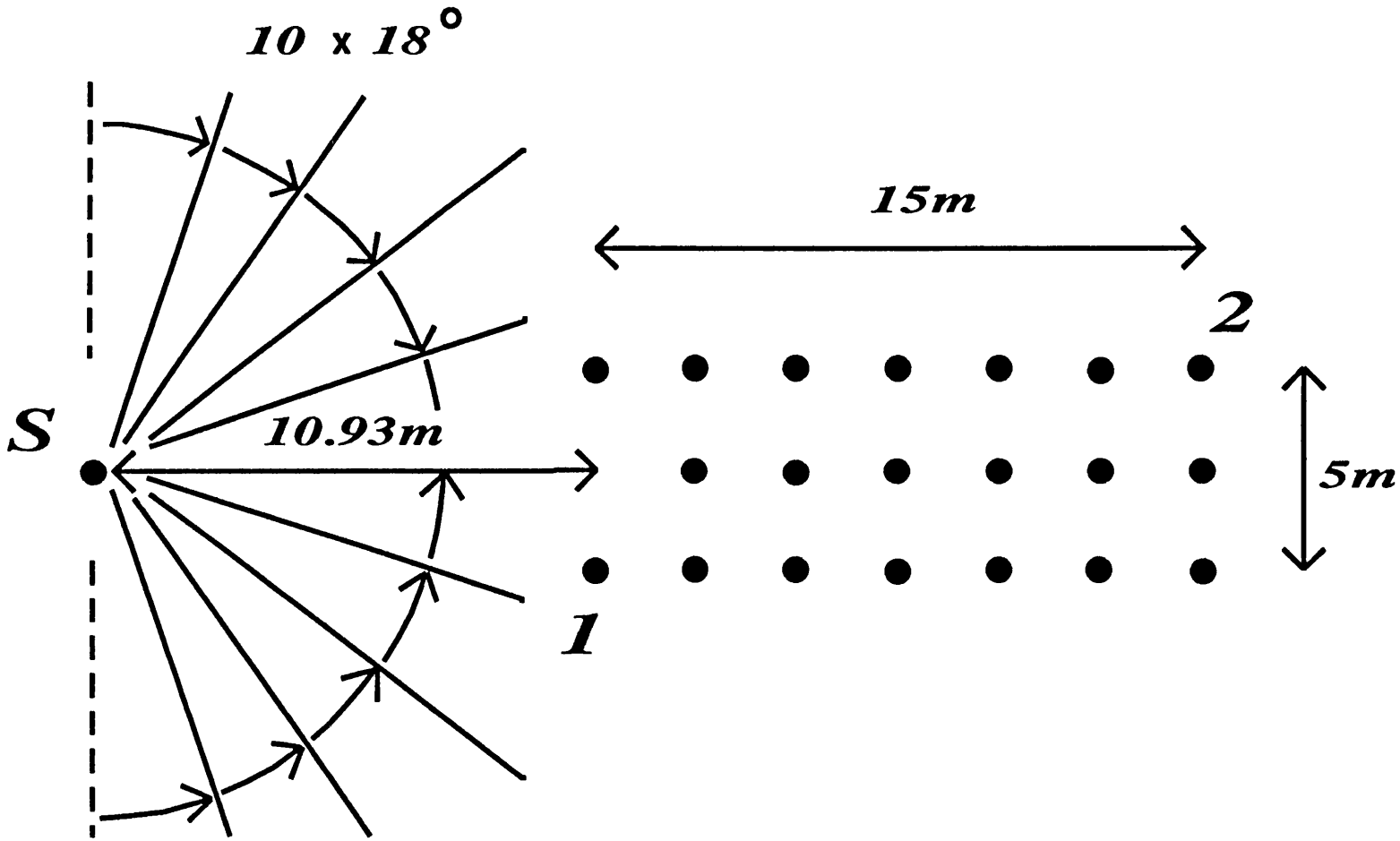


Fig. 7.2 Sea-Grid used for statistics

CHAPTER 8

RANDOM WAVE GENERATION ON PARALLEL COMPUTERS

CHAPTER EIGHT

CONTENTS

8.1 Introduction.

The 1980's saw an advance of parallel computer architectures, diverting attention from the classical Von Neumann design. Improved VLSI (very large-scale integrated circuits) technologies have undoubtedly led to the creation of very fast serial computers, but have also helped realise the limits of sequential processing. In terms of hardware the speed of accessing the data buffers seriously restricts the capabilities of the processor speed (Modi 1988), whereas in terms of programming the inherent parallelism of some algorithms is not exploited. A number of diverse disciplines have recently turned their attention to parallel processing. In particular cognitive science (Rumelhart and McClelland 1986), seeking in parallel distributed processing a framework to understand perception, contributed to the development of the theoretical bases of a new science. Applications like artificial intelligence, image and speech processing boosted the development of machines consisting of many processors, communicating with each other and thus representing a semantic network.

The aim of the present chapter is to study an implementation of the proposed algorithm for the generation of random wave kinematics, discussed in earlier chapters, on a parallel computer. An introduction to the main types of parallel machines and the basic concepts of concurrent programming will precede the main subject.

8.2 Parallel machines and concurrent programming.

Although the first operational parallel machines appeared in the early 70's, ILLIAC IV 1972, the vast variety of parallel computers has been developed in the last decade. The full classification exceeds the scope of this introduction (Kuck 1982), but a division of the existing machines into four categories, proposed quite early (Flynn 1966), is useful: 1) *single instruction stream - single data stream (SISD)*; mostly vector machines (Cray 1 - 3) fall in this class, 2) *single instruction stream - multiple data stream (SIMD)*; these are machines usually with a large number of processors operating with the same number on different data buffers, DAP (1974) is an example, 3) *multiple instruction stream - single data stream (MISD)*; not so common, and 4) *multiple instruction stream - multiple data stream (MIMD)*. A lot of parallel machines fall in this last category, starting from the older BBN butterfly (1981), the Intel Hypercube (1985) to today's computing surfaces, like the Meiko, consisting of Inmos Transputers (T800).

MIMD machines are usually divided into two further categories: *tightly coupled* and *loosely coupled*. In the first category of computers a number of independent processors are sharing a common memory bank; access to the memory is regulated by a control mechanism. The Encore Multimax (Gropp 1990) is an example. Computers of the second class on the contrary consist of independent processors that have each their own memory and they exchange occasionally information via the links that are connecting them; Intel's iPSC Hypercube and computing surfaces consisting of Inmos's Transputers are such cases. The parallel computer used in the present work belongs to this last class of loosely coupled MIMD processors; more details will be given in a following section.

As far as programming is concerned there are distinctions based on the level of parallelization. There exists a classification of numerical tasks originating from the collaborative effort of U.S. Govt. laboratories and universities in developing a set of standard program modules. This is the BLAS (Basic Linear Algebra Subprograms) classification. Although they predate the move to parallel computing, it will be expedient to relate the BLAS levels to the scale of the parallelisation. Thus, *Level 1* can be roughly equated with problems requiring fine grain parallelisation, *Level 2* with medium-grain and *Level 3* with coarse-grain. Networked processors can be used at every level, although the lowest level might be better handled by a vector-processor unit. At Level 2 the numerical task might be of the degree of complexity, say, of matrix inversion while at Level 3 it might be a problem of minimisation of a nonlinear function of several variables under linear constraints by the Frank-Wolfe algorithm.

A different classification of programming methodologies stems from the way an application is divided amongst the processors. The Occam User Group (Pritchard 1988) has introduced three programming paradigms: *Processor farms*, where all the processors carry identical code, produce their results independently and occasionally communicate with a collector processor; *Geometric parallelism*, where the processors are given almost identical code, but are handling different data and communicate with their neighbours; and *Algorithmic parallelism*, where the same data flows through the processors to be processed differently by each of them. Finally there are Hybrid methods that are combinations of more than one of the above.

The requirement for programming languages comprising features that allow exchange of information between processors running code in parallel was addressed in various ways. Existing high level languages like FORTRAN, C, ADA

etc. are designed to handle sequential code and therefore cannot be used, in their original form, for concurrent programming. The most radical is the design of an entirely new language; *OCCAM* (Bowler K., Kenway R. et.al 1987), (Pountain D. And May D. 1988), (Burns A. 1988), (Wexler J. 1989) is a language written specifically for the Transputer and as such is regarded the most complete concurrent programming tool. Its serious disadvantages are the lack of portability to different machines, the inability to parallelize easily existing and widely used programs, and (Downie M. and Bettess P. 1989) some features that render it inconvenient for programmers used in other languages. Therefore the modification of existing high level languages to include parallel features was a welcome development; 3L-FORTRAN and 3L-C were developed along these lines. Existing codes can easily become parallel with the addition of a few statements that handle communications or synchronize access to common data buffers. A third, hybrid, approach is to run the sequential programs on a single processor and speed-up time consuming calculations using an Occam harness to send parts of the code to a special multi-processor board.

In the application described in this chapter 3L-FORTRAN (3L User guide 1988) was the obvious choice, since the algorithm was originally written in FORTRAN. Nevertheless, for studies of the performance of the program, parts of the code had to be rewritten in a "quasi-Occam" language because this was the type of input required by the proprietary monitoring tool, Transim (Hart E. And Flavell S. 1990).

8.3 The parallelization of the wave simulation scheme.

The first question that a programmer should answer before attempting a parallel implementation of a given application, is whether the nature of the problem lends

itself to parallelization. The existence of relatively independent, computationally intensive processes is a favourable feature for the adoption of concurrent techniques. In programming terms this can be translated to the "computation over communication ratio", which must be large, and in any case greater than one, for a program to be considered amenable to parallelization.

In a simulation application the benefits of parallelisation may appear at the medium grain level, where typically the multiple processors may be used to speed up a particularly time consuming and often-called step of the simulation (for instance a parallel Blas-2 routine) while the main process executes on a single processor; or at the coarse grain level where the total simulation task is divided among the processors. The first type of usage benefits from the existence of libraries of parallelised mathematical routines which are currently available to purchase. In some applications the paradigm of a main serial program calling packaged parallel routines for standard numerical operations is sufficient. Here, the communications between slave processors would be the responsibility of the packaged software and need not preoccupy the user.

In the simulation of extensive dynamical systems however such an algorithmic model is too simplistic for two reasons: i. The simulation will fall into distinct computational phases with transfers of accumulated results between the phases; ii. each phase will be parallelisable at the coarse grain level, possibly with the parallel tasks themselves containing standard numerical sub-tasks (e.g. linear algebra tasks) for which packaged or own-make parallel routines may be used. In point i. not only the physical destination of the intermediate data is an issue, but also the rate of generation of such data on each parallel device: the prescription that the communication/computation ratio should be kept very low cannot always be put in practice. In certain standard numerical analysis tasks, such as the FFT, it has been

found that the inability to lower the ratio results in a poor efficiency index. As for the physical destination, with present day board architectures offering the faster DMA (direct memory access) mechanism this could be outside the processor network, i.e. on the host computer; in this case the transfer must be via the master processor on the network thus imposing a constraint on program flow. Alternatively the storage can be on the network itself; in this case the direct processor-to-processor links can be used. The destination can be another transputer/local memory module or a mass storage device such as a board-mounted hard disk. Again, there are resulting constraints on program flow.

Having decided to exploit parallelism at the "course grain" level, two possible lay-outs can be immediately visualized. One is the splitting of the wave directions amongst the processors. Each processor in this case generates time histories for the whole sea grid, but for one direction only. The contributions from all the directions are then added together on the root processor. Organized like this it becomes a pure "processor farming" application. A second idea, which is still "farming", incorporating some elements of geometric parallelism in the sense that different data are handled by each processor, is to allocate a part of the sea grid to each processor, which now generates the final time histories at the grid points, taking into account contributions from all directions. The root processor, apart from collecting the results from all the worker processors, can either be allocated a different kind of job, such as evaluating statistics, or be given its own part of the sea-grid to generate time histories. The second idea is adopted here and will be elaborated later; the hardware that was employed for the purpose will be described first.

8.3.1 Hardware facilities.

The Inmos T800 Transputer (Inmos manual 1988) is a 32-bit processor with an integrated 64-bit maths co-processor, four communications links and 4KB on-chip fast access memory. A Transtech TMB08 motherboard (TMB08 manual) containing seven Inmos T800 Transputers and hosted in an IBM PC/AT compatible machine is the basic parallel facility used here. Each transputer resides on a module (TRAM) with a DRAM memory bank (one module has 8MB and the other six 1MB each) and peripheral circuitry. The seven Transputers have two of their links hard-wired in a pipeline, whereas their remaining links are reconfigurable. This ability for software control of the topology of the transputer network is achieved by using a crossbar switch (IMSC004), provided on the motherboard. This switch can be set up by running an Occam PROGRAM on a T212 16-bit Inmos Transputer which is also provided on the board solely for this purpose.

Cascaded with the motherboard is a second (TMB03) board, containing a Transtech TTG3 graphics TRAM. This TRAM consists of a T800 Transputer, 2MB of DRAM and a designated graphics chip, the Inmos G300 Colour Video Controller. The G300 outputs to a separate, high resolution monitor attached to the system. Very fast or animated displays can be set up from data channelled to the TTG3 module directly from the Transputer network, thus avoiding the slow PC's data bus. The TMB03 board can be controlled from the PC directly, for example when it is used on its own, or when it is used as a "graphics engine" (this will be explained in detail in a later section); alternatively it can be part of the network and be controlled from the TMB08 motherboard.

8.3.2 Organization and monitoring of the distributed program.

The region of interest in the sea is covered by a 3-dimensional grid, and equal parts of this grid are allocated to the various processors. Five of the six workers carry identical code, namely a 2-thread task (producer and sender) and a simple multiplexor task for interleaving packets from upstream with packets originating locally; the sixth worker differs only inasmuch as it does not have a multiplexing task. The producer thread has two buffer arrays in which the output is placed in alternation. These are picked up by the sender thread using a swinging buffer technique.

The multiplexor is split in two sequential parts. The first part is responsible for broadcasting the initial information, that is global variables, filter coefficients, etc, generated at the Master task, to the producer task. Organizing the multiplexor this way, the use of separate channels for the initial flow of information and the passing of the processed information is avoided. This will allow the same programs to be used in cases where there is shortage of available links, like in the binary tree topology. When this role of the multiplexors is accomplished, they start the second phase, where they concentrate the packets of processed information generated at the producer tasks towards the reception point, which is a task residing in the root transputer.

The data is passed initially to a multiplexor residing on the root transputer itself, and subsequently to the Master task. The latter decodes the headers associated with the packets and stores the data in appropriate parts of a large (4 MB) storage array. A sender thread in the same task can channel the contents of these buffers either to a task performing statistical analysis of the results, or to an auxiliary graphical display system (a separate graphics tram driving its own

screen), or both. For the performance analyses that will be reported in a following paragraph, these subsidiary tasks were switched off; the data was merely placed in the buffer arrays and a minimum of communication was performed to inform the subsidiary processes of its availability.

The Master task is also connected to the "Filter", a standard i/o interfacing task between transputer and the PC., which will allow it to read the data files and communicate with the user. In the first part of the program, it will prepare all the constants that will be needed for the generations; then it will broadcast these to all the workers through the multiplexor mechanisms described earlier. When this is finished and all the producer tasks have the data they need to start the generation, the Master task will start its collecting role, described earlier.

As will become obvious in the following, the work load of the root processor is significant for the performance of the program. So far as the splitting of the sea area among the processors is concerned, two arrangements have been tried. In Arrangement 1 the root processor is not involved in the actual generation of wave kinematics. In the Arrangement 2 the master transputer has been allocated part of the surface grid in addition to the data handling duties described above.

In Figure 8.1 the organization described above is shown for the pipeline topology, whereas the three level binary tree topology is shown in Figure 8.2. It should be noted that for experiments with fewer processors (pipelines with one to five workers, two level binary tree), the basic organization is the same.

8.4 Studies of the performance using ad-hoc monitoring.

Studies of the performance of the parallel program for various typical cases are required in order to uncover possible deadlocks, flow disruptions and other factors

that may influence the efficiency of the program. In the present application it is additionally important to synchronise the delivery of information from the worker processors to the master.

A simple method was devised to monitor the performance of the distributed program. Timing information from the component processes is collected at run time and conveyed to the master process as part of the header buffers. The timing is done by planting calls to a standard timing function included in the run-time library (3L-Fortran) at strategic locations. Such locations are the beginning and the end of the calculation of a number of time steps, the passing of a buffer from the multiplexors and the arrival at the master process. This information is held in an array in the master task and later, in the end of the run, it is passed to the host computer for postprocessing. A post-processing package has been developed which uses the timing information from the actual run to calculate the total elapsed time of the run, statistics for the overall performance of the component programs, the time dependent arrival time of the information packets at the various stages on their route to the master, and profiles of activities at the processes and channels at regular time intervals.

8.4.1 Speedup and efficiency.

Timing results from the parallel implementation are summarised in Figs. 8.3 and 8.4. In these figures S1 and S2 refer to Arrangements 1 and 2 respectively, as described above. The individual curves in each group represent different problem sizes: thus 60 pts. refers to a grid of 60 points overall, divided among 2, 3, 4 etc. processors. The curves labelled S1 show an initial drop in speedup followed by an almost linear rise. The drop is due to the fact that in Arrangement 1, using 2 transputers, the computational work is not shared at all,

since the master transputer only handles communications. Thereafter adding processors produces proportional gains in speed. However the gains are seen to stabilise only when the number of gridpoints is increased to beyond 120 in this instance. For smaller amounts of computation, the fixed communication overheads dominate the timings and the speedup ratio reflects this loss. In Arrangement 2 (labelled S2), the trend is similar, but there is an immediate increase in speedup with the number of processors. It is worth noting that speedup has been defined here as the ratio of throughput from the multi-transputer system to throughput from a *sequential version* of the program running on a single transputer. This sequential version does not contain the sender thread and the multiplexor task but stores its output in buffer arrays directly as it is produced. This explains why curves S1 start slightly below the value 1.

The corresponding Efficiency curves (defined as ideal multi-processor timing over actual multi-processor timing) show that this implementation reaches an asymptotic efficiency value of 0.75 as the number of processors increases (Fig. 8.4). This asymptotic value is also the slope of the speedup curve (for a sufficiently large grid). The slope is less than unity because the parallel programs have to contain inter-process message passing steps which would be non-existent in single-task programs. It is conjectured that this value is a consequence of the computation to communication ratio characteristic of this simulation project, and of the minor programming decisions that have been adopted, such as buffer sizes etc. After stabilisation, this ratio would be approximately independent of the number of gridpoints since increasing the number of gridpoints also increases the volume of data to be transmitted.

For the three level binary tree topology shown in Fig. 8.2 the maximum efficiency reached was 0.90. The table below shows the speedup and efficiency achieved for cases corresponding to these shown in Figs.3 and 8.4:

Table 8.1: 6 Worker processors, 1000 time steps generated.

Nr. of Grid Points	Speedup	Efficiency
12 pts - S1	5.14	0.73
60 pts - S1	6.38	0.90
120 pts - S1	6.39	0.91

The tree topology appears to be more suitable for the present application than the pipeline. Nevertheless the flow disruption studies reported in the next section were performed on a pipeline, because these phenomena are more pronounced there, and because this topology can have wider applications.

8.4.2 Data flow disruption

The parameters whose variation can affect balance of the data flow are studied here. In the present case the number of time steps produced and buffered in the worker before the packet is forwarded to the master processor is thought to be a significant parameter. The number of grid points processed per worker is also of importance. The total length of a run is also important when transient behaviour is expected.

The number of time steps alters the communication over calculation ratio in a rather complicated manner since it affects both the work load of the process and the buffer length. To simplify the observations, the buffer length is varied only

by adding zeros to a fixed number of calculated time steps. Alternative the computational burden can be solely varied by filling a fixed length buffer to a changing degree with actual generated time steps.

Performing a series of experiments on a 7 transputer pipeline implementation the postprocessing package produced pictures such as Figures 8.5-8.10 from which the progress of the algorithm can be assessed.

