

ON THE DESIGN OF LOWPASS DIGITAL FIR FILTER COEFFICIENTS

A DISSERTATION

*submitted in partial fulfilment of the
requirements for the award of the degree*

of

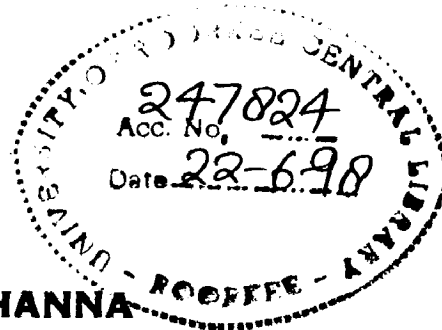
MASTER OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING
(With Specialization in Television Technology)

By

BINDU KHANNA



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
UNIVERSITY OF ROORKEE
ROORKEE – 247 667 (INDIA)

MARCH, 1997

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this dissertation, "**On the Design of Lowpass Digital FIR Filter Coefficients**", in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Electronics and Communication Engineering** with specialization in Television Technology, submitted in Department of Electronics and Computer Engineering, University of Roorkee, Roorkee is authentic record of my work carried out during the period of August, 1996 to March, 1997, under the supervision of **Dr. S.K. Varma**, Professor, Department of Electronics and Computer Engineering, University of Roorkee, Roorkee.

The matter embodied in this dissertation report has not been submitted by me for award of any other degree.

Date : 19th March, 1997

Place : Roorkee

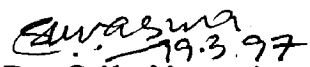

(BINDU KHANNA)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date : 19th March, 1997

Place : Roorkee


(Dr. S.K. Varma)
Professor
E & C Engg. Department
University of Roorkee,
Roorkee - 247 667.

ACKNOWLEDGEMENT

Inspiration and guidance are indispensable in all walks of life, but they become very crucial in the academic field. No task however, small, can be completed without proper guidance and encouragement.

I take this opportunity to express my sincere gratitude to **Dr. S.K. Varma, Professor, Department of Electronics and Computer Engineering, University of Roorkee, Roorkee**, for his keen interest, methodical approach, enthusiastic guidance, timely suggestions and discussion throughout the dissertation period. He has displayed unique tolerance and understanding at every step of progress and encouraged me incessantly. I deem it my privilege to have carried out the dissertation work under his able guidance.

I am grateful to teaching staff of this department for having been imparted with knowledge.

I express my regards to my parents and guardians who have been a constant source of inspiration to me.

I am thankful to **Mr. S.M. Deshpande, Research Scholar** and all my colleagues, especially **Mr. S.M. Akbar**, for their valuable help during the development of the software.

Thanks are also due to my friends for their constant encouragement throughout the work. One of the pleasure is to acknowledge the many persons, whose names may not appear on the cover but without whose efforts, cooperation and encouragement, this dissertation could not have been completed.


(**BINDU KHANNA**)

ABSTRACT

In this dissertation, designing of two types of digital FIR filters are studied using three techniques. Firstly, analysis of response-error due to finite precision is considered. Then different types and structures of FIR filters are given. Different approximation criteria used for FIR filter designing are also discussed.

In the first technique, equispaced samples of desired frequency response are taken. From these samples of frequency response, the filter coefficients are calculated. In second technique, algorithm is presented to minimize the error in frequency response when finite precision filter coefficients are evaluated by rounding-off the infinitely precision coefficients.

At last, maximally flat passband linear phase FIR filters are considered. In this technique, designing of linear phase FIR digital filter having symmetric impulse response is formulated as a constrained minimization problem. The constraints express the maximal flatness of the frequency response at the origin. The objective function, which is quadratic in filter coefficients, is formed as a convex combination of two objective functions. The two objective functions represent the energy of the error between the frequency response of the designed filter and a scaled version of the frequency response of the ideal filter in both stop and passbands.

CONTENTS

TOPIC	Page No.
CANDIDATE'S DECLARATION	(i)
ACKNOWLEDGEMENT	(ii)
ABSTRACT	(iii)
1. PREVIEW	
1.1 Introduction	1
1.2 Advantages of Digital Filters	1
1.3 Types of Digital Filters	1
1.4 Frequency Response of Digital Filters	3
1.5 Phase and Group Delay	4
1.6 Filter Design	4
1.7 Properties of FIR Filters	5
1.8 Applications of FIR Filters	5
1.9 Disadvantages of FIR Filter	6
1.10 Digital FIR Filters-Brief History	7
1.11 Statement of Problem	9
1.12 Organization of Dissertation	10
2. FINITE PRECISION EFFECTS ON DIGITAL FILTERS	
2.1 A-D Noise	14
2.1.1 Rounding	17
2.1.2 Truncation	17
2.1.3 Sign-magnitude truncation	17
2.2 Round off Noise	22

2.3 Filter Response Error	23
2.4 Fixed Point Arithmetic	25
3. STRUCTURES AND TYPES OF FIR FILTERS	
3.1 Structures of FIR Filters	27
3.2 Structure for Linear Phase Filters	27
3.3 Types of Linear Phase FIR Filters	29
4. APPROXIMATION CRITERIA	
4.1 Approximation Criteria	35
4.2 Advantages and Disadvantages of Different Criteria	35
4.3 Frequency Sampling Design Technique	36
4.4 Chebyshev Approximation	38
4.5 Maximally Flat Approximation	41
4.5.1 Maximal Flatness Constraints	42
4.5.2 Mean Squared Error Criterion	44
5. DESIGNING OF FIR FILTER COEFFICIENTS	
5.1 Frequency Sampling Design of Linear Phase FIR Filters	55
5.2 FIR Filter Design using Chebyshev Approximation Criterion and Linear Programming	58
5.3 Filter Designing using Maximally Flat Approximation	64
6. RESULTS, CONCLUSION AND FUTURE SCOPE	
6.1 Results	69
6.2 Conclusion	94
6.3 Future Scope	99
REFERENCES	100
APPENDIX (Software Listing)	

PREVIEW

1.1 INTRODUCTION

Digital filters first were simulations of analog filters on general-purpose computers. As computer technology advances, it provides faster multipliers, more memory, and good analog-to-digital converters. Some of these computer simulations were implemented with special hardware to replace the analog filters.

1.2 ADVANTAGES OF DIGITAL FILTERS [2]

Digital filter has following advantages over analog filter:

1. Programmable
2. Reliable and repeatable
3. Free from component drift
4. No tuning required
5. No precision components, no component matching
6. Superior performance (linear phase)

1.3 TYPES OF DIGITAL FILTERS

Digital filters can be classified into two types:

- (i) Finite Impulse response (FIR) filters
- (ii) Infinite Impulse response (IIR) filters

If the input to the discrete-time system is $x(n)$ then the output is represented as $y(n)$ and the impulse response of the (causal) system is $h(n)$. Output can be represented in terms of input & impulse response as:

$$y(n) = \sum_{m=-\infty}^n x(m) h(n-m) \quad \dots(1.1)$$

or

$$y(n) = \sum_{m=0}^{\infty} h(m) x(n-m) \quad \dots(1.2)$$

This is as shown in Fig. 1.1

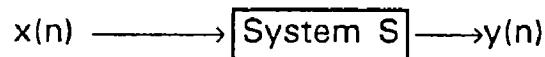


Fig.1.1 A block diagram of discrete-time system

The z-transform of a digital filter can be expressed as a rational polynomial in z^{-1} , i.e.,

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{i=0}^N a_i z^{-i}}{\sum_{i=0}^N b_i z^{-i}}$$

and $b_0 \triangleq 1$

The z-transform of impulse response is also known as its transfer function.

In non-recursive type of system current output $y(n)$ depends entirely on a finite number of past and present values of input. The above equation, in terms of filter impulse response $h(n)$, is written as:

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{n=0}^{N-1} h(n) z^{-n}$$

$$y(n) = h(0)x(n) + h(1)x(n-1) + \dots + h(N-1)x(n-N+1)$$

If the impulse response has infinite duration as in Eqs.(1.1) and (1.2) then the filter is an infinite-duration impulse-response (IIR) filter[2].

If the impulse response of a causal system is zero for all $n > N-1$ Eq.(1.2) becomes:

$$y(n) = \sum_{m=0}^{N-1} h(m) x(n-m). \quad \dots(1.3)$$

and the filter is known as finite-duration impulse-response (FIR) filter.

1.4 FREQUENCY RESPONSE OF DIGITAL FILTERS

If the input to a causal, linear, stationary system is $e^{j\omega n}$ (complex exponential) with frequency ω then from Eq.(1.2)

$$\begin{aligned} y(n) &= \sum_{m=0}^{\infty} h(m) e^{j\omega(n-m)} \\ &= e^{j\omega n} \left[\sum_{m=0}^{\infty} h(m) e^{-j\omega m} \right] \end{aligned} \quad \dots(1.4)$$

$$y(n) = e^{j\omega n} \cdot H(\omega)$$

$$\text{where } H(\omega) = \sum_{m=0}^{\infty} h(m) e^{-j\omega m} \quad \dots(1.5)$$

$H(\omega)$ is called the frequency response because it describes the change in magnitude and phase with frequency ω .

As compared to continuous-time system, the frequency response of a discrete-time system is always periodic with a period of sampling frequency, which is generally normalized to 2π radians per second.

Frequency response of a discrete time system may be written as

$$H(\omega + 2\pi) = \sum_{m=0}^{\infty} h(m)e^{-j(\omega + 2\pi)m}$$

$$= \sum_{m=0}^{\infty} h(m)e^{-j\omega m} e^{-j2\pi m}$$

$$H(\omega + 2\pi) = H(\omega)$$

So, discrete-time system is periodic with a period of 2π radians per second.

$$H^*(\omega) = H(-\omega) \quad \dots(1.6)$$

The equation (1.6) shows that frequency response has an even magnitude function and an odd phase function.

Hence, frequency-response plots need only be drawn for positive frequencies only.

1.5 PHASE AND GROUP DELAY

Letting $|H(\omega)|$ and $\theta(\omega)$ as magnitude and phase of $H(\omega)$ respectively, the phase shift is written as

$$\tau_p = -\frac{\theta(\omega)}{\omega}$$

and the group delay is given by

$$\tau_g = \frac{-d\theta(\omega)}{d\omega}$$

For a bandpass signal, τ_p represents the delay of carrier and τ_g represents the delay of the envelope of the signal.

1.6 FILTER DESIGN

Filters are designed in two steps :

- (i) Approximation problem
- (ii) Realization problem

The approximation part of the problem deals with the choice of parameters or coefficients in the filter's transfer function to approximate an ideal response. The approximation is performed by using the frequency response. The realization part of the filter design deals with choosing a structure to implement the transfer function. This structure may be in the form of a circuit diagram using components or it may be a program to be used on a general-purpose computer or a signal-processing microprocessor. In this dissertation approximation problem is mainly considered.

1.7 PROPERTIES OF FIR FILTERS [1][6]

1. FIR filters have exactly linear phase.
2. Length-N FIR filter has a pole of order N-1 at the origin of z plane. A pole at origin does not affect the magnitude of the frequency response of the filter.
3. FIR filters have an impulse response that is symmetrical around the $(N-1)/2$ point.
4. FIR filters are usually implemented in a non-recursive way. This guarantees stability.
5. Round-off noise, which is inherent in realizations with finite precision arithmetic, can easily be made small for non-recursive realizations of FIR filters.
6. FIR filters can be more easily altered in a dynamic fashion to implement adaptive filtering functions.

1.8 APPLICATIONS OF FIR FILTERS [2][8][16]

1. FIR filters have been used as band-select filters in FDM/TDM translators and in touch-tone receivers.
2. FIR filter can be used as a matched filter in radar and as echo cancellor in satellite communication. FIR filters appear in

cascade with other filters, such as in long-distance communication channel with repeater stations.

3. Optimal multi-dimensional FIR filters can be designed easily from 1-D prototypes which can be used in image processing applications.
4. Digital signal processing is used in various stages of digital television receivers. In television transmission luminance & chrominance signals are interleaved with each other which can be separated by using FIR bandpass and comb filters. FIR filters can also be used to remove ghost to improve picture quality.
5. Digital filters are often used in speech processing systems such as compact disk players.
6. Linear phase filters are used where frequency dispersion due to non-linear phase is harmful, e.g., speech processing and data transmission.
7. FIR filters are also used in adaptive equalizers used for data communication having simple tap coefficient adjustment algorithms.

1.9 DISADVANTAGES OF FIR FILTER [1][6]

1. A large impulse response duration is required for sharp cut-off filters. Hence a large amount of processing is required to realize such filters.
2. The delay of linear phase FIR filters need not always be an integer number of samples. This non-integer delay can lead to problems in some digital signal processing applications such as in digital feedback control system.
3. Closed form design formula for FIR filters do not exist. Thus FIR filter design is always an approximation problem.

1.10 DIGITAL FIR FILTERS-BRIEF HISTORY

FIR digital filters were initially analyzed by Kaiser using window functions, which indicated that IIR filters were much more efficient than FIR filters. However, Stockham's work on FIR digital filtering, indicated that implementation of higher-order FIR filters could be made extremely efficient-computationally. Thus comparison between FIR and IIR filters are no longer strongly biased towards the latter ones. These results also inspired significant research for efficient designs for FIR filters [1].

A class of selective non-recursive digital filters with independently prescribed equiripple passband and stopband attenuation and linear phase (LP) are obtained by numerical solutions of a set of non-linear equations. Attenuation plots of linear phase filters and experimental results are also given[9]. However, non-linear optimization procedure of Herrmann [9] is relatively slow algorithm and limited to the design of filters with few parameters.

An analysis of the three possible types of quantization effects in the direct form realization of FIR digital filters is presented in [4]. Based on analysis of quantization, statistical bounds on the error incurred in the frequency response of a filter due to coefficient quantization were developed and verified by experimental data. Using these bounds, a procedure for applying known techniques for FIR filter design to the design of filters with finite word length coefficients were presented. In this, direct form is shown to be a very attractive structure for realizing FIR filters.

However, several papers have appeared on the subject of non-linear-phase (NLP) filters. In [10] NLP filters were studied for minimizing the order of a filter subjected to given gain response specifications. It was shown that most LP filters can be implemented more efficiently than NLP ones by taking into account the symmetry of their coefficients for filters

with very wide passband. For certain special purpose, filters such as CCD and those used for filtering a delta-modulated or ADPCM signal, an NLP implementation is usually more efficient.

Simultaneous design in both magnitude and group delay of FIR filters based on Multiple Criterion Optimization (MCO) was studied by G.Cortelazzo and M.R. Lightner. Examples of the optimal tradeoff filters, both FIR non-linear phase filters and IIR filters were presented in [11]. In this paper, Chebyshev norm of magnitude and Chebyshev norm of the group delay which are two design objectives used in design formulation. Minimizing Chebyshev norm of magnitude as well as group delay simultaneously are useful performance objectives but very difficult to deal with computationally.

K.Preuss gave algorithm that deals with complex error function, which depends linearly on the coefficients of the filter to be designed [12]. The magnitude of this error function is minimized in the Chebyshev sense using Remez exchange algorithm. The method can be used to design complex-valued-selective systems. This method takes large design time.

In [21], iterative algorithm for designing FIR filters by complex Chebyshev approximation method introduced by Preuss was considered again and a modification was proposed yielding a significant acceleration.

The algorithms by Preuss and Schulist do not converge. When these algorithms do converge, it is not shown that they converge to optimal filters. In [20], algorithm to design the FIR filter with real coefficients that best approximates an arbitrary complex valued frequency response is presented. This algorithm computes the optimal filter by solving the dual to the filter design problem. It is guaranteed to converge theoretically.

S.C.Pei & J.J.Shu gave the design of real FIR filters with arbitrary complex frequency responses by two real Chebyshev approximations [13].

In [19], procedure for the design of finite-length impulse response filters with linear phase is presented. The algorithm obtains the optimum Chebyshev approximation on separate intervals corresponding to passbands and or stopbands, and is capable of designing very long filters.

When digital filters are implemented on a computer or with special-purpose hardware, each filter coefficient has to be represented by a finite number of bits. The simplest and most widely used approach to the problem is the rounding of the optimal infinite precision coefficients to its b-bit representation. The standard methods of optimal FIR digital filter design, namely the Remez algorithm [19] do not work when finite precision restriction is imposed. In [15], general purpose Integer-Programming computer program for the design of optimal finite wordlength FIR digital filters is described.

Medlin, Adams and Leondes [17] designed maximally flat linear phase filters by minimizing the energy of the error between the desired and designed frequency responses only over the stopbands subjected to flatness constraints at zero frequency. In [7], maximally flat linear phase filters were designed by minimizing the energy of error between the desired and designed frequency responses over the stopband as well as over the passband subjected to flatness constraints.

1.11 STATEMENT OF PROBLEM

In this dissertation, FIR filters are designed using three different techniques

- (i) Frequency sampling
- (ii) Chebyshev approximation
- (iii) Maximally Flat Approximation.

Designing of FIR filters using Frequency sampling method requires following specifications:

- (i) Filter length 'n'.

- (ii) Pass band cutoff frequency 'fp'.
- (iii) Number of output frequency samples required 'k'.

Designing of FIR filters using Chebyshev approximation method which uses Linear Programming requires following specifications:

- (i) Filter length 'n'.
- (ii) Number of bands 'nbnd'.
- (iii) Grid length 'lgrd'.
- (iv) Band edges 'edg' (array).
- (v) Desired frequency response 'fx' (array).
- (vi) Weighting factor 'wt' (array).
- (vii) Number of bits 'b'.

Designing of FIR filters using Maximally Flat Approximation requires following specifications:

- (i) Filter length 'N'.
- (ii) Number of constraints at zero frequency 'z'.
- (iii) Stopband (mean square) error scale factor 'alpha'.
- (iv) Normalized passband cutoff frequency 'wp'.

Designed filter frequency response and filter coefficients using above three methods are given in Chapter 6. The specifications of the filters to be designed by three methods are given in Tables 1.1 to 1.3.

1.12 ORGANIZATION OF DISSERTATION

In this chapter an introduction to the subject and a brief history of FIR filter designing over years is discussed.

Chapter two deals with effects of rounding infinite precision coefficients in FIR filter designing along with the analysis of all possible types of quantization errors. Fixed point arithmetic used for realization of filter coefficients is also described.

Table-1.1 Specifications for the design of low-pass FIR filters using frequency sampling technique

	Filter-1	Filter-2	Filter-3	Filter-4
n	20	21	30	31
fp	0.25	0.25	0.25	0.25
k	80	84	120	124

Table-1.2 Specifications for the design of lowpass FIR filters using Chebyshev approximation criterion & Linear Programming technique

	Filter-1	Filter-2	Filter-3	Filter-4
n	20	21	31	40
nbnd	2	2	2	2
lgrd	16	20	16	8
edg	PB: 0.00-0.20	PB: 0.00-0.20	PB: 0.00-0.20	PB: 0.00-0.20
	TB: 0.20-0.25	TB: 0.20-0.25	TB: 0.20-0.25	TB: 0.20-0.25
	SB: 0.25-0.50	SB: 0.25-0.50	SB: 0.25-0.50	SB: 0.25-0.50
f _x	PB: 1.0	PB: 1.0	PB: 1.0	PB: 1.0
	SB: 0.0	SB: 0.0	SB: 0.0	SB: 0.0
wt	PB: 1.0	PB: 1.0	PB: 1.0	PB: 1.0
	SB: 1.0	SB: 1.0	SB: 1.0	SB: 1.0
b	7	7	7	10

PB : Passband, TB : Transition band, SB : Stopband

**Table 1.3 Specifications for the design of lowpass FIR filters
using maximally flat approximation method**

Filter Number	N	z	alpha	wp
Filter -1	33	3	0.01	0.15
Filter -2	33	3	0.50	0.15
Filter -3	33	3	0.80	0.15
Filter -4	33	3	1	0.15
Filter -5	21	3	1	0.15
Filter -6	22	3	1	0.15
Filter -7	33	3	1	0.15
Filter -8	40	3	1	0.15
Filter -9	21	2	1	0.15
Filter -10	21	3	1	0.15
Filter -11	21	4	1	0.15
Filter -12	21	5	1	0.15
Filter -13	33	3	0.01	0.10
Filter -14	33	3	0.01	0.15
Filter -15	33	3	0.01	0.20
Filter -16	33	3	0.01	0.25
Filter -17	33	2	0.50	0.15

Different forms of FIR filter and linear phase filter structures are given in Chapter three. Four types of linear phase filters are also given.

In Chapter Four, analysis of different approximation criteria used for linear phase FIR filter designing is given. Advantages and disadvantages of different criteria are also described.

Chapter Five deals with designing of FIR filter coefficients using Frequency sampling, Chebyshev approximation and Maximally Flat Approximation methods. Flow charts for the above three methods are also given.

In Chapter Six, results and conclusion are discussed, impulse response tables and frequency responses of some designed FIR filters are also shown.

Software listing is given in Appendix.

FINITE PRECISION EFFECTS ON DIGITAL FILTERS

FIR filters are implemented with a digital computer or digital hardware. In these implementations, the signal and coefficient values can no longer be represented with arbitrary precision and unlimited amplitude. Numbers must be represented as members of a finite set of values in a digital processor. There are several schemes for approximately representing real numbers digitally, but generally floating point and fixed-point representations are used. The minimum computing time can be obtained by fixed-point arithmetic.

Finite precision effects may be divided into three different categories:

- (i) Errors due to quantization of input samples known as A-D noise.
- (ii) Errors due to finite-precision arithmetic operations of addition, multiplication and storage known as round off noise.
- (iii) Errors in representing coefficients as finite fixed-point numbers known as filter response errors.

2.1 A-D NOISE

An analog-to-digital (A/D) converter is a device that operates on the analog waveform, i.e., samples the continuous time signals to produce a digital output consisting of a sequence of numbers. The sequence of numbers thus produced approximates a corresponding sample of the input waveform. Fig 2.1 shows the block diagram of the A/D converter. The first stage, i.e., sampler output is the sequence $s(n) = s(t)|_{t=nT}$ is created where

$s(n)$ is expressed to infinite precision. In the second stage the numerical equivalent of each sample of $s(n)$ is expressed by a finite number of bits giving the sequence $s_Q(n)$. The difference signal $e(n) = s(n) - s_Q(n)$ is called quantizing noise or A/D conversion noise.

The input analog signal $s(t)$ must be band limited in order for the quantized output $s_Q(n)$ to be meaningful representation of $s(t)$. So, a presampling or antialiasing analog lowpass filter precedes the A/D converter as shown in Fig 2.2(b).

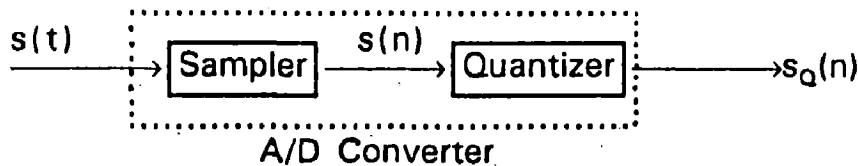


Fig. 2.1 Block diagram of A/D converter

In Fig. 2.2 (a), first block shows the conversion of continuous-time signals to discrete-time signals. This signal is then processed by digital processor and the resultant signal is converted to continuous-time signals. The digital processor may be a digital filter. Continuous-time signals can be converted to digital form by using a sampler, analog to digital convertor and an anti-aliasing filter. In most cases, it is desirable to minimize the sampling rate of a digital system. If the input is not band limited, the Nyquist frequency of the input is too high. So, prefiltering is often used. A more realistic model of digital filtering of continuous-time signals is as shown in Fig. 2.2(b), where digital processor is a digital filter[3].

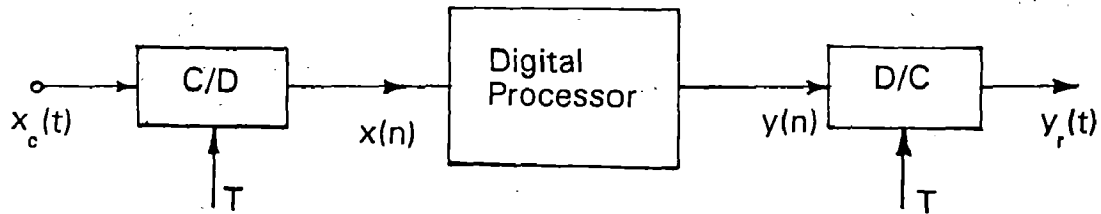


Fig. 2.2(a) Block diagram showing ideal digital filtering of continuous-time signals

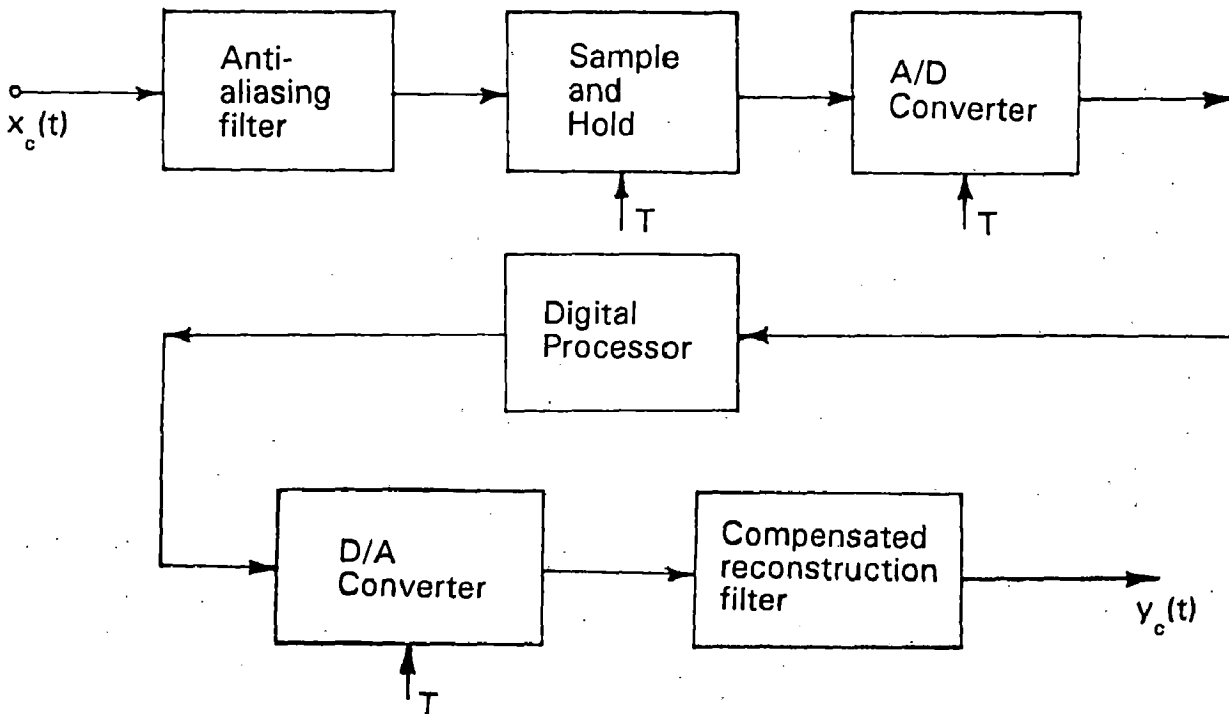


Fig. 2.2(b) A more realistic model of digital filtering of continuous-time signals

Depending on the way in which $s(n)$ is quantized, different distributions of quantization noise may be obtained.

Quantization can be done in three ways

- (i) Rounding
- (ii) Truncation
- (iii) Sign-magnitude truncation

2.1.1 Rounding [1]

If the smallest quantization step size used in the digital representation of $s_Q(n)$ is Q , then the relation between $s_Q(n)$ and $s(n)$ for rounding is as shown in Fig. 2.3. Since there are only a finite number of quantization levels, all signals either exceeding the largest (E_{\max}) or falling below the smallest ($-E_{\max}$) quantization level are rounded to these numbers. Generally such overflows or underflows are avoided by judicious choice of quantization step Q and by careful scaling of the analog input waveform.

The error signal satisfies the relation

$$-Q/2 \leq e(n) \leq Q/2 \quad \text{for all } n$$

If the distribution of the error signal is uniform then the probability distribution of the quantization error for rounding is as shown in Fig. 2.4.

2.1.2 Truncation

In this signal is represented by the highest quantization level that is not greater than the signal. Fig 2.5 shows the relation between $s_Q(n)$ and $s(n)$ for truncation.

Since truncation is equivalent to rounding less one-half a quantization step, the probability distribution of the error signal[1] is as shown in Fig. 2.6.

2.1.3 Sign-magnitude truncation

It is identical to truncation for positive signals and negative signals are approximated by the nearest quantization level that is greater than the signal. Fig 2.7 shows the relation between $s_Q(n)$ and $s(n)$ for sign-magnitude truncation. Thus, depending on whether $s_Q(n)$ is positive or negative, the distribution of Fig. 2.6 or it's mirror image is used[1].