Figures 8.5 and 8.6 show the time taken by a typical packet to traverse the various stages from the generation point to the collection point. The times are not absolute, but relative to the start time of the generation of the packet. Note that packet generation is an uninterrupted sequential calculation, therefore in all cases the generation time will be a constant. The case considered in Figure 8.5 is one where the time to produce the packet is greater than the times required for each of the communications and buffering services performed on that packet by other tasks. On the other hand Figure 8.6 shows a case where the production takes less time than one of the subsequent tasks, namely the final receiver task in the root transputer. In Figure 8.6 the production time has been deliberately shortened to 1/4 of the previous case, but the packet lengths have been kept constant, in order to avoid any changes in the communications burden. In this case the output of the remote workers takes longer to reach the final storage buffers, and the delay is seen to grow as the distance of the worker from the root transputer. More insight into this phenomenon is given by Figures 8.7 and 8.8, where the same information is displayed, but for packets originating from one processor only (Figure 8.7 is for the remotest, Figure 8.8 is for the 4th worker away from the root) and this time packet-by-packet rather than averaged across the packets. In Figure 8.7, after an initial transient phase the successive packets are seen to spend the same amount of time in transit,

whereas in Figure 8.8 the time shows rapid fluctuations between two levels. Incidentally, Figure 8.7 also shows that the transit through the 5 intermediate multiplexors is fast; much of the transmission delay is incurred at the root transputer. In this particular case this delay is seen to be roughly half the production time of the packet (since no intensive computation was taking place at the collection site, the reason for the delay may be that the root transputer has more tasks to perform in parallel, including 3L's "filter" task communicating with the host computer).

Another way of looking at the phenomenon is to plot absolute times of arrival into the buffer at the receiving task for all the packets in the run. In Figure 8.9 the packets from all transputers are seen to reach the receiver in clusters containing one packet from each processor (i.e. the processors remain in step with each other), whereas in Figure 8.10 processor 5 manages to send about 30 packets in the time that processors 1-4 finish 225, and processor 6 (the remotest) only sends one packet in that time. The overall run time is lengthened due to the fact that the last two processors have to do most of their work after the others have finished theirs, hence efficiency is lost. Some imbalance in the rates of delivery from the workers might be expected in a communication-heavy problem, with the downstream workers finding less time to complete their production quota; in the present case the imbalance is drastic, and it affects the upstream workers rather than the opposite.

Flow disorder can also appear when the delay is not on the receiving processor, but in elements of the pipeline itself. In a typical worker processor, as described earlier, there are three processes running in parallel, namely the producer and the sender threads and the multiplexor task. These three are timeslicing. It was observed that the transmission of data from the producing thread to the

multiplexor was taking significantly more time than would normally be required for the relatively small arrays. This phenomenon was also repeated on a one master - one worker configuration, indicating that its origin need not necessarily be associated with the pipeline.

It is assumed that timeslicing between the three modules running in parallel at the same (low) priority is the cause of the delay. One of them, the producer, is running an intense, compared with the communications' CPU requirements, calculation task, probably with few "natural" descheduling points (Inmos Transputer Reference Manual). Thus it is not paying attention to the communication modules which are nevertheless ready to perform their duties. This results in a significant delay of the transmission of an array, which will be felt in the whole pipeline.

In order to quantify the transmission delays assume that a multiplexor needs T_{mux} units of time to transmit a data packet (delays included), and the total number of worker processors in the pipeline is ***Nproc***. A packet of data originating from processor ***ID2*** will arrive at processor ***ID1*** (numbering from the pipe-head to the pipe-end) after $T_{mux}(ID1\text{-}ID2)$. If this time is larger that the time needed for the production of one packet, T_{prod}, then the "next generation" packet produced at ***ID1*** will enter the multiplexor, preventing ***ID2*** from delivering its own. This delay will start quickly amplifying upstream which may result in the total blockage of processors near the pipe-end, until the ones near the pipe-head have completed their run.

The above form of disorder can be diminished if the communications modules are prioritized. The Transputer offers two levels of priority, high and low. A process running on high priority will accomplish its task without pausing, so in

a multiplexor or a sender thread when the transmission of an array is ready to happen, it will take place immediately, forcing the producing thread to pause for a short period.

An alternative method is to artificially increase the number of descheduling locations by introducing interrupts in an intensive calculation module, like the producing thread, in order to allow for the communications to take place. These interrupts, which can be statements like timer inputs, can be inserted in the middle of a calculation loop. In practice it was found that very few such commands suffice to significantly reduce the communication's delay (a factor of 1/5 was achieved by inserting one timer call in the middle of the production of one time-step of data).

The above observations confirm the fact that flow disruption depends on the ratio of calculation over total transmission time of a packet of data. If changing the program in one of the described ways in order to reduce the communications delays is not desirable, care should be taken to increase the work load on the producing thread so that:

$$T_{prod} > T_{transit} = T_{mux} Nproc \qquad (8.1)$$

were the time spent in a multiplexor T_{mux} includes the delays due to the fact that the calculations are not interrupted, as mentioned earlier. Preparing more time-steps per packet is an easy way out to improve the performance, although one should bear in mind this will result in some increase of the transmission time as well, due to longer arrays of data.

8.5 Transim simulation

In addition to the ad-hoc data flow analysis by timing probes mentioned in the previous section a simulation of the whole parallel algorithm was carried out using the proprietary package Transim (Hart an Flavell 1990). The main advantage of the use of such a package is that more extensive distributed algorithms can be analysed than one has the hardware for. Also new ideas that the programmer feels could improve the performance of a distributed program can be tried first in Transim. An additional bonus is the fact that Transim allows debugging statements to be incorporated in the program, these producing information lines in a log file.

The major independent computational units should be identified, as well as the communications that are required between these units. The computational units will then have to be timed, in terms of CPU usage, whereas the communication buffer lengths have also to be specified. A skeletal program comprising only the computational units in terms of time delays, the channel communications and the buffer lengths can be fed into the package. The output of Transim will then contain the total elapsed time of the simulated run of the user's program, as well as information on profiles of the run at regular time intervals. The latter, showing the percentage of usage of the total time by the individual processes and by channel communications, is useful in identifying deadlocks and idle units.

Calibration.

Transim, as mentioned in an earlier paragraph, requires timings of the independent computational units and buffer lengths. The latter can be easily provided if the transputer code already exists. The timings can be estimated from sample runs of the program. So the time required to calculate elevations for N_g grid points, for a fixed number of wave directions, per timestep, can be found by solving the

equation

$$(T_1 + T_2 N_g) N_{steps} = T_{prod} \tag{8.2}$$

where T_{prod} is the production time per packet, and can be measured as displayed in the previous section. Two measurements of T_{prod} suffice to estimate the unknowns. Similarly the delay at the collection point can be timed from the equation:

$$T_3 + T_4 N_{buff.length} = T_{arrival-last.max} \tag{8.3}$$

where $T_{arrival\text{-}last.max}$ is again measured from previous experiments, for two cases.

Using these equations, unit timings were obtained from 3 calibration trials which were "normal" cases (i.e. free of flow disruption). The production times in these trials ranged between 4800 and 7200 ticks per packet (approximately). With all 6 transputers producing exactly identical parts of the sea, there would be one packet every 800 - 1200 ticks, trying to enter the root transputer (on average, and ignoring the fact that the arrivals are clustered) This should match the slope of Figure 8.9. Note that the figure of 930 is not the (average) time between entrances of packets into the receiver: it is the time one particular packet spends on the last stage of its journey. The transmission of a packet through a multiplexor was assumed to be instantaneous. The values for the timings found were (in low priority timer ticks; x 64 = microseconds, to be used in Transim's SERV)

T_1	T_2	T_3	T_4
34	29	930	0.5

Programming.

Version 3.2(7) of Transim as used here only accepts an input program in quasi-Occam; therefore the Fortran code was recast in this language. This gave rise to a difficulty inasmuch as the target program used 3L Fortran's *threads*. Threads are parallel processes which may share data with the main thread and with each other by accessing the data placed in COMMON. A system of semaphores exists in order to regulate the access to the data. The semaphores are also variables in COMMON, which the threads may read and modify. Access regulation for the semaphores themselves is presumably handled by the package. Since there is no comparable mechanism in Occam (in fact sharing of variables is strongly discouraged) it was necessary to emulate the semaphores by some explicit Occam code. This emulation does not cover all the functions of a semaphore but is more in the nature of a Boolean flag variable, shared, set and reset by the processes which require it. This facility is used, for example, in the swinging buffer operation between the production thread and the sender thread as discussed above. In this way, our Transim code has been kept as close to the original 3L Fortran as possible. The Quasi-Occam algorithm devised to resemble the threads and semaphores of 3L-Fortran, as they are used in our distributed program, is given below. The code is stripped of the parts that are irrelevant to Boolean flags mechanism. It should be noted that the purpose of the WAIT(1) that appears in the checking of the values of the Booleans, is to momentarily deschedule the running thread (process) so as to allow its parallel counterpart to resume execution and probably change the flag before the latter is checked again.

```
BOOL sender.can.start, sender.finished:
SEQ i worker -- workers numbered from 1 to 6
{
 sender.can.start := FALSE
 sender.finished := FALSE
 work := 1
 PAR
 {
  SEQ i ptop    -- 1. Producing thread
  {
      IF
      {
       k = marr
         SEQ
         {
          safebuffer.forsender:=work
          sender.can.start := TRUE
         }
       TRUE
         SEQ
         {
          WHILE NOT sender.finished
            WAIT(1)
          SEQ
          {
            sender.finished := FALSE
            safebuffer.forsender:=work
            sender.can.start := TRUE
          }
         }
      }
      IF
      {
       work = 1
         work := 2
       work = 2
         work := 1
      }
    }
  } -- end of SEQ (end of ptop)
  SEQ i sender  -- 2. Sender thread
  {
   WHILE kloc < ntotal
     SEQ
     {
      WHILE NOT sender.can.start
        WAIT(1)
      SEQ
      {
       sender.can.start := FALSE
       work := safebuffer.forsender
       to.mux ! body i nbytes
       sender.finished := TRUE
      }
     }
  } -- end of SEQ (end of sender)
```

Results.

The correct calibration of the Transim program can be demonstrated in the following example: In a run 10 gridpts per worker were done; 50 packets at 20 timesteps per packet were produced (total of 1000 timesteps). Packet length was 200 (i.e. it corresponded to the actual work produced). This represented a "normal" case. The total run times were as follows: From transputer run, as reported in the .TIM file: 368279 ticks x 64 = 2356986 microseconds. As predicted by transim: 23629552 microseconds. In this transim run SET NODIM was used to non-dimensionalise the hardware, and the following unit timings: TSL=1024 , EICRS=0.2 , ICS=2.

The main objective of the Transim runs performed was to verify that the flow disruption and attendant problems described above were reproducible even when the code was reduced to its bare essentials and run outside the original hardware environment. This was found to be so when the realistic task loads were specified to the simulation: the disruption of the order of the packets took almost the same form as in the transputer run, resulting in increased run times as before. Also the remedies proposed in a previous section, such as prioritising the communications modules, or planting disruption points in the computational mode, were seen to work in Transim as well.

Independent communication modules.

Another idea that was tried in Transim was to split the Multiplexor task which resided on each processor ("Mux" in Fig. 8.1) into two parallel modules; one, "Mux1", to handle the incoming communications from either upstream or from the processor's calculating module, and a second, "Mux2", to export the packets downstream. It was thought that this arrangement could better utilise the

Transputers ability to handle the communications through the hardware links in parallel with the CPU activity. At the same time, as far as the alternative input to "Mux1" is concerned, messages coming from upstream were prioritised with respect to messages coming from its own processor. Indeed the performance of the programme benefits from this arrangement: In Figure 8.11 a case is shown which suffers from gross flow disruption if handled by the unmodified program. The work load per packet is too small for each processor, resulting into a programme which behaves like a sequential; almost all the upstream workers are waiting until a downstream neighbour finishes its quota. The total run time is much longer than normal. In Figure 8.12 the modification described in this paragraph was tried on the same example. It can be seen that balance is restored almost completely and the overall run time is four times smaller than the previous case. The fact that packets arriving from worker 6 are now arriving at a slightly faster rate is due to the way the communications were prioritised, as described earlier.

8.6 Real time display.

The existence of the TTG3 graphics TRAM allows the setting up of very fast displays during run time. The basic requirement in terms of software is a graphics server that can accept instruction from programs running on Transputers. This server, TTGS, consists of a number of graphics primitives [Ref. manual TTGS] able to manipulate lines, arcs, polygons, etc. It is loaded on the T800 Transputer that resides on the TTG3 directly from the PC. The wave generation programs on the other hand are loaded on the Transputer network separately. These two separate actions are necessary since the library (server), compiled in an environment different from the programs, OCCAM/TDS, cannot be downloaded from the network. A link communication is established between the root

Transputer and the TTG3 which then acts as a graphics engine, driving its own screen. The master program running on the root transputer is thus interfaced with the server via a two way channel and can use its various drawing functions.

The organization of the distributed program follows the description given in an earlier section (Fig 8.1). The graphics commands are issued by a sender thread which runs in parallel with the receiver thread on the master task. This thread reads the data it needs from the common storage area. Contention between the two threads trying to access the same zone in the common buffer is avoided using the 3L Semaphore mechanism described earlier. The drawing takes place in a continuous advance of frames, each frame representing the wave elevation on the whole sea surface at one time instant. The screen is firstly divided in to a coarse grid, representing the sea surface grid where elevations have been generated, and on top of this a finer 4x4 mesh is superimposed, where elevations are estimated using a bilinear interpolation scheme. Each rectangle of the finer grid is painted a colour corresponding to the elevation at the enclosed grid point. Two screens are used, one of which is displayed while the next frame is being prepared on the other, hidden screen. Continuous flipping of the two screens is used to give an animated impression, although speed is greatly increased if rectangles are only painted when they change colour. This "contour like" method emphasizes the moving wave impression.

The speed of the wave generation is nevertheless reduced when high quality graphics are displayed. As an example, 2000 timesteps of surface elevation at 300 gridpoints, representing a record of 33.3 minutes of real time, were generated in 4.69 minutes when the visual display was off, thus giving a ratio of "*real over*

simulated time" of 7.1. The same record was generated in 7.74 minutes when the visual display was on, so the simulation was running just 4.3 times faster than real time.