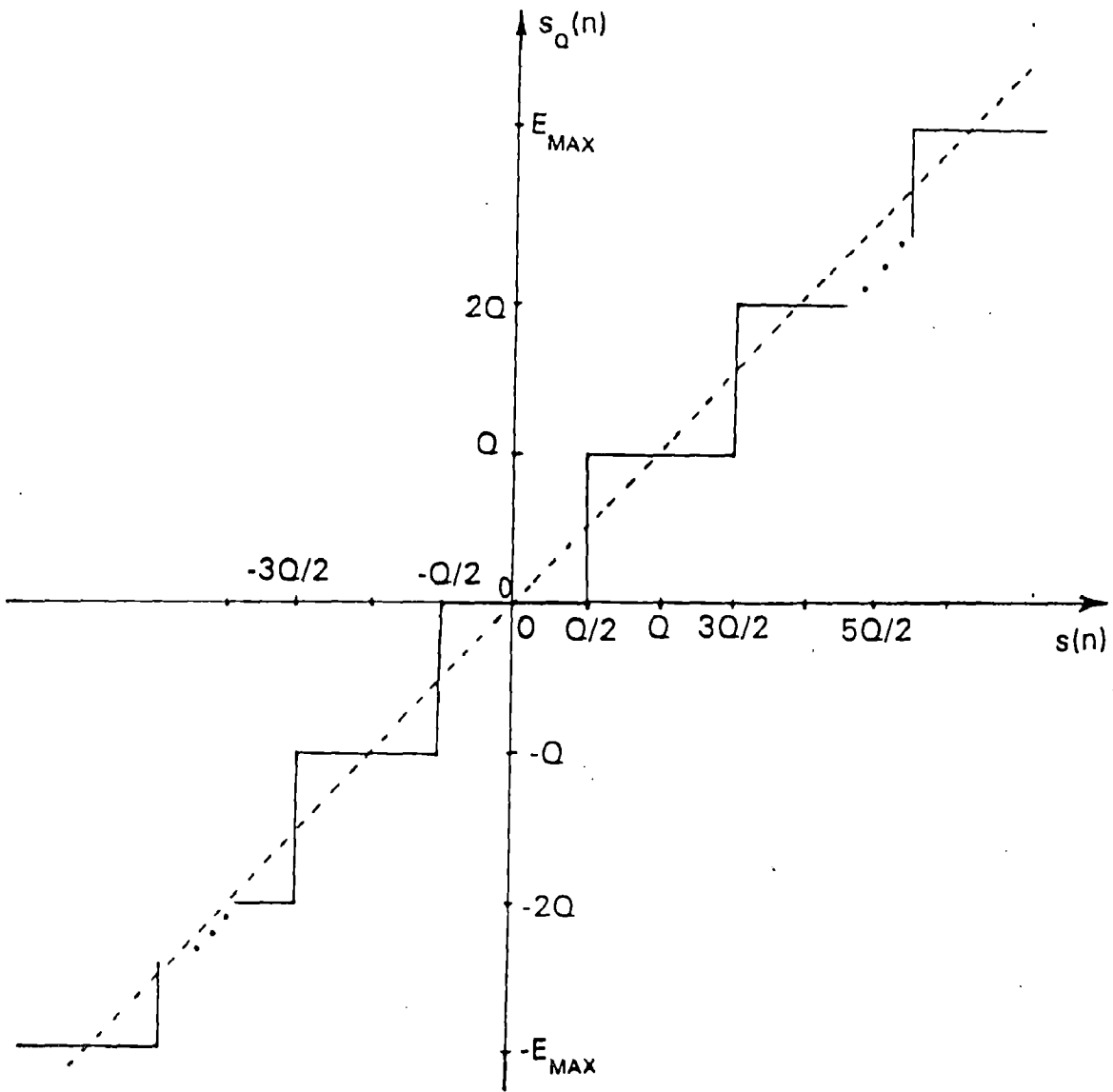


Fig. 2.3 Quantizer Characteristic with rounding

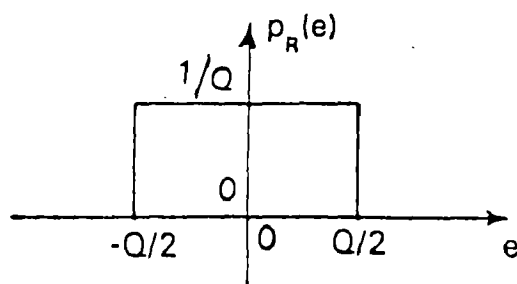


Fig. 2.4 Probability density function for roundoff error

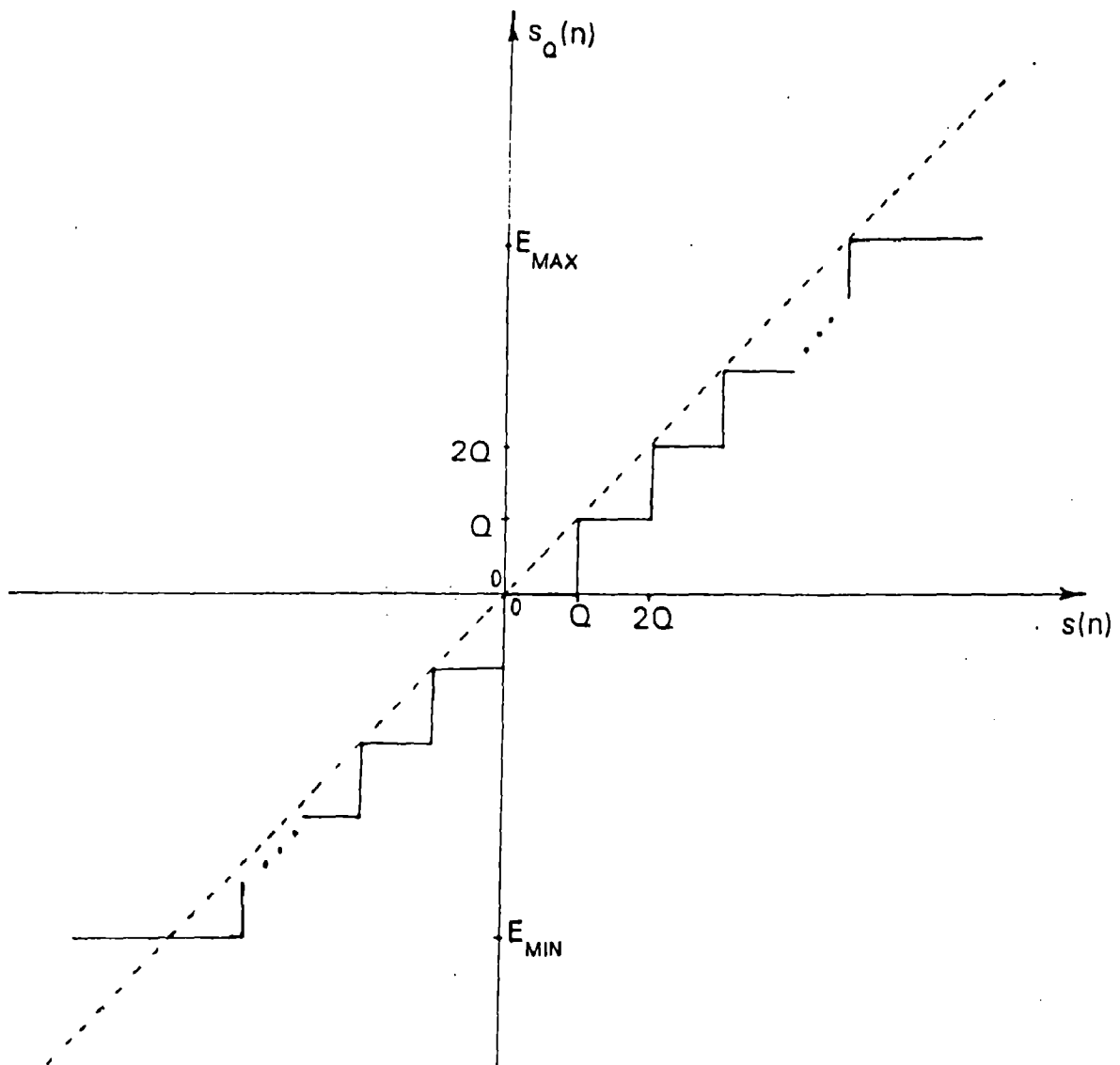


Fig. 2.5 Quantizer characteristic with truncation

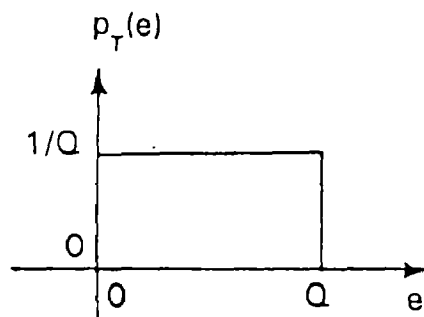


Fig. 2.6 Probability density function for truncation error

The quantization error has a mean value of 0 for rounding and sign-magnitude truncation and $Q/2$ for straight truncation.

The quantized signal is considered to be the true signal $s(n)$ with an added noise $e(n)$ as shown in Fig. 2.8.

The variance [2] of a random variable n is given by

$$\sigma_e^2 = E \left\{ e(n) - E \{ e(n) \} \right\}^2$$

Where $E\{x\}$ is the expected value of x . For rounding, the noise has zero mean, $E\{e(n)\}=0$, and the variance

$$\sigma_e^2 = \int p(e) e^2 de$$

$$\text{Probability density } p_R(e) = \frac{1}{Q}$$

$$\sigma_e^2 = \frac{1}{Q} \int_{-Q/2}^{Q/2} e^2 de$$

$$\sigma_e^2 = Q^2/12$$

The variance for truncation is also $Q^2/12$. The variance for sign-magnitude truncation is $Q^2/3$ or four times that of rounding or straight truncation. Due to statistical considerations, in most practical situations rounding is to be preferred to the other rules for quantizing a signal.

The errors that are made in converting a continuous-amplitude signal into a discrete representation may be evaluated in terms of signal-to-noise ratio (SNR). The signal must be scaled [2] to limit the possibility of overflow with the use of a scale factor or gain factor G , as shown in Fig.2.9. A small value of G will ensure that overflow never occurs, but the SNR will be reduced because the quantization noise level is fixed and a

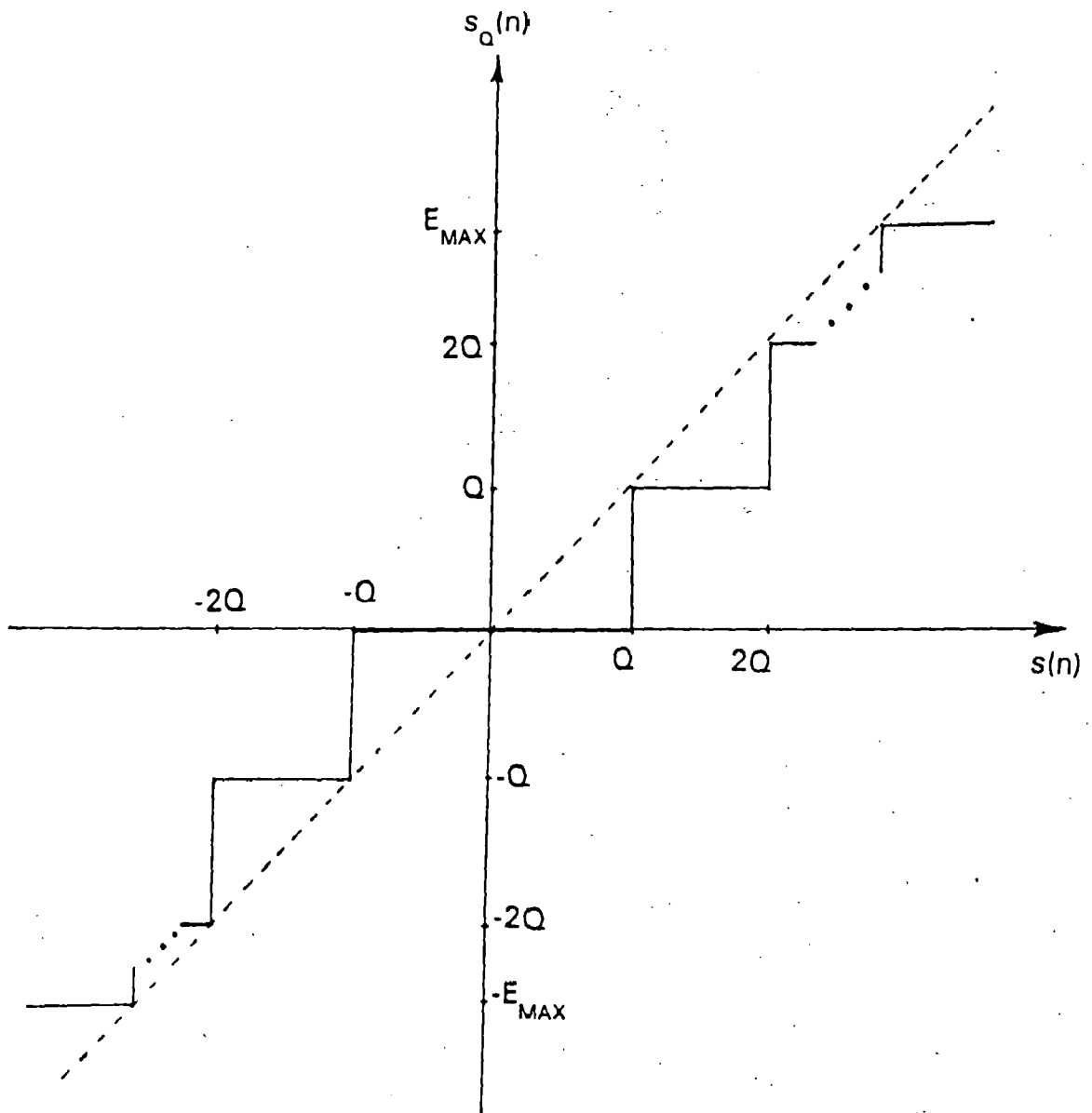


Fig.2.7 Quantizer characteristic with sign-magnitude truncation

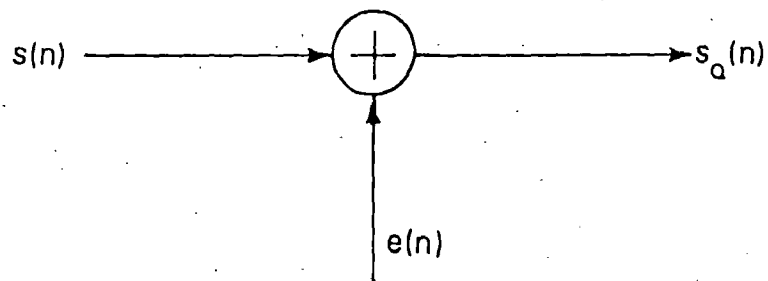


Fig. 2.8 Linear model of quantization noise

small value of G reduces the signal component. If occasional overflow is allowed, then the signal component will be larger and thus the SNR will be increased. This trade-off between overflow and quantization noise is always necessary when using fixed-point arithmetic.

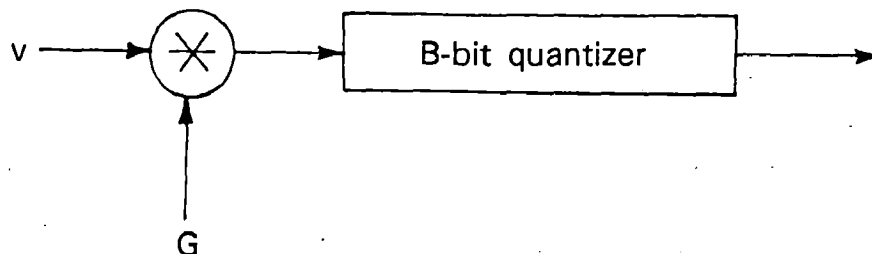


Fig. 2.9 A block diagram showing signal scaling before quantization

2.2 ROUND OFF NOISE [4]

Quantization of results of arithmetic operations within the filter causes errors in the filter output, referred to as round off noise. To each rounding point in the filter is associated a zero-mean white noise source of variance $Q^2/12$ and all noise sources are assumed to be uncorrelated with each other and with the input signal.

The round off noise at the output of a direct form FIR filter depends on the location of points in the filter where rounding is performed. If all multiplication products are represented exactly and rounding is performed only after they are summed (at the filter output) then only one noise source is present in the filter and it superimposes noise directly onto the output signal. Thus independent of filter order, the output roundoff noise is uniformly distributed between $-Q/2$ and $Q/2$ with mean zero and variance $Q^2/12$, where Q is the step size.

If on the other hand all, the multiplication products are rounded before they are summed, the output noise is the sum of all noise sources at each multiplication point. Let $\{e_i(n)\}$ the noise sequence produced by the

i_{th} noise sequence and $\{E(n)\}$ the noise sequence at the filter output.

$$E(n) = \sum_{n=0}^{(N-1)/2} e_i(n)$$

The mean of the output noise is zero and variance is

$$\sigma_L^2 = \left(\frac{N+1}{2} \right) \frac{Q^2}{12}$$

for a linear phase direct form FIR filter.

2.3 FILTER RESPONSE ERROR [3]

Direct form FIR system is expressed as

$$H(z) = \sum_{n=0}^{N-1} h(n)z^{-n}$$

If the coefficients $\{h(n)\}$ are quantized the resulting new set of coefficients are $\{\hat{h}(n) = h(n) + \Delta h(n)\}$. The transfer function for the quantized system is

$$\hat{H}(z) = \sum_{n=0}^{N-1} \hat{h}(n)z^{-n} = H(z) + \Delta H(z)$$

where

$$\Delta H(z) = \sum_{n=0}^{N-1} \Delta h(n)z^{-n}$$

Hence the quantized system can be represented as shown in Fig. 2.10. In this, unquantized system is considered to be in parallel with an error system, whose impulse response is the sequence of quantization error samples $\{\Delta h(n)\}$ and whose system function is the corresponding z-transform, $\Delta H(z)$.

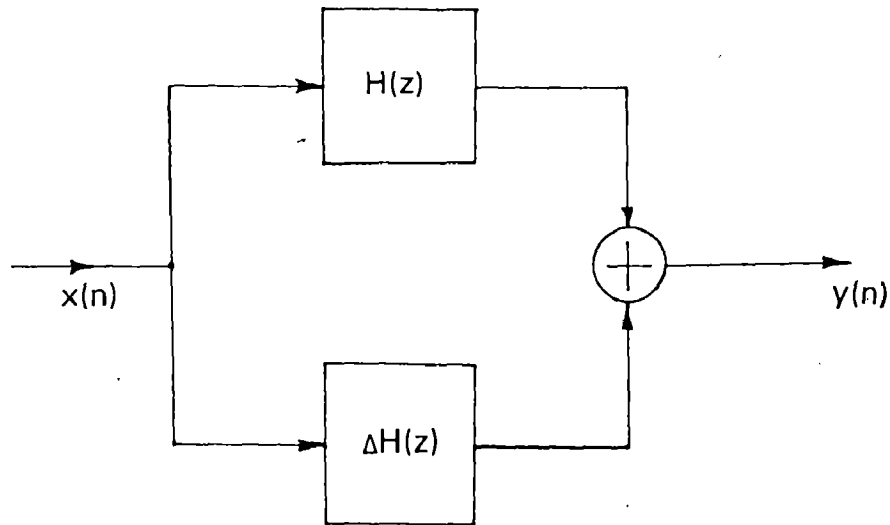


Fig. 2.10 Representation of coefficient quantization in FIR systems

Consider the case of odd length (N) FIR filter.

The frequency response is given by:

$$\begin{aligned}
 H(e^{j\omega}) &= \sum_{n=0}^{(N-3)/2} h(n) \left(e^{-j\omega n} + e^{-j\omega(N-1-n)} \right) + h \left(\frac{N-1}{2} \right) e^{-j\omega(N-1)/2} \\
 &= \left[\sum_{n=0}^{(N-3)/2} 2h(n) \cos \left[\left(\frac{N-1}{2} - n \right) \omega \right] + h \left(\frac{N-1}{2} \right) \right] e^{-j\omega(N-1)/2}
 \end{aligned}$$

$$H(e^{j\omega}) = \bar{H}(e^{j\omega}) e^{-j\omega(N-1)/2}$$

where $\bar{H}(e^{j\omega})$ denotes the magnitude function. Assume that rounding is used as quantization process. Let $\{h^*(n)\}$ be the sequence that results when $\{h(n)\}$ is rounded to a quantization step size of Q . Then $h^*(n) = h(n) + e(n)$ and $h^*(N-1-n) = h^*(n)$ for $0 \leq n \leq (N-1)/2$, where $e(n)$ for each n is a number that satisfies $|e(n)| \leq Q/2$. Let $H^*(z)$ be the z-transform of $\{h^*(n)\}$ and let $\bar{H}^*(e^{j\omega}) = H^*(e^{j\omega}) e^{j\omega(N-1)/2}$.

Then the error function is defined as:

$$E_L(e^{j\omega}) = \bar{H}^*(e^{j\omega}) - \bar{H}(e^{j\omega})$$

$$E_L(e^{j\omega}) = \sum_{n=0}^{(N-3)/2} 2e(n) \cos\left[\left(\frac{N-1}{2} - n\right)\omega\right] + e\left(\frac{N-1}{2}\right)$$

Since $|e(n)| \leq Q/2$, a upper bound on $E_L(e^{j\omega})$ is given by:

$$|E_L(e^{j\omega})| \leq \sum_{n=0}^{(N-3)/2} 2|e(n)| \left| \cos\left[\left(\frac{N-1}{2} - n\right)\omega\right] \right| + \left| e\left(\frac{N-1}{2}\right) \right|$$

$$\leq \frac{Q}{2} \left[1 + 2 \sum_{n=1}^{(N-1)/2} |\cos n\omega| \right]$$

$$E_L \leq \frac{Q}{2} \left[1 + 2 \cdot \left(\frac{N-1}{2}\right) \right] \quad \text{as } \cos n\omega \Big|_{\max} = 1$$

$$E_L \leq N \cdot \frac{Q}{2}$$

2.4 FIXED POINT ARITHMETIC [1]

Consider a word length of b bits is chosen to represent the numbers in a digital filter. So, 2^b different numbers may be represented exactly with a b -bit word. In fixed point representation, it is assumed that the position of the binary point is fixed. The bits to the right represent the fractional part of the number and those to the left represent the integer part. Let $b=7$, the number with base 10 say -0.375 is represented as 10.01100 where the binary point is assumed to be located following the second bit.

In most fixed point arithmetic realizations of digital filters, the position of the binary point is assumed to be just to the right of the first bit. Thus the range of numbers that can be represented is from -1.0 to $1.0 \cdot 2^{-(b-1)}$ where b is the number of bits in the word. The input signal

may be scaled to be within desired range as given in section 2.1. In case of digital filter coefficients, the position of the binary point is often moved further to the right to allow filter coefficients with magnitudes greater than 1.0.

In the next chapter different forms of FIR filter structure used for implementing FIR filter and four types of FIR filter are described.

STRUCTURES AND TYPES OF FIR FILTERS

3.1 STRUCTURES OF FIR FILTERS [3]

Causal FIR systems have only zeros and the difference equation is given by

$$y[n] = \sum_{m=0}^{N-1} h(m) x(n-m)$$

and the transfer function is given by

$$H(z) = \sum_{m=0}^{N-1} h(m)z^{-m}$$

The direct form realization of an FIR system is as shown in Fig. 3.1. There is a chain of delay elements across the top of the diagram. So, this structure is also referred to as a tapped delay line structure or a transversal filter structure. As shown in Fig. 3.1, the signal at the adder input is weighted by the appropriate coefficient (i.e., impulse response value) and the resulting product is summed to form the output.

The transposed direct form for the FIR case is as shown in Fig 3.2.

3.2 STRUCTURE FOR LINEAR PHASE FILTERS [4]

For the case of odd length filter, the transfer function is given by

$$H(z) = \sum_{n=0}^{(N-3)/2} h(n) [z^{-n} + z^{-(N-1-n)}] + h \left(\frac{N-1}{2} \right) z^{-(N-1)/2}$$

where $h(n) = h(N-1-n) \quad 0 \leq n \leq N-1$

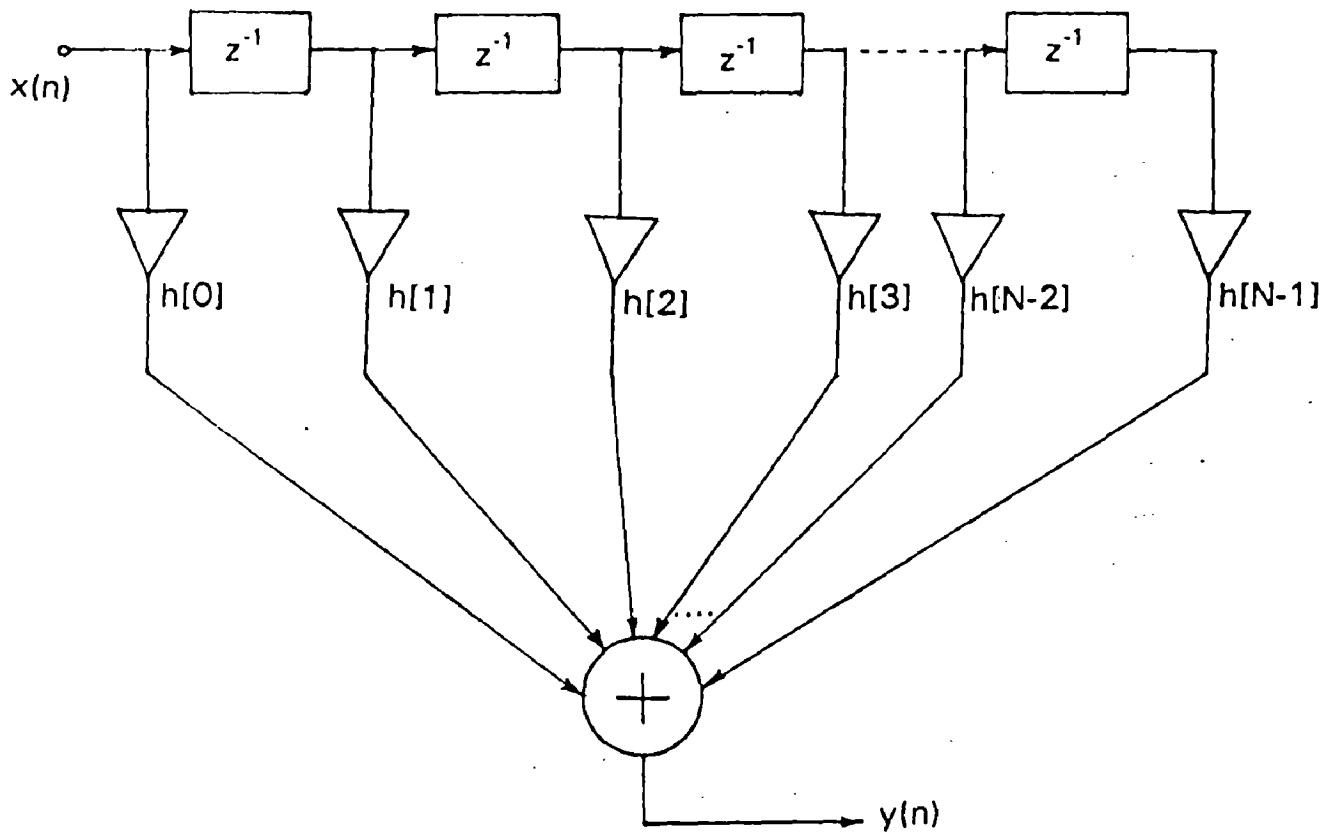


Fig. 3.1 Direct form realization of an FIR system

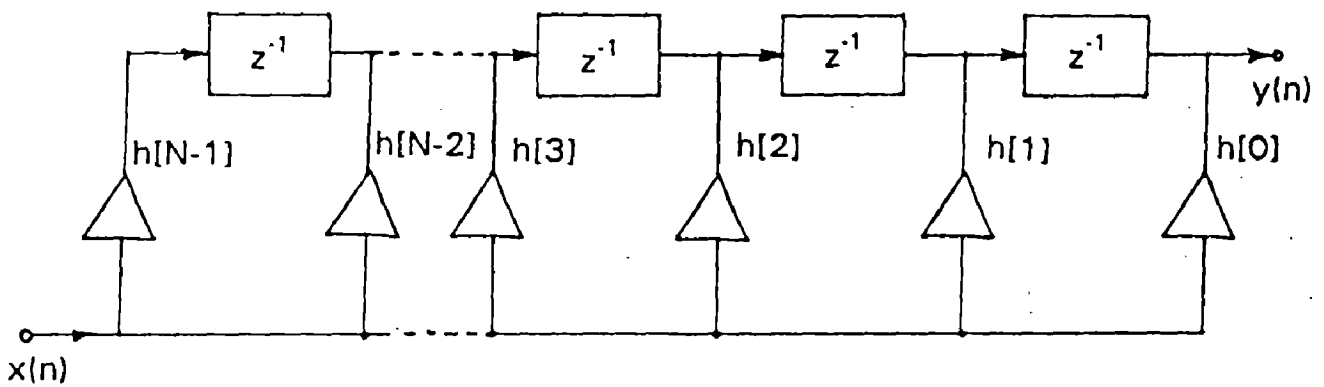


Fig 3.2 Transposition of the network of Fig. 3.1

From above equation it is seen that linear phase filter requires approximately half as many multipliers as that required for an arbitrary phase filter. A block diagram of the linear phase direct form is shown in Fig. 3.3.

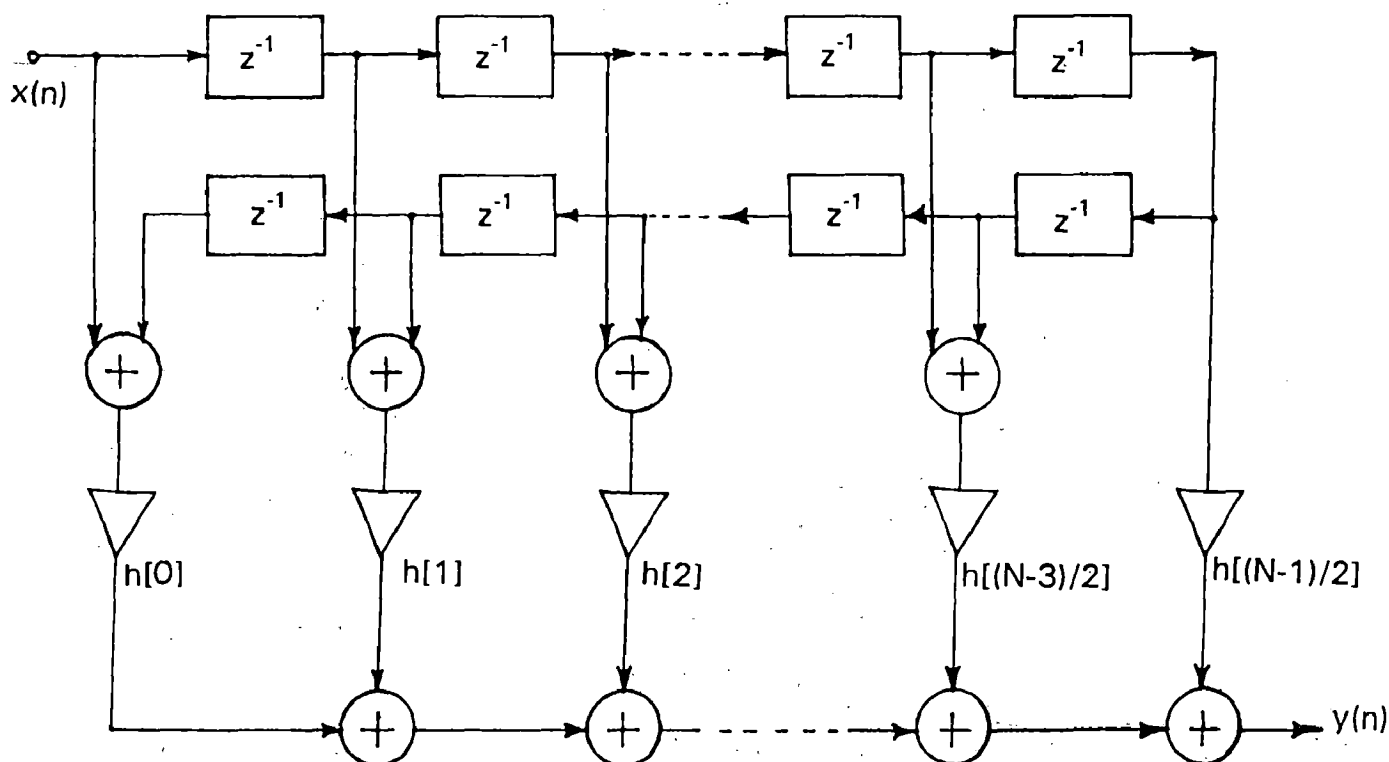


Fig. 3.3 Block diagram of linear phase direct form FIR filter

3.3 TYPES OF LINEAR PHASE FIR FILTERS [2]

The discrete fourier transform (DFT) can be used to evaluate the frequency response at certain frequencies. DFT of (length-N) impulse response, $h(n)$ is defined as

$$C(k) = \sum_{n=0}^{N-1} h(n)e^{-j2\pi nk/N}, \quad k=0, 1, 2, \dots, N-1 \quad \dots(3.1)$$

or

$$C(k) = H(\omega) \Big|_{\omega = 2\pi k/N}$$

$$= H\left(\frac{2\pi k}{N}\right), \quad k=0, 1, \dots, N-1 \quad \dots(3.2)$$

The DFT of $h(n)$ gives N samples of the frequency response function $H(\omega)$

$$H(\omega) = R(\omega) + jI(\omega)$$

$$\text{or } H(\omega) = M(\omega) e^{jd(\omega)}$$

$$M(\omega) = |H(\omega)| \\ = \sqrt{R^2 + I^2}$$

$$d(\omega) = \arctan \left(\frac{I}{R} \right)$$

where $M(\omega)$, $d(\omega)$ are defined as magnitude and phase function respectively. $M(\omega)$ is not analytic and $d(\omega)$ is not continuous.

So, $A(\omega) = \pm M(\omega)$ or $|A(\omega)| = M(\omega)$ is chosen to solve the above problem. Magnitude and amplitude as well as phase for a linear-phase FIR filter is as shown in Fig. 3.4. Now, $H(\omega)$ is given by

$$H(\omega) = A(\omega) e^{j\theta(\omega)} \quad \dots(3.3)$$

For FIR filter to be linear, the phase function $\theta(\omega) = K_1 + K_2\omega$ where K_1 and K_2 are arbitrary constants. Frequency response of a length- N FIR filter is given as

$$H(\omega) = \sum_{n=0}^{N-1} h(n)e^{-j\omega n} \quad \dots(3.4)$$

$$H(\omega) = e^{-j\omega M} \sum_{n=0}^{N-1} h(n)e^{j\omega(M-n)}$$

$$\text{where } M = \frac{N-1}{2} \text{ or } M = N-M-1$$

$$H(\omega) = e^{-j\omega M} \left[h_0 e^{j\omega M} + h_1 e^{j\omega(M-1)} + \dots + h_{N-1} e^{j\omega(M-N+1)} \right] \quad \dots(3.5)$$

From Eq.(3.3)

$$H(\omega) = A(\omega) e^{j(K_1 + K_2\omega)} \quad \dots(3.6)$$

where $A(\omega)$ is real-valued function of ω .

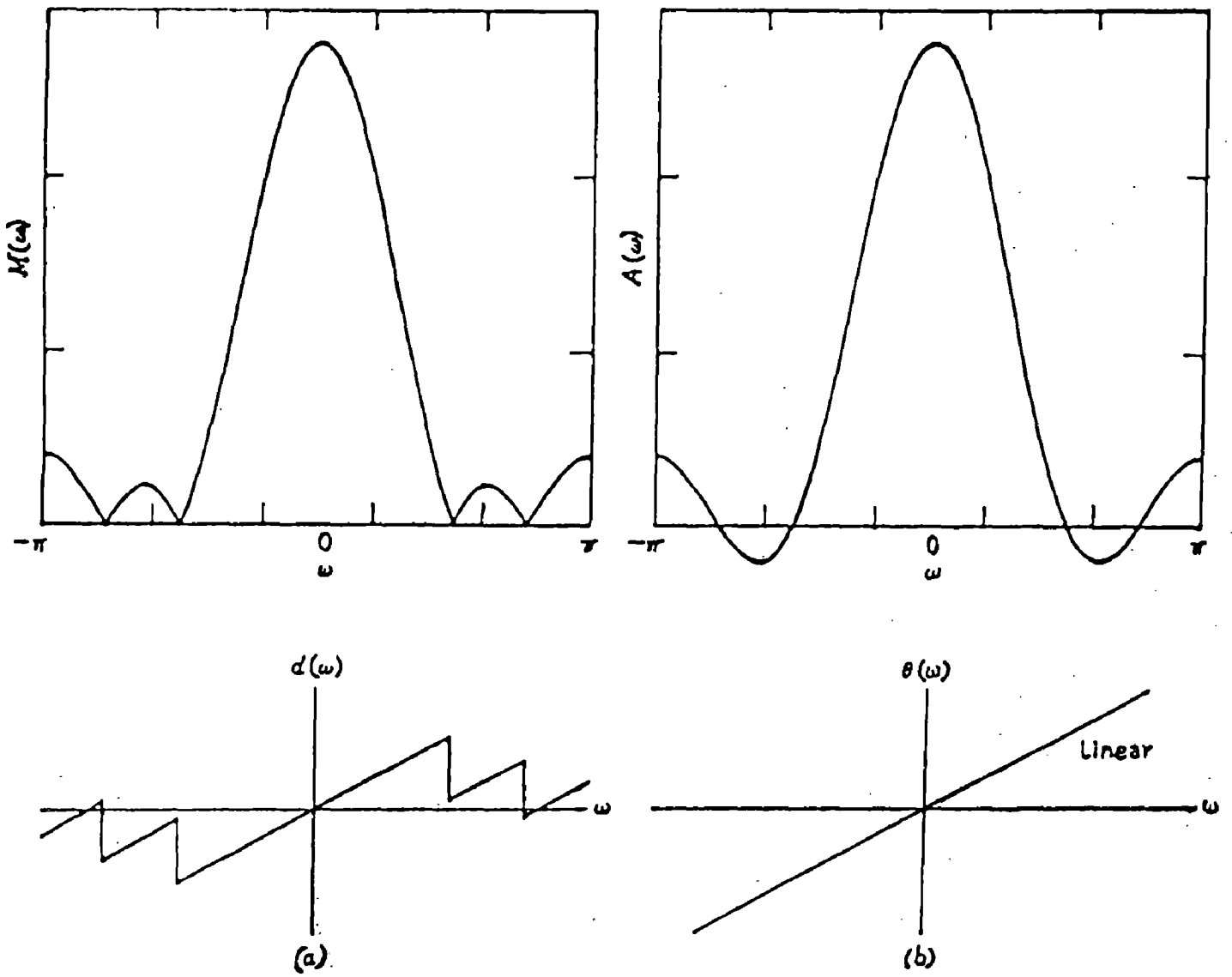


Fig. 3.4 The magnitude and amplitude of a linear-phase FIR filter

(a) magnitude and phase (b) amplitude and phase

From Eq.(3.5)

$$H(\omega) = e^{-j\omega M} \left\{ \left(h_0 + h_{N-1} \right) \cos(\omega M) + j \left(h_0 - h_{N-1} \right) \sin(\omega M) \right. \\ \left. + \left(h_1 + h_{N-2} \right) \cos\{\omega(M-1)\} + j \left(h_1 - h_{N-2} \right) \sin\{\omega(M-1)\} + \dots \right\} \dots(3.7)$$

Eq.(3.7) can be put in the form of Eq.(3.6) if $K_1 = 0$ or $K_1 = \pi/2$

(i) For $K_1 = 0$

$$h(n) = h(N-n-1)$$

$$H(\omega) = A(\omega) e^{jK_2\omega} \dots(3.8)$$

Eq.(3.7) can be written as

$$H(\omega) = e^{-j\omega M} A(\omega)$$

(a) for N odd (type-1)

$$A(\omega) = \sum_{n=0}^{M-1} 2h(n) \cos\{\omega(M-n)\} + h(M) \dots(3.9)$$

(b) for N even (type-2)

$$A(\omega) = \sum_{n=0}^{N/2 - 1} 2h(n) \cos\{\omega(M-n)\} \dots(3.10)$$

(ii) For $K_1 = \pi/2$

$$h(n) = -h(N-n-1)$$

$$H(\omega) = jA(\omega) e^{-jM\omega} \dots(3.11)$$

(a) for N odd (type-3)

$$A(\omega) = \sum_{n=0}^{M-1} 2h(n) \sin\{\omega(M-n)\}$$

(b) for N even (type-4)

$$A(\omega) = \sum_{n=0}^{N/2 - 1} 2h(n) \sin\{\omega(M-n)\}$$

The type-1 and type-2 formulae are useful in the design of low-pass filters, type-3 and type-4 formulae are useful in the design of differentiators and Hilbert transformers. The lowpass filter design using these formulae are given in Chapter 5. Different methods of designing FIR lowpass filters are given in the next chapter.

APPROXIMATION CRITERIA

Many techniques are available for design of FIR filters. The basic filter designing involves the following steps:

1. Choose a desired ideal response, usually described in the frequency domain.
2. Choose a length - N FIR filter.
3. Establish an error criterion for the response of a filter compared to the desired response.
4. Develop a method to find the best member of the linear phase FIR filters designed by varying length N (or other parameters).

When the best filter is designed and evaluated, the desired response or error criterion might be changed and the filter would then be redesigned. The approach is generally used iteratively.

In this chapter mainly different criteria used for designing of lowpass filters are considered. As shown in Fig. 4.1(b), the pass band extends from $\omega=0$ to $\omega=\omega_1$ and stopband from $\omega=\omega_2$ to $\omega=\pi$. The region between pass-band and stopband, i.e., $(\omega_2-\omega_1)$ is called transition region.

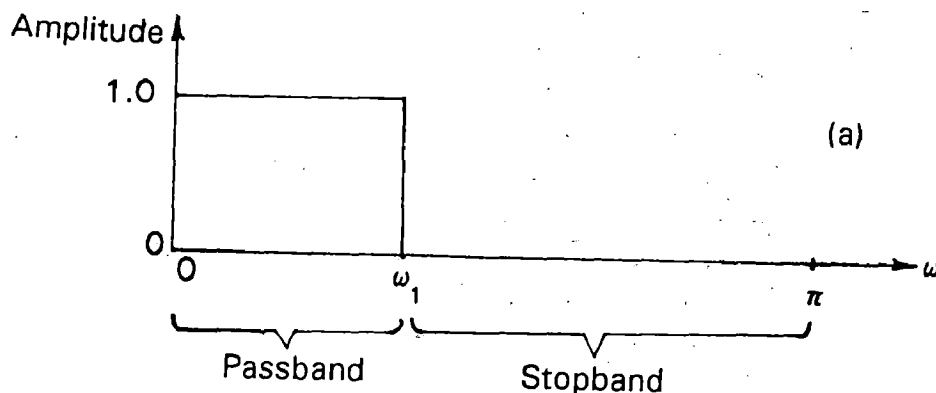


Fig.4.1(a) Ideal lowpass FIR filter frequency response without transition region

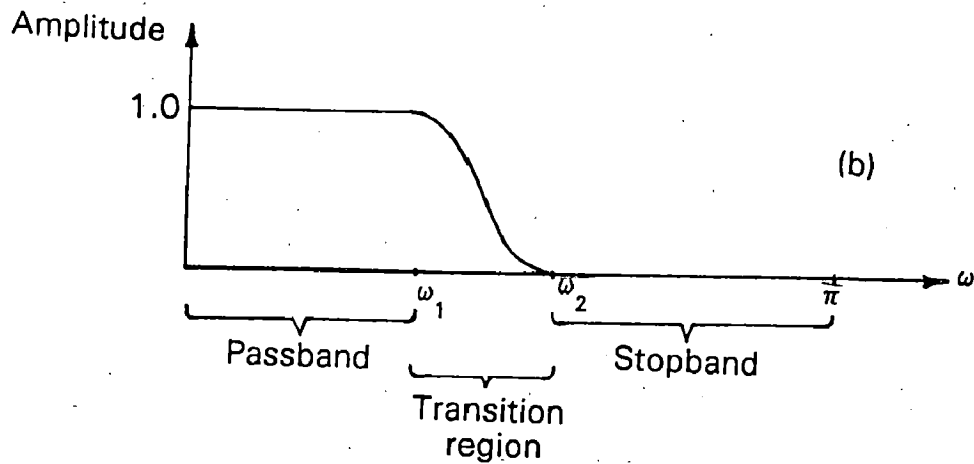


Fig.4.1(b) Ideal lowpass FIR filter frequency response with transition region

4.1 APPROXIMATION CRITERIA

Four error measures are generally used in FIR filter design.

- (i) Minimization of the peak stopband ripple called frequency sampling design.
- (ii) Minimization of the average of the squared error in the frequency-response, called least squared (LS) approximation.
- (iii) Minimization of the maximum of the error over specified regions of the frequency response, called Chebyshev approximation.
- (iv) Taylor series approximation to the desired response, called Butterworth or maximally flat approximation.

4.2 ADVANTAGES AND DISADVANTAGES OF DIFFERENT CRITERIA

For designing FIR filters, frequency-sampling method is fast and simple. It is useful for adaptive filters or for an intermediate stage in a more complicated algorithm. But it provides least control over the total frequency response.

The LS error method uses an error criterion that is related to the energy of the signal or noise. The design equations are linear. But the

design sometimes have oscillations or overshoot in the frequency response. The oscillations or overshoot are undesirable.

Design algorithm based on Chebyshev error can be slow. If smoothness of the response is needed, the maximally flat approximation is used.

4.3 FREQUENCY SAMPLING DESIGN TECHNIQUE [1]

FIR filters can be uniquely specified by impulse response coefficients $\{h(n)\}$ or by DFT coefficients $\{C(k)\}$.

As given in equation (3.1)

$$C(k) = \sum_{n=0}^{N-1} h(n) e^{-j2\pi nk/N}$$

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} C(k) e^{j2\pi nk/N} \quad \dots(4.1)$$

DFT samples $C(k)$ for a FIR sequence can be regarded as samples of the filter's z-transform evaluated at N points equally spaced around the unit circle, i.e.,

$$C(k) = H(z) \Big|_{z=e^{j2\pi k/N}} \quad \text{where } k=0, 1, \dots, N-1$$

The z-transform of a FIR filter can be expressed in terms of its DFT coefficients by substituting Eq. (4.1) into the z-transform as given below:

$$H(z) = \sum_{n=0}^{N-1} h(n) z^{-n}$$

$$= \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{k=0}^{N-1} C(k) e^{j2\pi nk/N} \right] z^{-n}$$

Summation over the index n is interchanged with summation over index k which gives:

$$\begin{aligned} H(z) &= \sum_{k=0}^{N-1} \frac{C(k)}{N} \sum_{n=0}^{N-1} \left[e^{j2\pi nk/N} z^{-1} \right]^n \\ &= \sum_{k=0}^{N-1} \frac{C(k)}{N} \frac{(1 - e^{j2\pi k/N} z^{-N})}{(1 - e^{j2\pi k/N} z^{-1})} \end{aligned}$$

Since $e^{j2\pi k} = 1$, the above equation reduces to

$$H(z) = \frac{(1 - z^{-N})}{N} \sum_{k=0}^{N-1} \frac{C(k)}{(1 - z^{-1} e^{j2\pi k/N})}$$

The above equation shows that sampling in frequency at N equispaced points around the unit circle and evaluating the continuous frequency response by interpolation of the sampled frequency response gives frequency response having exactly zero approximation error at the sampling frequencies and finite in between them.

The smoother the frequency response being approximated, the smaller the error of interpolation between the sample points. This is as shown in Fig. 4.2.

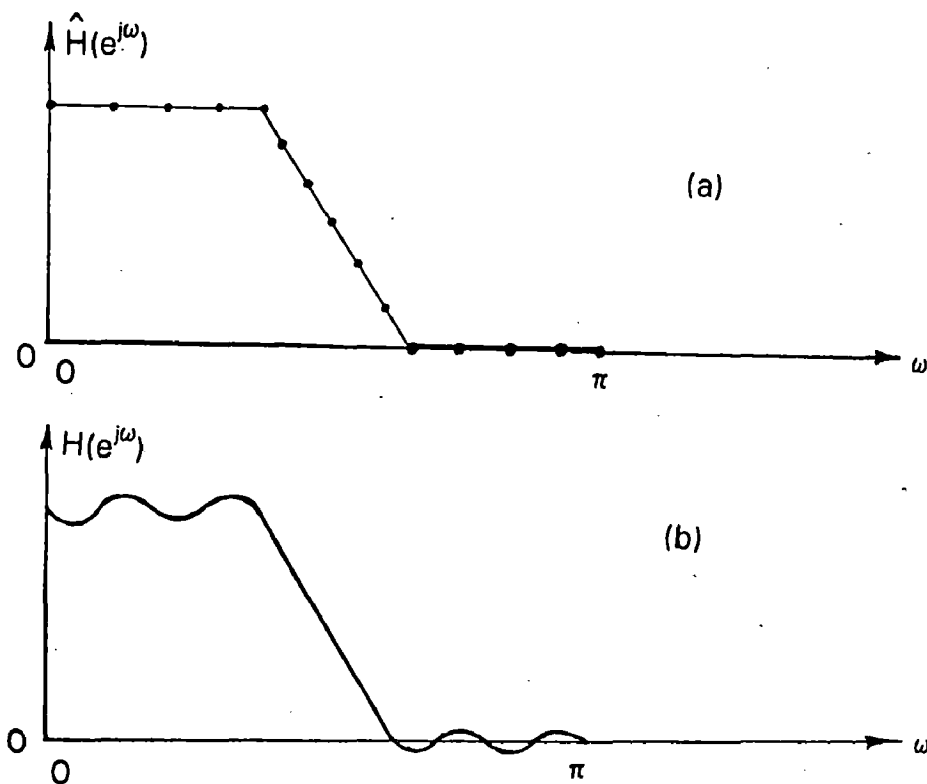


Fig. 4.2 (a) Desired frequency response of a low pass filter
 (b) Frequency response of filter using frequency sampling technique

The above procedure is used directly to design an FIR filter. To improve the quality of the approximation, i.e., to make the approximation error smaller, a number of frequency samples can be made unconstrained variables.

4.4 CHEBYSHEV APPROXIMATION [5][15]

Assume a desired magnitude response as $D(e^{j2\pi f})$, designed filter magnitude response as $H(e^{j2\pi f})$ and a weighting function $W(e^{j2\pi f})$ which is continuous on a compact subset $F \subset [0, 0.5]$ where F represents normalized frequency. Here sampling rate is assumed to be 2π radians. For a linear phase filter, maximum absolute weighted error is to be minimized. Maximum absolute weighted error is given by

$$\|E(e^{j2\pi f})\| = \max_{f \in F} W(e^{j2\pi f}) |D(e^{j2\pi f}) - H(e^{j2\pi f})| \quad \dots(4.2)$$

This error is to be minimized over a set of coefficients of $H(e^{j2\pi f})$ which are represented as $\hat{h} = [h_0, h_1, \dots, h_N]^T$. Minimization of Eq.(4.2) is known as Chebyshev approximation problem.

In other words, it may be stated as to find the filter coefficients \hat{h} of $H(e^{j2\pi f})$ such that the Chebyshev-norm, i.e., $\max |E(e^{j2\pi f})|$ is minimized.

Mathematically it may be represented as

$$\min_h \left\{ \max_{f \in F} |E(e^{j2\pi f})| \right\}$$

$$\text{where } E(e^{j2\pi f}) = W(e^{j2\pi f})[D(e^{j2\pi f}) - H(e^{j2\pi f})]$$

The above problem may be formulated as:

Minimize E

$$\text{Subjected to : } H(e^{j2\pi kf}) - E/W(e^{j2\pi kf}) \leq D(e^{j2\pi kf})$$

$$- H(e^{j2\pi kf}) - E/W(e^{j2\pi kf}) \leq - D(e^{j2\pi kf})$$

where $k=0, 1, 2, \dots, \text{ngrd}$.

and ngrd is number of frequency points on which ideal frequency response is sampled.

For simplicity $H(e^{j2\pi f})$ is represented as $H(f)$ and in term of filter coefficients it is given by

$$H(f) = \sum_{k=0}^{N-1} h(k)e^{-j2\pi kf}$$

where $k=0, 1, 2, \dots, N-1$

As given by Eqs. (3.8) and (3.11) $H(f)$ can always be written as

$$H(f) = G(f) \exp\left[\frac{jL\pi}{2} - \frac{(N-1)}{2} \right] 2\pi f$$

where $G(f)$ is a real valued function and $L=0$ or 1 . As given in section 3.3 $G(f)$ can be expressed in terms of coefficients $h(k)$ as

(i) Type 1 : N odd , $L=0$

$$G(f) = h(n) + \sum_{k=1}^n h(n-k)2g(k,f) \quad \dots(4.3)$$

where $n = (N-1)/2$

$$g(k,f) = \cos(2\pi kf)$$

(ii) For type 2, 3 and 4

$$G(f) = \sum_{k=1}^n h(n-k)2g(k,f)$$

where $n = N/2$ for N even

$n = (N-1)/2$ for N odd

$$(a) \text{ for type 2 } g(k,f) = \cos\left(2\pi\left(k - \frac{1}{2}\right)f\right) \quad N \text{ even, } L=0$$

$$(b) \text{ for type 3 } g(k,f) = \sin(2\pi kf) \quad N \text{ odd, } L=1$$

$$(c) \text{ for type 4 } g(k,f) = \sin\left(2\pi\left(k - \frac{1}{2}\right)f\right) \quad N \text{ even, } L=1$$

In terms of desired frequency response $D(f)$ and a positive weighting function $W(f)$, both continuous on a compact subset $F \subset [0, 0.5]$, Eq. 4.2 can be written as

$$\|E(f)\| = \max_{f \in F} W(f) |D(f) - G(f)|$$

and the Chebyshev approximation problem can be written (for type 1) as

Minimize E subject to constraints

$$h(n) + \sum_{k=1}^n h(n-k)2g(k,f) - E/W(f) \leq D(f)$$

$$-h(n) - \sum_{k=1}^n h(n-k)2g(k,f) - E/W(f) \leq -D(f) \quad \dots(4.4)$$

where $f \in F$

Similarly it can be written for other types also. Using this approximation criteria, filters having different lengths are designed. Filter length can be increased for having more attenuation in stopband.

Other way of achieving more attenuation in stopband is by using maximally flat filters in cascaded with these filters.

4.5 MAXIMALLY FLAT APPROXIMATION

Maximally flat filters are the filters that achieves a specified degree of flatness at $\omega = 0$ (for lowpass filters) or at reference frequency in passband (for bandpass filters). It means that this type of filters have maximally flat passband.

This type of filter has an advantage of trade-off between more flatness in the passband and better attenuation in the stopband. Linear phase FIR filters having very flat passbands and equiripple stopbands are important for several applications. Consider an example of removal of high frequency noise from a low frequency signal by low-pass filtering. In order to reduce the distortion of the signal introduced by the filter, a filter having a very flat passband is desirable and to maximize the stopband attenuation, a filter having equiripple stopband is desirable [8]. Filters having very flat passbands are also useful in applications in which a filter appears in cascade with other filters, such as in a long-distance communication channel with repeater stations.

The design of linear phase FIR filters having symmetric impulse response is formulated as constrained minimization problem. The constraints express the maximal flatness of the frequency response at the origin.

The objective function, which is a quadratic form in the filter coefficients is formulated as a convex combination of two criteria representing the energy of the error between the frequency response in both

the pass and stop bands. The constraints express the maximal flatness of the frequency response at zero frequency for the case of a low-pass filter.

4.5.1 Maximal Flatness Constraints [7]

For linear phase FIR filters (having symmetric impulse response) the frequency response is given by

$$H(e^{j\omega}) = e^{-j\omega(N-1)/2} H_a(\omega) \quad \dots(4.5)$$

where the amplitude function $H_a(\omega)$ can be expressed as

$$H_a(\omega) = s^T(\omega).x \quad \dots(4.6)$$

$$\text{where } x^T = \begin{cases} \{h(0.5(N-1)) & 2h(0.5(N-1)-1) & \dots & 2h(1) & 2h(0)\} & N \text{ odd} \\ \{2h(0.5N-1) & 2h(0.5N-2) & \dots & 2h(1) & 2h(0)\} & N \text{ even} \end{cases} \quad \dots(4.7)$$

$$s^T(\omega) = \begin{cases} \{1 \cos\omega \cos 2\omega & \dots \cos[0.5(N-1)\omega]\} & N \text{ odd} \\ \{\cos(0.5\omega)\cos(1.5\omega)\cos(2.5\omega) & \dots \cos[0.5(N-1)\omega]\} & N \text{ even} \end{cases} \quad \dots(4.8)$$

As FIR filters with symmetric impulse response is used for designing low-pass filters with maximally flat passband, the following flatness conditions will be applied at zero frequency.

$$\left. \frac{d^k H_a(\omega)}{d\omega^k} \right|_{\omega=0} = 0, \quad k \text{ integer} \quad \dots(4.9)$$

From (4.6) and (4.9)

$$\frac{d^k H_a(\omega)}{d\omega^k} = \left(\frac{d^k s(\omega)}{d\omega^k} \right)^T x \quad \dots(4.10)$$

For odd order derivatives (i.e. $k=1,3,5,\dots$), the above equation becomes equal to zero at $\omega=0$ because the k th derivative of $\cos(n\omega)$ is

$(n^k)\sin(n\omega)$ when k is odd. Hence at $\omega=0$, it is equal to zero. It may be expressed as

$$\frac{d^k s(\omega)}{d\omega^k} \Big|_{\omega=0} = 0 \text{ for } k \text{ odd}$$

For even order derivatives (i.e., $k=2,4,6, \dots$), Eq. (4.10) becomes

$$\frac{d^k s(\omega)}{d\omega^k} \Big|_{\omega=0} = \begin{cases} (-1)^{0.5k} \{ 0 & 1 & 2^k & [0.5(N-1)]^k \}^T & N \text{ odd} \\ (-1)^{0.5k} \{ (0.5)^k & (1.5)^k & (2.5)^k & [0.5(N-1)]^k \} & N \text{ even} \end{cases} \quad \dots(4.11)$$

The following normalization condition is always imposed:

$$H_a(\omega) \Big|_{\omega=0} = 1.0 \quad \dots(4.12)$$

From (4.6) and (4.8), the above condition becomes:

$$(1 \ 1 \ 1 \dots 1) x = 1 \quad \dots(4.13)$$

Constraint (4.13) & (z-1), i.e., $k/2$ constraints of the form of (4.9), given by Eq. (4.11) for $k=2,4, \dots, 2(z-1)$ can be expressed as

$$Cx = K \quad \dots(4.14)$$

or

$$\begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 1 & 2^2 & 3^2 & \dots & [0.5(N-1)]^2 \\ 0 & 1 & 2^4 & 3^4 & \dots & [0.5(N-1)]^4 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 2^{2(z-1)} & 3^{2(z-1)} & \dots & [0.5(N-1)]^{2(z-1)} \end{bmatrix} \begin{bmatrix} h[0.5(N-1)] \\ 2h[0.5(N-1)-1] \\ 2h[0.5(N-1)-2] \\ \vdots \\ 2h(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \dots(4.14a)$$

where N is odd.

Similarly matrix C can be written for even N as

$$C = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 3^2 & 5^2 & \dots & (N-1)^2 \\ 1 & 3^4 & 5^4 & \dots & (N-1)^4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 3^{2(z-1)} & 5^{2(z-1)} & \dots & (N-1)^{2(z-1)} \end{bmatrix} \quad \text{for N even} \quad \dots(4.14b)$$

C is matrix for $z \times m$ & K is z-dimensional where $m = (N+1)/2$ for odd N and $z \leq m$.

4.5.2 Mean Squared Error Criterion [7]

The error criterion will be derived as a quadratic form expressing the energy between the frequency response of the designed filter and a scaled version of the desired frequency response. In this, both pass & stop bands are taken into account.

The amplitude of ideal low-pass filter is given by

$$H_d(\omega) = \begin{cases} 1 & 0 \leq \omega \leq \omega_p \\ 0 & \omega_s < \omega \leq \pi \end{cases} \quad \dots(4.15)$$

where ω_p and ω_s are the passband & stopband cutoff frequencies.

The weighted error in the stopband is

$$E_s = \int_{\omega_s}^{\pi} W(\omega) e_s^2(\omega) d\omega \quad \dots(4.16)$$

where $W(\omega)$ is a positive weighting function and $e_s(\omega)$ the stopband error given by:

$$\begin{aligned} e_s(\omega) &= H_a(\omega) - H_d(\omega) \\ &= H_a(\omega) \\ e_s(\omega) &= s^T(\omega) x \quad \dots(4.17) \end{aligned}$$

Substituting (4.17) in (4.16)

$$E_s = x^T \left(\int_{\omega_s}^{\pi} W(\omega) s(\omega) s^T(\omega) d\omega \right) x$$

or

$$E_s = x^T P_s x \quad \dots(4.18)$$

where
$$P_s = \int_{\omega_s}^{\pi} W(\omega) s(\omega) s^T(\omega) d\omega$$

For $W(\omega) = 1$

$$P_s = \int_{\omega_s}^{\pi} s(\omega) s^T(\omega) d\omega \quad \dots(4.19)$$

For N odd,

$$P_s = \int_{\omega_s}^{\pi} \begin{bmatrix} 1 \\ \cos \omega \\ \cos 2\omega \\ \vdots \\ \cos(0.5(N-1)\omega) \end{bmatrix}_{r' \times 1} \left[1 \cos \omega \cos 2\omega \dots \cos(0.5(N-1)\omega) \right]_{1 \times r'} d\omega$$

where $r' = 0.5(N+1)$

$$P_s = \int_{\omega_s}^{\pi} \begin{bmatrix} 1 & \cos \omega & \cos 2\omega & \dots \cos(0.5(N-1)\omega) \\ \cos \omega & \cos^2 \omega & \cos \omega \cos 2\omega & \dots \cos \omega \cos[0.5(N-1)\omega] \\ \cos 2\omega & \cos 2\omega \cos \omega & \cos^2 2\omega & \dots \cos(2\omega) \cos[0.5(N-1)\omega] \\ \vdots & \vdots & \vdots & \ddots \vdots \\ \cos(0.5(N-1)\omega) & [\cos(0.5(N-1)\omega) \cos \omega] & [\cos(0.5(N-1)\omega) \cos 2\omega] & \dots \cos^2[0.5(N-1)\omega] \end{bmatrix} d\omega$$

$$(P_s)_{r,c} = (\pi - \omega_s) \quad \text{for } r = c = 1$$

$$(P_s) |_{r,c} = \int_{\omega_s}^{\pi} \cos^2[(r-1)\omega] d\omega \quad \text{for } r = c \neq 1$$

$$= \frac{1}{2} \int_{\omega_s}^{\pi} \{ \cos[2(r-1)\omega] + 1 \} d\omega$$

$$= \frac{1}{2} \left[\frac{\sin(2(r-1)\omega)}{2(r-1)} + \omega \right]_{\omega_s}^{\pi}$$

$$(P_s) |_{r,c} = \frac{1}{2} \left[\frac{-\sin(2(r-1)\omega_s)}{2(r-1)} + \pi - \omega_s \right]$$

$$(P_s) |_{r,c} = 0.5\pi - 0.5\omega_s - \frac{0.25 \sin(2(r-1)\omega_s)}{(r-1)} \quad \text{for } r=c \neq 1$$

$$(P_s) |_{r,c} = \int_{\omega_s}^{\pi} \cos[(c-1)\omega] \cos[(r-1)\omega] d\omega \quad \text{for } r \neq c, r \neq 1 \text{ \& } c \neq 1$$

$$= 0.5 \int_{\omega_s}^{\pi} \cos[(r+c-2)\omega] + \cos[(r-c)\omega] d\omega$$

$$(P_s) |_{r,c} = -0.5 \left[\frac{\sin(r+c-2)\omega_s}{(r+c-2)} + \frac{\sin((r-c)\omega_s)}{(r-c)} \right]$$

Hence for N-odd, P_s can be compactly expressed as

$$(P_s) |_{r,c} = \begin{cases} \pi - \omega_s & \text{for } r = c = 1 \\ 0.5(\pi - \omega_s) - \frac{0.25 \sin(2(r-1)\omega_s)}{(r-1)} & \text{for } r = c \neq 1 \\ -0.5 \left[\frac{\sin[r+c-2]\omega_s}{(r+c-2)} + \frac{\sin[(r-c)\omega_s]}{(r-c)} \right] & \text{for } r \neq c \end{cases} \quad \dots(4.20a)$$

For N even,

$$P_s = \int_{\omega_s}^{\pi} \begin{bmatrix} \cos(0.5\omega) \\ \cos(1.5\omega) \\ \cos(2.5\omega) \\ \vdots \\ \cos(0.5(N-1)\omega) \end{bmatrix}_{r' * 1} \left[\cos(0.5\omega) \cos(1.5\omega) \cos(2.5\omega) \dots \cos(0.5(N-1)\omega) \right]_{1 * r'} d\omega$$

where $r' = 0.5 N$

$$P_s = \int_{\omega_s}^{\pi} \begin{bmatrix} \cos^2(0.5\omega) & \cos(0.5\omega)\cos(1.5\omega) & \dots [\cos(0.5\omega)\cos(0.5(N-1)\omega)] \\ \cos(1.5\omega)\cos(0.5\omega) & \cos^2(1.5\omega) & \dots [\cos(1.5\omega)\cos(0.5(N-1)\omega)] \\ \cos(2.5\omega)\cos(0.5\omega) & \cos(1.5\omega)\cos(2.5\omega) & \dots [\cos(2.5\omega)\cos(0.5(N-1)\omega)] \\ \vdots & \vdots & \vdots \\ [\cos(0.5(N-1)\omega)\cos(0.5\omega)] & [\cos(1.5\omega)\cos(0.5\omega(N-1))] & \dots \cos^2[0.5(N-1)\omega] \end{bmatrix} d\omega$$

$$(P_s) |_{r,c} = \int_{\omega_s}^{\pi} \cos^2[(r-0.5)\omega] d\omega \quad \text{for } r = c$$

$$= \frac{1}{2} \int_{\omega_s}^{\pi} \{ \cos[2(r-0.5)\omega] + 1 \} d\omega$$

$$= \frac{1}{2} \left[\frac{\sin((2r-1)\omega)}{2r-1} + \omega \right]_{\omega_s}^{\pi}$$

$$(P_s) |_{r,c} = \frac{1}{2} \left[\frac{-\sin((2r-1)\omega_s)}{2r-1} + \pi - \omega_s \right]$$

$$(P_s) |_{r,c} = \int_{\omega_s}^{\pi} \cos[(c - 0.5)\omega] \cos[(r-0.5)\omega] d\omega \quad \text{for } r \neq c$$

$$= 0.5 \int_{\omega_s}^{\pi} \cos[(r+c-1)\omega] + \cos[(r-c)\omega] d\omega$$

$$(P_s) |_{r,c} = -0.5 \left[\frac{\sin((r+c-1)\omega_s)}{(r+c-1)} + \frac{\sin((r-c)\omega_s)}{(r-c)} \right]$$

Hence for N-even, P_s can be compactly expressed as

$$(P_s) |_{r,c} = \begin{cases} 0.5(\pi-\omega_s) - \frac{0.5 \sin((2r-1)\omega_s)}{(2r-1)} & \text{for } r = c \\ -0.5 \left[\frac{\sin[(r+c-1)\omega_s]}{(r+c-1)} + \frac{\sin[(r-c)\omega_s]}{(r-c)} \right] & \text{for } r \neq c \end{cases} \quad \dots(4.20b)$$

The weighted square error measure in the passband is

$$E_p = \int_0^{\omega_p} W(\omega) e_p^2(\omega) d\omega \quad \dots(4.21)$$

where $e_p(\omega) = H_a(\omega) - \gamma H_d(\omega)$

$$e_p(\omega) = x \cdot s^T(\omega) - \gamma H_d(\omega) \quad \dots(4.22)$$

where γ is a scale factor defined as

$$\gamma = \frac{H_a(\omega_0)}{H_d(\omega_0)} = \frac{s^T(\omega_0) \cdot x}{H_d(\omega_0)} \quad \dots(4.23)$$

Where ω_0 is the reference frequency usually taken as the frequency corresponding to the maximum of $H_d(\omega)$. For the case of lowpass filters $\omega_0 = 0$.