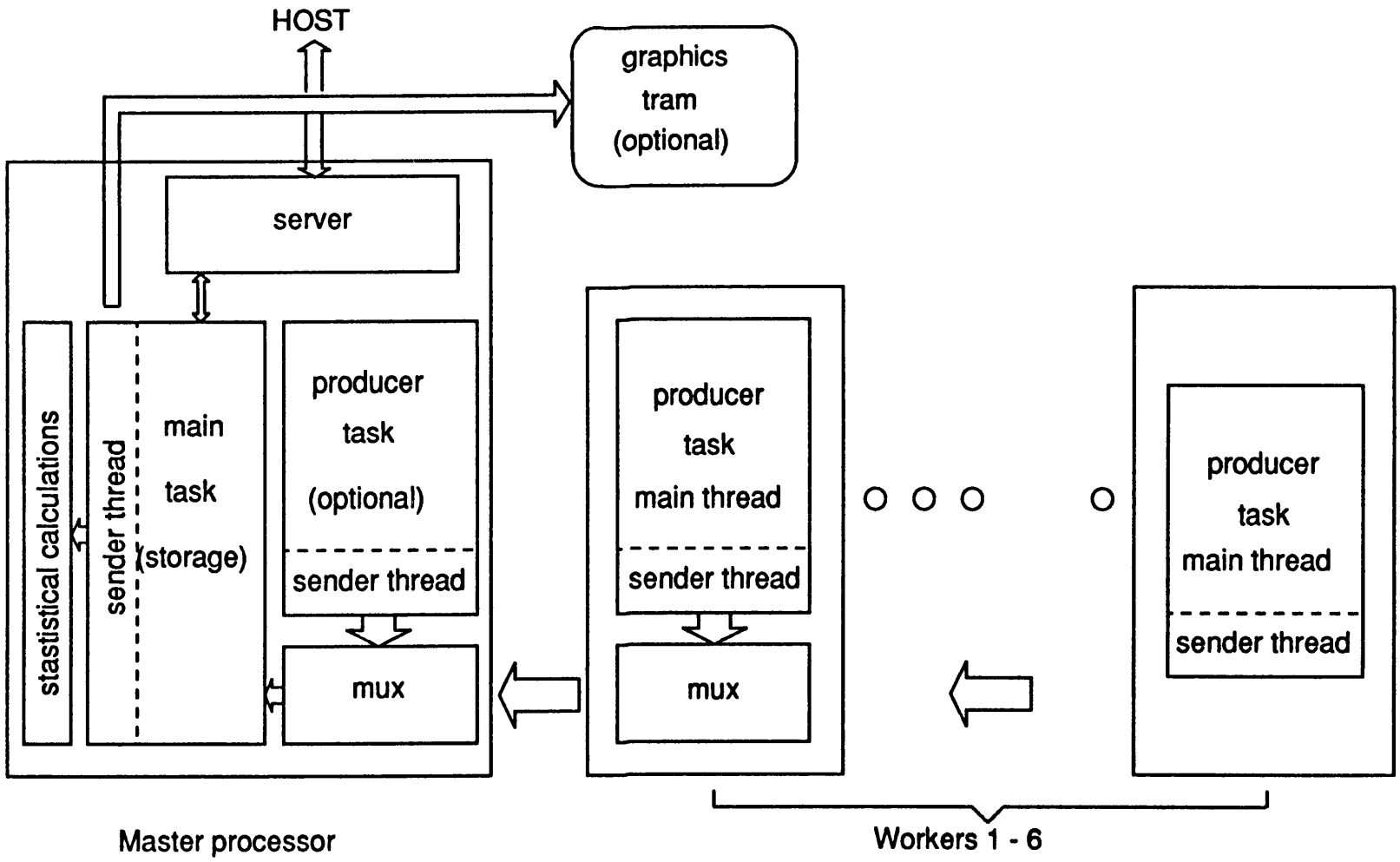


Fig. 8.1 Block diagram 1: Pipeline

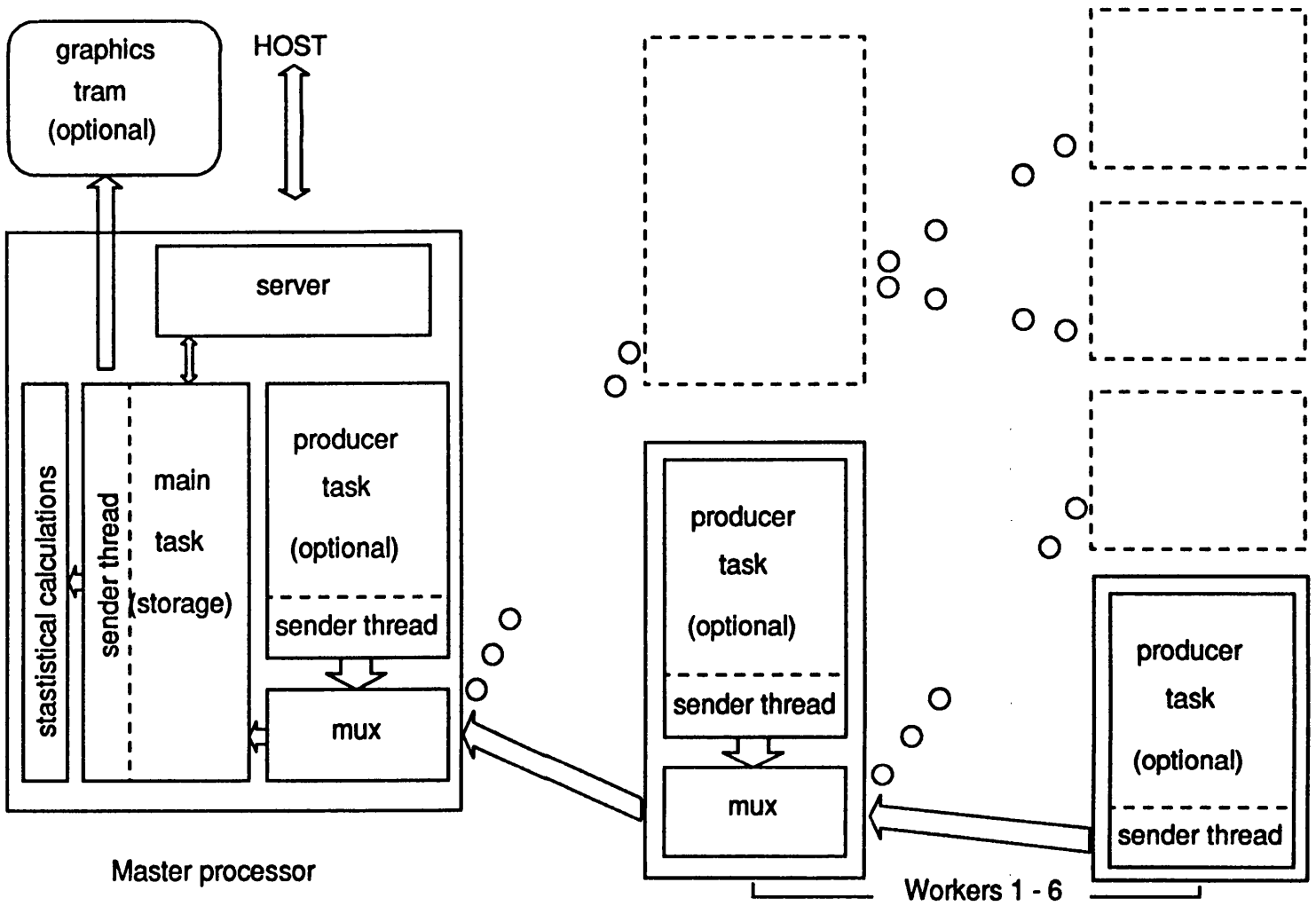
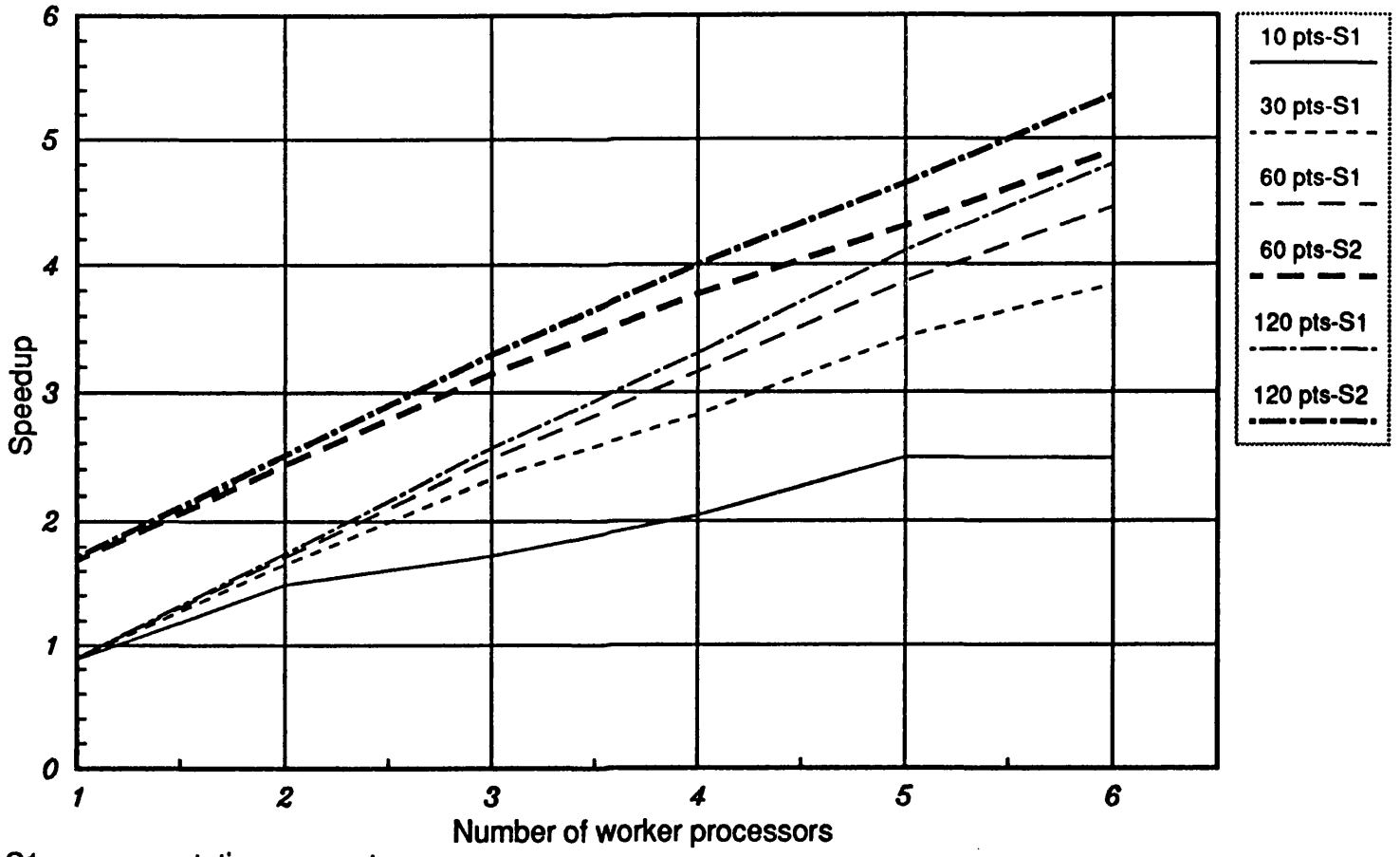
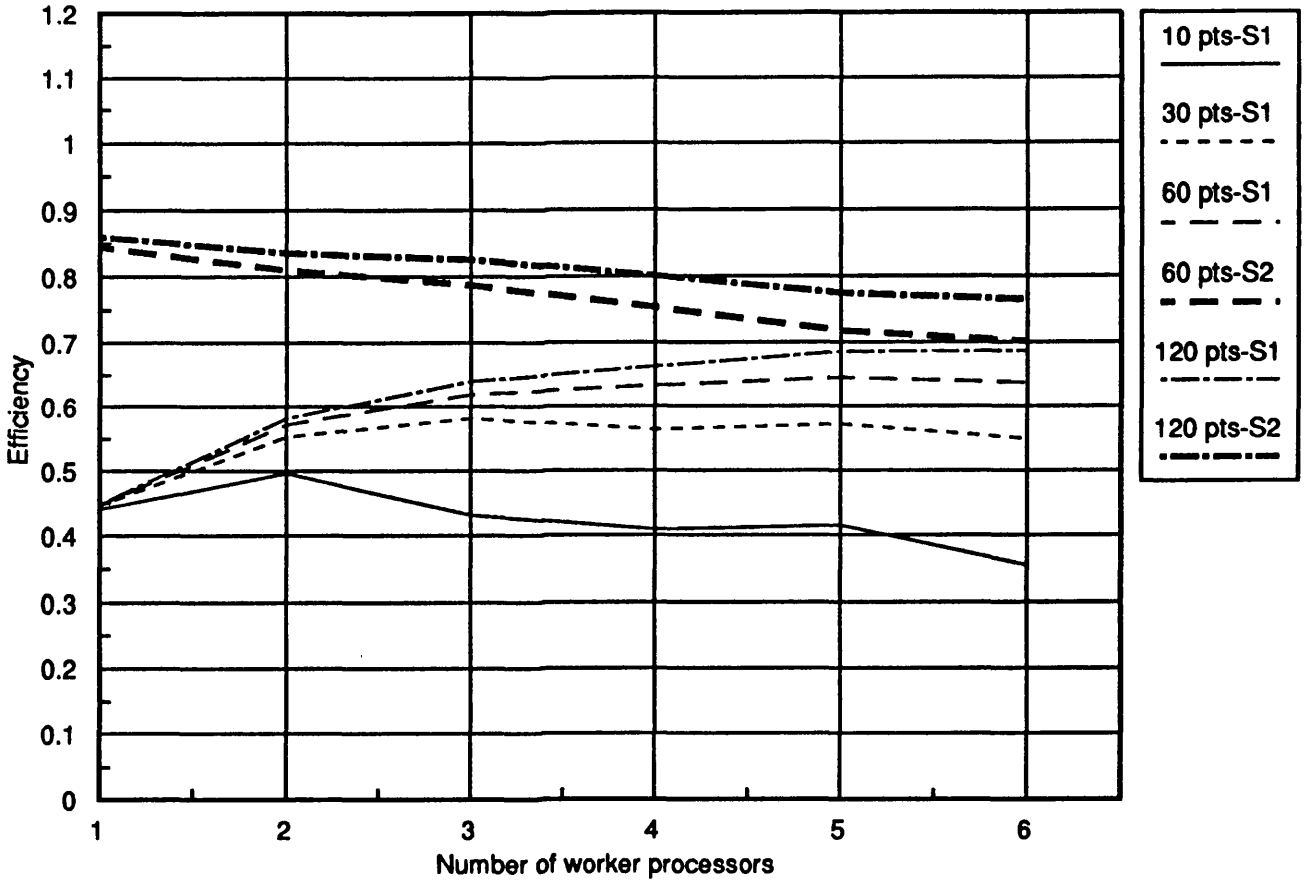