From (4.22) & (4.23)

$$e_p(\omega) = x \left[s^T(\omega) - \frac{s^T(\omega_0)}{H_d(\omega_0)} H_d(\omega) \right] \quad \dots(4.24)$$

Substituting (4.24) in (4.21)

$$E_p = x^T \int_0^{\omega_p} \left\{ W(\omega) \left[s(\omega) - \frac{s(\omega_0)}{H_d(\omega_0)} H_d(\omega) \right] \left[s^T(\omega) - \frac{s^T(\omega_0)}{H_d(\omega_0)} H_d(\omega) \right] \right\} d\omega * x$$

$$E_p = x^T P_p x \quad \dots(4.25)$$

where

$$P_p = \int_0^{\omega_p} W(\omega) \left[s(\omega) - \frac{s(\omega_0)}{H_d(\omega_0)} H_d(\omega) \right] \left[s(\omega) - \frac{s(\omega_0)}{H_d(\omega_0)} H_d(\omega) \right]^T d\omega \quad \dots(4.26)$$

Putting $\omega_0=0$, $W(\omega) = 1$, lowpass (odd length) filter desired response is unity at reference frequency ω_0 & in the complete passband. Now Eq.(4.26) can be written as

$$P_p = \int_0^{\omega_p} \left[s(\omega) - s(\omega_0) \right] \left[s^T(\omega) - s^T(\omega_0) \right] d\omega \quad \dots(4.27)$$

For N odd, substituting (4.8) in (4.27)

$$(P_p)_{r,c} = \int_0^{\omega_p} \begin{bmatrix} 0 \\ (\cos \omega) - 1 \\ \vdots \\ \cos[0.5(N-1)\omega] - 1 \end{bmatrix} * [0 (\cos \omega - 1) \dots (\cos[0.5(N-1)\omega] - 1)] d\omega$$

where P_p is a matrix of $r' * c'$ and $r' = c' = 0.5 (N+1)$

$$(P_p)_{r,c} = \int_0^{\omega_p} \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & (\cos\omega-1)^2 & \dots & \{\cos[0.5(N-1)\omega]-1\}(\cos\omega-1) \\ 0 & (\cos2\omega-1)(\cos\omega-1) & \dots & \{\cos[0.5(N-1)\omega]-1\}(\cos2\omega-1) \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \{\cos[0.5(N-1)\omega]-1\}(\cos\omega-1) & \dots & \{\cos[0.5(N-1)\omega]-1\}^2 \end{bmatrix} d\omega \quad \dots(4.28)$$

$$(P_p)_{r,c} = 0 \text{ for } r = 1 \text{ or } c = 1$$

$$(P_p)_{r,c} = \int_0^{\omega_p} \{ \cos[(r-1)\omega] - 1 \}^2 d\omega \text{ for } r=c \neq 1$$

$$= \int_0^{\omega_p} \{ \cos^2[(r-1)\omega] + 1 - 2\cos[(r-1)\omega] \} d\omega$$

$$= \int_0^{\omega_p} \left\{ \frac{1}{2} [\cos [2(r-1)\omega] + 1] + 1 - 2 \cos [(r-1)\omega] \right\} d\omega$$

$$= \int_0^{\omega_p} \left\{ \frac{1}{2} \cos [2(r-1)\omega] + \frac{3}{2} - 2 \cos [(r-1)\omega] \right\} d\omega$$

$$(P_p)_{r,c} = \frac{1}{2} \left[\frac{\sin[2(r-1)\omega_p]}{2(r-1)} + 3\omega_p \right] - \frac{2 \sin[(r-1)\omega_p]}{(r-1)}$$

$$(P_p)_{r,c} = \int_0^{\omega_p} \{ \cos[(c-1)\omega] - 1 \} * \{ \cos[(r-1)\omega] - 1 \} d\omega \text{ for } r \neq c, r \neq 1, c \neq 1$$

$$= \int_0^{\omega_p} \{ \cos[(c-1)\omega] \cos[(r-1)\omega] - \cos[(c-1)\omega] - \cos[(r-1)\omega] + 1 \} d\omega$$



$$= \int_0^{\omega_p} \left\{ 0.5 \cos[(r+c-2)\omega] + 0.5 \cos[(r-c)\omega] - \cos[(c-1)\omega] - \cos[(r-1)\omega] + 1 \right\} d\omega$$

$$(P_p)_{r,c} = \frac{0.5 \sin[(r+c-2)\omega_p]}{(r+c-2)} + \frac{0.5 \sin[(r-c)\omega_p]}{(r-c)} - \frac{\sin[(c-1)\omega_p]}{(c-1)} - \frac{\sin[(r-1)\omega_p]}{(r-1)} + \omega_p$$

Hence for N-odd, P_p can be compactly expressed as

$$(P_p)_{r,c} = \begin{cases} 0 & \text{for } r=1 \text{ or } c=1 \\ 1.5 \omega_p + \frac{0.25 \sin[2(r-1)\omega_p]}{2(r-1)} - \frac{2 \sin[(r-1)\omega_p]}{(r-1)} & \text{for } r=c \neq 1 \\ \omega_p + \frac{0.5 \sin[(r+c-2)\omega_p]}{(r+c-2)} + \frac{0.5 \sin[(r-c)\omega_p]}{(r-c)} - \frac{\sin[(r-1)\omega_p]}{(r-1)} - \frac{\sin[(c-1)\omega_p]}{(c-1)} & \text{for } r \neq c, r \neq 1, c \neq 1 \end{cases}$$

...(4.29a)

For N even, substituting (4.8) in (4.27)

$$(P_p)_{r,c} = \int_0^{\omega_p} \begin{bmatrix} \cos(0.5\omega) - 1 \\ \cos(1.5\omega) - 1 \\ \vdots \\ \cos[0.5(N-1)\omega] - 1 \end{bmatrix} [(\cos(0.5\omega) - 1)(\cos(1.5\omega) - 1) \dots (\cos(0.5(N-1)\omega) - 1)] d\omega$$

where P_p is a matrix of $r' \times c'$ and $r' = c' = 0.5N$

$$(P_p)|_{r,c} =$$

$$\int_0^{\omega_p} \begin{bmatrix} \{\cos(0.5\omega)-1\}^2 & \{\cos(0.5\omega)-1\}\{\cos(1.5\omega)-1\} \\ \{\cos(1.5\omega)-1\}\{\cos(0.5\omega)-1\} & \{\cos(1.5\omega)-1\}^2 \\ \vdots & \vdots \\ \{\cos[0.5(N-1)\omega]-1\}\{\cos(0.5\omega)-1\} & \{\cos[0.5(N-1)\omega]-1\}\{\cos(1.5\omega)-1\} \end{bmatrix}$$

$$\begin{bmatrix} \dots \{\cos(0.5\omega)-1\}\{\cos(0.5(N-1)\omega)-1\} \\ \dots \{\cos(1.5\omega)-1\}\{\cos(0.5(N-1)\omega)-1\} \\ \vdots \\ \dots \{\cos(0.5(N-1)\omega)-1\}^2 \end{bmatrix} d\omega$$

$$(P_p)|_{r,c} = \int_0^{\omega_p} \left\{ \cos[(r-0.5)\omega] - 1 \right\}^2 d\omega \quad \text{for } r = c$$

$$= \int_0^{\omega_p} \left\{ \cos^2[(r-0.5)\omega] + 1 - 2\cos[(r-0.5)\omega] \right\} d\omega$$

$$= \int_0^{\omega_p} \left\{ 0.5[\cos[2(r-0.5)\omega] + 1] + 1 - 2\cos[(r-0.5)\omega] \right\} d\omega$$

$$= \int_0^{\omega_p} \left\{ 0.5\cos[(2r-1)\omega] + 1.5 - 2\cos[(r-0.5)\omega] \right\} d\omega$$

$$(P_p)|_{r,c} = 0.5 \left[\frac{\sin[(2r-1)\omega_p]}{(2r-1)} + 3\omega_p \right] - \frac{2\sin[(r-0.5)\omega_p]}{(r-0.5)}$$

$$\begin{aligned}
(P_p)_{r,c} &= \int_0^{\omega_p} \{\cos[(c-0.5)\omega]-1\} * \{\cos[(r-0.5)\omega]-1\} d\omega \text{ for } r \neq c \\
&= \int_0^{\omega_p} \{\cos[(c-0.5)\omega]\cos[(r-0.5)\omega]- \cos[(c-0.5)\omega]-\cos[(r-0.5)\omega]+1\} d\omega \\
&= \int_0^{\omega_p} \{ 0.5\cos[(r+c-1)\omega]+0.5\cos[(r-c)\omega]-\cos[(c-0.5)\omega]-\cos[(r-0.5)\omega]+1 \} d\omega
\end{aligned}$$

$$\begin{aligned}
(P_p)_{r,c} &= \frac{0.5 \sin[(r+c-1)\omega_p]}{(r+c-1)} + \frac{0.5 \sin[(r-c)\omega_p]}{(r-c)} - \frac{\sin[(c-0.5)\omega_p]}{(c-0.5)} \\
&\quad - \frac{\sin[(r-0.5)\omega_p]}{(r-0.5)} + \omega_p
\end{aligned}$$

Hence for N-odd, P_p can be compactly expressed as

$$(P_p)_{r,c} = \begin{cases} 1.5 \omega_p + \frac{0.5 \sin[(2r-1)\omega_p]}{(2r-1)} - \frac{2\sin[(r-0.5)\omega_p]}{(r-0.5)} & \text{for } r=c \\ \omega_p + \frac{0.5\sin[(r+c-1)\omega_p]}{(r+c-1)} + \frac{0.5\sin[(r-c)\omega_p]}{(r-c)} - \frac{\sin[(r-0.5)\omega_p]}{(r-0.5)} - \frac{\sin[(c-0.5)\omega_p]}{(c-0.5)} & \text{for } r \neq c \end{cases}$$

...(4.29b)

In Eqs. (4.20) & (4.29) $r, c = 1, 2, \dots, m$ where $m = 0.5(N+1)$ for N odd and $m = 0.5N$ for N even. From Eqs.(4.20) & (4.29), it is evident that the matrices P_s and P_p are at least positive semidefinite. However, they are positive definite except for the case of N odd where matrix P_p becomes positive semidefinite as given by Eq. (4.28).

The total mean squared error, E , is a convex combination of the mean squared errors as given below:

$$E = \alpha E_s + (1-\alpha) E_p \quad \text{where } 0 \leq \alpha \leq 1 \quad \dots(4.30)$$

Substituting (4.18), (4.25) in (4.30)

$$E = x^T P x \quad \dots(4.31)$$

$$\text{where } P = \alpha P_s + (1-\alpha) P_p \quad \dots(4.32)$$

The symmetric matrix P is positive definite except in the extreme cases of $\alpha=0$ and $N=\text{odd}$ where P becomes positive semidefinite matrix. So α is restricted to the interval $0 < \alpha \leq 1$.

The linear system of Eq. (4.14) is undetermined since z is less than the length of vector x . So linear system (4.14) is always consistent. So, the problem can be written as:

Minimize $E = x^T P x$ subjected to constraint

$$C x = K \quad \dots(4.33)$$

where matrix P is positive definite and symmetric.

Linear phase maximally flat FIR filters are needed in many practical situations, particularly in applications requiring extremely high attenuation in stopband. Besides direct applications, maximally flat FIR filters have been used as a building blocks to improve the performance of equiripple filters [14].

DESIGNING OF FIR FILTER COEFFICIENTS

Many criteria are used for designing of FIR filters. Some of the methods used for designing FIR filters are already given in Chapter 4. In this chapter, designing of FIR filter coefficients is done using Frequency sampling method, Chebyshev approximation method using Linear Programming and Maximally Flat Approximation method.

5.1 FREQUENCY SAMPLING DESIGN OF LINEAR PHASE FIR FILTERS[2]

In this technique, N (filter length) equispaced samples of desired frequency response are taken. From these N samples of frequency response, the filter coefficients $\{h(n)\}$ are calculated.

N equally spaced samples of frequency response are given by

$$C_k = H(\omega) \Big|_{\omega=2\pi k/N} = H(2\pi k/N) \quad k = 0, 1, 2, \dots, N-1 \quad \dots(5.1)$$

and FIR filter (linear phase) coefficients are given by

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} C_k e^{j2\pi kn/N} \quad \dots(5.2)$$

From (3.9)

$$A_k = \sum_{n=0}^{M-1} 2h(n) \cos \left(\frac{2\pi(M-n)k}{N} \right) + h(M) \quad \dots(5.3)$$

where $M=(N-1)/2$ for N odd and $h(n)=h(N-n-1)$

From (5.1) and (3.8)

$$C_k = H \left(\frac{2\pi k}{N} \right) = A_k e^{-j2\pi kM/N} \quad \dots(5.4)$$

From (5.2) and (5.4)

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} e^{-j2\pi kM/N} A_k e^{j2\pi nk/N}$$

$$h(n) = \frac{1}{N} \sum_{k=0}^{N-1} A_k e^{j2\pi(n-M)k/N} \quad \dots(5.5)$$

$$= \frac{1}{N} \left[A_0 + A_1 e^{j2\pi(n-M)/N} + A_{N-1} e^{j2\pi(n-M)(N-1)/N} + A_2 e^{j2\pi(n-M)2/N} \right. \\ \left. + A_{N-2} e^{j2\pi(n-M)(N-2)/N} + \dots \right]$$

$$h(n) = \frac{1}{N} \left[A_0 + A_1 e^{j2\pi(n-M)/N} + A_{N-1} e^{-j2\pi(n-M)/N} + A_2 e^{j4\pi(n-M)/N} \right. \\ \left. + A_{N-2} e^{-j4\pi(n-M)/N} + \dots \right]$$

As $h(n)$ is real and symmetric, $A_k = A_{N-k}$ where $k=1, 2, \dots, M$

$h(n)$ can be expressed as:

$$h(n) = \frac{1}{N} \left[A_0 + \sum_{k=1}^M 2A_k \cos\left(\frac{2\pi(n-M)k}{N}\right) \right] \quad \dots(5.6a)$$

where $M=(N-1)/2$ for N odd

Due to symmetric property, i.e., $h(n)=h(N-n-1)$ only M of $h(n)$ need to be calculated. Equation (5.6a) is used for calculating the filter coefficients, where A_k (for $k=0,1,2,\dots, M$) are the samples of desired frequency response.

From (3.10)

$$A_k = \sum_{n=0}^{N/2-1} 2h(n) \cos\left(\frac{2\pi(M-n)k}{N}\right) \quad \dots(5.7)$$

where $M=(N-1)/2$ for N even and $h(n)=h(N-n-1)$

From Eq. (5.5)

$$h(n) = \frac{1}{N} \left[A_0 + A_1 e^{j2\pi(n-M)/N} + A_{N-1} e^{j2\pi(n-M)(N-1)/N} + A_2 e^{j2\pi(n-M)2/N} \right. \\ \left. + A_{N-2} e^{j2\pi(n-M)(N-2)/N} + \dots + A_{(0.5N-1)} e^{j2\pi(n-M)(0.5N-1)/N} \right. \\ \left. + A_{(0.5N+1)} e^{j2\pi(n-M)(N-(0.5N-1))/N} + A_{0.5N} e^{j\pi(n-M)} \right] \text{ for } N \text{ even.}$$

For type-2 filters i.e. when N is even

$$A_{0.5N} = 0$$

and $A_k = A_{N-k}$ for $k = 1, 2, \dots, ((N/2)-1)$

So,

$$h(n) = \frac{1}{N} \left[A_0 + A_1 e^{j2\pi(n-M)/N} + A_{N-1} e^{-j2\pi(n-M)/N} + A_2 e^{j4\pi(n-M)/N} \right. \\ \left. + A_{N-2} e^{-j4\pi(n-M)/N} + \dots + A_{(0.5N-1)} e^{j2\pi(n-M)(0.5N-1)/N} \right. \\ \left. + A_{(0.5N+1)} e^{-j2\pi(n-M)(0.5N-1)/N} \right]$$

$$h(n) = \frac{1}{N} \left[A_0 + \sum_{k=1}^{N/2-1} 2A_k \cos \left(\frac{2\pi(n-M)k}{N} \right) \right] \quad \dots(5.6b)$$

For odd symmetry, i.e., $h(n) = -h(N-1-n)$

(i) N odd

$$A_k = \sum_{n=0}^{M-1} 2h(n) \sin \left(\frac{2\pi(M-n)k}{N} \right)$$

$$h(n) = \frac{1}{N} \left[\sum_{k=1}^M 2A_k \sin\left(\frac{2\pi(M-n)k}{N}\right) \right]$$

(ii) N even

$$A_k = \sum_{n=0}^{N/2-1} 2h(n) \sin\left(\frac{2\pi(M-n)k}{N}\right)$$

$$h(n) = \frac{1}{N} \sum_{k=1}^{N/2-1} 2A_k \sin\left(\frac{2\pi(M-n)k}{N}\right) + A_{N/2} \sin(\pi(M-n))$$

After having calculated the filter coefficients, the designed filter frequency response can be obtained using Eqs. (5.3) or (5.7) for the case of even symmetric low pass filters.

Flow chart of filter designing using frequency sampling method is given in Fig.5.1. Desired frequency response is sampled at m_1 points, where $m_1 = (N+1)/2$ for N odd and $m_1 = N/2$ for N even. Desired response is saved in array `dis[]`. From the desired response and using Eq. (5.6a) for odd N or Eq. (5.6b) for even N, the filter coefficients are obtained. The frequency response of the designed filter is then obtained by using function `fresp()`.

5.2 FIR FILTER DESIGN USING CHEBYSHEV APPROXIMATION

CRITERION AND LINEAR PROGRAMMING [15]

When digital filters are implemented on a computer or with special-purpose hardware, each filter coefficient has to be represented by a finite number of bits. The simplest and most widely used approach to the problem is the rounding of the infinite precision coefficients to its b-bit representation.

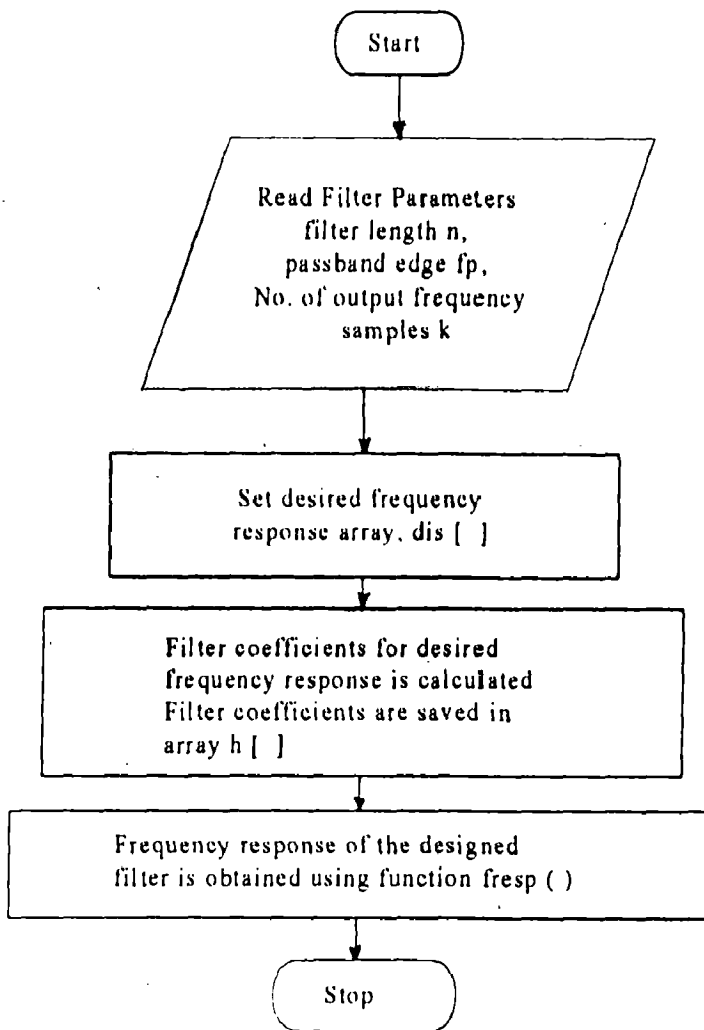


Fig. 5.1 : FLOW CHART OF FILTER DESIGNING USING FREQUENCY SAMPLING TECHNIQUE

To formulate the Chebyshev approximation problem required input specifications are as described in section 4.4.

The set F must be replaced by finite set of points for implementation on computer. A dense grid of points is used with spacing between points. Spacing between points is given by $(0.5/(lgrd * n))$ where n is the number of cosine functions as given in section 4.4 and $lgrd$ determines the number of frequency points at which constraints are to be formed. Both $D(e^{j2\pi f})$ and $W(e^{j2\pi f})$ are evaluated on this grid by functions $eff1()$ and $wt1()$ respectively as given in Appendix.

The coefficients $\{h(n)\}$ are found by solving Eq.(4.4) by linear programming. The coefficients found are of finite precision. For usual non-amplifying device $|h(k)| \leq 1$. So, we can express it in terms of $2^{-(b-1)}$, where b is the number of bits.

$$0 \leq |h(k)| \leq (1 - 2^{-(b-1)})$$

or

$$0 \leq |h(k)| \leq (2^{b-1} - 1) 2^{-(b-1)} \quad \dots(5.8)$$

$$k = 0, 1, \dots, n$$

Since most linear-programming techniques require that discrete variables are non-negative, so, the following substitution is used.

$$h^*(k) = 2^{(b-1)}(h(k) + 1) \quad \dots(5.9)$$

and the constraints in Eq. (4.4) for type-1 can be written as:

$$2^{(b-1)} \left[h(n) + 1 + \sum_{k=1}^n (1 + h(n-k)) \cdot 2g(k, f) - E/W(f) \right]$$

$$\leq 2^{(b-1)} \cdot \left(D(f) + 1 + \sum_{k=1}^n 2g(k, f) \right)$$

where $n = (N-1)/2$

or

$$h^*(n) + \sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq 2^{(b-1)} \cdot \left(D(f) + 1 + \sum_{k=1}^n 2g(k,f) \right)$$

Now the Chebyshev approximation problem for type-1 FIR filter becomes:

Minimize E subjected to constraints:

$$h^*(n) + \sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq 2^{(b-1)} \left(D(f) + 1 + \sum_{k=1}^n 2g(k,f) \right)$$

$$-h^*(n) - \sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq -2^{(b-1)} \cdot \left(D(f) + 1 + \sum_{k=1}^n 2g(k,f) \right) \quad \dots(5.10)$$

where $h^*(k) \in [1, 2, 3, \dots, 2^b - 1]$

The constraints in Eq. (4.4) for type-2 FIR filters can be written as:

$$2^{(b-1)} \left\{ \sum_{k=1}^n \left[h(n-k) + 1 \right] 2g(k,f) - E/W(f) \right\} \leq 2^{(b-1)} \cdot \left(D(f) + \sum_{k=1}^n 2g(k,f) \right)$$

where $n = N/2$

or

$$\sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq 2^{(b-1)} \cdot \left[D(f) + \sum_{k=1}^n 2g(k,f) \right]$$

Now the Chebyshev approximation problem for type-2 becomes:

Minimize E. subjected to constraints:

$$\sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq 2^{(b-1)} \left(D(f) + \sum_{k=1}^n 2g(k,f) \right)$$

$$- \sum_{k=1}^n h^*(n-k) \cdot 2g(k,f) - 2^{(b-1)} \cdot E/W(f) \leq - 2^{(b-1)} \left(D(f) + \sum_{k=1}^n 2g(k,f) \right) \quad \dots(5.11)$$

Flow chart of filter designing using Chebyshev approximation criterion and linear programming is given in Fig. 5.2. In this, number of frequency points at which constraints are to be formed is calculated and saved in variable ngrd. Constraints are formed by using Eqs. (5.10) and (5.11) for type-1 and type-2 filters respectively.

The Chebyshev approximation problem is solved for type-1 and type-2 filters using function malpp(). In function malpp(), two sub-function dspp1() and spp2() are used for calculating the pivoted row and column using dual simplex and simplex methods. After selecting the pivot using functions dsrow() and dscol() or scol() and srow(), condensed table is formed using function pctab(). When optimum solution is found the filter

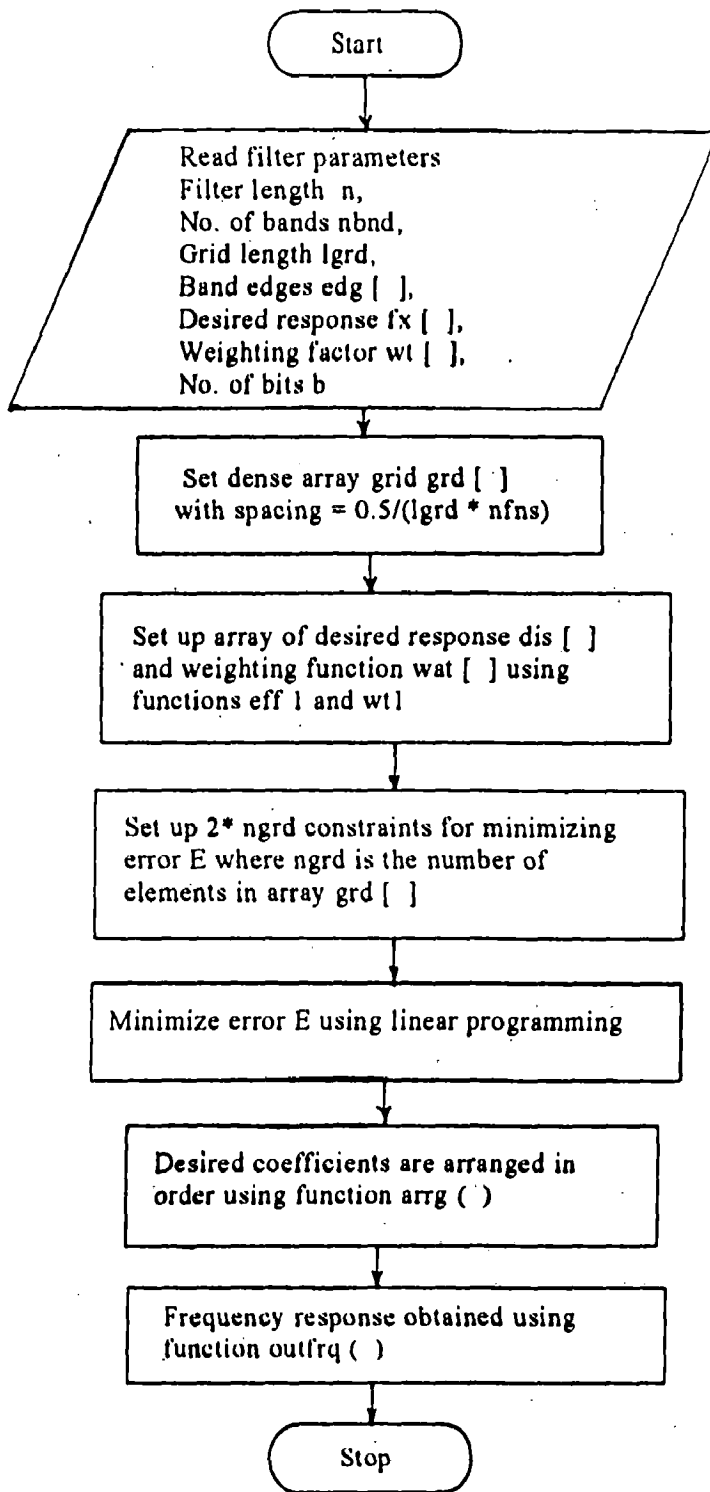


Fig. 5.2 : FLOW CHART OF FILTER DESIGNING USING CHEBYSHEV APPROXIMATION CRITERION AND LINEAR PROGRAMMING

coefficients are arranged in order using function `arrg()` and then the frequency response of designed filter is obtained using function `outrfq()`.

5.3 FILTER DESIGNING USING MAXIMALLY FLAT APPROXIMATION [7][17][18]

In this technique, matrix C , K , P_s and P_p are calculated using Eqs.(4.14), (4.20) and (4.29), where matrix C is of order of $z * m$, K is of order of $z * 1$ and P_s and P_p are of order of $m * m$. Number of constraints (one normalization condition plus even order derivative constraints) are z and $m=(N+1)/2$ for N odd or $m=N/2$ for N even. Using P_s and P_p matrix P is formed by using Eq.(4.32).

The linear system of (4.14) is undetermined since z is less than the length of vector x . So, linear system (4.14) is always consistent. And the problem can be written as:

Minimize $E = x^T P x$ subjected to constraints

$$Cx = K \quad \dots(5.12)$$

where matrix P is positive definite and symmetric.

This has family of solutions for $z < m$. To select the member of this family, Lagrange multipliers technique is used to get the unique solution.

The above problem (5.12) can be solved by the association of a Lagrange multiplier, λ , with passband constraints in Eq. (4.9). The Lagrange multiplier vector is then $\lambda = [\lambda_1 \ \lambda_2 \ \dots \ \lambda_z]^T$, where z denotes the number of constraints at a particular frequency, i.e., $\omega = 0$ for low-pass filters.

As objective function E and constraints Cx are differentiable w.r.t. x , a function $d(x)$ differentiable w.r.t. x and defined by $d(x) = Cx - K$ is introduced. Then the problem can be restated as

Minimize $E = x^T P x$ subject to the constraints:

$$d(x) = 0$$

For finding the necessary and sufficient condition for a minimum value of E , a new function is formed by introducing a Lagrange multiplier λ , as

$$L(x, \lambda) = x^T P x - \lambda^T d(x)$$

$$L(x, \lambda) = x^T P x - \lambda^T (Cx - K)$$

The vector λ is unknown constant vector and the function $L(x, \lambda)$ is called the Lagrangian function, with Lagrange multiplier λ . The necessary condition for a minimum of E subject to $d(x) = 0$ are thus given by

$$\frac{\partial L(x, \lambda)}{\partial x} = 0$$

and

$$\frac{\partial L(x, \lambda)}{\partial \lambda} = 0$$

or

$$\nabla_x L = 0 \quad \dots(5.13)$$

$$\nabla_\lambda L = 0 \quad \dots(5.14)$$

Now the partial derivatives are given by

$$\frac{\partial L}{\partial x} = \frac{\partial E}{\partial x} - \lambda^T \frac{\partial d}{\partial x}$$

$$\frac{\partial L}{\partial \lambda} = -d$$

where L , d stands for $L(x, \lambda)$, $d(x)$ respectively.

The necessary conditions for a minimum solution are given by

$$\frac{\partial E}{\partial x} = \lambda^T \frac{\partial d(x)}{\partial x} \quad \dots(5.15)$$

$$-d(x) = 0 \quad \dots(5.16)$$

The necessary condition becomes sufficient condition for a minimum if the objective function is convex and side constraints are in the form of equalities.

As the objective function of Eq. (5.12) is convex and constraints are in the form of equalities, Eqs.(5.15) and (5.16) are the necessary and sufficient conditions for a minimum of E.

From Eqs.(5.15) and (5.16)

$$\begin{aligned} \frac{1}{2} Px &= \lambda C^T \\ -Cx + K &= 0 \end{aligned}$$

or these can be written in matrix form as

$$\begin{bmatrix} \frac{1}{2} P & -C^T \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ -K \end{bmatrix}$$

$$\text{or } \frac{1}{2} Px - C^T \lambda = 0 \quad \dots(5.17)$$

$$-Cx + K = 0 \quad \dots(5.18)$$

$$\text{From Eq. (5.18) } x = C^{-1} K \quad \dots(5.19)$$

$$\text{From Eq. (5.17) } x = P^{-1} C^T \cdot 2\lambda \quad \dots(5.20)$$

$$\text{or } Px = C^T \cdot 2\lambda \quad \dots(5.21)$$

From Eqs. (5.19) and (5.21)

$$PC^{-1}K = C^T \cdot 2\lambda$$

$$\text{or } 2\lambda = (C^T)^{-1} PC^{-1} K$$

$$\text{or } 2\lambda = (CP^{-1}C^T)^{-1} K \quad \dots(5.22)$$

From Eqs. (5.20) and (5.22)

$$x = P^{-1}C^T(CP^{-1}C^T)^{-1}K \quad \dots(5.23)$$

So, the solution of problem (5.12) is given by Eq.(5.23) and the solution is unique as matrix C is assumed to be having full row rank and matrix P^{-1} is non-singular and consequently $(CP^{-1}C^T)$ is non-singular. From vector x the filter coefficients are calculated as given by Eq.(4.7).

Flow chart of designing of filter using maximally flat approximation criterion is given in Fig.5.3. Filter coefficients are obtained using Eqs.(5.23) and (4.7). From the designed filter coefficients, filter frequency response is obtained. Program listing using MATLAB for odd and even length filters are given in Appendix.

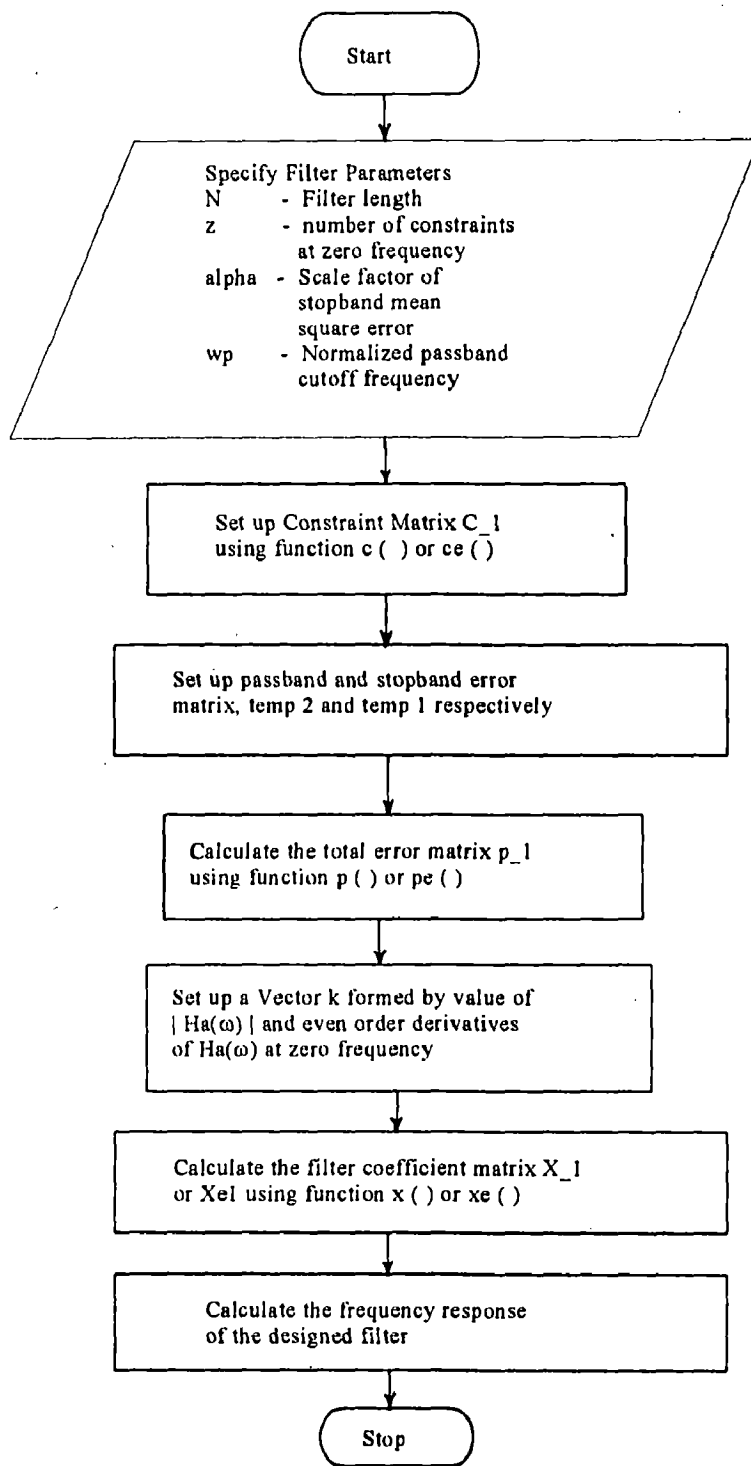


Fig. 5.3 : FLOW CHART OF FILTER DESIGNING USING MAXIMALLY FLAT APPROXIMATION TECHNIQUE

RESULTS, CONCLUSION AND FUTURE SCOPE

6.1 RESULTS

The impulse response for the designed filters using different techniques are given in Tables 6.1 to 6.7 as per the specifications mentioned in chapter-1 (Tables 1.1 to 1.3).

Four lowpass filters are designed using frequency sampling technique as given in section 5.1. The impulse response of these filters are given in Table 6.1 and the obtained frequency responses are shown in Figs.6.1 to 6.4.

Four lowpass filters are also designed using Chebyshev approximation criterion and Linear Programming technique as described in section 5.2. The obtained frequency responses are shown in Figs.6.5 to 6.8. The impulse response of the designed filters are given in Table 6.2. It is found that the desired specifications are met using above mentioned techniques.

Finally, seventeen more filters are designed using Maximally Flat Approximation Technique as described in section 5.3. The obtained frequency responses are shown in Figs.6.9 to 6.16. The corresponding impulse response of the designed filters are given in Tables 6.3 to 6.7. In Fig. 6.9, different curves are plotted for scale factor $\alpha=0.01, 0.5, 0.8, 1$; when the filter length $N=33$, number of constraints $z=3$ and the passband cutoff frequency $w_p=0.15$. In these graphs, log magnitude is plotted against the normalized frequency (frequency is normalized to 2π).

Figs.6.10 to 6.13 are the plots for lowpass filters with $z=3, w_p=0.15, \alpha=1$ and $N=21, 22, 33, 40$. In Fig.6.14, different curves are plotted for lowpass filters with $N=21, w_p=0.15, \alpha=1$ and $z=2, 3, 4, 5$. In

Fig.6.15, different curves are plotted for lowpass filters with $N=33$, $z=3$, $\alpha=0.01$ and $w_p=0.1, 0.15, 0.2, 0.25$. Fig.6.16 is the graph for lowpass filter with $N=33$, $z=2$, $w_p=0.15$ and $\alpha=0.5$.

By looking at the Figs.6.9 to 6.16, it is found that best results are obtained for filter length $N=33$, $w_p=0.15$ when $\alpha=0.5$ and number of constraints $z=2$.

Table-6.1 Finite impulse response of lowpass filters
Technique - 1 (Designing using frequency sampling technique)

	Filter-1	Filter-2	Filter-3	Filter-4
Length of filter n	20	21	30	31
h(0)	-0.032573	-0.032480	-0.023603	-0.023410
h(1)	0.043843	0.038188	0.023864	0.021257
h(2)	0.020711	0.028817	0.024402	0.026350
h(3)	-0.057021	-0.047619	-0.025247	-0.019651
h(4)	-0.005159	-0.026427	-0.026453	-0.030492
h(5)	0.076751	0.065171	0.028104	0.018447
h(6)	-0.022339	0.024916	0.030329	0.036624
h(7)	-0.120711	-0.106999	-0.033333	-0.017551
h(8)	0.111910	-0.024078	-0.037453	-0.046441
h(9)	0.484588	0.318607	0.043277	0.016904
h(10)		0.523810	0.051918	0.064348
h(11)			-0.065771	-0.016466
h(12)			-0.091068	-0.106513
h(13)			0.150672	0.016212
h(14)			0.450364	0.318446
h(15)				0.483871

Note : As the filter is symmetric, $h(i) = h(n-i-1)$

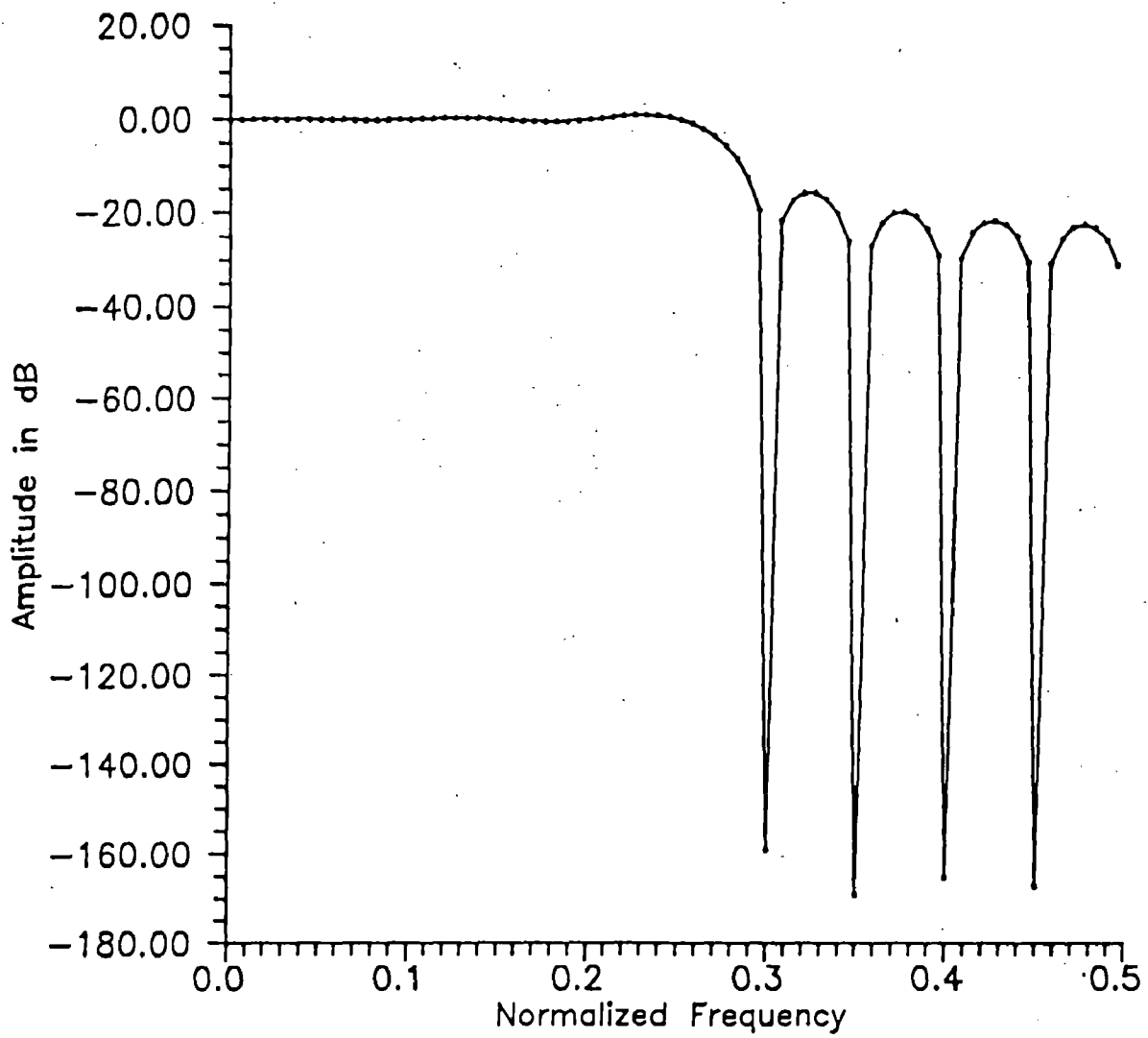


Fig. 6.1 Length-20 low-pass FIR filter frequency response by frequency sampling method

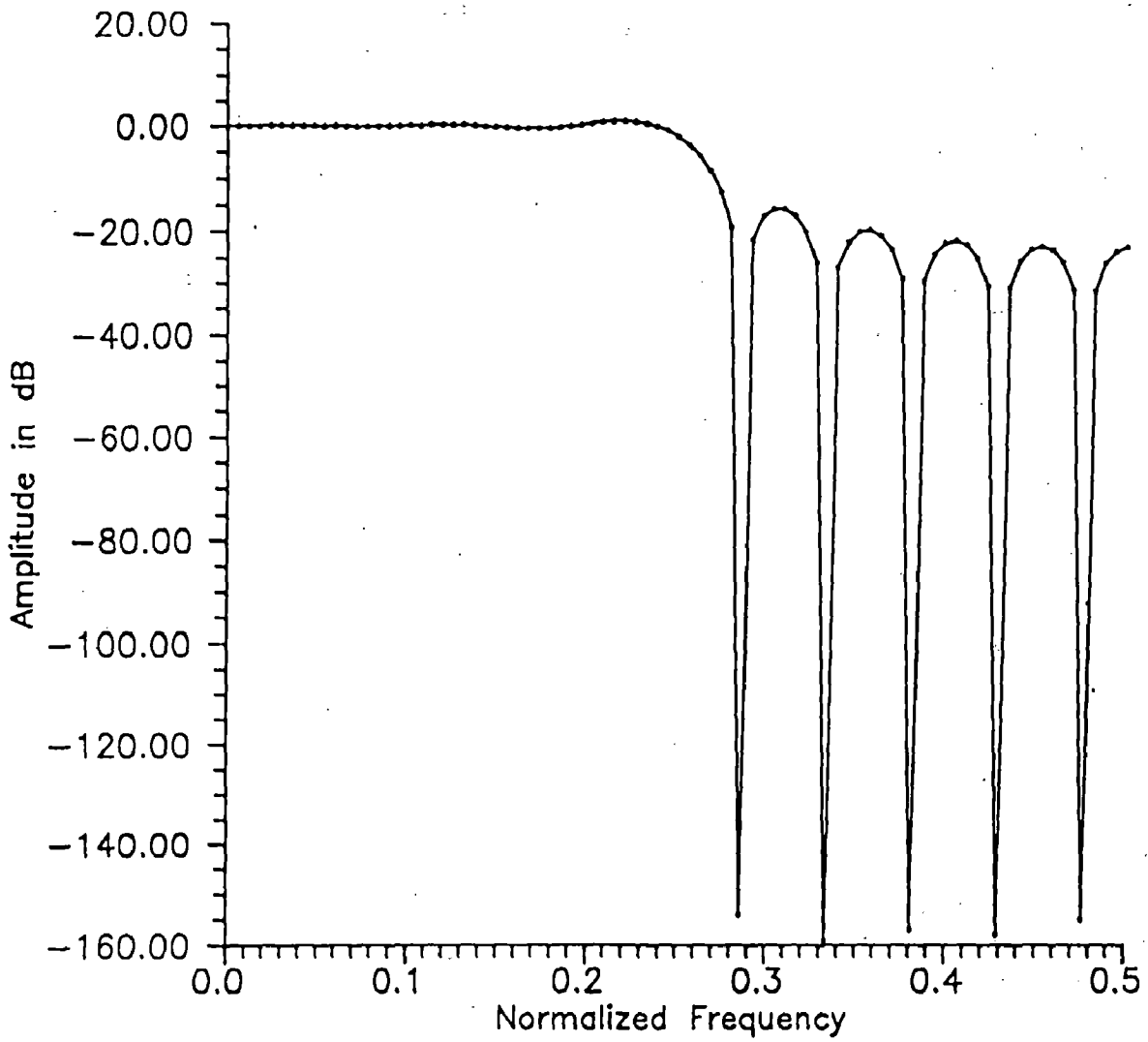


Fig. 6.2 Length-21 low-pass FIR filter frequency response by frequency sampling method

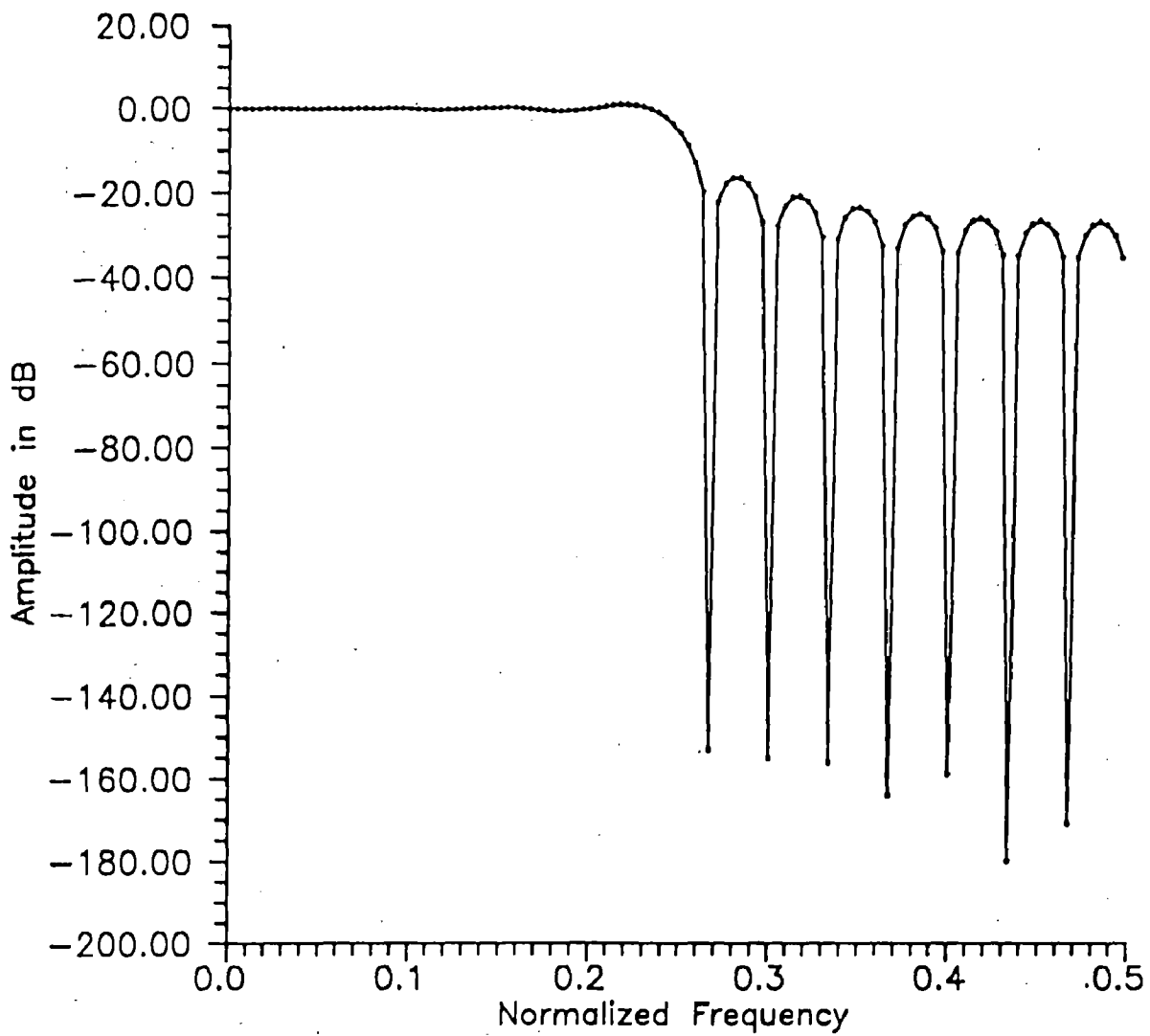


Fig. 6.3 Length-30 low-pass FIR filter frequency response by frequency sampling method

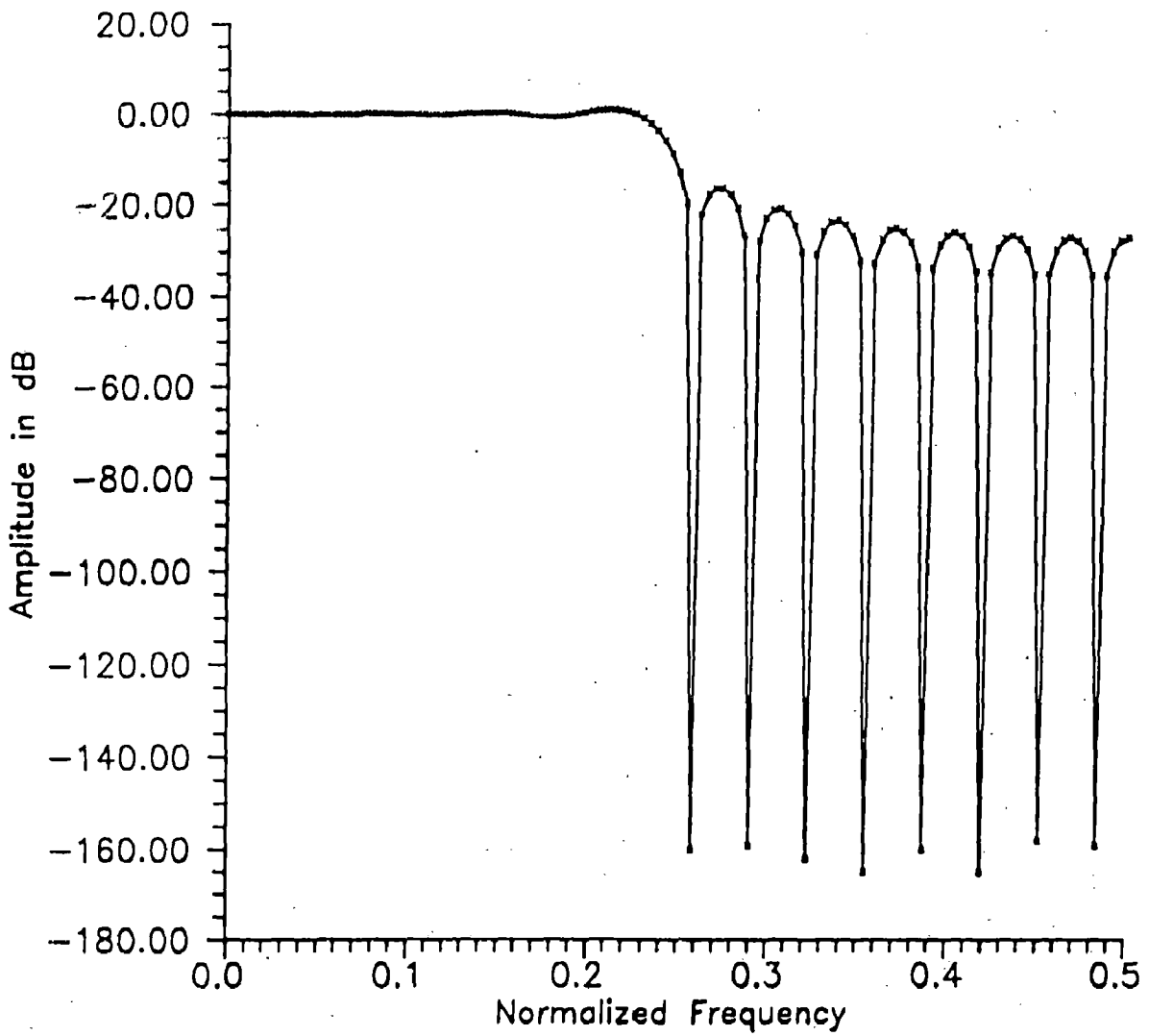


Fig. 6.4 Length-31 low-pass FIR filter frequency response by frequency sampling method

**Table-6.2 Finite impulse response of lowpass filters
Technique-2 (Designing using Chebyshev approximation
criterion & Linear Programming)**

	Filter-1	Filter-2	Filter3	Filter-4
Length of filter (n)	20	21	31	40
h(0)	2	1	1	3
h(1)	1	0	1	3
h(2)	-3	-2	0	-3
h(3)	0	0	-1	-4
h(4)	3	3	0	2
h(5)	1	2	1	6
h(6)	-6	-4	0	0
h(7)	-4	-5	-2	-9
h(8)	11	4	-2	-5
h(9)	27	20	2	9
h(10)		28	3	9
h(11)			-3	-9
h(12)			-6	-17
h(13)			3	6
h(14)			20	27
h(15)			29	1
h(16)				-44
h(17)				-22
h(18)				93
h(19)				210

Note : As the filter is symmetric, $h(i) = h(n-i-1)$

Rounded b bit coefficients multiplied by 2^b

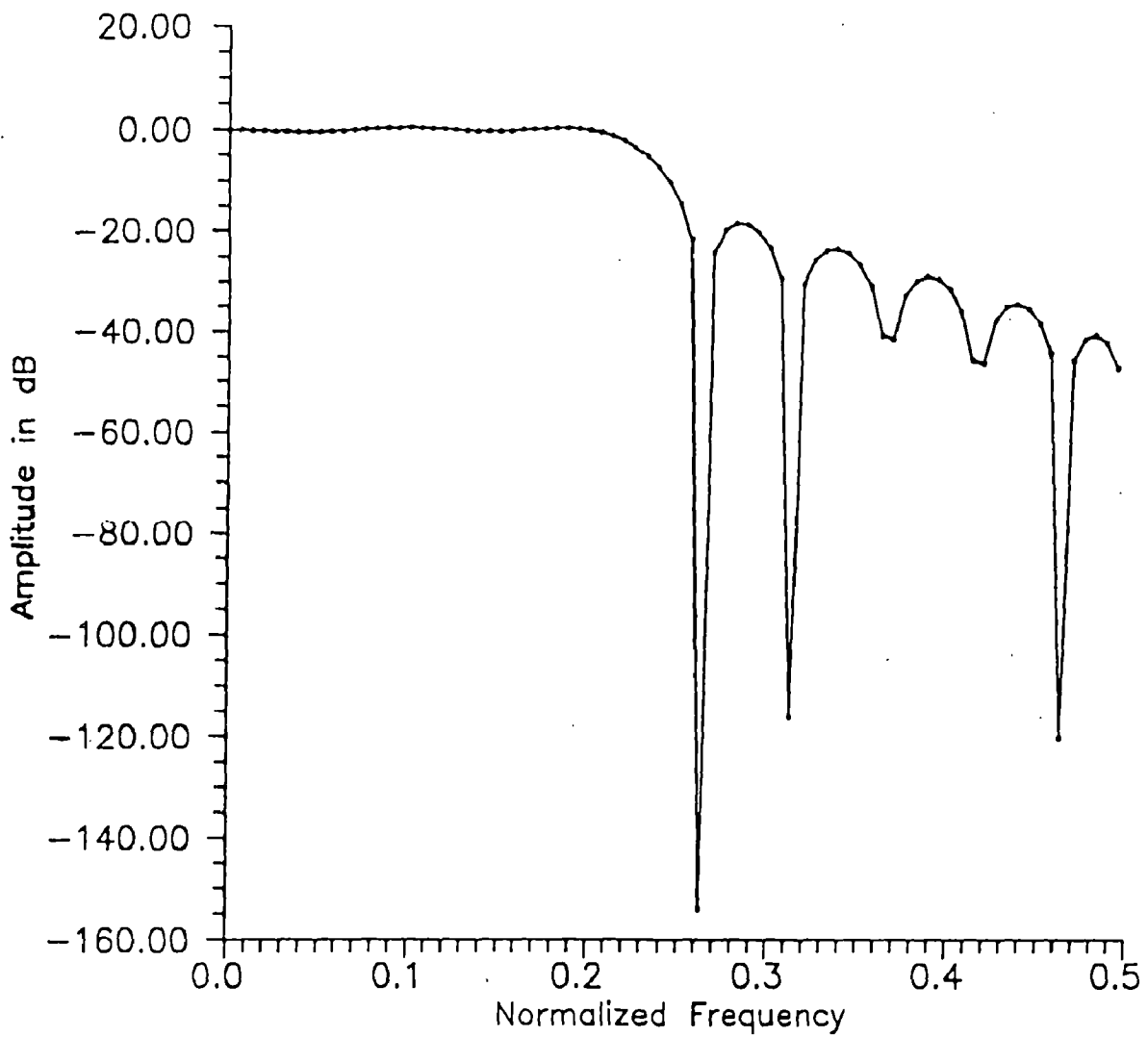


Fig. 6.5 Length-20 low-pass FIR filter frequency response by Chebyshev approximation method and Linear Programming Technique

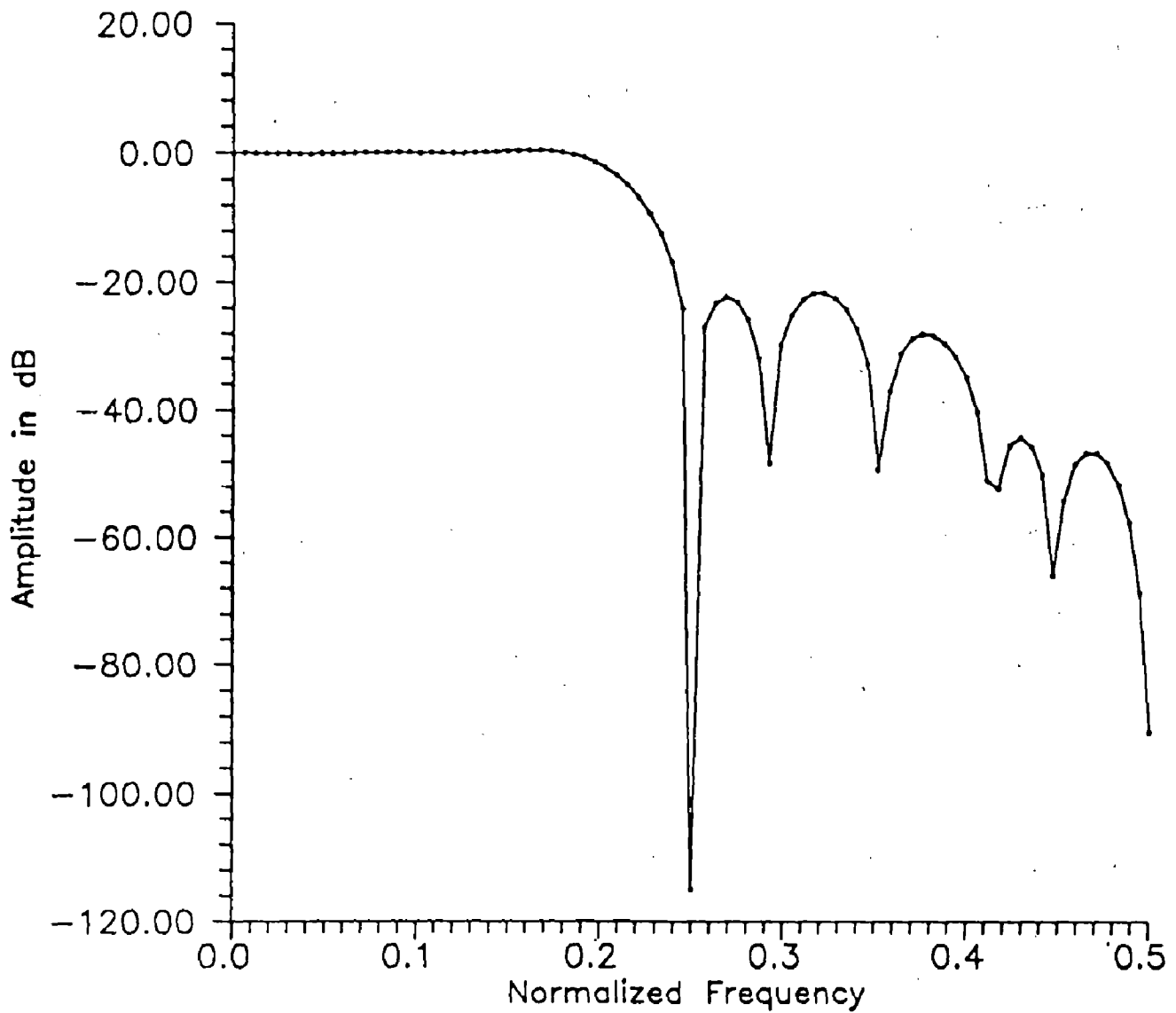


Fig. 6.6 Length-21 low-pass FIR filter frequency response by Chebyshev approximation method and Linear Programming Technique