Fig. 8.2 Block diagram 2: Tree



S1 : no computation on master processor
S2 : computation on all processors

Fig. 8.3 Speedup curves for pipeline topology



S1 : no computation on master processor
S2 : computation on all processors

Fig. 8.4 Efficiency curves
U=40 (Knots) N=1000 Ndir=9

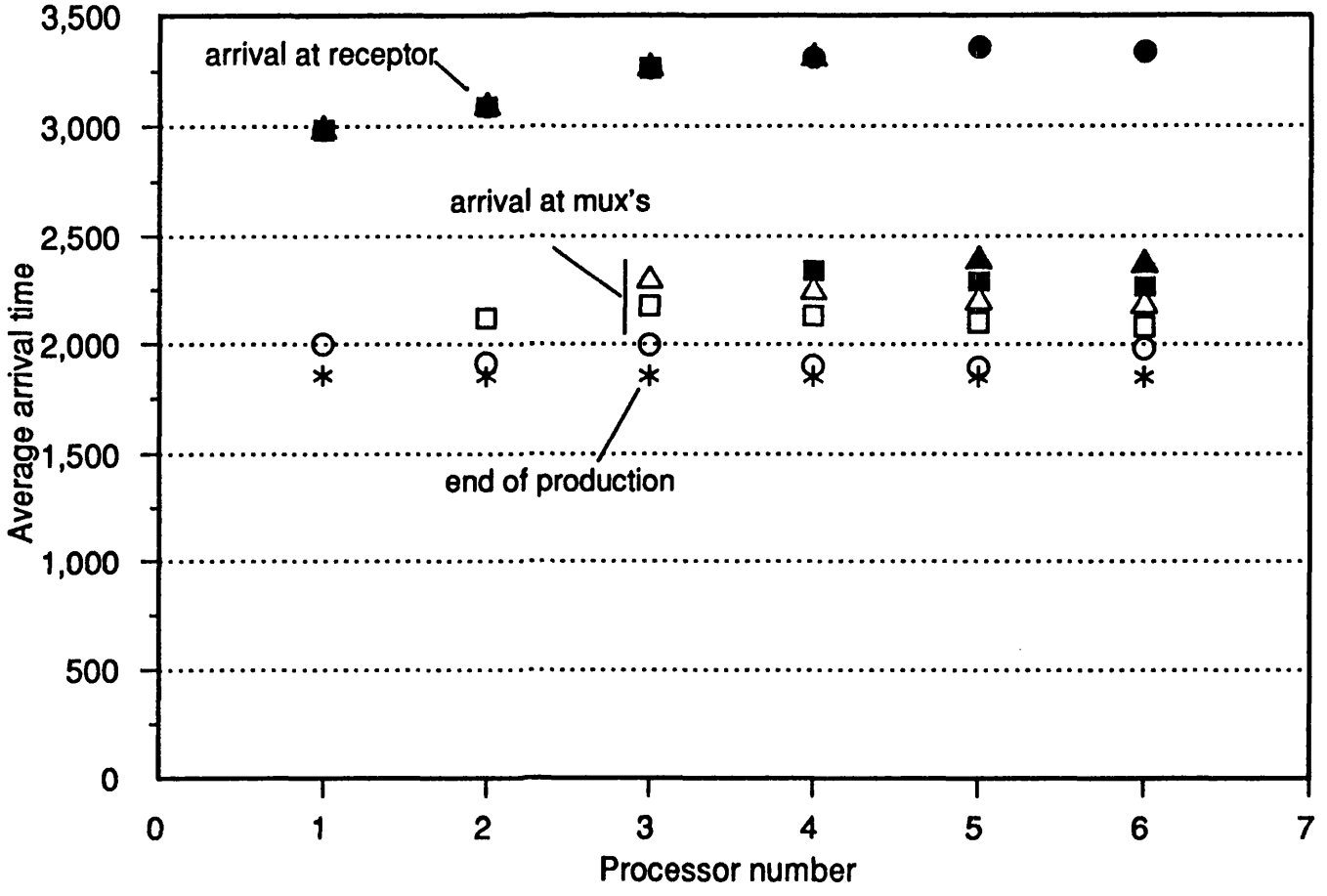


Fig. 8.5 Average timings for packets from workers 1-6
normal case

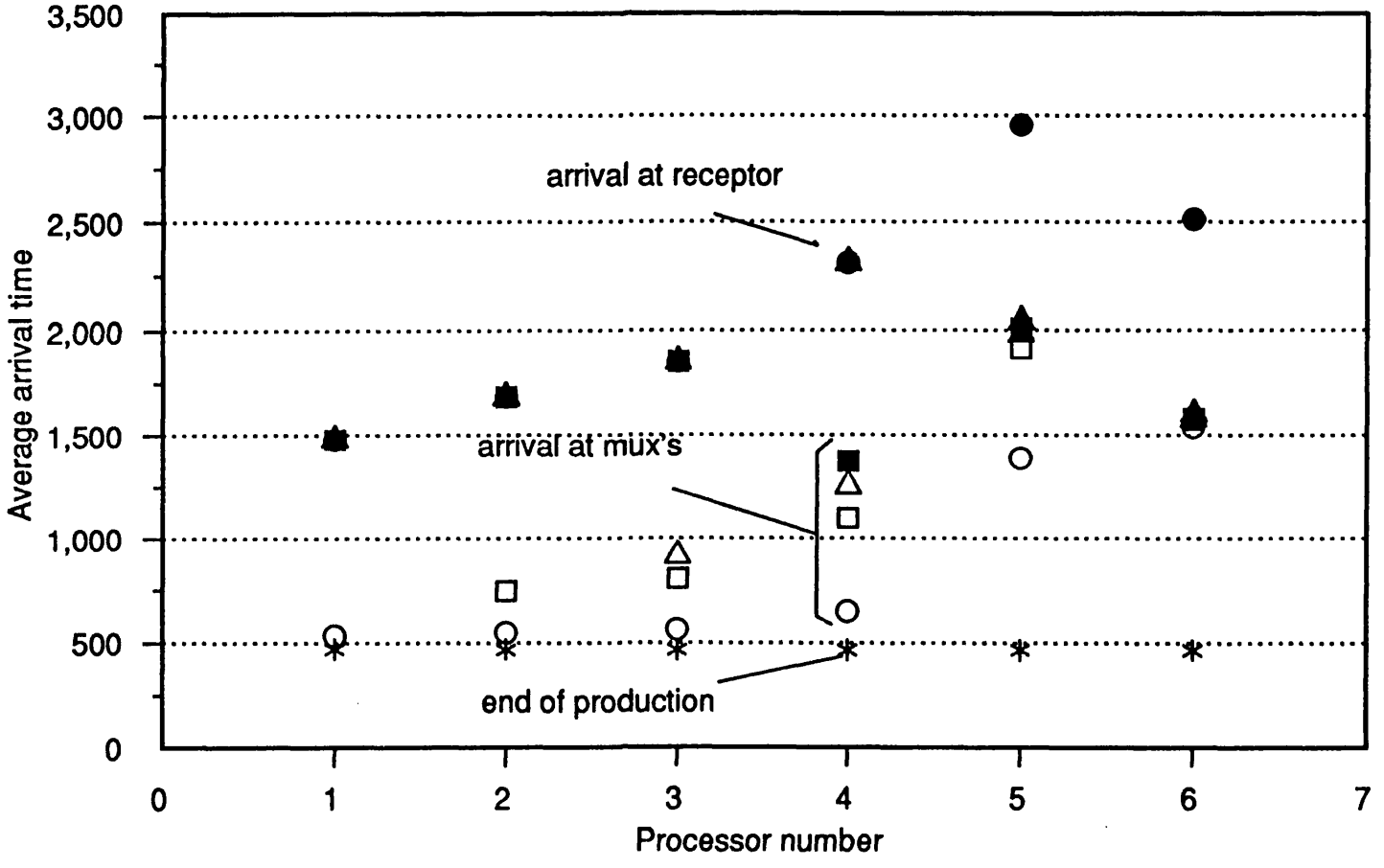


Fig. 8.6 Average timings for packets from workers 1-6

flow disruption

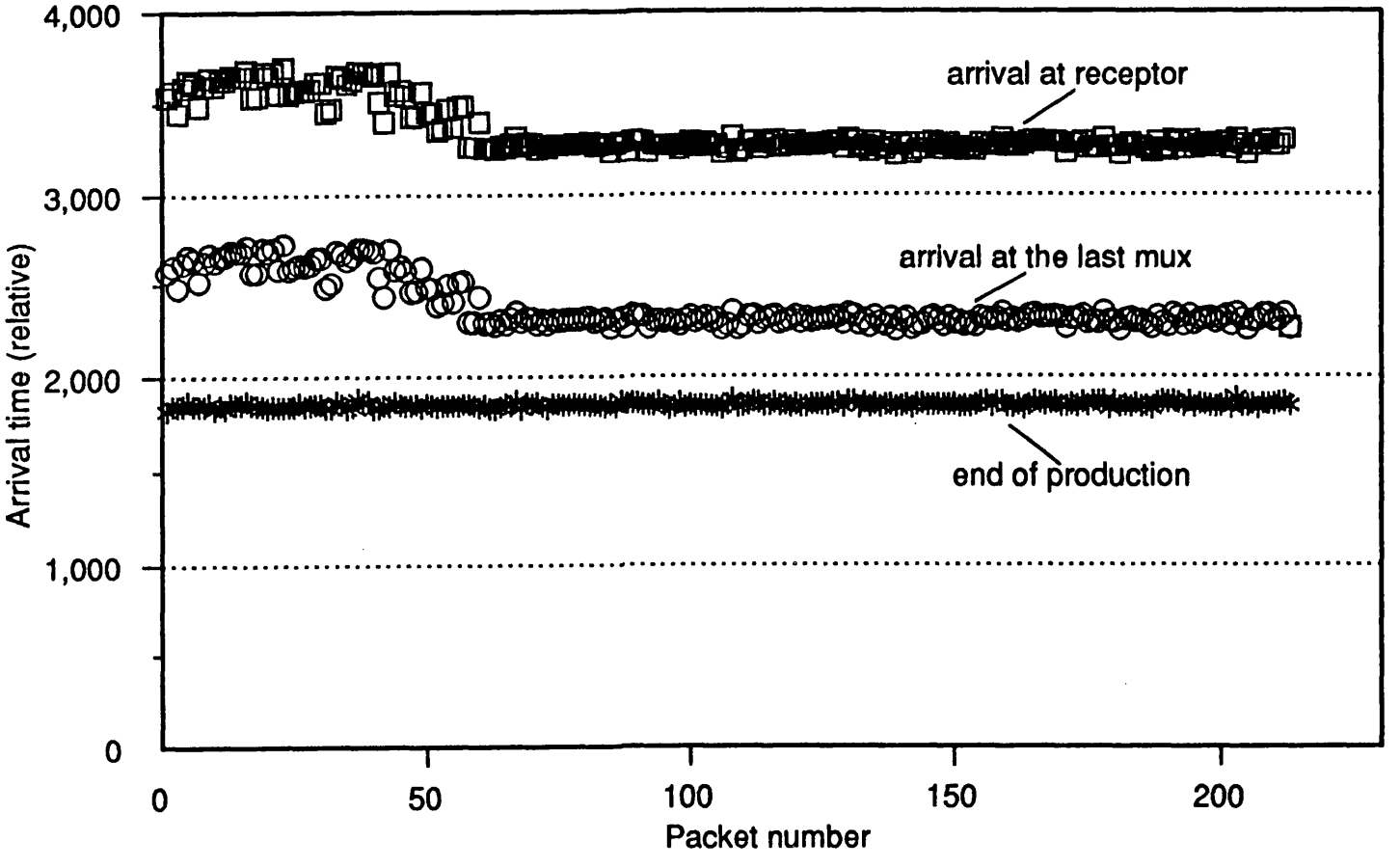


Fig. 8.7 Transit times for packets from worker 6
normal case

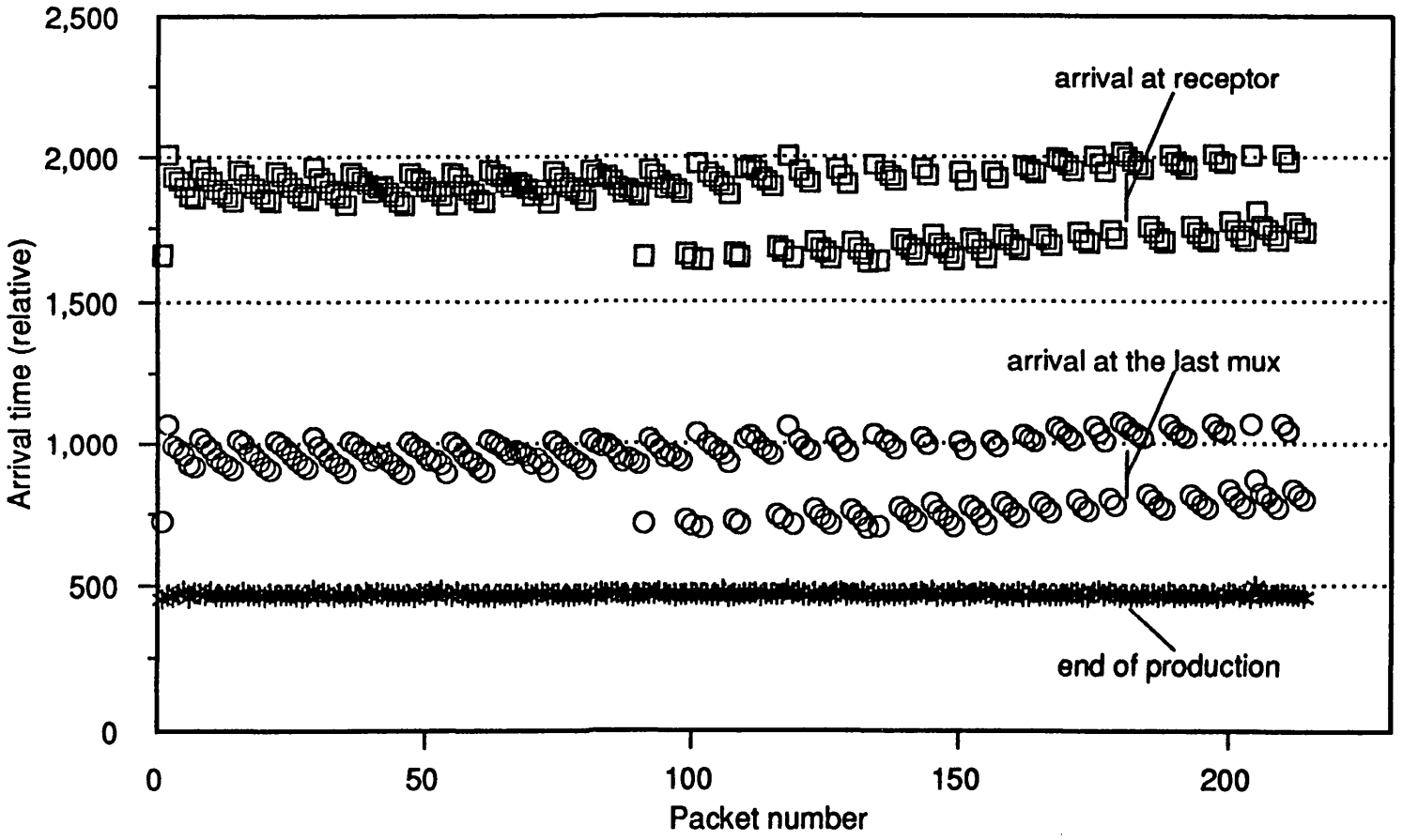


Fig. 8.8 Transit times for packets from worker 4
flow disruption

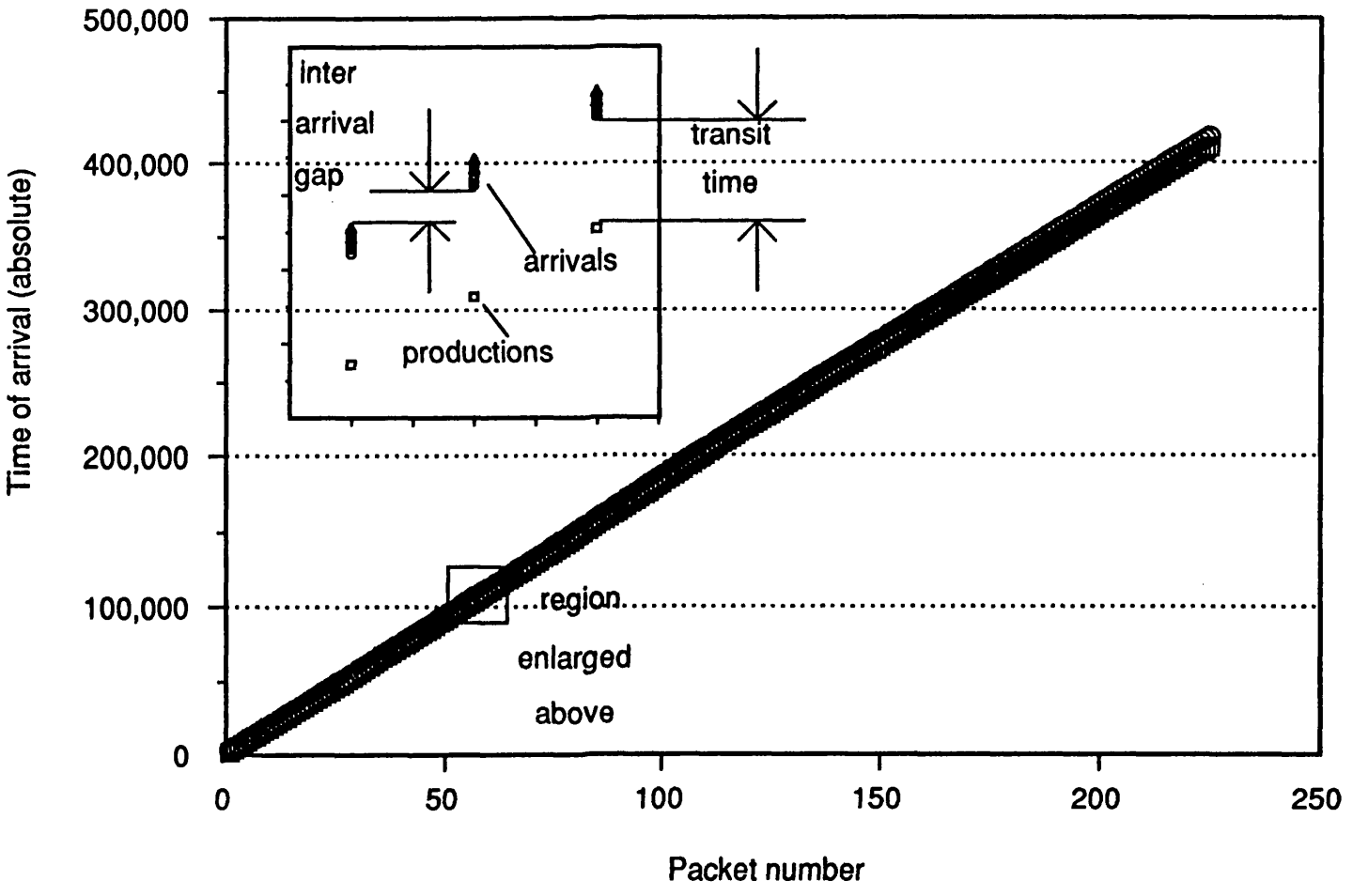


Fig. 8.9 Arrival times at receptor site
normal case

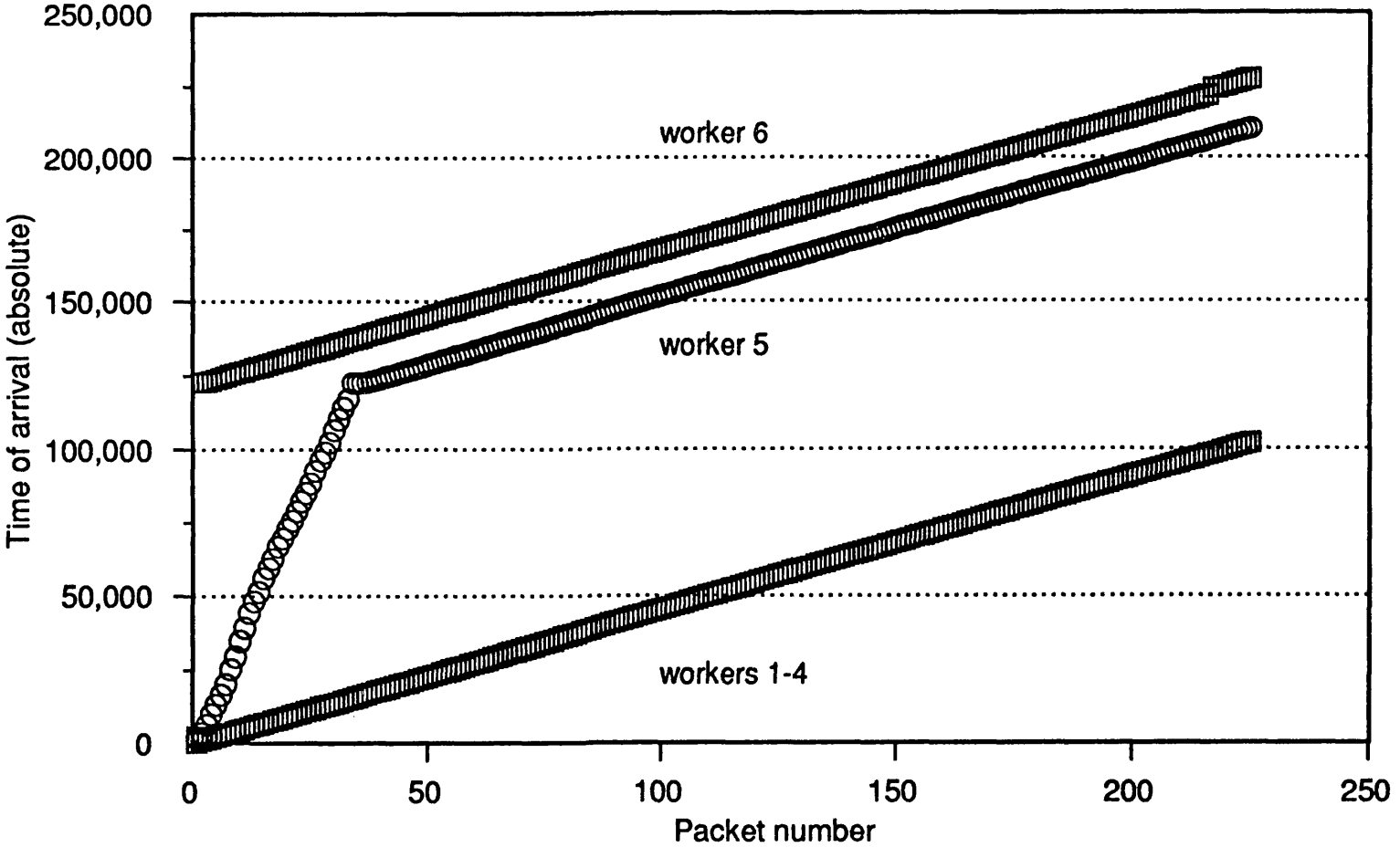


Fig. 8.10 Arrival times at receptor site
flow disruption

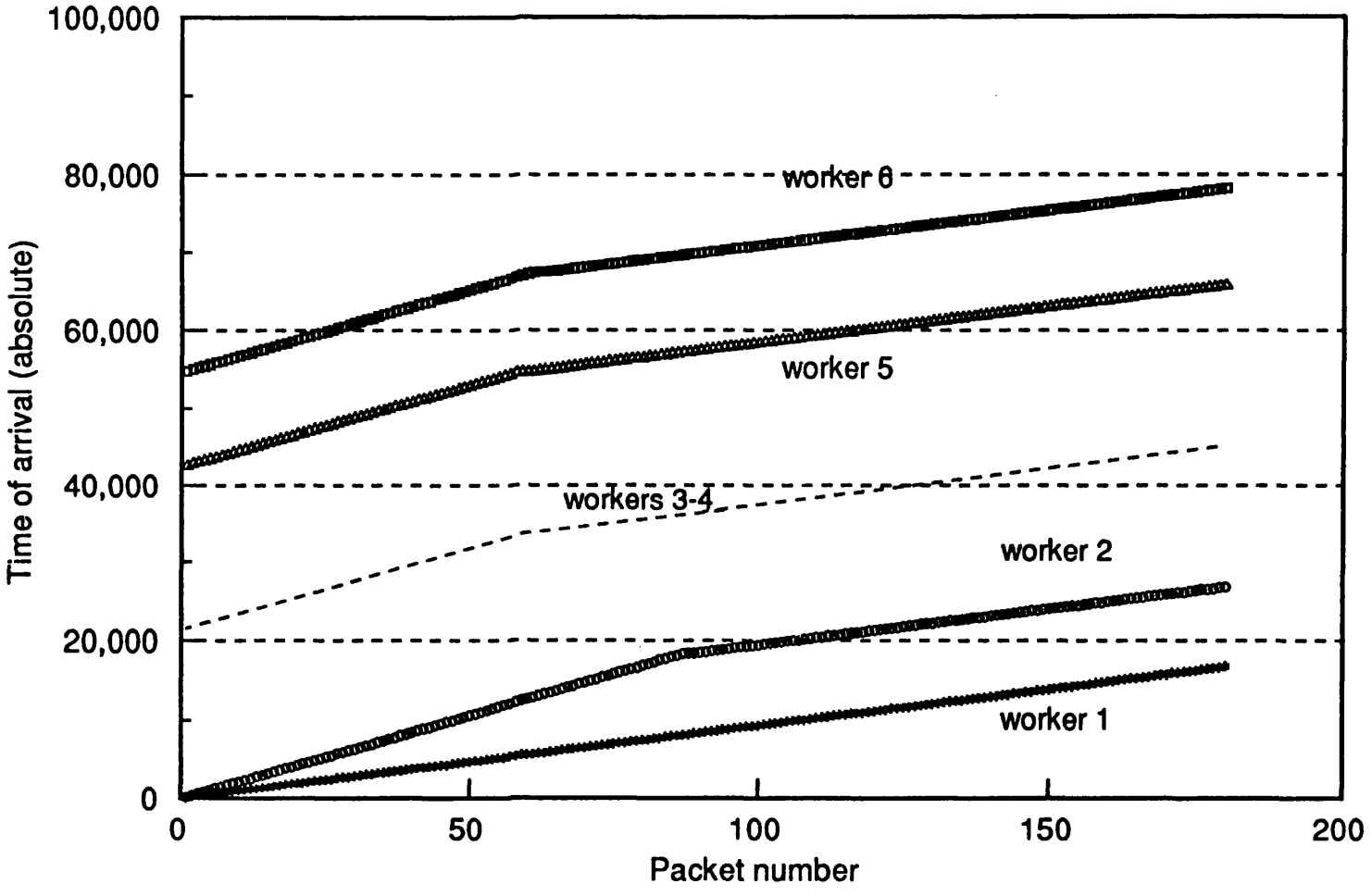


Fig. 8.11 Simulated arrival times at receptor site
Flow disruption

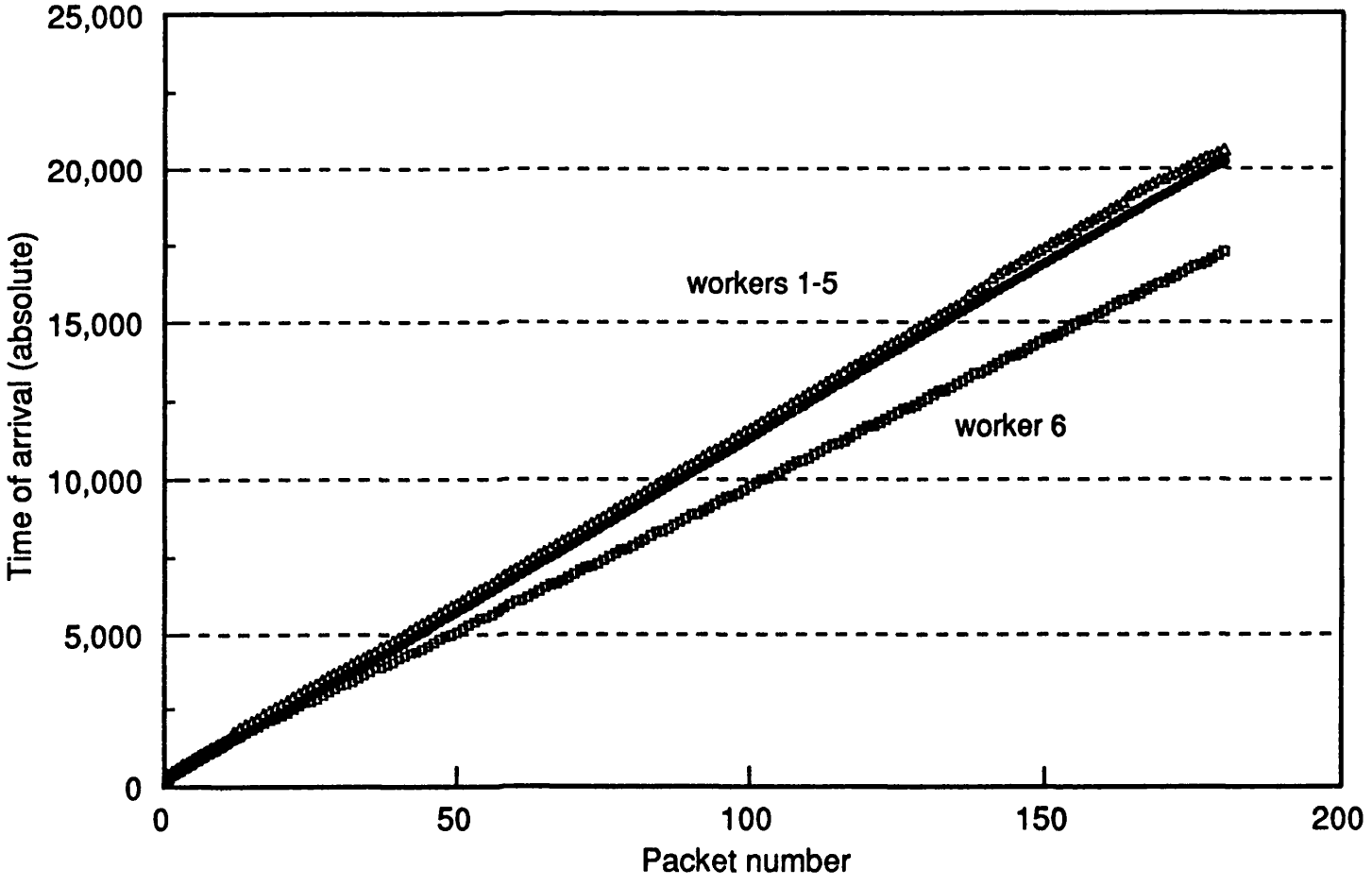


Fig. 8.12 Simulated arrival times at receptor site

Two parallel communication modules

CHAPTER 9

CONCLUSIONS

A Summary of main results.