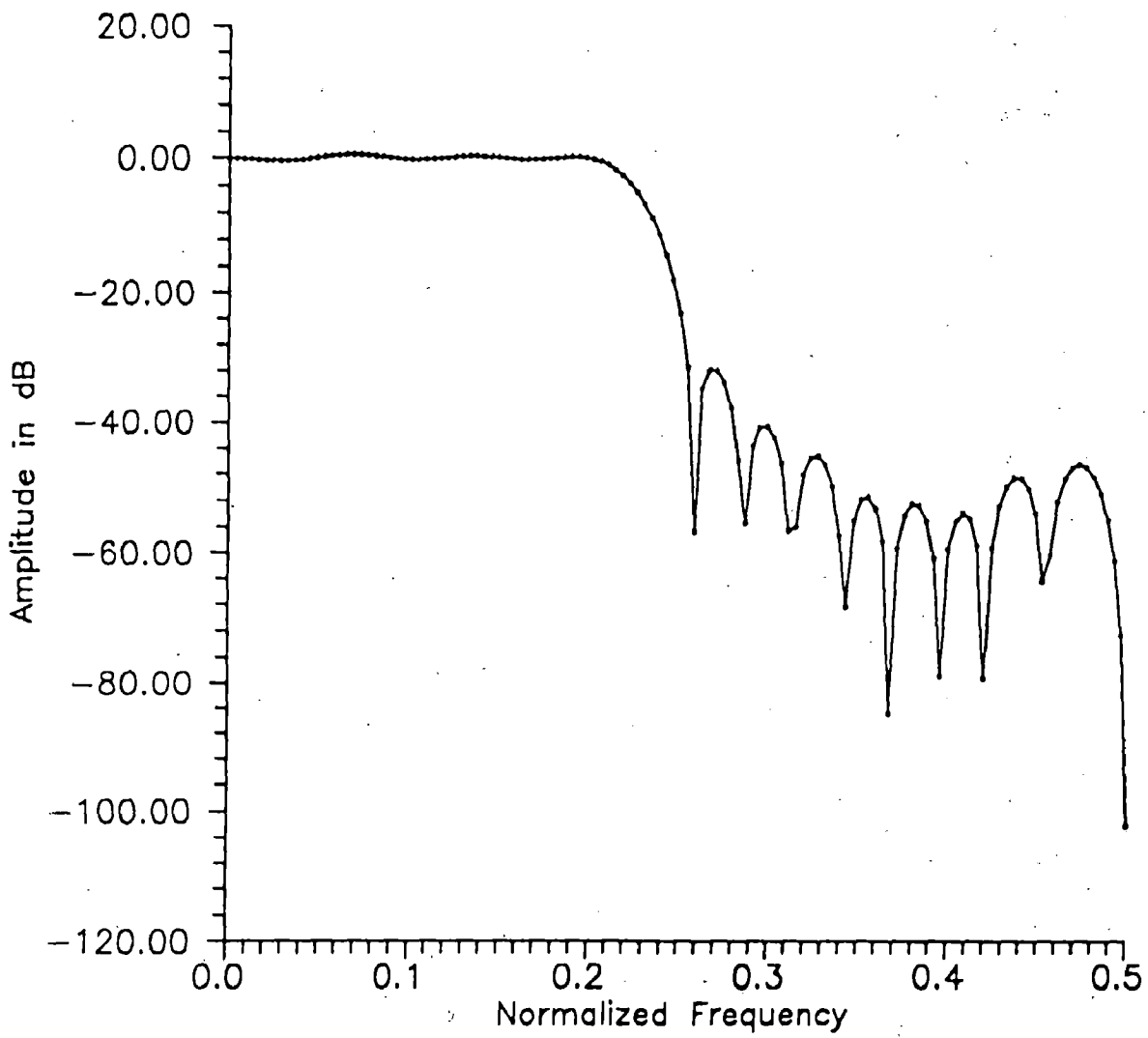


Fig. 6.7 Length-31 low-pass FIR filter frequency response by Chebyshev approximation method and Linear Programming Technique

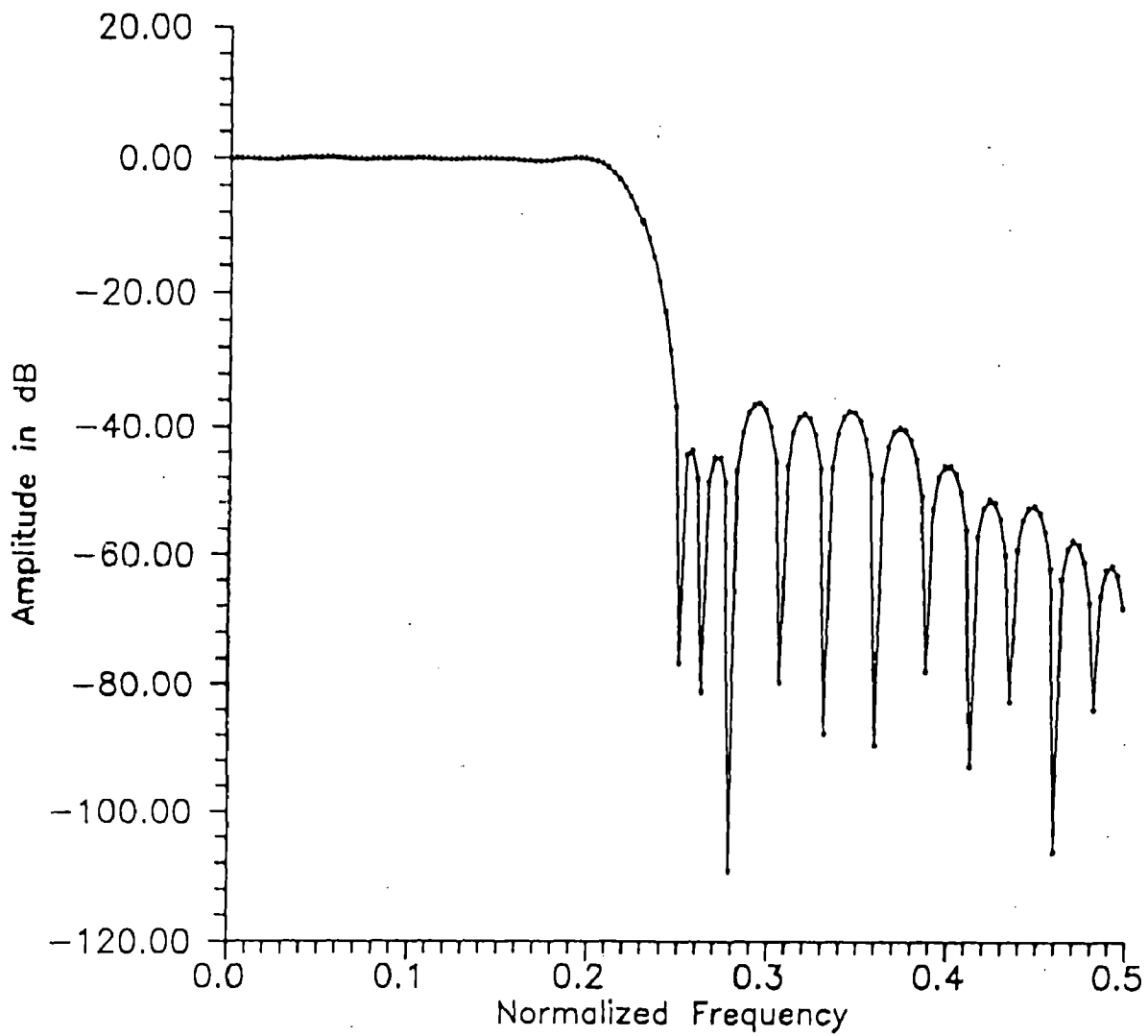


Fig. 6.8 Length-40 low-pass FIR filter frequency response by Chebyshev approximation method and Linear Programming Technique

Table-6.3 Finite impulse response of lowpass filters
Technique-3 (Designing using maximally flat approximation method)
For N=33, z=3, wp=0.15

	Filter-1	Filter-2	Filter3	Filter-4
Scale factor 'alpha'	0.01	0.5	0.8	1
h(0)	3.4052e-3	-3.9687e-3	7.0355e-3	3.1783e-4
h(1)	-1.7453e-2	1.1961e-2	7.6266e-3	1.0237e-3
h(2)	8.1769e-3	8.9008e-3	-3.713e-3	2.0070e-3
h(3)	2.3451e-2	-8.6018e-3	-1.8814e-2	2.6828e-3
h(4)	6.4041e-3	-2.3981e-2	-2.4157e-2	2.0407e-3
h(5)	-2.2609e-2	-2.0822e-2	-1.1050e-2	-9.4821e-4
h(6)	-2.9166e-2	3.1804e-3	1.5806e-2	-6.592e-3
h(7)	-2.0280e-3	3.1807e-2	3.8736e-2	-1.3696e-2
h(8)	3.3649e-2	4.0628e-2	3.8124e-2	-1.9200e-2
h(9)	3.8740e-2	1.6149e-2	7.6916e-3	-1.8640e-2
h(10)	-1.2615e-3	2.9932e-2	-3.6743e-2	-7.5330e-3
h(11)	-5.5860e-2	-6.3309e-2	-6.2811e-2	1.6689e-2
h(12)	-6.7647e-2	-4.7279e-2	-4.0084e-2	5.2973e-2
h(13)	2.6563e-3	3.1539e-2	3.9223e-2	9.6006e-2
h(14)	1.3943e-1	1.4954e-1	1.5095e-1	1.3715e-1
h(15)	2.7465e-1	2.5534e-1	2.4856e-1	1.6686e-1
h(16)	3.3094e-1	2.977e-1	2.8723e-1	1.7770e-1
h(17)				

Note : As the filter is symmetric, $h(i) = h(N-i-1)$

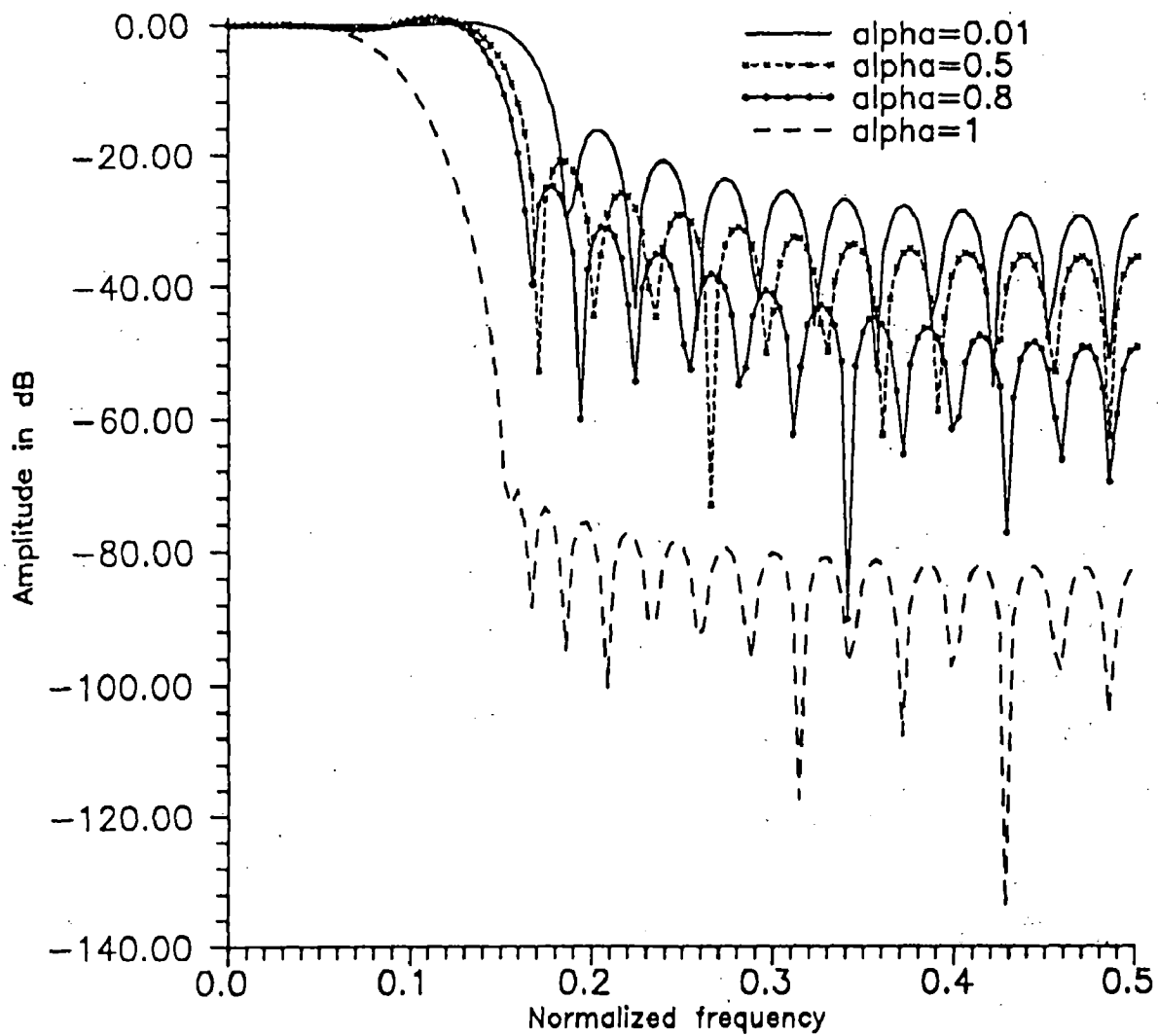


Fig. 6.9 Log amplitude plots for low-pass filters with $N=33$, $z=3$, $w_p=0.15$

Table-6.4 Finite impulse response of lowpass filters.
Technique-3 (Designing using maximally flat approximation method)
For $z = 3$, $\alpha = 1$, $w_p = 0.15$

	Filter-5	Filter-6	Filter-7	Filter-8
Length of filter 'N'	21	22	33	40
h(0)	8.7938e-3	6.8862e-3	3.1783e-4	3.6544e-5
h(1)	4.7021e-3	5.3027e-3	1.0237e-3	1.7251e-4
h(2)	-7.0270e-3	-3.1267e-3	2.0070e-3	4.8838e-4
h(3)	-2.2130e-2	-1.6603e-2	2.6828e-3	1.0182e-3
h(4)	-3.0182e-2	-2.7505e-2	2.0407e-3	1.6541e-3
h(5)	-1.8844e-2	-2.4665e-2	-9.4821e-4	2.0607e-3
h(6)	1.9205e-2	1.1761e-3	-6.5920e-3	1.6748e-3
h(7)	8.0392e-2	5.1335e-2	-1.3696e-2	-1.3914e-4
h(8)	1.4925e-1	1.1612e-1	-1.92e-2	-3.7603e-3
h(9)	2.0367e-1	1.7705e-1	-1.8640e-2	-8.8872e-3
h(10)	2.2435e-1	2.1430e-1	-7.5330e-3	-1.4191e-2
h(11)			1.6689e-2	-1.7267e-2
h(12)			5.2973e-2	-1.5038e-2
h(13)			9.6006e-2	-4.6226e-3
h(14)			1.3715e-1	1.5555e-2
h(15)			1.6686e-1	4.4797e-2
h(16)			1.7770e-1	7.9720e-2
h(17)				1.1469e-1
h(18)				1.4306e-1
h(19)				1.5898e-1
h(20)				

Note : As the filter is symmetric, $h(i) = h(N-i-1)$

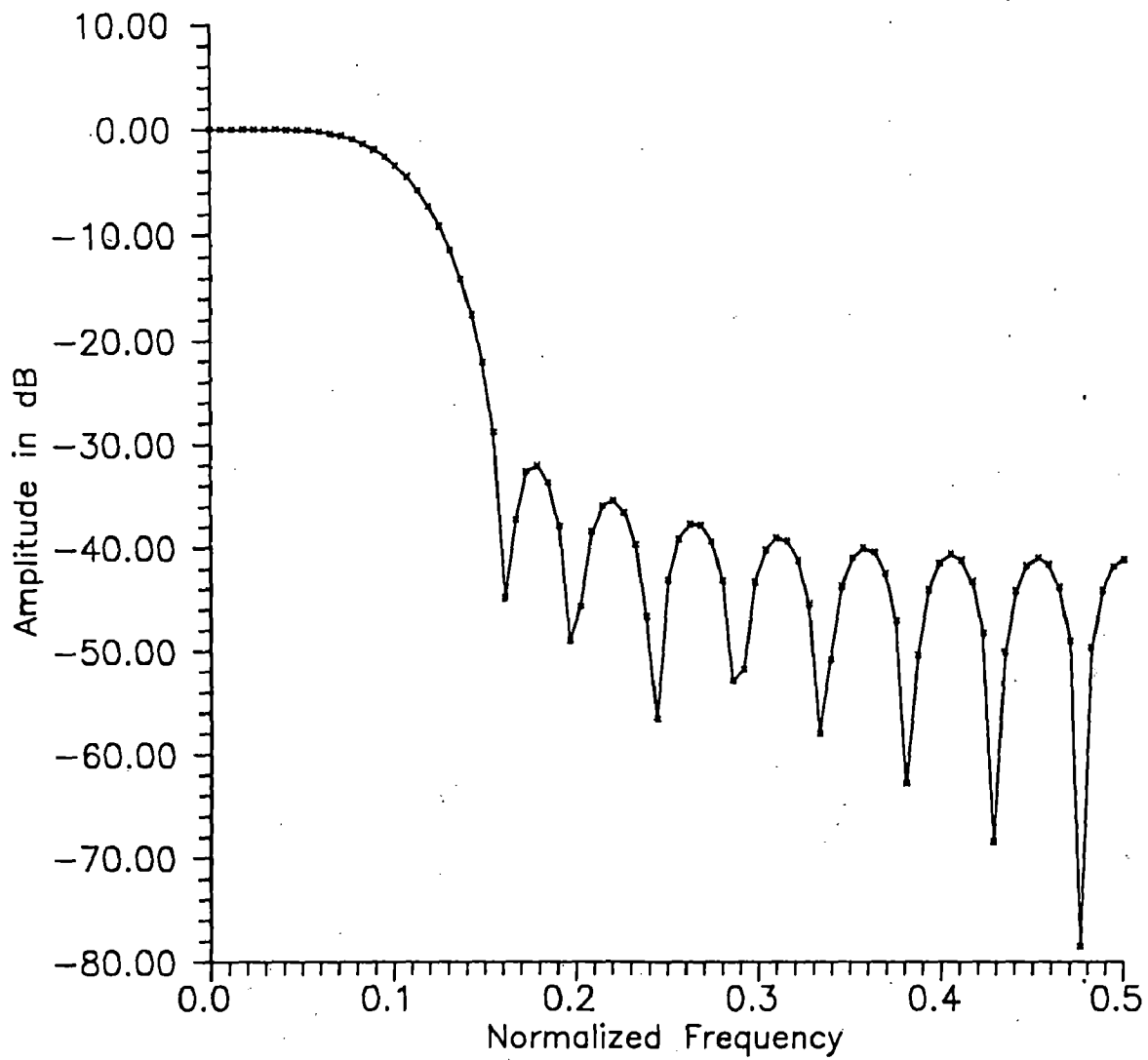


Fig. 6.10 Log amplitude plot for a low-pass filter with $N=21$, $w_p=0.15$, $\alpha=1$, $z=3$

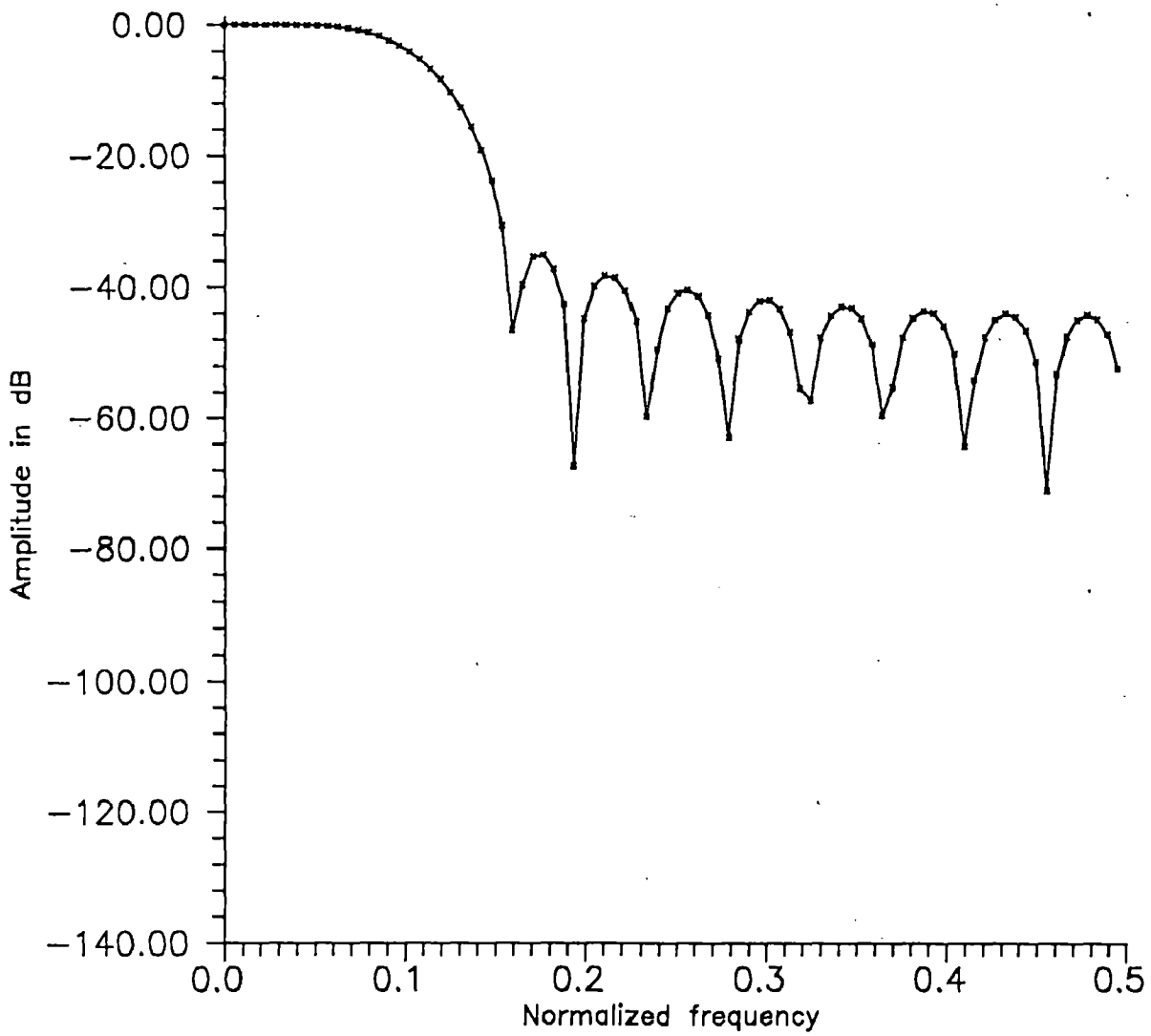


Fig. 6.11 Log amplitude plot for a low-pass filter with $N=22$, $z=3$, $w_p=0.15$, $\alpha=1$.

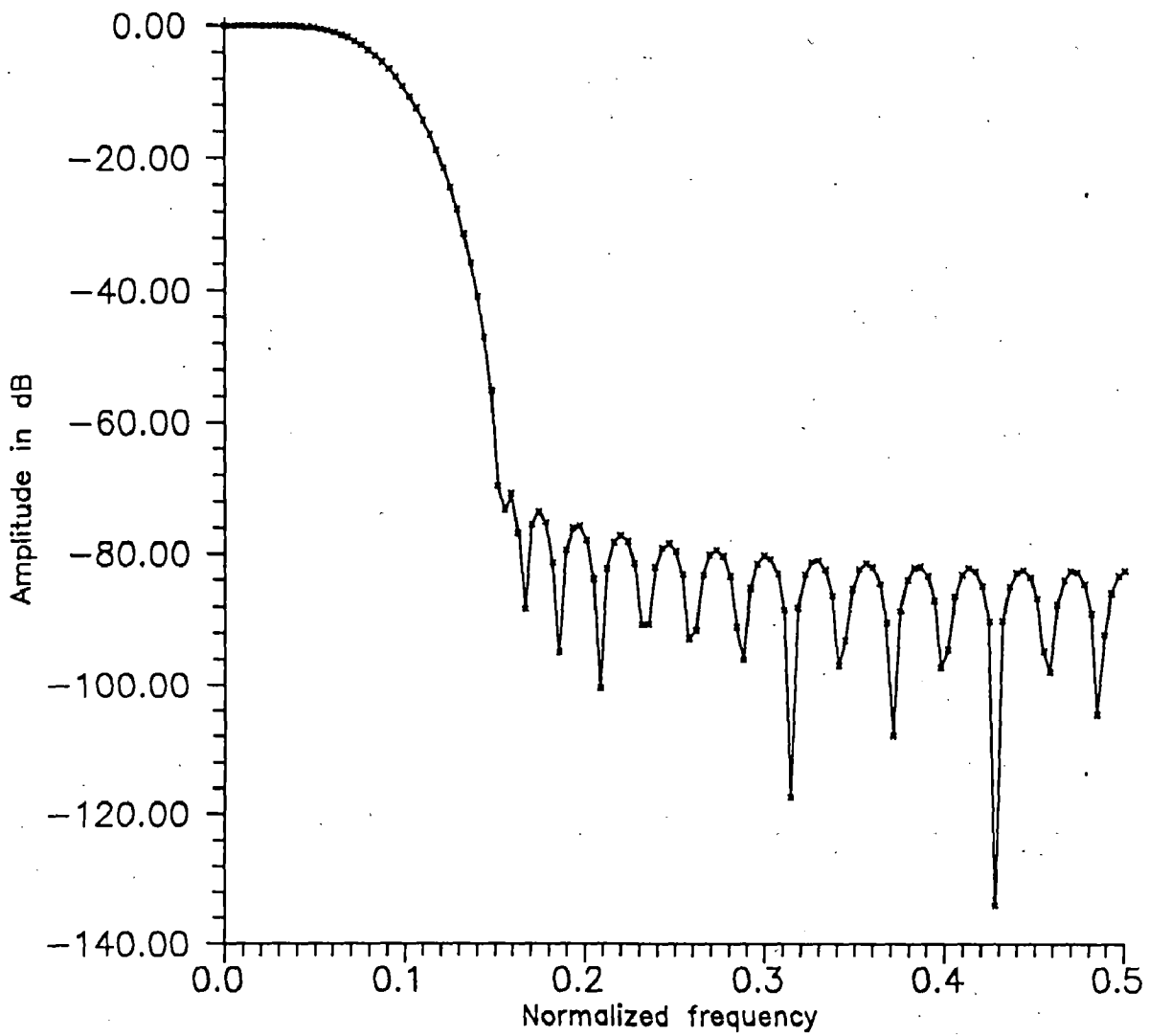


Fig. 6.12 Log amplitude plot for a low-pass filter with $N=33$, $z=3$, $w_p=0.15$, $\alpha=1$

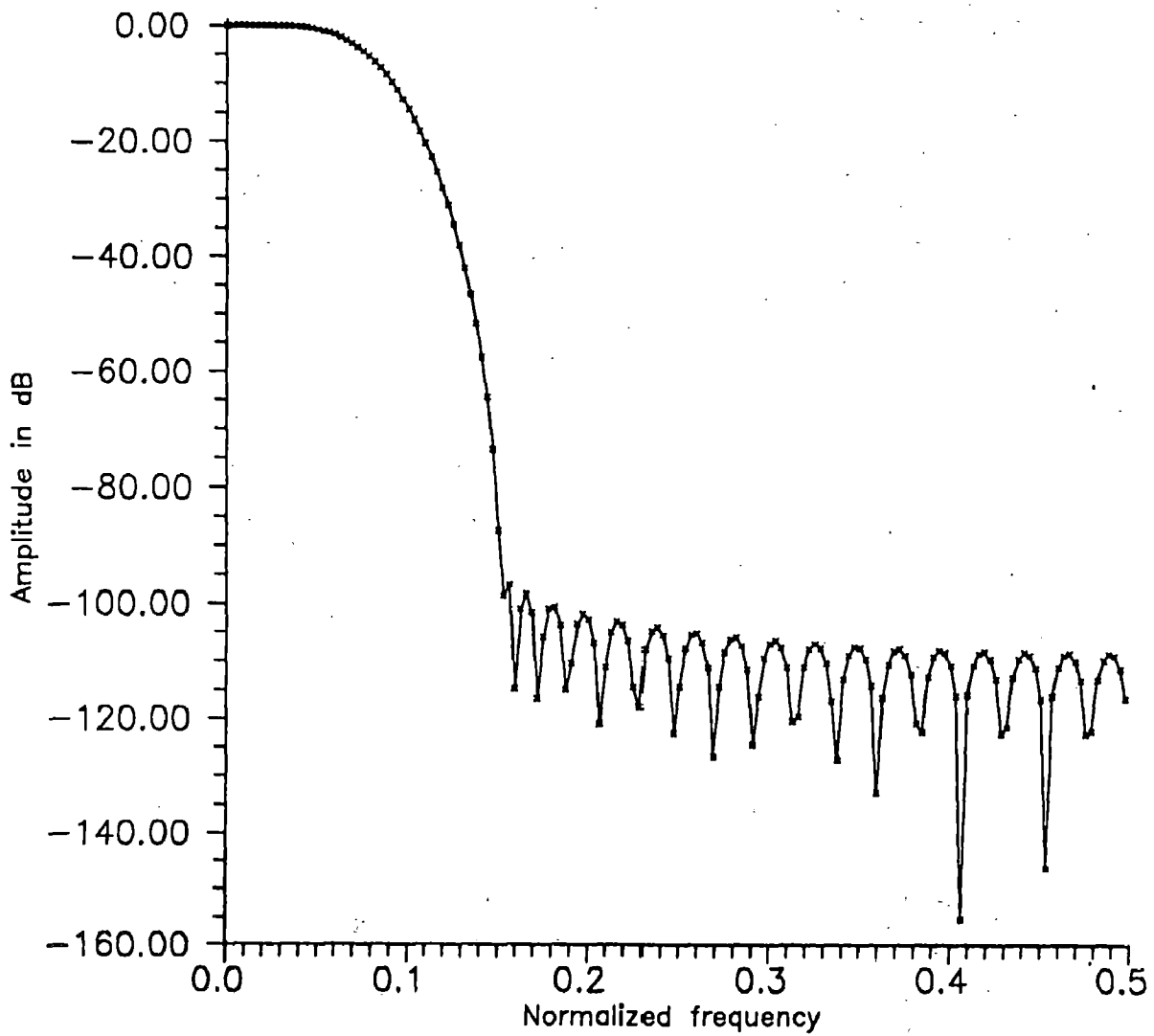


Fig. 6.13 Log amplitude plot for a low-pass filter with $N=40$, $z=3$, $w_p=0.15$, $\alpha=1$