White noise filtering methods for the generation of wave motion at gridpoints in a region of interest are discussed in this thesis. These methods allow a continuous simulation run of arbitrary length to be made. For the generation of surface elevation at the reference point, a simulation program has been developed based on a method due to Spanos. This method employs short ARMA filters (order 4-5) to generate time histories which possess the prescribed (Pierson - Moskowitz) Power Spectrum.

In contrast to the majority of the digital filtering methods where correlation matrices are matched in order to simulate the wave field, in the present work the random field is generated by using the transfer functions of the medium. With regard to the problem of transmitting wave elevation to another point by digital filtering, earlier research has used a FIR filter based on the impulse response function obtained by the transform method. In general, this type of filter has considerable length. The filter lengths required vary with the distance of transmission, but over 75 coefficients are required in all cases. More compact filters of the recursive (ARMA) type have been developed for the above problem, using optimization in the time-domain, reducing the number of coefficients roughly by a factor of 5. The competence of these filters is also demonstrated by comparing the error committed by the recursive filter as designed here, with the error of a FIR filter of the same length. Such comparisons are given here in the

form of graphs; these graphs show that the recursive filter enjoys an order-of-magnitude advantage in the error measure. Tables of worked-out filter coefficients are presented for a variety of horizontal transmission distances.

In addition, a frequency-domain design formulation based on allpass filters has been investigated. Insistence on the exact matching of the magnitude characteristics of the designed filters led to limited success. The phase could be matched over a limited bandwidth only and additional bandpass filters had to be introduced to suppress the rest. In a revised form, the frequency domain design techniques might be competitive with the time domain design, but more work is needed to explore that possibility.

The impulse response function for the transmission of wave kinematics as obtained by the transform method using linear Airy's wave theory has been compared with a function derived directly from the hydrodynamical theory and their equivalence has been demonstrated (this is presented in Appendix A).

The transfer function for the scattered waves around a large vertical cylinder was derived using first order diffraction theory and digital filters were constructed to implement the necessary convolutions. The quality of the filters was shown in terms of the wave runup which they generate around the cylinder.

The quality of the generated time histories for the overall simulation scheme, for all the components of the wave kinematics, has been demonstrated by evaluating their statistical and cross-statistical moments at various points of a spatial grid. The very good agreement with the corresponding theoretical values was shown.

A parallel implementation was made of the simulation program on an eight transputer board hosted in a PC. A pipeline and a binary tree topologies were tried. A processor farming combining elements of geometric parallelism was used. The

efficiency of the distributed program reached a maximum of 0.75 for the pipeline and 0.90 for the tree topology. The performance was found to depend on a number of programming decisions. In particular the number of time steps of the generated kinematic quantities that were buffered before communicating with neighbouring processors governed the nature of the flow of data to the collector. Disruption and sometimes a total blockage of a number of processors could result from a non-optimum packeting strategy.

B Further research.

Although the simulation of the wave field was based on linear (Airy's) theory, the method used need not necessarily remain confined there. The undisturbed wave field can be simulated under second order perturbation (Stokes) theory. The second order term in the equation of a kinematic quantity can be considered as an additional filter in cascade with the first order filter. Even more important is the incorporation of second order diffraction theory in the simulations of the waves around a large body. Isaacson and Cheung (1991) are using an integral equation method based on a Green's theorem to obtain a second order solution for the field around a large cylinder. The time-invariant coefficients which they obtain for regular waves are then used in convolutions for time domain simulations. This can be extended to cover the case of the multi-directional random seas examined in the present thesis. The filter design methods developed here can be utilized resulting to high speed simulations.

The recursive digital filters designed in this work correspond to a particular predetermined time step. Certain restrictions could follow from this fact. For example the generated time histories cannot be fed as inputs to a structure whose impulse response function needs to be sampled at a different timestep; also the

energy contents of higher frequency components, which could be important for certain structures, are omitted. This problem can be remedied with the introduction of digital *interpolators* or *decimators*, if smaller or larger sampling interval is required respectively. Li and Kareem (1990) have used these additional filters in simulating the wind field. In their case the *trigonometric* interpolators proposed by Saunders and Collings (1980) were used due to their very short length. The same interpolators could be used in conjunction with the filters developed in the present work.

Finally the experience acquired from the organization and the study of the performance of the distributed program can be utilized in the parallelization of more general applications. The data flow management, although applied in this work to a farming application, is also useful in the simulation of MDOF dynamical systems. These systems can be described by multiple data stream inputs passing through a chain of component processes and emerging as multiple outputs. For example, the wave kinematical quantities generated from the present program can be used to generate forces at the nodes of an offshore structure and in chain the output can be fed into a dynamic analysis program to produce response time histories at selected points on the structure. Similar examples exist in transport studies, gas distribution network analysis and other engineering applications.

REFERENCES

REFERENCES.

1. Abramowitz M. and Stegun I. A. (eds) (1970), Handbook of Mathematical Functions, Dover Puplications, 9th ed.

2. Bertran,M. (1975), "Approximation of digital filters in one and two dimensions", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-23, pp438-443.

3. Borgman,L.E. (1969), "Ocean wave simulation for engineering design", Journal of the Waterways and Harbors division, Proc. Of the ASCE, 4,pp.557-583

4. Borgman,L.E. (1979), "A technique for computer simulation of ocean waves", ASCE, Proc. Of the Specialty Conf. On Probabilistic Mechanics and Structural Reliability, Ang and Shinozuka,M. Eds., Tuscon, pp.200-204

5. Bowler K. C., Kenway R. D., Pawley G. S. and Roweth D. (1987) An Introduction to Occam 2 Programming, Studentlitteratur, Chartwell-Bratt.

6. Bratley,P., Fox,B.L., and Shrage,L.E. (1983), A Guide to Simulation, Springer-Verlag.

7. Brophy,F. and Salazar,A.C. (1974), "Recursive digital filter synthesis in the time domain", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-22, pp45-55.

8. Bryden,I.G. And Greated,C.A. (1984), "Generation of three-dimensional random waves", J.Phys.D:Appl.Phys. **17**, pp.2351-2366

9. Bryden,I.G. And Linfoot, B.T. (1991), "Generation of Short Crested Seas in Laboratoty Wave Tanks", Proceedings First International Offshore and Polar Engineering Conference, Vol III, Edinburgh.

10. Burke,B.G. And Tighe,J.T. (1972), "A time series model for dynamic behaviour of offshore structures", Society of Petroleum Engineers Journal, pp.156-170

11. Burns A. (1988) Programming in Occam 2 Addison-Wesley Publishing Company.

12. Burrus,C.S. and Parks,T.W. (1970), "Time domain design of recursive digital filters", Digital Signal Processing, ed. Rabiner,.L and Rader,C. IEEE Press, pp138-142.

13. Cadzow, J.A. (1976), "Recursive digital filter synthesis via gradient based algorithms", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-24, pp349-355.

14. Chen, C.T. (1979), One-Dimensional Digital Signal Processing, Marcel Dekker.

15. Claerbout, J. F. (1985), Fundamentals of Geophysical Data Processing, Blackwell Scientific publications, 2nd Ed.

16. Dean R. G. and Dalrymple R. A. (1984), Water Wave Mechanics for Engineers and Scientists, Prentice- Hall Inc.

17. Dommermuth,D.G., (1986), "Time series analysis of ocean waves", M.I.T Report J.K.Vandiver.

18. Downie M. J. and Bettess P. (1989), "The discrete vortex method and parallel processing using transputers", Engineering Computations, Vol 6, pp. 142-150.

19. Eatock Taylor R. and Hung S. M. (1987), "Second order diffraction forces on a vertical cylinder in regular waves", Applied Ocean Research, Vol 9, pp. 19-30.

20. Elgar,S.,Guza,R.T. And Seymour,R.J. (1985), "Wave group statistics from numerical simulations of a random sea", Applied Ocean Research 7(2),pp.93-96

21. Fenton A. G. and Vanmarcke E. H., (1990), "Simulation of Random Fields via Local Average Subdivision", Proc. ASCE, Journal of Engineering Mechanics, Vol. 116, 8.

22. Gersch,W. and Liu,R.S-Z. (1976), "Time series methods for the synthesis of random vibration systems", Trans. of the ASME, Journal of Applied Mechanics, pp.159-165

23. Gersch,W. and Yonemoto,J. (1977), "Synthesis of multivariate random vibration systems: a two stage least squares AR-MA model approach", Journal of Sound and Vibration, 52,(4), pp.553-565

24. Gropp W. D. and Smith E. B. (1990), "Computational Fluid Dynamics on Parallel Processors", Computers & Fluids, Vol 18, 3, pp. 289-304.

25. E. Hart and S. Flavell (1990) "Prototyping Transputer Applications", in Real Time Systems with Transputers ed. H. Zedan, pp. 241-247, Amsterdam, Occam User Group, IOS Press.

26. Hastings-James,R. and Mehra,S.K. (1977), "Extensions of the Pade-approximant technique for the design of recursive digital filters", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-25, pp501-509.

27. Haykin S. (ed.) (1979, 1983), Nonlinear Methods of Spectral Analysis, Springer-Verlag.

28. Hogben N. and Standing R. (1974), "Wave Loads on Large Bodies", Proc. Int. Symp. on the Dynamics of Marine Vehicles and Structures in Waves, UCL, London, pp. 258 - 277.

29. Isaacson M. (1977), "Shallow Wave Diffraction around Large Cylinder", Journal of the Waterway, Port and Coastal Engineering Division, ASCE, Vol 103, pp. 69 - 82.

30. Isaacson M. (1978a), "Wave Runup around Large Circular Cylinder", Journal of the Waterway, Port and Coastal Engineering Division, ASCE, Vol 104, pp. 69 - 79.

31. Isaacson M. (1978b), "Vertical Cylinders of Arbitrary Section in Waves", Journal of the Waterway, Port and Coastal Engineering Division, ASCE, Vol 104, pp. 309 - 324.

32. Isaacson M. and Cheung K. F. (1991), "Time-Domain Solution of Second-Order Runup and Wave Forces", Proceedings First International Offshore and Polar Engineering Conference, Vol III, Edinburgh.

33. Isobe,M. Kondo,K. Horikawa,K. (1984), "Extension of MLM for estimating directional wave spectrum", Sym. on Description and Modelling of Directional Seas, Paper no. A-6, Danish Hydraulic Institute.

34. Jefferys, E. R. (1987), "Directional seas should be ergodic", Applied Ocean Research, Vol9, 4, pp. 186-191.

35. Kareem A. and Li Y. (1991), "Simulation of Multi-Variate Stationary and Nonstationary Random Processes: A Recent Development", Proceedings of the First International Conference in Computational Stochastic Mechanics, Corfu.

36. Kuck, D. J. (1982) "High-speed Machines and their Compilers", In Parallel Processing Systems (ed. D. Evans). Cambridge University Press.

37. Li, Y. And Kareem, A. (1990), "ARMA systems in wind engineering", Probabilistic Engineering Mechanics, Vol. 5, No. 2, pp 50-59.

38. Li, Y. and Kareem, A. K. (1989), "On Stochastic Decomposition and its Application in Probabilistic Dynamics", Proceedings of the 5th International Conference on Structural Safety and Reliability (ICOSSAR '89), San Fransisco.

39. Lin,N.K. And Hartt,W.H. (1984), "Time series simulations of wide-band spectra for fatigue tests of offshore structures", Trans. of the ASME, Journal of Energy Resources Technology, **106**, pp.466-472

40. Mignolet,M.P. And Spanos,P.D. (1987), "Recursive simulation of multivariate random processes", Parts I and II, Journal of Applied Mechanics, ASME.

41. Mignolet,M.P. (1987), ARMA Simulation of Multivariate and Multidimensional Random Processes, Ph.D. Dissertation, Rice University, Houston, Texas.

42. Miles,M.D. And Funke,E.R. (1987), "A comparison of methods for synthesis of directional seas", Proc.of the 6th Int. OMAE Sym., Houston, pp.247-255

43. Modi Jagdish J. (1988), Parallel Algorithms and Matrix Computation, Clarendon Press, Oxford.

44. Mullis,C.T. and Roberts,R.A. (1976), "The use of second-order information in the approximation of discrete time linear systems", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-24, pp226-237.

45. Nunes,D. (1987), "Generation of random waves in a model basin", Proc.of the 6th Int. OMAE Sym., Houston, pp.273-278

46. Oppenheim,A.V. and Schafer,R.W. (1974), Digital Signal Processing, Prentice-Hall.

47. Papoulis, A. (1984), Probability, Random variables and Stochastic processes McGraw-Hill, 2nd Ed.

48. Pountain D. and May D. (1988), A Tutorial Introduction to Occam Programming BSP Professional books, INMOS.

49. Pritchard, D. J. (1988), "Mathematical Models of Distributed Computation", in Parallel Programming of Transputer Based Machines, Proc. of Occam User Group, 14-16 Sept. 1987, Grenoble, France, ed. T. Muntean, IOS Amsterdam.

50. Rabiner,L.R. and Gold,B. Theory and Application of Digital Signal Processing, Prentice-Hall, ISBN No. 0-13-914101-4

51. Rabiner,L.R., Graham,N.Y. and Helms,H.D. (1974), "Linear programming design of IIR filters with arbitrary magnitude function", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-22, pp117-123.

52. Rebaudengo Lando L., Scarsi G., Taramasso A.C. (1991), "Statistical Properties of Wave Fronts In Directional Seas". Proceedings First International Offshore and Polar Engineering Conference, Vol III, Edinburgh.

53. Rivero M. G. (1987) "Random Wave Simulation Using ARMA Models". MSc. Thesis, MIT.

54. Rumelhart D. E. and McClelland J. L. (Eds.) (1986), Parallel Distributed Processing, (Explorations in the Microstructure of Cognition), Vols I & II, MIT Press.

55. Samaras, E. F. (1983), Time Series Generation Using the Auto-Regressive Moving-Average Model, Ph.D Thesis, Columbia University.

56. Samaras,E., Shinozuka,M., And Tsurui,A. (1985), "ARMA representation of random processes", Journal of Engineering Mechanics, **111**, (3), pp.449-461

57. Samii,K., And Vandiver,J.K. (1984), "A numerically efficient technique for the simulation of random wave forces on offshore structures", Proc. Of the 1984 Offshore Technology Conference, Paper No. OTC 4811, pp. 301-308

58. Sarpkaya T. and Isaacson M. (1981). Mechanics of Wave Forces on Offshore Structures. Van Nostrand Reinhold, New York.

59. Saunders, L. R. and Collings, G. (1980), "Efficient Solution of Model Equations with Arbitrary Loadings", Engineering Structures, 2, pp. 35-48

60. Schueller,G.I. And Shinozuka,M. (Eds.) (1987), Stochastic Methods in Structural Dynamics Martinus Nijhoff.

61. Shinozuka,M. And Jan,C.M. (1972), "Digital simulation of random processes and its applications", Journal of Sound and Vibration, 25(1), pp.111-128

62. Sneddon,I.N. (1951), Fourier Transforms McGraw-Hill.

63. Sorensen,J.D. and Thoft-Christensen,P. (1985), "Modelling and simulation of wave loads", Instituttet for Bygningsteknik, Aalborg University, ISSN 0105-7421 R8503

64. Spanos,P-T.D. and Hansen,J.E. (1981), "Linear prediction theory for digital simulation of sea waves", Trans. of the ASME, Journal of Energy Resources Technology, **103**, pp.243-249

66. Spanos,P-T.,D. (1983), "ARMA algorithms for ocean wave modelling", Trans. of the ASME, Journal of Energy Resources Technology, **105**, pp.300-309

67. Steiglitz,K. (1970), "Computer-aided design of recursive digital filters", in Digital Signal Processing, ed. by Rabiner,L. and Rader,M., IEEE Press, pp 143-149.

68. Steiglitz,K. (1977), "On the simultaneous estimation of poles and zeros in speech analysis", IEEE Trans. Acoust., Speech, Signal processing, Vol. ASSP-25, pp229-234

69. Wang,H.T. (1986), "Simulations of velocities and slopes of random ocean waves", Proc. Of the 5th Int. OMAE Sym., Tokyo, pp.382-389

70. Wexler J. (1989) Concurrent Programming in Occam 2, Ellis Horwood.

71. Wu G. X. and Eatock Taylor R. (1989), "The second order diffraction forces on horizontal cylinders", J. of Hydrodynamics, Vol 1, pp. 55-65.

72. Yahagi,T. (1981), "New methods for the design of recursive digital filters in the time domain", IEEE Trans. Acoust., Speech, Signal processing, Vol.ASSP-29, pp245-253.

73. Yarimer E. and Solomonidis C. (1991), "Random Wave Simulation by Digital Filtering Using an Industry-Standard Parallel Computing Facility", Proceedings ISOPE-91 International Offshore and Polar Engineering Conference, Vol. III, Edinburgh, 11-16 August 1991.

74. Yarimer E. and Solomonidis C. (1991), "Random field simulation by the digital filtering method", Proceedings Transputer Applications 91, Glasgow, 28-30 August 1991.

USER GUIDES - MANUALS

1. Parallel Fortran User Guide (1988), 3L Ltd., Scotland.

2. Transputer Reference Manual (1988), INMOS Ltd., Prentice-Hall.

3. Transputer Development System (1988, 1990), INMOS Ltd, Prentice Hall

4. TMB08 Installation and User Manual, Transtech Devices Ltd, Bucks.

APPENDIX A

IMPULSE RESPONSE FUNCTION BASED ON HYDRODYNAMICS.

A.1 Transmission of an initial disturbance.

An impulse response function for the problem at hand may be derived also from physical considerations in the time domain. Thus one may take h(t) to be the solution of a hydrodynamical initial-value problem involving a displacement pulse at x=0 at time t=0.

Sneddon (1951) gives the solution of the following problem: given an initial elevation of the sea surface at a location, to predict the subsequent motion at a distance x from this point, using the equations of hydrodynamics. If the spatial distribution of the initial elevation is f(x) at time t=0 ,then the elevation of the free surface at any subsequent time t, at a distance x is :

$$\eta(t)=\frac{1}{\sqrt{2\pi}}\int_0^{\infty} F(\xi)\left[\cos(\xi x+g^{1/2}\xi^{1/2}t)+\cos(\xi x-g^{1/2}\xi^{1/2}t)\right]d\xi \qquad \text{(A.1.)}$$

In this equation $F(\xi)$ is the spacewise Fourier transform of f(x). The integral represents an inverse transform operation from ξ to the space variable x at fixed time t. The result will be the surface elevation at location x as a function of time, which, for the time being, will be taken to be the impulse response at x due to the initial disturbance at t=0. Further discussion of this point will be made below.

When the initial raised profile is confined to the immediate vicinity of the origin, $f(x)=\varepsilon\delta(x)$, the above integral can be evaluated exactly:

$$\eta(t)=\varepsilon t\sqrt{\frac{g}{2\pi x^3}}\left[\cos\left(\frac{gt^2}{4x}\right)C\left(\frac{1}{\sqrt{\pi}}t\sqrt{\frac{g}{2x}}\right)+\sin\left(\frac{gt^2}{4x}\right)S\left(\frac{1}{\sqrt{\pi}}t\sqrt{\frac{g}{2x}}\right)\right] \qquad \text{(A.2.)}$$

where ε represents the intensity of the delta function.

If however there is a certain spatial distribution of the initial elevation f(x), with spatial Fourier transform $F(\xi)$, one can use Kelvin's stationary phase approximation, which yields the following value for the integral of

$$F(\xi)\cos\{\xi[x - tf(\xi)]\}$$

over ξ :

$$\frac{F(\xi)\cos\left(\xi^2 tf'(\xi) + \frac{1}{4}\pi\right)}{\sqrt{-2\pi t[2f'(\xi) + \xi f''(\xi)]}}$$

evaluated[1] at $\xi = gt^2/4x^2$. Using this result, Sneddon obtains:

$$\eta(t) \approx \left(\frac{gt^2}{2x^3}\right)^{1/2} F\left(\frac{gt^2}{4x^2}\right)\cos\left(\frac{gt^2}{4x} - \frac{\pi}{4}\right) \qquad \textbf{(A.3.)}$$

In the case where the initial distribution has a Gaussian shape:

$$f(x) = \frac{e^{-x^2/2\varepsilon^2}}{\varepsilon\sqrt{2\pi}} \quad \Leftrightarrow \quad F(\xi) = e^{-\xi^2\varepsilon^2/2} \qquad \textbf{(A.4.)}$$

the result is:

$$\eta(t) = \sqrt{\frac{gt^2}{2x^3}}\, e^{-\varepsilon^2 g^2 t^4/32x^4}\cos\left(\frac{gt^2}{4x} - \frac{\pi}{4}\right) \qquad \textbf{(A.5.)}$$

Eq. A.2 may also be used when the initial displacement is a delta-function: $f(x) = \varepsilon\delta(x)$ One gets the result:

---

1 This value of ξ renders the phase $\xi[x - tf(\xi)]$ stationary with respect to ξ, and it yields the largest contributions to the integral.

$$\eta(t) = \varepsilon\left(\frac{gt^2}{4\pi x^3}\right)^{(1/2)} \cos\left(\frac{gt^2}{4x} - \frac{\pi}{4}\right) \qquad \text{(A.6.)}$$

It is noted that the solutions based on Kelvin's approximation lead to small errors for $t/\sqrt{x} \ll 1$ (Sneddon).

A.2 Equivalence of the hydrodynamical impulse response functions with the transmission filter of Chapter 3.

Observing that there is a factor t in front of expressions (A.2) and (A.5 or A.6), it is not apparent at first sight whether they are equivalent to Eq.3.10. The Fourier sine and cosine transforms of the above functions were evaluated in order to examine their frequency characteristics.

$$I_1 = \int_0^{\infty} h(\theta)\cos(\omega\theta)d\theta$$

$$I_2 = \int_0^{\infty} h(\theta)\sin(\omega\theta)d\theta \qquad \text{(A.7.)}$$

Written in terms of I_1 and I_2, the transfer function $H(\omega)$ of the system under consideration is : $H(\omega) = I_1 + iI_2$, and this should have the properties:

$\sqrt{I_1^2 + I_2^2} = 1$ and $\tan^{-1}(I_2/I_1) = -\omega^2 x/g$

In the calculation of I_1 and I_2 one needs the function h(t), or $\eta(t)$ in the present notation: as discussed above, in a few special cases this is available in closed form. If not, numerical integration of Eq.A.1 must be carried out at successive values of t. One set of 1400 h(n) were obtained in this way; for $F(\xi)$ as given in Eq.A.4 this set of digitised h(n) values is shown in Fig A.1.

In most cases there exists no analytical solution for the integrations for I_1 and I_2 even when $\eta(t)$ is available in closed form. The fast Fourier transform can be used with values of $\eta(t)$ discretised on a dense grid. It is necessary to keep the interval Δt very small because of the highly oscillatory nature of the function $\eta(t)$ at large t. In this study, numerical integration of Eq. (A.1) was performed. Although this is less efficient than the FFT, in a case such as this it would allow changing the stepsize progressively along the t axis according to need.

i. Using approximate expression for h(t)

The first case examined is Kelvin's approximation assuming an initial displacement f(x) of infinitely small length (Eq.A.2). In order to obtain convergence it was necessary to replace the delta function type spatial distribution of the initial displacement by a more spread shape. The Gaussian-shaped f(x) already introduced in Eq.A.3 was used. The resulting h(t) is shown in Fig. A.2, and is seen to resemble the exact function of Fig. A.1 closely.

Values of the recovered phase ($\tan^{-1}(I_2/I_1)$) over the expected are plotted against a range of frequencies ω, for five values of ε, in Fig. A.3. The magnitude $\sqrt{I_1^2+I_2^2}$ is shown in Fig. A.4 for ε=0.2. It can be seen that the phase behaviour of this filter is correct and almost independent of the parameters of the initial rise f(x), but the magnitude, which should be a constant, varies instead as a linear function of ω. It will be shown later that a further transformation removes this discrepancy.

ii. Using exact expression for h(t)

The frequency domain characteristics of the exact solutions given in Eqs. A.1, A.2 are also examined using similar methods. The function h(t) for

delta-shaped spatial distribution of the initial rise of the water surface is given in Eq.A.2. When introduced into the integrals I_1 and I_2 it will yield the expressions:

$$I_1 = A(I_{1a} + I_{1b})$$

$$I_2 = A(I_{2a} + I_{2b})$$

where $$A = \frac{\varepsilon}{x}\sqrt{\frac{g}{2x\pi}}$$

The integrals that appear in these are:

$$I_{1a} = \int_0^{\infty} \theta \cos\left(\frac{g\theta^2}{4x}\right) C\left(\frac{1}{\sqrt{\pi}}\sqrt{\frac{g}{2x}}\right) \cos(\omega\theta) d\theta$$

$$I_{1b} = \int_0^{\infty} \theta \sin\left(\frac{g\theta^2}{4x}\right) S\left(\frac{1}{\sqrt{\pi}}\sqrt{\frac{g}{2x}}\right) \cos(\omega\theta) d\theta$$

$$I_{2a} = \int_0^{\infty} \theta \cos\left(\frac{g\theta^2}{4x}\right) C\left(\frac{1}{\sqrt{\pi}}\sqrt{\frac{g}{2x}}\right) \sin(\omega\theta) d\theta$$

$$I_{2b} = \int_0^{\infty} \theta \sin\left(\frac{g\theta^2}{4x}\right) S\left(\frac{1}{\sqrt{\pi}}\sqrt{\frac{g}{2x}}\right) \sin(\omega\theta) d\theta$$

However, these integrations would not converge. Clearly, it is necessary to spread the delta-function initial rise thus introducing a "damping" effect in the filter. As before a Gaussian-shaped distribution is the best choice. When the spacewise Fourier transform of f(x) is introduced into Eq.A.1, h(t) becomes:

$$h(t) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} e^{-\xi^2\varepsilon^2/2} \cos(\xi x) \cos(\sqrt{g}\, t\sqrt{\xi}) d\xi \qquad \text{(A.8.)}$$

where ξ is the spacewise Fourier transformed domain of x.

The integration was performed numerically by fixing t at closely spaced points, yielding the h(n) values shown in Fig.A.1. Thereafter I_1 and I_2 were found by a further numerical integration.

iii. Results.

When the integrations were carried out, either using the function in Eq.A.2 (exact solution, using a delta-shaped initial displacement *f(x)*), or the functions defined by Eqs.A.5 or A.6 (using Kelvin's approximations), it was found that the phase behaviour of the filter was correct and almost independent of the parameters of the initial rise f(x), but the magnitude, which should be a constant, varied instead as a linear function of ω. It will be shown below that a further transformation removes this discrepancy.

A.3 The use of an integrator in cascade.

In all the preceding cases the expressions from hydrodynamical theory failed to match the target impulse response function because the magnitude of their transfer functions was not unity, but increased linearly with ω. It was found that this problem can be eliminated by cascading one of the previous expressions with an *integrator*, suggesting that the impulse used by Sneddon is a higher order singularity than the impulse defined from Eq.3.10 through the inverse Fourier transform. The frequency characteristics of an integrator are:

$$|H(\omega)| = \frac{1}{\omega} \quad \text{and} \quad arg\,[H(\omega)] = -90^{o}$$

In the time domain, taking into account the causality of the system, this corresponds to the integration:

$$h_1(t) = \int_0^t h(\tau)d\tau$$

In the numerical verification, the $\eta(t)$ function that was used in conjunction with an integrator was the expression from Kelvin's approximation for a delta-shaped distribution of the water rise, Eq. A.6.

Integrating $\eta(t)$ according to the preceding discussion, the new impulse response function $h_1(t)$ is, after manipulation, given by the expression:

$$h_1(t) = \varepsilon\sqrt{\frac{1}{2g\pi x}}\left[\sin\left(\frac{gt^2}{4x}\right) - \cos\left(\frac{gt^2}{4x}\right) + B\right] \qquad \text{(A.9.)}$$

where B is the constant of the integration. B must be taken as 0 in order to obtain oscillation with a zero mean value at large t. When compared with Eq.3.10 it is seen that this function is locally phase-shifted by $\pi/2$, but otherwise has the same general form. The phase shift may be explained as a by-product of the integration operation. Furthermore, the amplitudes of the two expressions have precisely the same x- dependence. Choosing the constant $\varepsilon = g$ makes the amplitudes agree exactly.

It was thus demonstrated that after a further integration, the impulse response function derived by Sneddon was equivalent to the expression used in the main body of the paper (Eq.3.10). Conversely, differentiating Eq.3.10 results in a function similar to the exact solution of the discussed hydrodynamic problem (Eq.A.2).

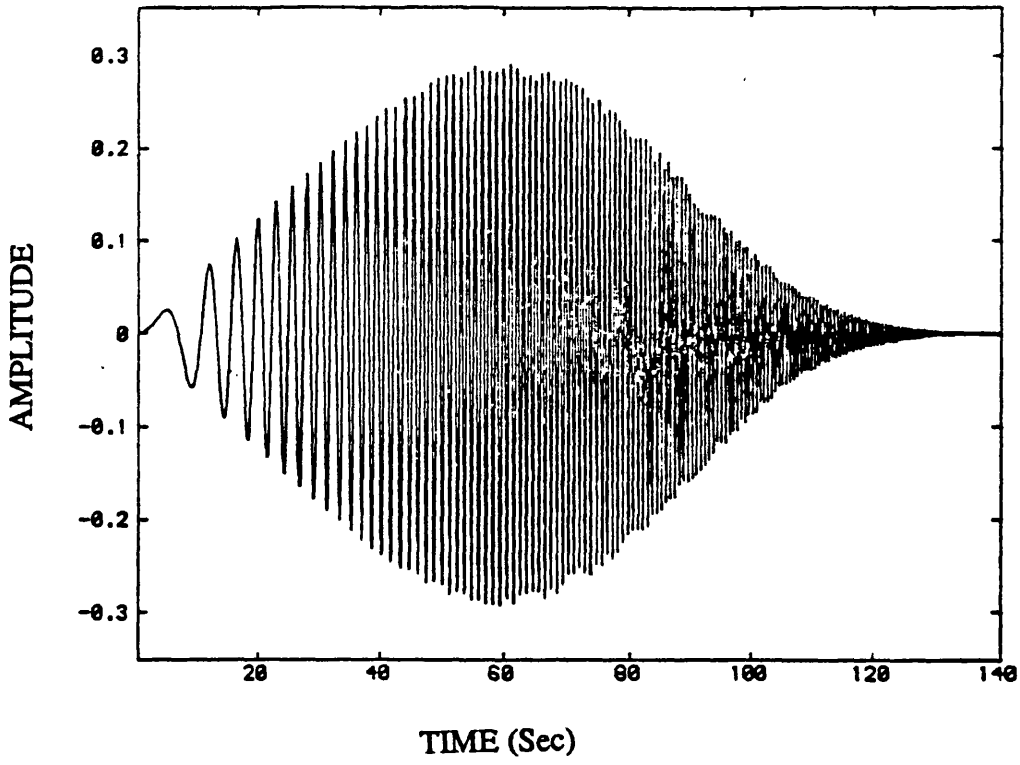


Fig. A. 1 Evolution of surface elevation from hydrodynamical theory: exact h(t) expression, exponential initial profile ($\varepsilon = 0.2$)

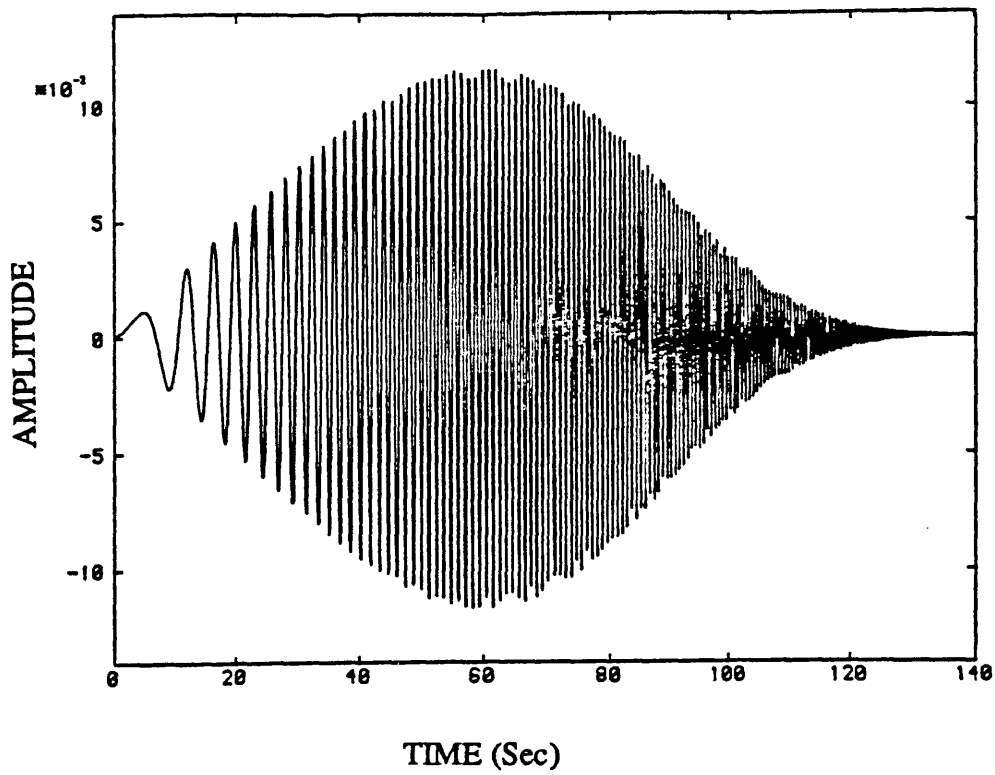


Fig. A .2 Evolution of surface elevation from hydrodynamical theory: Kelvin's approximation, exponential initial profile

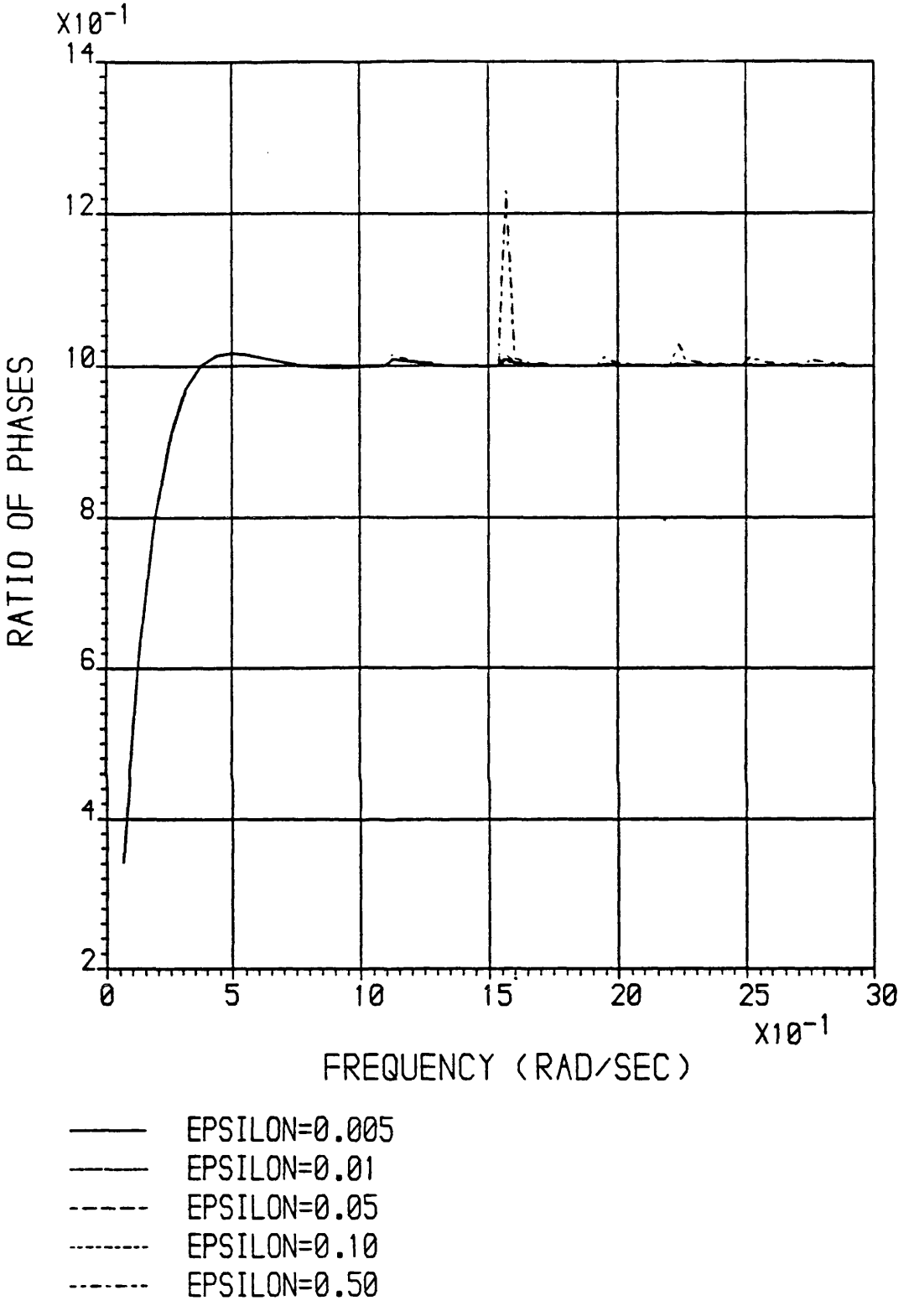


Fig. A .3 Phase angle ratio $\theta_{hydr.}/\theta_{true}$ for the function of Fig. A .2

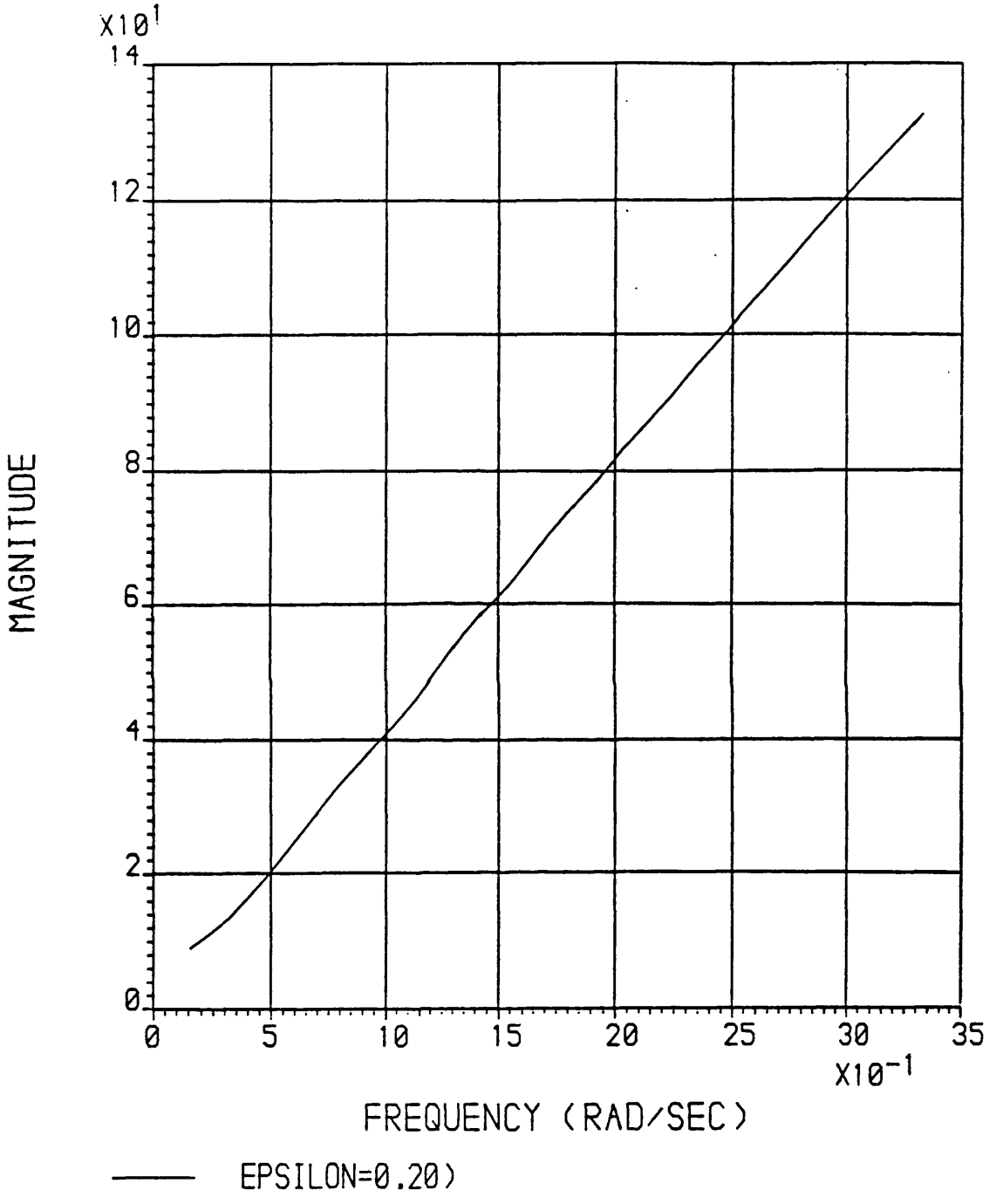


Fig. A .4 Magnitude of transfer function corresponding to Fig. A .2

APPENDIX B

COEFFICIENTS OF THE RECURSIVE FILTERS FOR THE HORIZONTAL TRANSMISSION OF WAVE MOTION.