Table-6.5 Finite impulse response of lowpass filters
Technique-3 (Designing using maximally flat approximation method)
For N=21, alpha = 1, wp = 0.15

	Filter-9	Filter-10	Filter-11	Filter-12
Number of constraints 'z'	2	3	4	5
h(0)	-2.9437e-3	8.7938e-3	-1.3421e-2	1.0412e-2
h(1)	-7.6981e-3	4.7021e-3	2.1201e-2	-3.9387e-2
h(2)	-1.2741e-2	-7.0270e-3	2.0243e-2	2.6956e-2
h(3)	-1.4057e-2	-2.2130e-2	-5.8195e-3	4.1512e-2
h(4)	-6.4670e-3	-30182e-2	-3.6902e-2	-6.1871e-3
h(5)	1.4151e-2	-1.8844e-2	-4.6960e-2	-5.9305e-2
h(6)	4.8330e-2	1.9205e-2	-1.5828e-2	-5.8145e-2
h(7)	9.1499e-2	8.0392e-2	5.8114e-2	1.9557e-2
h(8)	1.3451e-1	1.4925e-1	1.5339e-1	1.4666e-1
h(9)	1.6636e-1	2.0367e-1	2.3356e-1	2.6301e-1
h(10)	1.7812e-1	2.2435e-1	2.6485e-1	3.0984e-1

Note : As the filter is symmetric, $h(i) = h(N-i-1)$

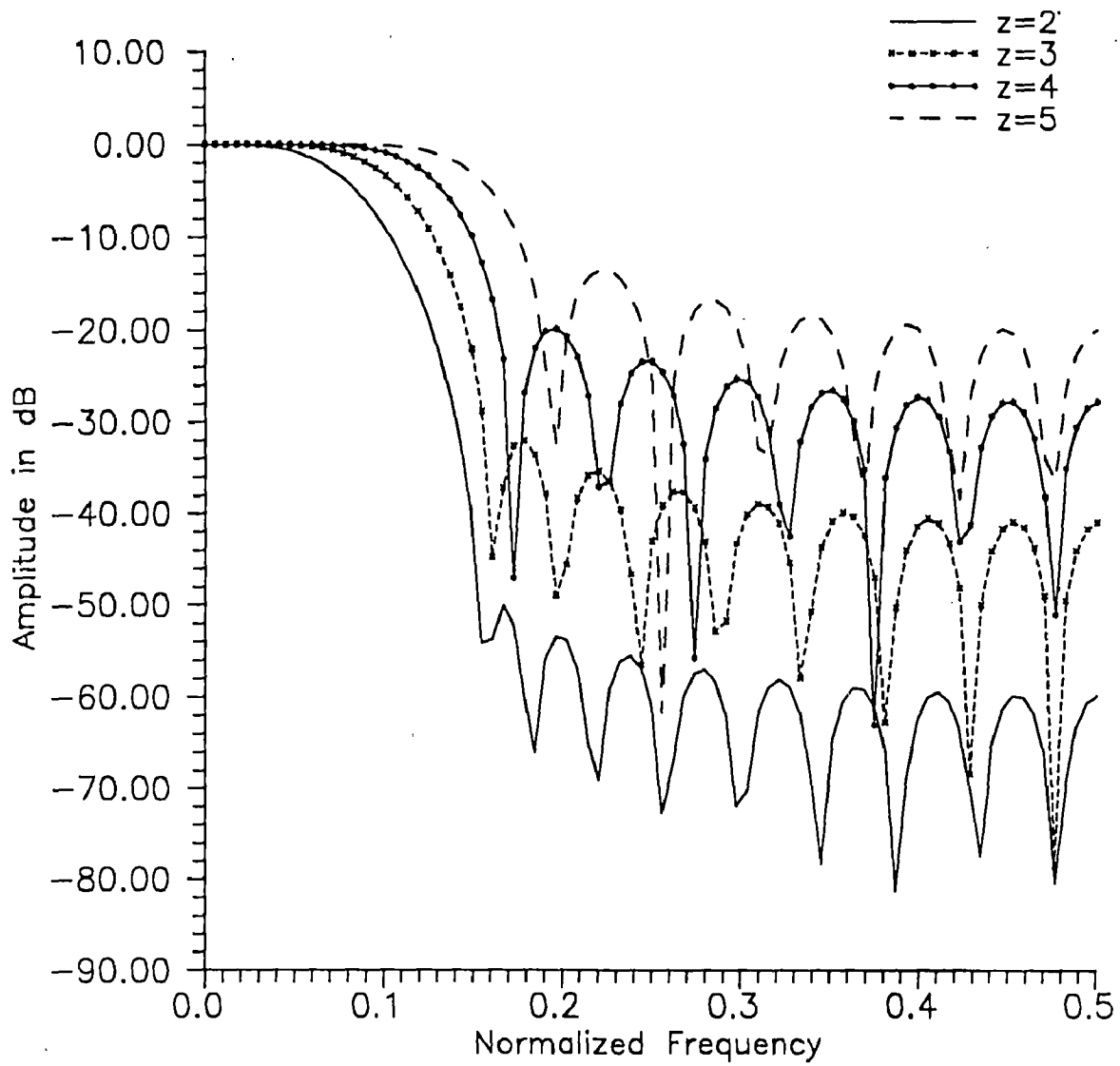


Fig. 6.14 Log amplitude plots for low-pass filters with $N=21$, $w_p=0.15$, $\alpha=1$

Table-6.6 Finite impulse response of lowpass filters
Technique-3 (Designing using maximally flat approximation method)
For N=33, z=3, alpha=0.01

	Filter-13	Filter-14	Filter-15	Filter-16
Passband cutoff frequency 'wp'	0.10	0.15	0.20	0.25
h(0)	1.9566e-2	3.4052e-3	-1.0429e-2	-5.0791e-3
h(1)	-1.6802e-2	-1.7453e-2	1.6528e-2	1.9793e-2
h(2)	-2.4544e-2	8.1769e-3	1.0947e-2	-2.3859e-2
h(3)	-1.0348e-2	2.3451e-2	-1.9618e-2	-2.4164e-3
h(4)	1.2412e-2	6.4041e-3	-1.718e-2	2.7660e-2
h(5)	2.9035e-2	-2.2609e-2	1.8132e-2	-9.7632e-3
h(6)	2.9264e-2	-2.9166e-2	2.6344e-2	-2.8689e-2
h(7)	1.1322e-2	-2.028e-3	-1.2845e-2	2.1522e-2
h(8)	-1.7333e-2	3.3649e-2	-3.9205e-2	2.9221e-2
h(9)	-4.283e-2	3.8740e-2	-1.6914e-3	3.5389e-2
h(10)	-5.0265e-2	-1.2615e-3	5.12e-2	-2.9185e-2
h(11)	-2.9680e-2	-5.5860e-2	2.8035e-2	5.6901e-2
h(12)	1.9601e-2	-6.7647e-2	-6.1880e-2	2.9226e-2
h(13)	8.7906e-2	2.6563e-3	-8.2639e-2	-1.0220e-1
h(14)	1.5789e-1	1.3943e-1	6.9510e-2	-2.9270e-2
h(15)	2.1010e-1	2.7465e-1	3.1053e-1	3.1696e-1
h(16)	2.2940e-1	3.3094e-1	4.2852e-1	5.2915e-1

Note : As the filter is symmetric, $h(i) = h(N-i-1)$

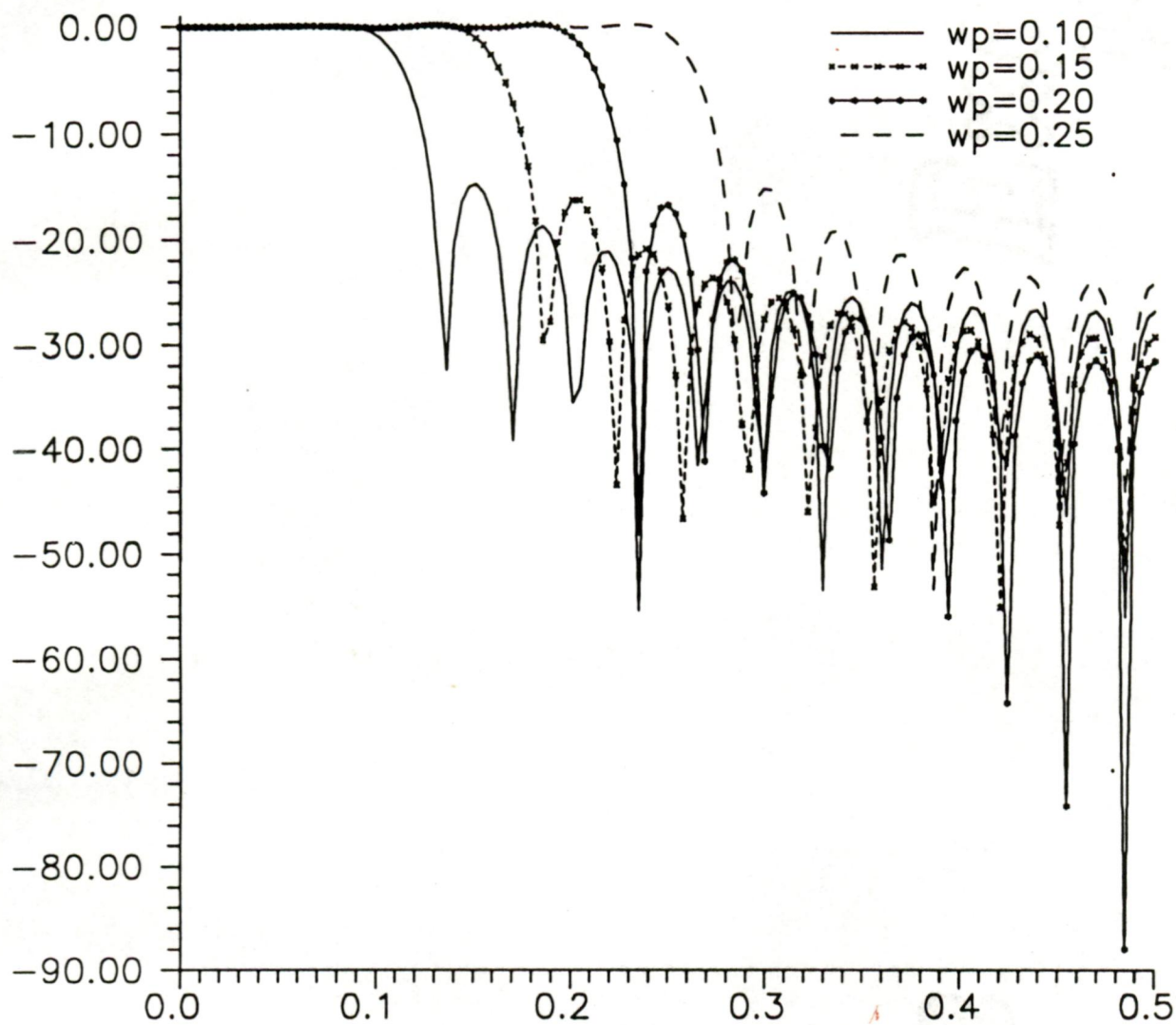


Fig. 6.15 Log amplitude plots for low-pass filters with $N=33$, $z=3$, $\alpha=0.01$

**Table 6.7 Finite impulse response of a lowpass filter
(Designing using maximally flat approximation method)**

For $N=33$, $z=2$, $\alpha=0.5$, $w_p=0.15$

	Filter - 17
h(0)	4.4900e-3
h(1)	1.5133e-2
h(2)	8.3210e-3
h(3)	1.1637e-2
h(4)	-2.8398e-2
h(5)	-2.5753e-2
h(6)	-1.5865e-3
h(7)	2.7711e-2
h(8)	3.7551e-2
h(9)	1.4302e-2
h(10)	-3.0465e-2
h(11)	-6.2548e-2
h(12)	-4.5336e-2
h(13)	3.4478e-2
h(14)	1.5324e-1
h(15)	2.5950e-1
h(16)	3.0201e-1

Note : As the filter is symmetric, $h(i) = h(N-i-1)$

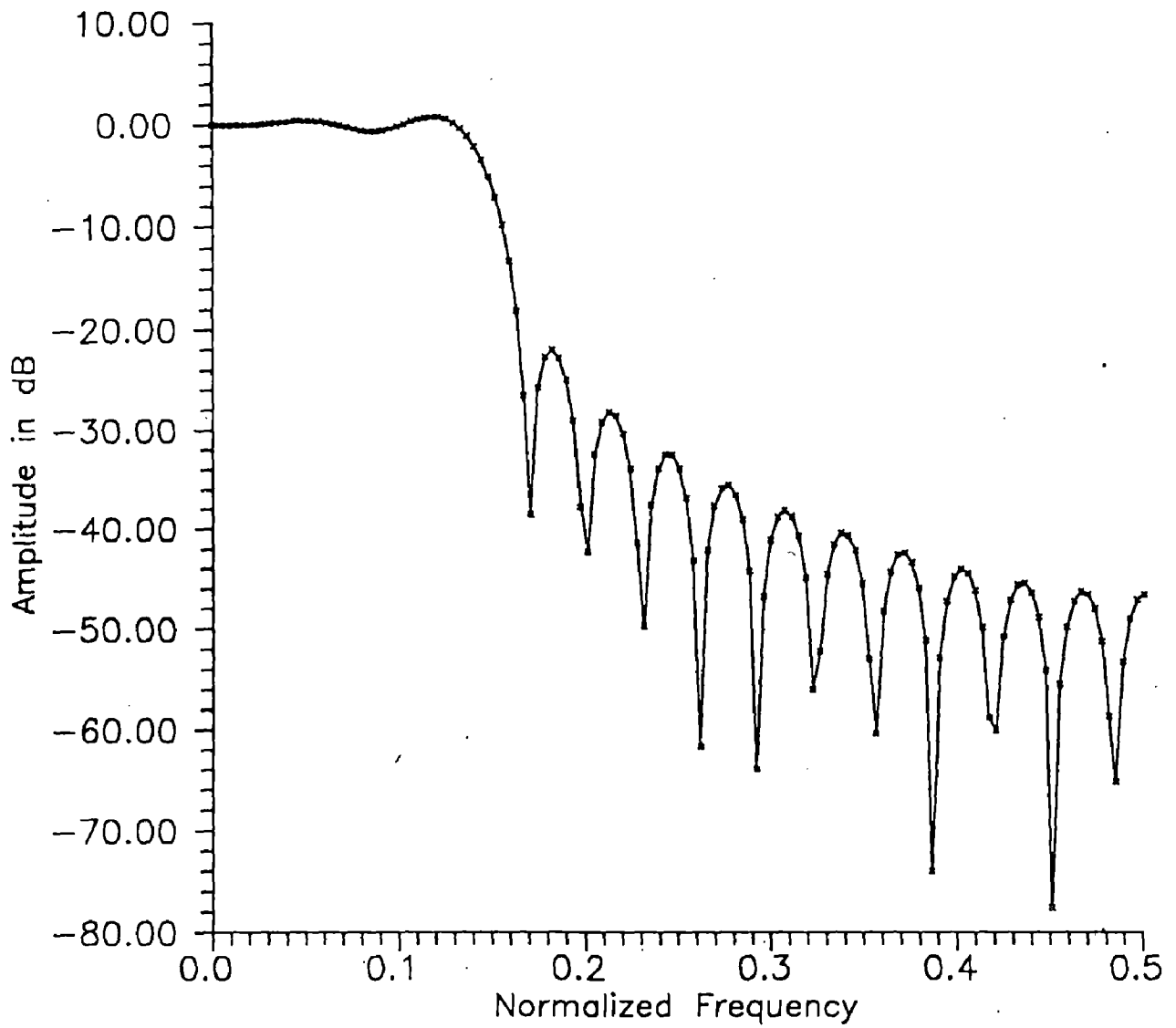


Fig. 6.16 Log amplitude plot for a low-pass filter with $N=33$, $w_p=0.15$, $\alpha=0.5$, $z=2$

6.2 CONCLUSION

In this dissertation, three methods of designing of lowpass digital FIR filter coefficients are reviewed which include recent procedures that lead to efficient implementations. Although many modified algorithms were introduced in latter years to minimize the error between responses of ideal and actual filters but they are not suitable for all classes of filters.

Frequency sampling design method is the simplest and the most straight forward design method. In this method, m_1 (where $m_1 = (N + 1)/2$ for odd N , and $m_1 = N/2$ for even N) samples of a desired frequency response are used to find the N filter coefficients. Graphs (Figs. 6.1 to 6.4) are plotted for passband cutoff frequency = 0.25 and varying filter length.

- (a) For $N = 20$, maximum amplitude of signal in stopband is -15.7dB and 3dB bandwidth = 0.269.
- (b) For $N = 21$, maximum amplitude of signal in stopband is -15.8dB and 3dB bandwidth = 0.256.
- (c) For $N = 30$, maximum amplitude of signal in stopband is -16.2dB and 3dB bandwidth = 0.246.
- (d) For $N = 31$, maximum amplitude of signal in stopband is -16.2dB and 3dB bandwidth = 0.238.

So, as the filter length is increased, the stopband attenuation increases. The passband and stopband performance for $N = 20$ is very close to that of filter with $N = 21$ but the location of passband edge for $N = 20$ is slightly higher as compared to that for $N = 21$. Similarly the passband and stopband frequency response for $N = 30$ is very close to that of filter with $N = 31$ but the band edge for $N = 30$ is slightly higher as compared to that for $N = 31$. In this method, the stopband attenuation increases slightly with increase in filter length. This happens, because rounding the filter coefficients to six places of decimal point does not guarantee that higher

order filters will be better than lower ones. So, lower order filter can be preferred to meet the same specifications.

In the second method, linear programming is used to design FIR lowpass filters which proved quite successful in solving the Chebyshev approximation problem. In this method, number of constraints are formed at dense frequency grid points and the objective is to minimize the maximum absolute weighted error between the desired and the actual filter frequency response.

Graphs (Figs. 6.5 to 6.8) are plotted for passband edge = 0.20, stopband edge = 0.25, weighting factor in passband as well as in the stopband = 1.0 and varying the filter length.

- (a) For $N=20$, maximum amplitude of signal in stopband is $=-18.4\text{dB}$.
- (b) For $N=21$, maximum amplitude of signal in stopband is $=-21.7\text{dB}$.
- (c) For $N=31$, maximum amplitude of signal in stopband is $=-31.8\text{dB}$.
- (d) For $N=40$, maximum amplitude of signal in stopband is $=-36.1\text{dB}$.

So, as the filter length increases the attenuation in stopband increases. The error measure using Cheybshev approximation criterion is less than design using frequency sampling method and error decreases as filter length is increased. The frequency sampling design method can be used directly to design FIR filters. However, it may also be used as a starting point or intermediate stage in a more complicated method as it is a simple and fast method.

Design using linear programming is costly in terms of computer time in designing higher order filters. In fact, computer time increases exponentially w.r.t. the order of filters. Linear programming is useful when there is a frequency response to be met with given tolerance limit using fixed coefficient wordlength.

In third method, design of linear phase lowpass FIR digital filters having symmetric impulse response is formulated as a constrained minimization problem. The constraints express the maximal flatness of the frequency response at the origin. The objective function, which is a quadratic form in the filter coefficients, is formed as a convex combination of mean squared errors over the stopband and passband.

As stopband mean squared error scale factor 'alpha' is increased, amplitude of the unwanted signal decreases. Graph (Fig.6.9) is plotted for filter length $N=33$, number of constraints = 3, passband cutoff frequency = 0.15 and varying scale factor alpha between zero and unity.

- (a) For $\alpha=0.01$, maximum amplitude of signal in stopband is = -16dB
- (b) For $\alpha=0.5$, maximum amplitude of signal in stopband is = -22dB
- (c) For $\alpha=0.8$, maximum amplitude of signal in stopband is = -24dB
- (d) For $\alpha=1$, maximum amplitude of signal in stopband is = -72dB

So, for more attenuation in stopband scale factor alpha can be increased. Maximum value of alpha can be set to unity. As shown in the above stated graph (Fig.6.9), the passband specifications are met for smaller values of alpha.

- (a) For $\alpha=0.01$, 3dB passband width = 0.159
- (b) For $\alpha=0.5$, 3dB passband width = 0.146
- (c) For $\alpha=0.8$, 3dB passband width = 0.143
- (d) For $\alpha=1$, 3dB passband width = 0.076

So, for maximum flatness of passband in frequency response, scale factor alpha is set near to zero.

Hence if more attenuation in stopband is required, scale factor alpha is unity or if maximum flatness in passband is more important then scale factor alpha is set near to zero. This happens, because when scale factor alpha is set to unity, mean squared stopband error is equal to total mean

squared error E , which is the objective function in the constrained minimization problem and is given by $E = \alpha E_s + (1-\alpha)E_p$. So, only stopband mean squared error is minimized and passband mean squared error is not considered in the minimization problem when scale factor alpha is equal to unity.

When scale factor alpha is set near to zero, passband mean squared error is contributing much more as compared to stopband mean squared error in the total mean squared error. So, passband mean squared error, in the objective function of constrained minimization problem is minimized much more as compared to stopband mean squared error.

As the filter length is increased, attenuation in stopband increases. Graphs (Figs.6.10 to 6.13) are plotted for number of constraints=3, passband cutoff frequency=0.15, scale factor alpha =1 and for various lengths of filter.

- (a) For filter length=21, maximum amplitude of signal in stopband is =-32dB
- (b) For filter length=22, maximum amplitude of signal in stopband is =-35dB
- (c) For filter length=33, maximum amplitude of signal in stopband is =-70dB
- (d) For filter length=40, maximum amplitude of signal in stopband is =-95dB

So, more attenuation in the stopband can be achieved by increasing the length of filter and other parameters of filter remains unchanged.

Passband bandwidth specification can be met by increasing the number of constraints at zero frequency. Graph (Fig.6.14) is plotted for filter length = 21, scale factor alpha = 1, passband cutoff frequency = 0.15 and varying the number of constraints.

- (a) Number of constraints = 5, 3dB passband width = 0.15
- (b) Number of constraints = 4, 3dB passband width = 0.123
- (c) Number of constraints = 3, 3dB passband width = 0.1
- (d) Number of constraints = 2, 3dB passband width = 0.077

For more flatness of frequency response throughout the passband, number of constraints at zero frequency can be increased, so as to satisfy passband bandwidth specification. As the number of constraints at zero frequency are increased, the stopband attenuation decreases. From the graph (Fig.6.14) drawn for filter length=21, scale factor $\alpha=1$, passband cutoff frequency=0.15 amplitude of signal in stopband are given below for varying the number of constraints.

- (a) Number of constraints=5, maximum amplitude of signal in stopband is =-14dB.
- (b) Number of constraints=4, maximum amplitude of signal in stopband is =-19dB.
- (c) Number of constraints=3, maximum amplitude of signal in stopband is =-32dB.
- (d) Number of constraints=2, maximum amplitude of signal in stopband is =-50dB.

So, for more attenuation in stopband, the length of filter can be increased or the number of constraints at zero frequency can be reduced. As the length of filter increases or as the number of constraints at zero frequency reduces, the solution space of $Cx=K$ has higher dimension and hence there is more freedom to minimize the objective function, $E=x^T Px$, resulting in an improved stopband frequency response.

Graph (Fig.6.15) is also plotted for filter length=33, scale $\alpha=0.01$, number of constraints=3 and for varying the passband cutoff frequency (normalized to 2π radians).

- (a) For passband cutoff frequency=0.10, 3dB passband width=0.113
- (b) For passband cutoff frequency=0.15, 3dB passband width=0.161
- (c) For passband cutoff frequency=0.20, 3dB passband width=0.210
- (d) For passband cutoff frequency=0.25, 3dB passband width=0.262

So, the passband specifications are approximately met for scale factor $\alpha = 0.01$.

Hence, there is a freedom of trade-off between the flatness of frequency response in the passband and higher attenuation in the stopband.

Graphs are also plotted for $N=33$, passband cutoff frequency $=0.15$, and for various combinations of 'alpha' and 'number of constraints'. It is found that best result is obtained when $\alpha = 0.5$ and number of constraints $=2$. As shown in Fig.6.16, the maximum amplitude of signal in stopband at these values is $= -22\text{dB}$ and the passband width $= 0.145$.

6.3 FUTURE SCOPE

The design of linear phase FIR filters with real coefficients can also be extended to design arbitrary function-both in magnitude and phase. The work can also be extended for the design of complex coefficient finite impulse response filters to attain specified arbitrary multi-band magnitude and linear or arbitrary phase responses.

This work can be extended to design 2-D FIR filters. 1-D Linear phase FIR filters can be transformed to 2-D linear phase FIR filters with the help of frequency transformation. The advantage is that less time is required to design 1-D filters.

REFERENCES

1. L.R. Rabiner and B. Gold, *"Theory and Application of Digital Signal Processing"*, Prentice Hall Limited, 1988.
2. T.W. Parks and C.S. Burrus, *"Digital Filter Design"*, John Wiley & Sons, Inc., 1987.
3. A.V. Oppenheim and R.W. Schaffer, *"Discrete Time Signal Processing"*, Prentice Hall Limited, 1994.
4. David S.K. Chan and L.R. Rabiner, "Analysis of Quantization Errors in the Direct Form for Finite Impulse Response Digital Filters", *IEEE Trans. Electroacoust.*, Vol. AU-21, pp. 354-366, Aug. 1973.
5. J.H. McClellan, T.W. Parks, L.R. Rabiner, "A Computer Program for Designing Optimum FIR Linear Phase Digital Filters", *IEEE Trans. Audio Electroacoust.*, Vol. AU-21, pp. 506-526, Dec. 1973.
6. B.A. Bowen, W.R., Brown, "VLSI System Design for Digital Signal Processing, Vol. I : Signal Processing and Signal Processers", Prentice Hall, Inc., pp. 95-96, 1982.
7. M.T. Hanna, "Design of Linear Phase FIR Filters with a Maximally Flat Passband", *IEEE Trans. Circuits and Systems - II : Analog and Digital Signal Processing*; Vol. 43, No. 2, pp. 142-147, Feb. 1996.
8. I.W. Selesnick, C.S. Burrus, "Exchange Algorithms for the Design of Linear Phase FIR Filters and Differentiators Having Flat Monotonic Passbands and Equiripple Stopbands", *IEEE Trans. Circuits and Systems-II : Analog and Digital Signal Processing* Vol. 43, No. 9, pp. 671⁻⁶⁷⁵, Sept. 1996.

9. O. Herrmann, "Design of Nonrecursive Digital Filters with Linear Phase", Electronics Letters Vol. 6, No. 11, pp. 328-329, May 1970.
10. Eli Goldberg, R. Kurshan, "Design of Finite Impulse Response Digital Filters with Nonlinear Phase Response", IEEE Trans. Acoustics Speech and Signal Processing Vol. ASSP-29, No. 5, pp. 1003⁻¹⁰¹⁰, Oct. 1981.
11. G. Cortelazzo, M.R. Lightner, "Simultaneous Design in Both Magnitude and Group-Delay IIR and FIR Filters Based on Multiple Criterion Optimization", IEEE Trans. Acoustics Speech Signal Processing, Vol. ASSP-32, No. 5, pp. 949⁻⁹⁶⁶, Oct. 1984.
12. K. Preuss, "On the Design of FIR Filters by Complex Chebyshev Approximation", IEEE Trans. Acoustic Speech and Signal Processing, Vol. 37, No. 5, pp. 702⁻⁷¹², May 1989.
13. S.C. Pei and J.J. Shyu, "Design of real FIR Filters with arbitrary complex frequency responses by two real Chebyshev approximations", Signal Processing, Vol. 26, No. 1, pp. 119⁻¹²⁹, Jan. 1992.
14. L.R. Rajagopal and S.C. Dutta Roy, "Optimal Design of Maximally Flat FIR Filters with Arbitrary Magnitude Specifications", IEEE Trans. Acoustics Speech Signal Processing, Vol. 37, No. 4, pp. 512⁻⁵¹⁸, Apr. 1989.
15. D.M. Kodek, "Design of Optimal Finite Wordlength FIR Digital Filters Using Integer Programming Techniques", IEEE Trans. Acoustics Speech Signal Processing, Vol. ASSP-28, No. 3, pp. 304-308, Jun. 1980.
16. H.C. Chiang, J. C. Liu, "Fast Algorithm for FIR Filtering in the Transform Domain", IEEE Trans. Signal Processing, Vol. 44, No. 1, pp. 126⁻¹²⁹, Jan. 1996.
17. G.W. Medlin, J.W. Adams, C.T. Leondes, "Lagrange Multiplier Approach to the Design of FIR Filters for Multirate Applications", IEEE Trans. Circuits and Systems, Vol. 35, No. 10, pp. 1210-1219, Oct. 1988.

18. K. Swarup, P.K. Gupta, Man Mohan, "Operation Research", Educational Publishers, pp. 311-312, 1994.
19. T.W. Parks, J.H. McClellan, "Chebyshev approximation for Nonrecursive Digital Filters with Linear Phase", IEEE Trans. Circuit Theory, Vol. CT-19, No. 2, pp. 189⁻¹⁹⁴, Mar. 1972.
20. A.S. Alkhairy, K.G. Christian, J.S. Lim, "Design and Characterization of Optimal FIR Filters with Arbitrary Phase", IEEE Trans. Signal Processing, Vol. 41, No. 2, pp. 559⁻⁵⁷², Feb. 1993.
21. M. Schulist, "Improvements of a Complex FIR Filter Design Algorithm", Signal Processing, Vol. 20, No. 1, pp. 81⁻⁹⁰, May 1990.