COEFFICIENTS OF THE RECURSIVE FILTERS FOR THE HORIZONTAL

TRANSMISSION OF WAVE MOTION.

B.1 Table 1: Filter coefficients for bandwith π

(Left: numerator coefficients Right: denominator coefficients)

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
12	12	1.00	1.00
-4.69532230589908492E-02		1.00000000000000000	
-1.05543978313235101E-02		1.47268568661014210	
4.23845652584330171E-02		-0.57080754285548318	
4.20422301613034219E-02		-1.53480780908312608	
1.52452636480244208E-02		-0.26282486699901342	
0.76369540348789522		0.32949996278242438	
1.51688805832475992		0.10880270460972380	
-0.20617816785144391		6.93940356919238357E-03	
-1.64624366366445507		1.29591163130827699E-03	
-0.47042226054720881		2.11460418458079289E-04	
0.38818686137666841		5.88088177172146296E-05	
0.15046728607480189		2.12038329764773890E-05	
Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	2.00
-4.49965978442761275E-02		1.00000000000000000	
-1.44565936297092901E-02		1.58467552700575398	
2.02103747700298104E-02		0.14928766243118990	
4.22301597663867628E-02		-0.67625540510429005	
6.77390769187590364E-02		-0.25660067520091029	
0.61445265492425427		2.97052807158362693E-02	
1.56572376303188299		8.73054650248542181E-03	
0.67516742682308461		1.04148923084748895E-03	
-0.74546160587279331		8.19591990095055035E-04	
-0.46033010637536481		6.90813787522601846E-05	
Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
7	7	1.00	3.00
-4.69356409644690678E-03		1.00000000000000000	
8.50490663745630954E-03		0.42838452733527799	
1.42165071749559093E-03		7.23953278975487530E-02	
3.23668635120106171E-02		3.20125657140086575E-02	
7.20374968051221698E-02		2.00783700535178298E-03	
0.42876917769292222		8.37089006363874941E-03	
0.99975517689287974		-1.50901371375921300E-03	
Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
8	8	1.00	4.00
3.41614358137122681E-02		1.00000000000000000	
8.21414807240342798E-03		1.20450236118922605	
2.58939242103883617E-02		0.39907485464327708	
2.58068817275981695E-02		0.12446607344557690	
0.13199469531316191		2.92038892797791688E-02	
0.40129463375493068		1.94221129726583411E-02	
1.23035678728802100		4.77921348519758701E-03	
0.96636822073548345		9.90141582803002497E-03	

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	5.00

3.95558777514402327E-02	1.00000000000000000
7.05458379432700837E-03	1.12348887822885590
5.85695802296221698E-03	-0.29681195478783862
2.34588546742392212E-02	-0.59493776269657761
0.11918931557314299	-0.21764383279398999
0.34120135424406278	-7.12531462814301048E-02
0.97265736971523609	-1.73583551306207284E-02
0.82553132680373598	-8.50761586112247503E-03
-0.68466925577266668	-2.01357348180841115E-03
-0.72651810395017202	-2.55067840004383596E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
8	8	1.00	6.00

1.23452751889479902E-02	1.00000000000000000
5.56622398504228160E-03	0.85660040561735051
2.44035132027885203E-02	0.32830391753756832
3.61808025828869381E-02	0.12601284926091469
0.12671184616220910	3.66656655232197176E-02
0.32786471030236702	2.32218427616781298E-02
0.85740004116163249	5.01093132628596238E-03
0.99933183643085210	8.34185179333149333E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	7.00

-1.92226649244142712E-02	1.00000000000000000
9.11625454903083714E-03	1.17729374454253399
1.78072755239688692E-02	0.24468846479258779
5.59528006067826003E-02	-3.44747318278263368E-02
0.12179187774839870	-3.66358816189746306E-02
0.35429392974369250	-4.96142735398122717E-03
0.80545396822941673	-2.27624145680369603E-03
1.18976732698048204	3.55299106670648292E-03
0.21775705520259309	8.53412929179285496E-04
-0.43896951480784080	3.83902478260005108E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	8.00

-2.69850345694832988E-02	1.00000000000000000
-2.21414408762932308E-02	2.36108090203055720
1.57285121793678886E-02	2.17753682750662580
8.14807962048055279E-02	1.16804749183644296
0.20439039880817680	0.49769397845240731
0.52470908050279663	0.19990445224590320
1.22152023423502198	8.65904361198799599E-02
2.24656725274863112	3.87841630085096201E-02
2.30630796183824094	1.62505617903053699E-02
0.88529108432193593	3.62768576291588700E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	9.00

-6.74767778178530543E-03	1.00000000000000000
1.11791370706163708E-02	1.35739198066383393
2.39566401733555812E-02	0.86083996314931632
6.94696851541994481E-02	0.41636938380849492
0.14744299754726639	0.16995019065674680
0.37140239737233632	7.91722233659416102E-02
0.79331610577314993	3.26699107262058974E-02
1.34122185494397494	2.12774299634696899E-02
1.09212265808676201	8.87475139277002154E-03
7.25200614289924633E-02	8.88756166908822101E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	10.00

2.10149259794562594E-02	1.00000000000000000
3.42672061605878500E-02	2.06008736132446302
5.49463265765106831E-02	1.75796080767993290
9.58747175882210811E-02	0.98690364827084887
0.20729887957444151	0.45455392715557841
0.46081501355584009	0.20174622058759359
1.00107139628188690	9.11581714905240481E-02
1.77200432830603294	4.43221121604425727E-02
2.05675998991344811	1.94658376333144804E-02
0.96465585353529859	6.44865076728785586E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	11.00

3.15800076272146485E-02	1.00000000000000000
4.03727940905923566E-02	1.89404947549847402
6.23598811170614498E-02	1.56582857363008099
9.53990102774347065E-02	0.90225506929586752
0.20449192944019859	0.42607836420911471
0.42749403993603341	0.20015030655521990
0.90760064860367473	9.15438414475731599E-02
1.56863753170190190	5.09384048352705121E-02
1.89524255187771695	2.57685991796753298E-02
0.98896168505717263	1.50718286592673202E-02

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
12	12	1.00	12.00

1.78196286049999604E-02	1.00000000000000000
3.08934135880240511E-02	1.92397270571079693
4.65415192554098980E-02	1.08943430882091996
8.34485879115859641E-02	-2.65085556610524693E-02
0.17735958704684929	-0.35174201599993382
0.37780881070087080	-0.26341571436296768
0.77119998714871008	-0.13265645163065301
1.29949671171018011	-5.83364648046069828E-02
1.46691857152845095	-2.36947495141698501E-02
0.43941768973266121	-9.80042339356018873E-03
-0.92386648091886925	-3.54616167253658885E-03
-0.66103007378700396	-1.06845452104171708E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
10	10	1.00	13.00

-5.40700784092021641E-03	1.00000000000000000
1.45403819150118803E-02	1.52204237617050908
3.44806876295525866E-02	1.24147218513103708
8.86787014773957466E-02	0.72074001411651511
0.17314340025416519	0.37443700813651720
0.37391112108585500	0.17193062116462421
0.72198059074150100	8.85194502744110490E-02
1.24096692590151503	3.19471862627626613E-02
1.52278111951212791	1.23971660549884299E-02
0.99758897490735399	-9.16733284544760628E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
11	11	1.00	14.00

-1.63677266151624513E-02	1.00000000000000000
-2.47406906600377207E-03	2.32157925402102094
3.64365514438674820E-02	2.44848150900961281
0.11411108801740530	1.74082697678756593
0.24349667166181341	0.98694187342007256
0.50744758802507706	0.50739918210495161
0.98837419878919153	0.24861853670889961
1.74610954827959808	0.12297585719496540
2.45044403886619389	5.37544231075955531E-02
2.31748168795581622	1.77440054419567492E-02
0.98794754823215769	-5.11649013633583283E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
11	11	1.00	15.00

-6.13518453648262928E-03	1.00000000000000000
1.22676826391590992E-02	2.14092941560373307
4.71471382750144608E-02	2.19575092185226017
0.11696385870051230	1.56508573929413597
0.23498892120336909	0.90195239319259302
0.47443332897989399	0.47496703283392538
0.90137133137729275	0.23738540616824280
1.56574893033418605	0.12029493801527399
2.19554195290080090	5.36840407543656070E-02
2.14040816682668300	1.94226139717906914E-02
0.99767383000642418	-2.14921268276071698E-05

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
13	13	1.00	16.00

1.47069615604722807E-02	1.00000000000000000
3.62228286230295293E-02	2.19673572681354701
6.48708432385131928E-02	1.75019583285427904
0.11380802276119940	0.52112310496131387
0.21460308391021771	-0.20037706311230821
0.41252238769687888	-0.32980211385168801
0.77192993664200649	-0.23388698613809511
1.29383318820168403	-0.12708663514367180
1.73538371180933892	-6.14148098329080974E-02
1.46197942965743599	-2.77119019943136390E-02
0.10867680469106671	-1.20925159335819807E-02
-1.11281025926240207	-4.58088124643636976E-03
-0.69483392089306584	-1.26424579556610604E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
13	13	1.00	17.00

2.70582338359747800E-02	1.00000000000000000
4.34923320460397431E-02	1.92001824442746294
6.20221888885393521E-02	1.23419297193490896
9.54683752013464199E-02	7.43610655764564282E-02
0.18210542386189049	-0.49242907396611452
0.34358693700147402	-0.49769489913477027
0.63656571983625088	-0.33205542670147059
1.03444064977661010	-0.18134918721607690
1.34309267349689598	-9.38759128080835087E-02
1.02918639433384107	-4.58521845662234820E-02
-0.17521134191388540	-2.53089404107510914E-02
-1.28515122947478400	-1.33942992259144800E-02
-0.81875174670777440	-7.01582926178319789E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
13	13	1.00	18.00

2.08812888655886195E-02	1.00000000000000000
3.43235804716459994E-02	1.69591654768021693
5.57782961099919719E-02	0.94189819064180302
8.75811005121371750E-02	-8.81142082920352565E-02
0.16574716339772880	-0.54821417251198024
0.30392188421865929	-0.51378617351700673
0.55231385634563890	-0.34295334046716241
0.87700937657932210	-0.18977914757100339
1.11928820052593103	-0.10173800470149380
0.83240718211795661	-5.06964977407754896E-02
-0.21329759823310709	-2.98315256832207311E-02
-1.24452542165340807	-1.61273198771339187E-02
-0.87070176747955086	-9.62348137536540166E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
13	13	1.00	19.00

3.38276002368297080E-03	1.00000000000000000
2.44095003295059995E-02	1.81850231239557303
5.66273608892473510E-02	1.53806677225798105
0.11193591673825499	0.79579951230781287
0.20377053373688289	0.27417474452653340
0.37575136536838849	5.22008064289509327E-02
0.66407519704859574	-1.23173260094186499E-03
1.08623307453417106	-6.51511091047668814E-03
1.49629307719448801	1.61936657954785904E-03
1.51917763337783307	3.83887245589236911E-03
0.76251708931682804	8.35480497256278071E-03
-0.33950680271608569	7.43548653713979688E-03
-0.56219146490087424	6.94154302202237058E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
14	14	1.00	20.00

-8.69093262600643436E-03	1.00000000000000000
9.91153682549430863E-03	2.51469288810188596
6.14731288183211488E-02	2.65928776778571585
0.14582777034405531	1.57026679364459709
0.27318962435091382	0.48264332783620190
0.49550904552487279	-4.71650100543880776E-02
0.86921111395472872	-0.16944322517025889
1.43442928058905705	-0.13283745541815251
2.06325403851053801	-7.69718032746775360E-02
2.33376119402800786	-3.62702625615874824E-02
1.62557153474769089	-1.44728754858503006E-02
1.08783996010826303E-02	-2.18736551279279101E-03
-1.10614415841713498	2.33542178635027121E-03
-0.65266881823846878	2.61996570630124289E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
15	15	1.00	21.00

-4.64923878478711895E-03	1.00000000000000000
1.30875576788533794E-02	3.24127031818761280
8.00717051531530250E-02	4.41652659016507787
0.19802947310303781	3.32285806286078111
0.37931381391291330	1.38427435854563607
0.67937186399341920	6.70088442603408302E-02
1.18486666709863608	-0.39646061771408741
1.96691340708373397	-0.39363767324712490
2.93736414739424578	-0.26213907047367779
3.63402515019070815	-0.14621907249541921
3.20011658858431192	-7.31131368448928870E-02
1.18237360505547806	-3.32790525361625622E-02
-1.20133279628445599	-1.29112985935892800E-02
-1.76958364872028495	-3.58538751448110897E-03
-0.67006126785002318	-3.74167681311507818E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
14	14	1.00	22.00

1.10399418458858607E-02	1.00000000000000000
3.47203141171343321E-02	2.03754582045943611
7.17586042663245494E-02	1.80029087123476095
0.12628539869540689	0.78545945755624946
0.21977994377208401	-2.17691868816873683E-02
0.38198424756494981	-0.33118985017616992
0.65262661002729816	-0.32612227575776448
1.03968196435787497	-0.22624243727269219
1.44794749700674008	-0.13293290006698971
1.57125794135925201	-7.28121050420606764E-02
0.99531271331504423	-3.78806129734073280E-02
-0.25684685566282822	-2.03235630147486010E-02
-1.17031249879376409	-9.94418608575187157E-03
-0.77228994554833885	-4.20030339479951584E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
15	15	1.00	23.00

2.37711509227650393E-02	1.00000000000000000
6.73236751734967442E-02	2.85654707742703318
0.12274017742690189	3.49936267855215588
0.19862816121138629	2.39374473814130218
0.33094411737208701	0.83119734867760142
0.56508622039827516	-0.14589535955077579
0.96320002427731100	-0.45364826837604327
1.56070382549662789	-0.40855619909429358
2.28623693354350088	-0.27415055270351402
2.79512822051216014	-0.15885344268461621
2.47300813277722620	-8.55037296683203857E-02
0.92918781111693705	-4.35091863633324111E-02
-1.03656871625137903	-2.13777861185456414E-02
-1.68926526785609710	-9.05808019577040470E-03
-0.73535145949016067	-2.76184283414986119E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
15	15	1.00	24.00

2.25301740406004709E-02	1.00000000000000000
6.05191223976863091E-02	2.62763176734353587
0.11049375108183430	2.98127254898884120
0.17787241970331999	1.83324441793647597
0.29540808208246028	0.41934692857171069
0.49820035459484607	-0.39070366585240968
0.83911963225483332	-0.59179859588820349
1.33907987260186490	-0.48873764008057990
1.93237373961239789	-0.32451538699449628
2.31949337045324189	-0.19100600012828631
1.99040661261908403	-0.10755025022588650
0.61409521148563684	-5.88871642426921915E-02
-1.13318440256024400	-3.34367792795275606E-02
-1.73066827225032793	-1.70880160876235102E-02
-0.79420153202204224	-6.59259465217719443E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
14	14	1.00	25.00

-1.87777223861855004E-03	1.00000000000000000
2.29557936975757007E-02	3.03421590202900004
9.10522802862601954E-02	4.77844200900450034
0.22530254755453999	5.19331830336130018
0.44041468362877001	4.46048079591519997
0.79232558239819995	3.25671243375840014
1.34643984053460009	2.14479265943900010
2.19351647319460019	1.31081250222340007
3.32878373073979983	0.76554114697353004
4.55361669889690024	0.41356476703196998
5.29920000838509964	0.19899123565102000
4.86777233069079962	6.42301043938145050E-02
3.08559512586690010	3.99003174309906991E-03
1.01131751518179991	-1.11930507291687995E-02

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
14	14	1.00	26.00

-2.59188234871081911E-03	1.00000000000000000
2.09173657003982189E-02	3.04165116684381420
8.72783034916623790E-02	4.79458274695270070
0.21934458138049470	5.21391833148908823
0.43117695098103820	4.47843787582443120
0.77765861913352408	3.26998765624617205
1.32299230396116707	2.15282645640483805
2.15663222112478703	1.31580762056330203
3.27405982614983904	0.76815570772268027
4.47967595344370473	0.41542411385965827
5.21333903582629787	0.19999339243112230
4.78767693951894913	6.52004044804768285E-02
3.03297666141172995	4.47312916975676437E-03
0.99268353809778198	-1.07919254821562203E-02

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
15	15	1.00	27.00

-2.74237048091811007E-03	1.00000000000000000
1.89028353546382005E-02	3.85283692554809987
0.10562192549927001	7.25024751248269972
0.29315959523385998	9.11212849655879964
0.61345273417276003	8.77011081768799983
1.12907712591340004	7.01987040308639987
1.94365565434400001	4.94637978120549970
3.19101150264449984	3.20251392527690015
4.94049127723789994	1.96342000233040004
7.01754574399650011	1.15887733771590007
8.76795557331490016	0.65404034913797004
9.10717421295209917	0.34007298889344001
7.24096822591879974	0.14918170491873001
3.84319115987700011	4.70681606766876967E-02
0.99502600936650998	7.09814798633038018E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
16	16	1.00	28.00

8.74405131440735700E-03	1.00000000000000000
4.34881210808127430E-02	2.92155563141523888
0.11172038203880710	3.97330745471097302
0.21733321928414601	3.30328641332564299
0.37517447942930893	1.75842053729136105
0.62256408238825489	0.44574661804754440
1.01095446661108701	-0.22933556604239469
1.58441161491469407	-0.40715170594933042
2.31502771524348594	-0.35274847074392801
3.00304194705782512	-0.24105849007591470
3.19898554509181787	-0.14572287288896729
2.37697764550006596	-8.13136139655268181E-02
0.51563455499793376	-4.27113911027998602E-02
-1.33413234449302198	-2.06805050843712999E-02
-1.73425840630435202	-8.53401702672064270E-03
-0.74553325185816210	-2.61417319032796087E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
16	16	1.00	29.00

2.10517622192657010E-02	1.00000000000000000
5.93970205265821824E-02	2.55804187534880478
0.11538654384827300	3.00100591460683086
0.18503278147558999	1.88264897361313910
0.29543125834288142	0.28602985292651611
0.46902317699574408	-0.80603139631183884
0.75019914000909404	-1.15464828392067509
1.14838716416094799	-1.04238628099817610
1.63057680650302395	-0.76105444853347837
2.00143669369005206	-0.50184690647465013
1.91121717888120402	-0.30799143823342989
0.96912366587319065	-0.19245920490428170
-0.71522804581606725	-0.12256179532616480
-2.17732599164489793	-8.68745439812570047E-02
-2.17354612988147178	-5.27977461321577080E-02
-0.91014590555694197	-2.34117239366515989E-02

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
17	17	1.00	30.00

2.32058882495259911E-02	1.00000000000000000
8.52104607023701083E-02	3.50990291559851819
0.18514234197618129	5.67149631833911005
0.32237382011169902	5.60994292047282439
0.52370145436914273	3.66466621017618088
0.83919970231162988	1.41584056099123101
1.34012766054940302	-7.46868098501969058E-02
2.08911113685184402	-0.67757927693525100
3.08736900620769594	-0.72779236168637340
4.15127825280017060	-0.56216636875157455
4.79908455345481944	-0.37256339440265918
4.30770956101133162	-0.22590231615513179
2.24628224931774012	-0.13036045565685470
-0.69612953607484829	-7.21387754908736495E-02
-2.63979005100791708	-3.76476243529610469E-02
-2.29690045370133822	-1.62754641897841501E-02
-0.76693806885132298	-4.48373239276315776E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
17	17	1.00	31.00

1.39757875024301304E-02	1.00000000000000000
6.07141375502020728E-02	3.35553691630337880
0.15096780684042679	5.31993525272759626
0.28905026849729121	5.29495955718216216
0.49311354407321828	3.61972583688518901
0.80316215086922360	1.64607378347688194
1.28106006134865003	0.28357373976654848
1.98555749137520299	-0.33307373218202557
2.92051467454067692	-0.46341299772432121
3.92919437426660112	-0.38425254949291809
4.59026548172489512	-0.26104657433144218
4.25996397715377295	-0.15793053375436220
2.51028726659908719	-8.82695452439714312E-02
-0.14290739913816070	-4.53322835902444724E-02
-2.08560411397620005	-2.08764410807206792E-02
-2.03126553858266812	-7.62871061711081679E-03
-0.73996897761381686	-1.84487026946845794E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
19	19	1.00	32.00

2.47892969902876016E-03	1.00000000000000000
3.44659116134487034E-02	3.33168607615830004
0.11638897218763999	4.57841274334580017
0.23970536924638999	2.97980225287709999
0.39640600738870002	1.84258889510791994E-02
0.61214288648024995	-1.96415685355809999
0.93463819610400001	-2.27820886172529979
1.40030847734030006	-1.62739037973889999
1.97643214548470003	-0.84740689402696001
2.48396616390580016	-0.31908879580080002
2.52896816440020000	-6.20095027640961000E-02
1.60104512371580010	2.76119746660545007E-02
-0.48571036962238001	4.20265247018518986E-02
-2.92621840132370004	3.27453151138695012E-02
-3.93672035060810011	2.05113266395648003E-02
-2.38303545245230008	1.14352926289581006E-02
0.28020371179274001	5.74970140326108020E-03
1.33155405817509997	2.58886237804565982E-03
0.57933224536275996	9.22208133355005956E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
18	18	1.00	33.00

-5.82623604434909887E-04	1.00000000000000000
2.73943443368645997E-02	4.11333984573290490
0.13220642077413020	8.06991049669687222
0.35046674069333811	10.22284386180042940
0.70425986400786056	9.50489795355893108
1.23582933548074991	6.98073836469129816
2.02566773018020996	4.25460539909571889
3.18225033331388918	2.24055283883024892
4.77551192724251017	1.06515751810441706
6.71198597039688316	0.48994823177452201
8.55370693270327998	0.24376596416846941
9.43668238502115209	0.14750666786659100
8.36357419994878448	0.10841058541600700
5.07058272131254295	8.72059314882825259E-02
0.88622799476933800	6.77139331542937622E-02
-1.79226873311226997	4.55702384613743405E-02
-1.81052330082515289	2.26417770780245713E-02
-0.59453713637749495	6.24597873052408579E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
18	18	1.00	34.00

7.32880924315118532E-03	1.00000000000000000
4.99034838842109599E-02	3.81399874529234983
0.15765713124153810	6.96582548075220487
0.34809710693236040	8.15492502557753340
0.63595841226257677	6.84506956493641461
1.05927995082171189	4.29971873132928462
1.68567932120748498	1.96607401282771810
2.59720727960401110	0.49754333141076118
3.83322448541413507	-0.16446389762797331
5.28949573881570512	-0.33634713573208769
6.58000922262163623	-0.29665840580062641
6.99233907577019309	-0.20231157595547919
5.75605277179521657	-0.11806529786815120
2.76517081228431705	-5.90881439191677932E-02
-0.74514347162070349	-2.37464557262233103E-02
-2.70347053606066590	-5.64417761660259473E-03
-2.21208582945565579	5.55467282014136354E-04
-0.70973750476935848	9.44374355326727888E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
18	18	1.00	35.00

1.86817296361789301E-02	1.00000000000000000
7.57032596828745435E-02	3.48857422376618187
0.18084305857376490	5.83521173920675462
0.33337778027372011	6.10140287111101731
0.54865874339418774	4.24974976587910369
0.86124613154762708	1.69873399614264398
1.32926008193359291	-0.25647629716895398
2.00603496762543898	-1.20824561817477205
2.90479245049759793	-1.37951875970121107
3.90774798302924520	-1.16371604722506694
4.67864090810920263	-0.84457130388620660
4.63691753569918141	-0.56496251043338252
3.22233568337367204	-0.36170227843602642
0.47913764072137482	-0.23053198783940759
-2.41400919778347411	-0.14619046149339471
-3.67378783509420614	-8.89769991243055114E-02
-2.65285292164986686	-4.37998974708184996E-02
-0.84197166888898101	-1.33918616125432693E-02

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
20	20	1.00	36.00

2.31720097062454895E-02	1.00000000000000000
8.71276305680308144E-02	3.49123245203524712
0.18162565416905871	5.17876831904890889
0.29086344774050049	3.83072752165553920
0.42835841393222301	0.44181675308050661
0.63217577507660128	-2.35290890432737498
0.94859482213565460	-3.21532738596455792
1.40451527892528594	-2.58907217324407402
1.97469079835628403	-1.50253017973190395
2.51499411841157405	-0.62446443970605869
2.70126405364091982	-0.12883553834372241
2.05238032332555509	7.32903435801771225E-02
0.21499440251182120	0.11744933310302860
-2.47953635763352009	0.10044262818833460
-4.61789790229859776	7.00998243196492010E-02
-4.42141855411852269	4.46905999953204902E-02
-1.69172345248376410	2.69644151440062603E-02
1.19663077029998410	1.52930875079528503E-02
1.77522423412731500	7.04720678872493376E-03
0.66184612237022189	1.99659855517424082E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
20	20	1.00	37.00

1.71856535941844989E-02	1.00000000000000000
7.13398675282705674E-02	3.40777831621792604
0.16496281113081759	5.12339649087765991
0.28828825872126779	4.16929118319411174
0.44758754303637849	1.31355459101216399
0.67189118830351313	-1.25680309892538311
1.00530791278655895	-2.32766530069648603
1.48054652314861590	-2.11921454162291401
2.08480787141698221	-1.39980597115801397
2.69604824199310311	-0.72794714067779520
3.02210022870823591	-0.29758152283638239
2.60773844859554504	-8.27909162033925034E-02
1.06757027789952108	3.71338444681047980E-04
-1.42229290398159502	2.19162856787945594E-02
-3.70351807472242989	2.04153730113131385E-02
-4.11313856244022436	1.39067563348152599E-02
-2.12855523925009793	7.82204332525671814E-03
0.47194613445010042	3.91372807315217269E-03
1.34824078295573591	1.49501558559477898E-03
0.58361146380986228	4.52761276325426012E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
19	19	1.00	38.00

6.77429511138021977E-03	1.00000000000000000
5.08959079498988998E-02	3.98718866766169988
0.16977253081103000	7.73485129219009959
0.38776067537212999	9.68150695015230056
0.71772333313903003	8.68752777029749978
1.18972231467809997	5.76170632654389969
1.86286609725790009	2.64898904451459982
2.81737948202209987	0.44862626620429002
4.10415376556730038	-0.64755833663801998
5.66013214126849995	-0.95514969524144999
7.18023972694379964	-0.86311760521484004
8.03676536825969912	-0.64444930282745005
7.39988196565099976	-0.43145264341681000
4.76009588154019969	-0.26978122041695002
0.68828553684025995	-0.15955576172156000
-2.92542403608810009	-9.01653378415339934E-02
-4.08288285092829994	-4.72251574985348005E-02
-2.68585067034350011	-2.24948259408781007E-02
-0.77346189652135999	-8.41627428217390020E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
20	20	1.00	39.00

1.73680736469232510E-03	1.00000000000000000
3.78027341372479120E-02	4.83039494535652914
0.17811890401429190	11.14996207553905980
0.49203423881858688	16.50781131404147930
1.02102141392259704	17.65305899705318990
1.80439571034435597	14.45162499673421050
2.91963191591536786	9.23353782722385930
4.49199991472392934	4.44506603468321160
6.64329927947994392	1.26562044897782799
9.37186583997674560	-0.31307502113890440
12.36041643308042910	-0.82316303267653279
14.78585943117060000	-0.80272756124526612
15.33478955166106060	-0.60205937537204657
12.73961939547795020	-0.39049223161363422
6.89161925357713390	-0.22630846998506751
-0.20342198601671810	-0.11628237072873160
-5.09459809484342330	-5.06565082810518513E-02
-5.60646444315950632	-1.68747437975978291E-02
-3.07000252465956702	-3.44919984131583299E-03
-0.72789574060314255	-1.69558926459048711E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
21	21	1.00	40.00

6.56883019087343284E-03	1.00000000000000000
4.99454021561091532E-02	3.84034935493199603
0.16056238010259741	6.67498594185290539
0.34215980262995199	6.68535052340701519
0.58381357167191983	3.64097282381834919
0.89735190517299757	-0.15465288110321160
1.32566847909159691	-2.55889681113220613
1.92118328464050991	-3.09636748463944000
2.69568843886409981	-2.47380435073423710
3.55446219492638482	-1.54486948786708700
4.21438262145989917	-0.79205207619032691
4.17587893900515184	-0.33194314167329991
2.86887601106819279	-0.10414857747536510
0.10479779421375431	-1.24821029350350599E-02
-3.32929318757076809	1.41553967575056300E-02
-5.58579679564225628	1.60870208853956496E-02
-4.96132390846037108	1.10278360071989802E-02
-1.84089579542981907	5.97098463597792086E-03
1.06254123214679708	2.40330177555922800E-03
1.58247116343971794	6.58769259606779454E-04
0.57965437450274504	1.29586055831342898E-05

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
21	21	1.00	41.00

1.65990442875528811E-02	1.00000000000000000
7.47044322401454464E-02	3.61818337963315884
0.18473574505557841	5.98238238937496725
0.33764935529747092	5.66873672894362102
0.53203142625003230	2.75539707088172392
0.79005521618217167	-0.59747962741817184
1.15564874451094490	-2.57436137735444115
1.66868729249337799	-2.88417389855884609
2.33163290778307619	-2.21639980362247702
3.05197157722471202	-1.33505453994320411
3.58135617629787983	-0.64791107861446862
3.48961311326089785	-0.24161486771945020
2.29924241911154503	-4.87184705379108268E-02
-0.13704769788357560	2.10860421980569694E-02
-3.12739396840142092	3.53227199907078723E-02
-5.07309850951714392	2.92502801746291098E-02
-4.47631960781683258	1.98705083866231903E-02
-1.63591540201703101	1.16573217064930602E-02
1.06173213545623302	6.46347395115143362E-03
1.58039580216473197	3.01272267736258205E-03
0.60703915634805505	1.43799581596033193E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
22	22	1.00	42.00

2.26121885942023315E-02	1.00000000000000000
0.10770007225501881	4.37659430084868983
0.27379095790216701	8.70431498069117993
0.50720156857585352	10.15096654279729020
0.80012736379817573	6.98886149137422397
1.18268743285605304	1.43366060760494896
1.72158927041999510	-3.08889602050740697
2.48794081412604484	-4.91355764715174459
3.51011468530377302	-4.49008057995794729
4.69927050717728356	-3.09230720926588321
5.75407905671224462	-1.71252042829673701
6.08946208814242507	-0.75866281003905633
4.93453402757772341	-0.23802744944100951
1.76461084657953204	-1.11289754484936408E-02
-2.95050402683139978	5.98523619666553527E-02
-7.25580219676333282	6.44717031284167547E-02
-8.47750804482391196	4.88367699708038075E-02
-5.50768174860633764	3.14918310485999414E-02
-0.58348286080236933	1.85563892144275690E-02
2.39356365089225287	9.81813769193719786E-03
2.05648800711289015	4.99189793278981796E-03
0.59191185061840512	1.87526527553531297E-03

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
23	23	1.00	43.00

1.94278898137497900E-02	1.00000000000000000
7.91131587293926047E-02	3.44040593177408782
0.17170235237945930	4.84586088344873911
0.27193799238601518	2.84931908840494286
0.37728417104035222	-1.18010721226727511
0.51550241203549352	-3.97649535059177506
0.72496150314225538	-4.08913065595758862
1.02383034114215699	-2.47697769574553517
1.38736610376959502	-0.72713628298706856
1.71384199010120009	0.31838778012141822
1.79450053678840904	0.64238030288962700
1.31887402098014195	0.56608658399747691
1.08894403249398496E-02	0.37301875491682268
-2.06106986629710098	0.20523534760671541
-4.12057070923773239	9.78237349690312258E-02
-4.74508437701981212	4.09791635994303777E-02
-2.84630921083055011	1.46548130350975404E-02
0.81148240269582827	4.23606932824975406E-03
3.45676898265638988	5.51930230195458570E-04
2.91321953925059818	-2.61464766219746602E-04
0.43326831772353219	-4.08619372192166612E-04
-0.89867046225488190	-1.79131990481428311E-04
-0.47164236432874751	-1.51030440868126208E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
22	22	1.00	44.00

1.04199435819473697E-02	1.00000000000000000
6.44499904421654557E-02	4.09078342148890606
0.19638529347542269	7.81537392151917576
0.41503108277840728	9.01498530479321580
0.71188307151606178	6.47202718701538160
1.09553188464379492	2.00286507885500997
1.60707627680347209	-1.75049981708751790
2.30452222826979192	-3.48311012737518189
3.21791440543100782	-3.45474478966031295
4.28497503443672922	-2.57063044065633894
5.26827432850481703	-1.58025855223217104
5.69745489674908789	-0.83311881582499203
4.93198998893389717	-0.38046721041231812
2.48347351654738580	-0.14839401042795200
-1.40576450380366702	-4.61484865990971632E-02
-5.30521420045989522	-8.78119097145470262E-03
-7.02955556486429067	1.25863989671714908E-03
-5.30919064662560203	1.51582392138833490E-03
-1.44330504020539596	-1.23078717346041796E-04
1.45548833251317911	-1.29240541510711703E-03
1.68154872827906510	-1.14071796712052697E-03
0.57115541823316773	-5.41262823194217798E-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
20	20	1.00	45.00

4.14616675283503971E-03	1.00000000000000000
5.10245959499307997E-02	5.42231753403680017
0.23551284791933999	14.91255550533599990
0.69593805929743002	27.79192677385299960
1.57756457139980011	39.65618715109300040
3.02505221354930010	46.43313817563200270
5.20576545930399970	46.71689225001399850
8.34465235711270026	41.76178485869200330
12.71490420740500050	34.03671042370900320
18.55284529796799830	25.80385854695400030
25.85191039416099910	18.46945712361900150
34.05666082509500110	12.58785820528900070
41.75805077304600130	8.17056047986950062
46.68972656076500270	4.99084157783390037
46.38302865521300330	2.79031900586889980
39.58832454674399770	1.35609887695990006
27.72091100736900060	0.52289799089836997
14.85675436461800030	0.13009678247022999
5.39292118942710008	6.51455977075734993E-03
0.99183775112792005	-5.88002303848370999E-03

B.2 Table 2: Selected filter coefficients for bandwidth 2π

(Left: numerator coefficients Right: denominator coefficients)

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
12	12	0.50	5.00
9.40334902520775957E-03		1.00000000000000000	
0.15162635113552000		2.68787508259199992	
0.48201104340919998		3.20420781340399996	
1.05547929953029995		2.38369940758609999	
1.96652955824570008		1.28715512855089997	
3.32601198895140016		0.56688717733486005	
4.92329453457819977		0.22877440999054999	
5.90125732308830031		0.10240039901054000	
4.86426460596929999		5.64540610698842024E-02	
1.64108027537600010		3.80884013819332984E-02	
-1.14083256938409994		2.32130151149992983E-02	
-0.97405570463579005		9.80591556956155082E-03	
Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
22	22	0.50	10.00
9.41542902901829D-04		1.0000000000000	
3.30018842081780D-02		4.6647528207596	
0.15650938643860		10.529500505332	
0.42648254540408		15.414911317435	
0.86905724263516		16.482133776832	
1.5099111229618		13.673770049197	
2.4173166046214		9.0427200250806	
3.7339305817175		4.7249991839664	
5.6774122103201		1.7711487576803	
8.5132889204266		0.21155567024249	
12.462849827942		-0.38804556415991	
17.510306841075		-0.48504929839206	
23.078946695842		-0.38790918372589	
27.700124439680		-0.25309567688482	
29.023478830457		-0.14044729570701	
24.721019885385		-6.20744599776413D-02	
14.415840881972		-1.37290773616497D-02	
1.4469470528239		1.19936321125766D-02	
-8.0634432591849		2.06293068401235D-02	
-9.8346043741164		1.76011389309782D-02	
-5.7030723145131		9.27937409139937D-03	
-1.4306458780549		2.50575928748800D-03	

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
31	31	0.50	15.00

3.06900833585920D-03	1.0000000000000
3.98359915450573D-02	5.0848513926382
0.17234028429154	11.988050501720
0.44307046397611	16.751556279009
0.82364920777087	13.871035099418
1.2434006255149	3.4474114513384
1.6692112979408	-8.5398041259988
2.1562071679094	-15.666418594610
2.8286319989932	-15.741825700251
3.8268506560301	-10.925226029230
5.2545884960598	-4.8782207407309
7.1350878852642	-0.22487845512286
9.3466443019909	2.2051285956181
11.523246788895	2.8225186041223
12.932621312864	2.4384195288296
12.421898019473	1.7264929757383
8.5876870120281	1.0637057823302
0.36966029638780	0.58334748798125
-11.891770216916	0.28564355062889
-25.323905862850	0.12217119495659
-34.359743859350	4.22263459806136D-02
-32.796172300790	7.78888331698885D-03
-18.167901756006	-4.23510419097026D-03
4.2306355077288	-6.78212348479102D-03
22.392885294216	-5.78299910576879D-03
25.772468636744	-4.13105449172633D-03
14.710919109120	-2.55712167173077D-03
0.51665634999783	-1.51891797698808D-03
-5.9569883368523	-7.61116833057022D-04
-4.3140506604431	-3.95979158631084D-04
-1.1087859769656	-1.15954229971485D-04

Num. Order N	Denom. Order M	Time step (sec)	Distance x (m)
37	37	0.50	20.00

1.34648681806207D-02	1.0000000000000
0.14198055501137	7.3980991565408
0.69209928133639	25.743727169873
2.1390639488912	55.161372415869
4.7963977897357	78.712365721077
8.4455767589922	71.673106685716
12.380545803203	24.158314433352
15.936851946870	-45.374325772393
19.040943193431	-101.57027314523
22.277951373897	-117.69447764109
26.420460310383	-91.882910688472
31.755231617201	-43.307385080357
37.559219991314	3.7319277378941
41.855718084591	33.512901025325
41.438950712825	42.936829915381
32.228513153630	37.644609203907
10.196578379053	25.822031479075
-26.833537013678	13.868314780206
-76.532982151419	4.9954181368406
-129.38497684004	-0.18705029270729
-167.91789521920	-2.4446986119273
-170.43707021319	-2.9031744952762
-120.68812799853	-2.5044181454056
-20.976849606623	-1.8470873187442
99.218977468317	-1.2304225569738
189.58069049011	-0.75981076778527
204.16982875529	-0.44129024486952
133.61627392329	-0.24311499493533
18.680557501523	-0.12765596901404
-73.519534226472	-6.40003075401518D-02
-97.889402986094	-3.06051519005244D-02
-62.674447089445	-1.38945420523229D-02
-13.266147182289	-5.93507534810803D-03
13.209011150795	-2.34125179844884D-03
13.769213571890	-8.27495840319982D-04
5.6815097295642	-2.41233943813211D-04
0.95932327286687	-4.87646780563652D-05