APPENDIX

Software listing

```

/* *****/
/* Program to obtain the filter coefficients */
/* as well as frequency response using */
/* frequency sampling design technique */
/* *****/

/*      Input Parameters      */
/* Filter length 'n', Passband edge 'fp', */
/* Number of output frequency samples 'k' */

#include <stdio.h>
#include <math.h>
#define PI2 6.28318530717959
void fresp(float *,double *,int , int );

int main()
{
    int n,dc,k,m,m1,i,n2,y,np,j,z;
    float fp,am,f;
    double q,xt;
    float h[101],dis[101];
    double ab[1000];
    FILE *ifp = stdin;
    FILE *ofp = stdout;
    char *in;
    char *out;
    in = "in1.dat";
    out = "out1.dat";

    if((ifp = fopen(in,"r")) == NULL)
    {
        fprintf(stderr,"inpo.c : could not open %s\n",in);
        return 1;
    }
    if((ofp = fopen(out,"w")) == NULL)
    {
        fprintf(stderr,"inpo.c : couldn't open %s\n",out);
        fclose(ifp);
        return 1;
    }

    for(z = 0; z < 2; z++)
    {
        printf("If you want to enter new data yes = 1,no = 0\n");
        scanf("%d",&y);
        printf("\nFreq. sampling design of a lowpass filter\n");
        if(y == 1)
        {
            printf("Enter :N,FP,DC,K : ");
            scanf("%d %f %d %d",&n,&fp,&dc,&k);
            fseek(ifp,12,SEEK_SET);
        }
        else
            fscanf(ifp,"%d %f %d %d",&n,&fp,&dc,&k);
        printf("%d,%f,%d,%d.\n",n,fp,dc,k);
        m = (n-1)/2;
        am = ((float)n - 1.0 )/2.0;printf("am = %f\n",am);
    }
}

```

```

m1 = n/2 + 1;
q = PI/2/n;
n2 = n/2;
if(dc != 0)
    np = n*fp + 0.5;
else
    np = n*fp + 1;

/* Set desired frequency response array dis[] */
for(j = 0; j < np; j++)
    dis[j] = 1.0;
for(j = np; j < m1; j++)
    dis[j] = 0.0;

/* Finding the filter coefficients */
if(dc == 0)
{
    for(j = 0; j <= m; j++)
    {
        xt = dis[0]/2.0;
        for(i = 1; i <= m; i++)
            xt = xt + dis[i]*cos(q*(am-j)*i);
        h[j] = 2.0*xt/n;
    }
}
else
{
    for(j = 0; j <= m; j++)
    {
        xt = 0;
        for(i = 0; i < n2; i++)
            xt = xt + dis[i]*cos(q*(am-j)*(i+0.5));
        if(am == m)
            xt = xt + dis[m]*cos(3.141592654*(am-j)/2);
        h[j] = 2*xt/n;
    }
}

/* Output frequency response of designed filter */
for(j = 0; j <= m; j++)
{
    if(j%5 == 0)
        printf("\n");
    printf("%f\t", h[j]);
}
printf("\n");
fresp(h, ab, n, k);
for(j = 0; j < k+1; j++)
{
    if(j%1 == 0)
        fprintf(ofp, "\n");
    f = 0.5*j/k;
    fprintf(ofp, "%f %e %.2 e", f, fabs(ab[j]), 20*(log10(fabs(ab[j]))));
}
fprintf(ofp, "\n");
} /* end of for loop */

```

```

if(ferror(ifp))
{
    fprintf(stderr,"inpo.c:error in reading input \n");
    return 1;
}
fclose(ifp);
fclose(ofp);
return 0;
} /* end of main */

/* Function to calculate frequency response */
void fresp (float *h,double *ab,int n,int k)
{
    float am = ((float)n -1.0)/2.0;
    int m = (n-1)/2;
    int n2 = n/2;
    double q = PI2/(2.0*k);
    double at;
    int i,j;
    for(j = 0;j<k+1;j++)
    {
        at =0;
        if(am == m)
            at = 0.5 *h[m];
        for(i =0;i<n2;i++)
        {
            at = at+h[i]*cos(q*(am-i)*j);
        }
        ab[j] =2*at;
    }
} /* end of function */

```



```

/* *****/
/* Program to obtain the constraints used for minimizing */
/* the error in Chebyshev Approximation method*/
/* *****/

/*      Input      Parameters      */
/* Filter length 'n', Number of bands(ie. passband and */
/* stopband in frequency response) 'nbnd', */
/* Grid length 'lgrd'( it determines the number of */
/* frequency points at which constraints are to be */
/* formed),Band edges(of pass and stop bands)'edg'(array)*/
/* Desired response in different bands are saved in */
/* array 'fx', Weighting factor for pass and stop bands */
/* are saved in array 'wt', Number of bits used to */
/* represent the filter coefficients 'b' */

#include <stdio.h>
#include <math.h>
#define PI 3.1415926
#define PI2 6.283185
#define SPACE " "
#define NMAX 80
#define ED 50
#define RES 1000
float eff1(float ,float *,int ,int );
float wt1(float ,float * ,float * ,int ,int );
void error();

int main()
{
    int n,ltyp,nbnd,lgrd,lb,i,j,b,ng,nd,nfns,lbnd,nf,m;
    int no,mm,nn,k[ED],l[RES],i1,j1,ngrd;
    float edg[ED],fx[ED],wt[ED],grd[RES],dis[RES],wat[RES];
    float gf[RES][ED],dlf,fp,tmp,y[RES][ED],gsum;
    FILE *in = stdin;
    FILE *out = stdout,*ot2 = stdout;
    char *in1 = "rx2.dat",*ot1 = "o3.dat";
    char *out1 = "out2.dat";
    if((in = fopen(in1,"r")) == NULL)
    {
        printf("main2:couldn't open %s\n",in1);
        return 1;
    }
    if((out = fopen(out1,"w")) == NULL)
    {
        printf("main2:couldn't open %s\n",out1);
        fclose(in);
        return 1;
    }
    if((ot2 = fopen(ot1,"w")) == NULL)
    {
        printf("main2:couldn't open %s\n",ot1);
        fclose(in);fclose(out);fclose(ot2);
        return 1;
    }
    /* Input parameters are defined */
    fscanf(in,"%d %d %d %d",&n,&ltyp,&nbnd,&lgrd);
    printf("%d,%d,%d,%d",n,ltyp,nbnd,lgrd);

```

```

/* Check whether the input parameters are in desired limits */
if((n>NMAX) || n<3)
{
    error();
    return 1;
}
if(nbnd<=0) nbnd = 1;
if(lgrd ==0) lgrd = 10;
lb = 2*nbnd;
for(i = 0;i<lb;i++)
fscanf(in,"%f",&edg[i]);
for(i = 0; i<nbnd; i++)
fscanf(in,"%f",&fx[i]);
for(i = 0;i<nbnd; i++)
fscanf(in,"%f",&wt[i]);
fscanf(in,"%d",&b);
if(ltyp ==0)
{
    error();
    return 1;
}

/* Calculate grid points(ie. frequency points
/* at which constraints are to be formed),
/* weight function & desired values arrays
/* at grid points
ng = 1;
if(ltyp ==1) ng = 0;
nd = n/2;
nd =n-2*nd; /* set for n odd reset for n even */
nfns =n/2; /* nfns = N/2 for N even */
if(nd == 1 && ng == 0)
nfns = nfns+1; /* nfns = (N+1)/2 for N odd */
grd[0] = edg[0];
dlf = (float)lgrd*nfns;

/* To set up spacing between two
/* adjacent frequency points at which
/* constraints are to be calculated
dlf = .5/dlf;printf("\t dlf = %f",dlf);
if((ng!=0) && (edg[0]<dlf)) grd[0] = dlf;
for(j = 0,i =0,lbnd =0; lbnd<nbnd ; )
{
    fp = edg[j+1];
    /* printf("fp = %f",fp); */
    printf ("\n desired values in band %d \n",lbnd);
    do{
        if(i%4 ==0) printf("\n");
        tmp = grd[i];
        dis[i] = eff1(tmp,fx,lbnd,ltyp);
        wat[i] = wt1(tmp,fx,wt,lbnd,ltyp);
        printf("%d %-6.2f : %-6.2f/%f %1s",i,dis[i],wat[i],grd[i],SPACE);
        grd[+ +i]=tmp+dlf;
    }while(grd[i] <= fp);
    printf("\n");printf("fp = %f",fp);printf("grd[i] = %f",grd[i]);
    grd[--i] = fp;
    dis[i] =eff1(grd[i],fx,lbnd,ltyp);
    wat[i] = wt1(grd[i],fx,wt,lbnd,ltyp);
    + +lbnd;
}

```

```

    j += 2;
    grd[ ++i ] = edg[j];
}
ngrd = i; printf("\ni = %d", i);
if( (ng == nd) && (grd[ngrd-1] > (.5-dlf) ) )
ngrd = ngrd-1;
printf("\nno. of grid pts = %d\t", ngrd);
if(nd == 1) nf = (n-1)/2;
else nf = n/2;
for( i=0; i <= ngrd; i++ )
{
    if( i%3 == 0 ) printf("\n");
    printf("%-6.2f: %-6.2f/%f%1s", dis[i], wat[i], grd[i], SPACE);
} printf("\n");

/* Forming the constraints */
if( ng <= 0 )
{
    for( i = 0; i < ngrd; i++ )
    {
        printf("i is = %d", i);
        for( j1 = 0, j = 1; j <= nf; j1++, j++ )
        {
            if( j%5 == 0 )
                printf("\n");
            if( nd == 1 )
                gf[i][j1] = 2*cos(PI2*(float)j*grd[i]);
            else
                gf[i][j1] = 2*cos(PI2*((float)j-.5)*grd[i]);
            printf(" %f\t", gf[i][j1]);
        }
        printf("\n");
    }
}
else
{
    for( i = 0; i < ngrd; i++ )
    {
        printf("i is = %d", i);
        for( j1 = 0, j = 1; j <= nf; j++ )
        {
            if( j%5 == 0 ) printf("\n");
            if( nd == 1 )
                gf[i][j1] = 2*sin(PI2*(float)j*grd[i]);
            else
                gf[i][j1] = 2*sin(PI2*((float)j-.5)*grd[i]);
            printf(" %f\t", gf[i][j1]);
        }
        printf("\n");
    }
}

m = 2*ngrd + 1;
no = nf + 3;
fprintf(out, "%d %d %d %d %d %d", n, ng, nd, b, ltyp, nf);
fprintf(out, " %d %d", m, no);
mm = m-1;
nn = no-1;
for( j = 0; j < nn; j++ )
k[j] = j;
for( i = 0; i < mm; i++ )

```

```

l[i] = nn + i;
l[mm] = 1000;
printf("k");

for(j = 0; j < nn; j++)
{
    if (j%12 == 0) fprintf(out, "\n");
    fprintf(out, "%3d", k[j]);
}
printf("\n");
for(i = 0; i < m; i++)
{
    if(i%12 == 0) fprintf(out, "\n");
    fprintf(out, "%5d ", l[i]);
}
printf("\n");

/* Form the table for applying Linear */
/* Programming using constraints */
/* saved in matrix gf[][] */
for(i = 0; i < mm; i++)
{
    if(ltyp == 1 && nd == 1)
        if(i < ngrd) y[i][0] = 1;
        else y[i][0] = -1;
    else y[i][0] = 0;
}
for(i1 = 0, i = 0; i1 < ngrd; i1++, i++)
{
    gsum = 0;
    for(j1 = 1, j = 0; j1 < no; j1++)
    {
        gsum = gsum + gf[i][j];
        y[i1][j1] = gf[i][j];
        j++;
        if(j >= nf) break;
    }
    y[i1][nn-1] = -pow(2.0, (b-1.0))/wat[i];
    if(nd == 1)
        y[i1][no-1] = pow(2.0, (b-1.0)) * (dis[i] + 1 + gsum);
    else
        y[i1][no-1] = pow(2.0, (b-1.0)) * (dis[i] + gsum);
}
for(i = 0, i1 = ngrd; i1 < mm; i++, i1++)
{
    gsum = 0;

    for(j = 0, j1 = 1; j1 < no; j1++)
    {
        y[i1][j1] = -gf[i][j];
        gsum = gsum + gf[i][j];
        j++;
        if(j >= nf) break;
    }
    y[i1][nn-1] = -pow(2.0, (b-1.0))/wat[i];
    if(nd == 1)
        y[i1][no-1] = -pow(2.0, (b-1.0)) * (dis[i] + 1 + gsum);
    else
        y[i1][no-1] = -pow(2.0, (b-1.0)) * (dis[i] + gsum);
}

```

```

for(j =0; j<no; j+ +)
y[mm][j] =0;
y[mm][nn-1] =-1;

for(i =0;i<m ; i+ +)
{
for(j = 0; j< no ; j+ +)
{
fprintf(ot2,"%f ",y[i][j]);
}
fprintf(ot2,"\n");
}
if(ferror(in))
{
printf("\nmain2:error in reading input\n");
return 1;
}
fclose(in);fclose(out);fclose(ot2);
return 0;
} /* end of main */

/* function to evaluate desired values at grid points */
float eff1(float tmp,float *fx,int lbnd,int ltyp)
{
if(ltyp == 2)
return fx[lbnd]*tmp;
else
return fx[lbnd];
}

/* function to evaluate weight values at grid points */
float wt1(float tmp,float *fx,float *wt,int lbnd,int ltyp)
{
if(ltyp == 2)
if(fx[lbnd]<.0001) return wt[lbnd];
else
return wt[lbnd]/tmp;
else
return wt[lbnd];
}

/* function to inform about errors */
void error()
{
printf("\n Error in input data file\n");
}

```

```

/* *****/
/* Program to minimize the error E in Chebyshev*/
/* approximation criterion using Linear */
/* Programming and the constraints formed by*/
/* previous program */
/* *****/

#include <stdio.h>
#include <math.h>
#define C 100
#define RES 1000
#define ED 100
#define CXL 4
#define HUN 100

int dsrow(float y[][C],int,int,int *);
int dscol(float y[][C],int,int,int,float *,int *);
int scol(float y[][C],int,int,int *);
int srow(float y[][C],int,int,int,float *);
void pctab(float y[][C],int *,int *,int,int,int,int);
void dspp1(float y[][C],int *,int,int,int *,int *,float *,int *);
void spp2(float y[][C],int *,int,int,int *,int *,float *);
void malpp(float y[][C],int,int,int *,int *,FILE *);
void arrg(int,int,int,int,float y[][C],int *,float *,float,FILE *);
void outrq(float *,double *,int,FILE *);
void frequy(float *,double *,int,int);

int main()
{
float err,y[RES][ED],x[1][ED],w[RES],xi[RES][CXL],h[HUN];
double ab[1001];
int n,ng,nd,b,ltyp,nf,m,int1,no,mm,nn,i,j,nn1,m1;
int k[ED],l[RES];
FILE *ifp1,*ifp2,*ofp1,*ofp2;
char *in1,*in2,*out1,*out2;
in1 = "out2.dat";
in2 = "o3.dat";
out1 = "wx5.dat";
out2 = "wx7.dat";
if((ifp1 = fopen(in1,"r")) == NULL)
{
printf("smp.c:couldn't open %s\n",in1);
return 1;
}
if((ifp2 = fopen(in2,"r")) == NULL)
{
printf("smp.c:couldn't open %s\n",in2);
fclose(ifp1);return 1;
}
if((ofp1 = fopen(out1,"w")) == NULL)
{
printf("smp.c :couldn't open %s\n",out1);
fclose(ifp1);fclose(ifp2);
return 1;
}
}

```

```

if((ofp2 = fopen(out2,"w")) == NULL)
{
printf("smp.c:couldn't open %s\n",out2);
fclose(ifp1);fclose(ifp2);fclose(ofp1);
return 1;
}
fscanf(ifp1,"%d %d %d %d %d %d %d %d",&n,&ng,&nd,&b,&ltyp,&nf,&m,&no);
mm = m-1;
nn = no-1;
printf("\nn = %d,ng = %d,nd = %d,b = %d,ltyp = %d,nf
= %d,mm = %d,nn = %d",n,ng,nd,b,ltyp,nf,mm,nn);
for(j=0;j<nn;j++)
{
fscanf(ifp1,"%3d",&k[j]);
}
printf("\n");
for(i=0;i<m;i++)
{
fscanf(ifp1,"%5d",&l[i]);
}
for(i=0;i<m;i++)
{
for(j=0;j<no;j++)
fscanf(ifp2,"%f",&y[i][j]);
}
malpp(y,m,no,l,k,ofp1);
printf("\n Problem is optimized");
arrg(mm,nn,nd,b,y,l,h,err,ofp1);
printf("\n m nn = %d, m nf = %d",nn,nf);
outfrq(h,ab,n,ofp2);
if(ferror(ifp1))
{
printf("smp.c:error in reading input 1\n");
return 1;
}
if(ferror(ifp2))
{
printf("smp.c:error in reading input2\n");
return 1;
}
fclose(ifp1);fclose(ifp2);fclose(ofp1);fclose(ofp2);
return 0;
} /* end of main */

/* function for minimizing the objective */
/* value using Linear programming */
void malpp(float y[][C],int m,int no,int *l,int *k,FILE *ofp)
{
int i,j,ipd,ipq,ip,iq ;
int w[RES];int v[C];int z[C];
int mm = m-1;int nn = no-1;
float dp,ddq;
ip = -2;
iq = -2;
dp = -2.0;
ddq = -2.0;
printf("\n ml nn = %d",nn);

```

```

if(l[mm] != 1000)
{
for(j =0; j<nn; j++)
y[mm][j] = -y[mm][j];
l[mm] = 1000;
}
for(i =0;i<mm;i++)
w[i] = 0;

for(j =0; j<no; j++)
v[j] = 0;

for(j =0; j<no; j++)
z[j] = 0;

for( ; ; )
{
dsp1(y,w,m,no,&ip,&iq,&ddq,z);
ipd = ip;
ipq = iq;
spp2(y,v,m,no,&ip,&iq,&dp);
if(iq == -1)
{
ip = ipd;
iq = ipq;
}
if(ip != -1)
{
printf("\n ddq = %f,dp = %f",ddq,dp);
if(ddq > dp)
{
ip = ipd;
iq = ipq;
printf("\t ddq ip = %d,iq = %d\n",ip,iq);
}
printf("\t ip = %d,iq = %d\n",ip,iq);
pctab(y,k,l,m,no,ip,iq);
}
else break;
}
for(i =0; i<m; i++)
w[i] =0;
for(j =0; j<no; j++)
v[j] =0;

for(i =0; i<m; i++)
for(j =0; j<no; j++)
{
if(fabs(y[i][j]-(int)y[i][j]) <= 1e-4)
y[i][j] = (float)((int)y[i][j]);
}
printf("\n Optimum Sol.");
}

```



```

/* function for finding pivoted row and      */
/* column using dual simplex method      */
void dspp1(float y[][C],int *w,int m,int no,int *ip1,int *iq,float
*ddq,int *z)
{
int ip = -1;
int nn = no-1;
int i;
int mm = m-1;
float ds;

for( ; ; )
{
*ip1 = dsrow(y,m,no,w);
ip = *ip1;
if(ip == -1)
{
*ddq = -1;
*iq = -1;
break;
}
else
{
*iq = dscol(y,m,no,ip,&ds,z);
if((*iq) != -1)
{
*ddq = fabs(y[ip][nn] * ds);
break;
}
else w[ip] = 1;
}
}
}

```

```

/* function for finding pivoted row and      */
/* column using simplex method      */
void spp2(float y[][C],int *v,int m,int no,int *ip,int *iq1,float *dp)
{
int iq,j;
float s;
int mm = m-1;

for( ; ; )
{
*iq1 = scol(y,m,no,v);
iq = *iq1;
if((*iq1) == -1) break;
*ip = srow(y,m,no,iq,&s);
if((*ip) != -1)
{
*dp = fabs(y[mm][iq]*s);
break;
}
else v[iq] = 1;
}
}

```

```

/* function for finding pivoted row      */
/* using dual simplex method           */

```

```

int dsrow(float y[][C],int m,int no,int *w)
{
    float b = 0;
    int i,mm,nn;
    int ip = -1;
    mm = m-1;
    nn = no-1;

    for(i = 0; i < mm ; i++)
    {
        if(((y[i][nn] + .1e-5) < 0) && (w[i] <= 0))
        {
            if((y[i][nn]-b) < 0)
            {
                b = y[i][nn];
                ip = i;
            }
        }
    }
    return ip;
}

```

```

/* function for finding pivoted column  */
/* using dual simplex method           */

```

```

int dscol(float y[][C],int m,int no, int ip,float *ds1,int *z)
{
    float r,ds;
    int j,mm,nn,iq = -1;
    *ds1 = 1.e19;
    nn = no-1;
    mm = m-1;
    ds = *ds1;

    for(j = 0; j < nn ; j++)
    {
        if(((y[ip][j] + .1e-5) < 0) && ((y[mm][j] + .1e-5) >= 0))
        {
            if(z[j] <= 0)
            {
                r = -y[mm][j]/y[ip][j];
                if((ds-r) > 0) /* select min ratio */
                {
                    *ds1 = r; /* ds = -y[mm][iq]/y[ip][iq] */
                    ds = *ds1;
                    iq = j;
                }
            }
        }
    }
    z[iq] = 1;
    return iq;
}

```

```

/* function for finding pivoted */
/* column using simplex method */
int scol(float y[][C], int m, int no, int *v)
{
    float b = 0;
    int j;
    int nn = no-1, mm = m-1, iq = -1;
    for(j = 0 ; j < nn; j++)
    {
        if((v[j] <= 0) && ((y[mm][j] + .1e-5) < 0))
        {
            if((y[mm][j]-b) < 0)
            {
                b = y[mm][j];
                iq = j;
            }
        }
    }
    return iq;
}

```

```

/* function for finding pivoted */
/* row using simplex method */
int srow(float y[][C], int m, int no, int iq, float *s1)
{
    int ip = -1, i, mm, nn;
    float r, s;
    *s1 = 1.e19;
    mm = m-1;
    nn = no-1;
    s = *s1;
    for(i = 0; i < mm; i++)
    {
        if(((y[i][iq]-.1e-5) > 0) && ((y[i][nn] + .1e-5) >= 0))
        {
            r = y[i][nn]/y[i][iq];
            if((s-r) > 0) /* select min. ratio */
            {
                *s1 = r;
                s = *s1;
                ip = i;
            }
        }
    }
    return ip;
}

```

```

/* function for arranging the */
/* filter coefficients in order */
void arrg(int mm, int nn, int nd, int b, float y[][C], int *l, float *h, float
err, FILE *ofp)
{
    int nn1 = nn-1, j1, i, j2, mm1 = mm-1, nf = nn-2;
    float h1[HUN];

    printf("\na nf = %d, a nn = %d", nf, nn);
}

```

```

for(j1 = 0; j1 < nn1; j1++)
{
for(i = 0; i < mm; i++)
{
j2 = nf - j1;
if(l[i] == j1)
{
h1[j2] = y[i][nn];
h1[j2] = h1[j2] - pow(2.0, (b-1));
h[j2] = h1[j2] / pow(2.0, (b-1));
if(h1[j2] < 0)
h1[j2] = h1[j2] - 0.5;
else
h1[j2] = h1[j2] + 0.5;
fprintf(ofp, "\nh1[%d] = %d\n", j2, (int)h1[j2]);
break;
}
else
{ h[j2] = 0; h1[j2] = 0; }
}
}
for(i = 0; i < mm; i++)
if(l[i] == nn1)
err = y[i][nn];
for(j2 = 0; j2 < nf; j2++)
fprintf(ofp, "h[%d] = %f\n", j2, h[j2]);
if(nd == 1)
fprintf(ofp, "h[%d] = %f\n", nf, h[nf]);
}

/* Write output frequency response */

void outfrq(float *h, double *ab, int n, FILE *ofp)
{
float f;
int k = 4 * n, j;
frequy(h, ab, n, k);
for(j = 0; j < k + 1; j++)
{
f = .5 * j / k;
fprintf(ofp, "%f %e %.2 e\n", f, fabs(ab[j]), 20 * (log10(fabs(ab[j]))));
}
}

/* function for calculating the */
/* frequency response */

void frequy(float *h, double *ab, int n, int k)
{
float am = ((float)n - 1.0) / 2.0;
int m = (n - 1) / 2, i, j, n2 = n / 2;
double q = 6.28318530717959 / (2.0 * k), at = 0;
for(j = 0; j < k + 1; j++)
{
at = 0;
if(am == m) at = 0.5 * h[m];
for(i = 0; i < n2; i++)

```

```

    at = at+h[i]*cos(q*(am-i)*j);
    ab[j] = 2*at;
}
}

/* function for forming the pivoted      */
/* condensed table                        */
void pctab(float y[][C],int *k,int *l,int m,int no,int ip,int iq)
{
    float r = 1/y[ip][iq];
    int x,i,j;
    for(j =0; j<no ; j++)
        y[ip][j] = r*y[ip][j];
    y[ip][iq] = r;
    for(i =0; i<m; i++)
    {
        if((i-ip)!=0)
        {
            for(j =0;j<no;j++)
                if((j-iq)!=0)
                    y[i][j] = y[i][j]-y[i][iq]*y[ip][j];
            y[i][iq] = -r*y[i][iq];
        }
    }
    x = k[iq];
    k[iq] = l[ip];
    l[ip] = x;
}

```

```

% *****
% Program to obtain frequency response of FIR lowpass odd
% length filter using maximally flat approximation method
% *****

```

```

% Input Parameters
% N : length of filter
% z : degree of flatness at  $\omega = 0$ 
% wp : passband cutoff frequency
% ws : stopband edge
% alpha : scale factor of stopband error
% % Example
% N = 33
% z = 3
% wp = 0.15
% alpha = 1
% ws = wp

```

```

N=input('enter the value of N (odd) =');
z=input('enter the value of z =');
wp=input('enter the value of wp lies between 0 & 0.5 =');
wp=wp./0.5 .*pi;
alpha=input('enter the value of alpha (0<alpha<=1) =');
ws=wp;

```

```

if rem(N,2) == 0 % check whether filter length is odd
    disp('N must be odd')
else
    m = (N+1)./2;
    L = 4.*N;
    n_2 = m-1;
    X_1 = x(N,z,wp,ws,alpha);
    h(1:m) = X_1(m:-1:1)./2;
    h(m) = 2.*h(m); % filter coefficients are obtained
    q = pi ./L;
    for j = 0 : L
        at = 0.5.*h(m);
        for i = 1:n_2
            at = at+h(i).*(cos(q.*(m-i).*j));
        end % for loop i
        ab(j+1) = 2 .*at;
    end % for loop j
    f1 = 0 :(0.5./L) : 0.5 ;
    lab(1:L+1) = 20.*(log10(abs(ab(1:L+1))));
    ml1 = [f1' lab'];
    plot(f1,20.*(log10(abs(ab))), 'w-')
    % frequency response of designed filter is obtained
end %end of if

```

% function to calculate the error matrix P

function y = p(ws,wp,alpha,N)

m = (N+1) ./2;

for i=1:m

for j=1:m

if (i == 1 & j == 1)

temp1(i,j) = pi-ws;

elseif (i == j & i ~ = 1 & j ~ = 1)

temp1(i,j) = (pi ./2)-(ws ./2)-(sin(2 .* (i-1) .* ws)/(4 .* (i-1)));

else

temp1(i,j) = -0.5 .* ((sin((i+j-2) .* ws)/(i+j-2)) + (sin((i-j) .* ws)/(i-j)));

% Matrix P_s is obtained

end %endif

end %end for j

end %end for i

for i=1:m

for j=1:m

if (i == 1 | j == 1)

temp2(i,j) = 0;

elseif (i == j & i ~ = 1 & j ~ = 1),

temp2(i,j) = ((wp ./2) .* 3) + (sin(2 .* (i-1) .* wp)/(4 .* (i-1)))-(2 .* sin((i-1) .* wp)/(i-1));

else

t_1 = (sin((i+j-2) .* wp)/(2 .* (i+j-2)));

t_2 = (sin((i-j) .* wp)/(2 .* (i-j)));

t_3 = (sin((i-1) .* wp) ./ (i-1));

t_4 = (sin((j-1) .* wp) ./ (j-1));

temp2(i,j) = wp + t_1 + t_2 - t_3 - t_4;

% Matrix P_p is obtained

end %endif

end %end for j

end %end for i

y = (alpha*temp1) + ((1-alpha)*temp2); % Matrix P is obtained

% function to calculate the constraint matrix C

```
function y = c(z,N)
    m = (N+1)./2;
    for i = 1:z
        for j = 1:m
            if (i == 1)
                y(i,j) = 1;
            else
                y(i,j) = (j-1) .^(2 .* (i-1));
            end %end if
        end %end for j
    end %end for i
```

% function to calculate the vector x

```
function y = x(N,z,wp,ws,alpha)
    k=zeros(1,z);
    k(1) = 1;
    k=k';
    p_1 = p(ws,wp,alpha,N);
    c_1 = c(z,N);
    p_2 = inv(p_1) * c_1' ;
    p_3 = inv(c_1 * (inv(p_1)) * c_1');
    y = p_2 * p_3 * k;
```



```

% *****
% Program to obtain frequency response of FIR lowpass even
% length filter using maximally flat approximation method
% *****
% Input Parameters
% N : length of filter
% z : degree of flatness at  $\omega = 0$ 
% wp : passband cutoff frequency
% ws : stopband edge
% alpha : scale factor of stopband error
% % Example
% N = 40
% z = 3
% wp = 0.15
% alpha = 1
% ws = wp

Ne = input('enter the value of N(even) = ');
if rem(Ne,2)~=0 % check whether filter length is even
    disp('N must be even')

else
    me = Ne./2;
    ze = input('enter value of z = ');
    wpe = input('enter the value of wp lies between 0 & 0.5 = ');
    wpe = wpe./0.5.*pi;
    alphe = input('enter the value of alpha(0<alpha<=1) = ');
    wse = wpe;
    Le = 4.*Ne;
    ame = me-0.5;
    Xe1 = xe(Ne,ze,wpe,wse,alphe);
    h(1:me) = Xe1(me:-1:1)./2; % filter coefficients are obtained
    qe = pi./Le;
    RLe=Le-1;

    for j = 0:RLe,
        at = 0;
        for i = 1:me
            at = at+h(i).*(cos(qe.*(ame-i+1).*j));
        end % end of for loop i
        abe(j+1) = 2.*at;
    end %end of for loop j
    fe1 = 0:(0.5/Le):(0.5-(0.5/Le));
    plot(fe1,20.*(log10(abs(abe))), 'w-')
    % frequency response of designed filter is obtained
    lab(1:RLe+1) = 20.*(log10(abs(abe(1:RLe+1))));
    ml1 = [f1' lab'];
end %end of if

```

% function to calculate the error matrix P

```
function ye = pe(wse,wpe,alphe,Ne)
me = Ne./2;
for i = 1:me
    for j = 1:me
        if (i == j)
            tea1 = (pi./2)-(wse./2);
            tea2 = sin((2.*i-1).*wse)./(2.*(2.*i-1));
            temp1(i,j) = tea1-tea2;
        else
            tea1 = sin((i+j-1).*wse)./(i+j-1);
            tea2 = sin((i-j).*wse)./(i-j);
            temp1(i,j) = -0.5.*(tea1+tea2);
        end % end of if
        % Matrix Ps is obtained
    end % end of for loop j
end % end of for loop i

for i = 1:me
    for j = 1:me
        if (i == j)
            teb1 = (1.5.*wpe) + (sin((2.*i-1).*wpe)./(2.*(2.*i-1)));
            teb2 = 2.*(sin((i-0.5).*wpe)./(i-0.5));
            temp2 = teb1-teb2;
        else
            td1 = wpe + (sin((i+j-1).*wpe)./(2.*(i+j-1)));
            td2 = sin((i-j).*wpe)./(2.*(i-j));
            td3 = sin((i-0.5).*wpe)./(i-0.5);
            td4 = sin((j-0.5).*wpe)./(j-0.5);
            temp2(i,j) = td1+td2-td3-td4;
        end % end of if
        % Matrix Pp is obtained
    end % end of for loop j
end % end of for loop i

ye = (alpha * temp1) + ((1-alpha)*temp2);
% Matrix P is obtained
```

% function to calculate the constraint matrix C

```
function ye = ce(ze,Ne)
me = Ne./2;
for i = 1:ze
    for j = 1:me
        if (i==1 | j==1 )
            ye(i,j) = 1;
        else
            ye(i,j) = (2.*j-1).^(2*(i-1));
        end
    end
end
```

% function to calculate the error matrix P

```
function ye = pe(wse,wpe,alphe,Ne)
me = Ne./2;
for i = 1:me
    for j = 1:me
        if (i==j)
            tea1 = (pi./2)-(wse./2);
            tea2 = sin((2.*i-1).*wse)./(2.*(2.*i-1));
            temp1(i,j) = tea1-tea2;
        else
            tea1 = sin((i+j-1).*wse)./(i+j-1);
            tea2 = sin((i-j).*wse)./(i-j);
            temp1(i,j) = -0.5.*(tea1+tea2);
        end
        % Matrix Ps is obtained
    end
end

for i = 1:me
    for j = 1:me
        if (i==j)
            teb1 = (1.5.*wpe) + (sin((2.*i-1).*wpe)./(2.*(2.*i-1)));
            teb2 = 2.*(sin((i-0.5).*wpe)./(i-0.5));
            temp2 = teb1-teb2;
        else
            td1 = wpe + (sin((i+j-1).*wpe)./(2.*(i+j-1)));
            td2 = sin((i-j).*wpe)./(2.*(i-j));
            td3 = sin((i-0.5).*wpe)./(i-0.5);
            td4 = sin((j-0.5).*wpe)./(j-0.5);
            temp2(i,j) = td1+td2-td3-td4;
        end
        % Matrix Pp is obtained
    end
end

ye = (alpha * temp1) + ((1-alpha)*temp2);
% Matrix P is obtained
```