Design of Half-Band Filters to Construct Orthonormal Wavelets

by
Sanjay Chandra Verma

A thesis submitted to the
Department of Electrical and Computer Engineering
in conformity with the requirements of
the degree of Master of Science (Engineering)

Queen's University
Kingston, Ontario, Canada
September, 1998

Abstract

Cooklev in his Ph.D thesis has presented a new method for half-band filter design (which structurally incorporates the regularity constraint into the design procedure) for constructing orthonormal wavelets. His design method however, suffered from certain limitations such as : splitting of the multiple zeros at $z = -1$ into simple zeros and the non-convergence of the magnitude response of the product filter.

This thesis deals with the elimination of both these limitations in Cooklev's design method. We deal with the zero-splitting problem in a very simple manner, by factoring out the zeros at $z = -1$. The problem of non-convergence of the magnitude response of the product filter is dealt with by using the Goldfarb-Idnani (GI) dual algorithm to achieve the nonnegative frequency response as is necessary for the construction of orthonormal wavelets.

We observe that not only does the GI-algorithm guarantee convergence of the magnitude response of the product filter, but it also helps to construct orthonormal wavelets even when the optimization takes place with respect to an odd number of coefficients, something that was thought of as being not possible before. The use of the GI-algorithm not only ensures that the new scaling and wavelet functions are more regular than those obtained using Cooklev's method, but it also in some cases is instrumental in achieving scaling and wavelet functions more regular than the celebrated Daubechies scaling and wavelet functions.

Acknowledgements

I dedicate this thesis to my late grandparents and my parents.

I would like to thank my supervisor, Dr. Christopher J.Zarowski, for introducing me to the field of wavelets and for his excellent guidance, encouragement, patience, and support during my time at Queen's. He, in my mind was the best supervisor I could have had and one of the most intelligent persons I have ever met.

Many thanks are also due to Dr. Troung Nguyen, Dr. Gilbert Strang for their excellent wavelet workshop that increased my enthusiasm in this field, to Dr. Berwin Turlach, for the help given to understand the GI-algorithm, to Dr. Eric Koelink for his help with orthogonal polynomials and others in the field of wavelets and Mathematics, especially Dr. P.P. Vaidyanathan, Dr. Ingrid Daubechies, Dr. Selesnick, who have patiently answered all my queries.

I would like to thank my parents who have been a constant source of love, encouragement and blessings and who have taught me that hard work and perseverance is the key to success. I also would like to thank my fiancee Ekta, for her encouragement, support and her prayers.

I thank my cousin Sujata and my brother-in-law Amit, who were always there whenever I needed support. To my friends Homiar and Monaz, who always encouraged me, my housemates Yogesh, Prasad, Saugata, George and Bhaskar who made life fun in Kingston, to Nigam, Monika, Raj, Sumita, Govind, Jyoti who have cheered me whenever things started getting tough, to Manpreet and Anita who believed in me, to Geoff, Chris and Jean for their constant help and support, to Jay, Bo, Mike, Kareem, Martin, Haseeb, Hasan, and Osama for always being a source of inspiration, thanks to all of them.

I would like to thank this wonderful Department of Electrical and Computer Engineering and all the Professors and other staff members each of whom had a little something to do to make my stay here as cheerful as possible.

Last, but not the least, to Susan, Bonnie and Cathy at the International center who were my only family in Kingston and without whom I probably would have long left this place, thanks a million.

This work was supported financially by the Natural Sciences and Engineering Research Council, and the School of Graduate Studies and Research.

Contents

List of Figures

Symbol Notation

Symbol	Definition
$\mathbf{L^2(R)}$	space of finite energy analog (i.e., continuous time) signals
$\mathbf{R}$	the set of real numbers
$\mathbf{C}$	the set of complex numbers
$\mathbf{Z}$	set of integers
$\psi(t)$	the wavelet function
$\phi(t)$	the scaling function
$\Psi(\omega)$	fourier transform of the wavelet function
$\Phi(\omega), \hat{\phi}(\omega)$	fourier transform of the scaling function
V_j	sequence of embedded subspaces
W_{j-1}	the orthogonal complement of V_{j-1} in V_j
$\delta(m)$	discrete Kronecker delta function
L	the number of vanishing moments of the wavelet
	(which is also the number of zeros at $z = = -1$)
N	filter length
K	the number of elements in the vector α

α	the vector, with respect to which optimization takes place
$h_k, h(k)$	impulse response of lowpass filter of the filter bank
$g_k, g(k)$	impulse response of highpass filter of the filter bank
p_k	impulse response of the product filter
$H(z)$	z-transform of h_k
$G(z)$	z-transform of g_k
$P(z)$	z-transform of the product filter
$S(z)$	a Laurent polynomial
T_0	some odd polynomial in the Theorem (2.1)
$H(e^{j\omega})$	the frequency response of h_k
$b_k^N(x)$	The kth Bernstein polynomial of degree N
$S(\alpha)$	set of constraints defined by Equation (3.13b)
G	is a n x n symmetric positive definite matrix
C	is a n x m matrix
W	denotes the set of indices of the constraints
A	the index of the matrix of normal vectors of the constraints in the active set
q	cardinality of A
$\tilde{N}$	the matrix of normal vectors of the constraints in the active set
E	perturbation parameter
$T_n(x)$	Chebyshev polynomial of the first kind
$s(\phi)$	Sobolev regularity of the scaling function

Chapter 1

Introduction

1.1 Introduction

Wavelets are functions that satisfy certain mathematical requirements and are used in representing data or other functions. This concept is not new. Approximation using the superposition of functions has existed since the early 1800's, when Joseph Fourier discovered that he could superpose sines and cosines to represent other (periodic) functions. However, in wavelet analysis, the scale that one uses in looking at data plays a special role. Wavelet algorithms process data at different scales or resolutions. If we look at a signal with a large "window," we would notice gross features. Similarly, if we look at a signal with a small "window," we would notice small discontinuities. The result in wavelet analysis is to "see the forest and the trees" [1].

For many decades, scientists have wanted more appropriate functions than the sines and cosines which comprise the bases of Fourier analysis, to approximate choppy signals. By their definition, these functions are non-local (stretch out to infinity), and therefore do a very poor job in approximating sharp spikes. But with wavelet analysis, we can use approximating functions that are compactly supported, or at least are

concentrated about some mean in time. Wavelets are well-suited for approximating data with sharp discontinuities.

Since the original signal or function can be represented in terms of a wavelet expansion (using coefficients in a linear combination of the wavelet functions), data operations can be performed using just the corresponding wavelet coefficients. And if you further choose the best wavelets adapted to your data, or truncate the coefficients below a threshold, your data is sparsely represented. This "sparse coding" makes wavelets an excellent tool in the field of data compression. Wavelets in general can be said to have three basic properties :

- Wavelets are building blocks for general functions.
- Wavelets have Time-Frequency localization (i.e., most of the energy of the signal is concentrated about a certain mean time and mean frequency which in turn implies that the rms duration and rms bandwidth of the signal are narrow).
- Wavelets have fast transform algorithms.

It must be pointed out that these three properties are not unrelated. For example, if the wavelet basis is orthogonal, then the coefficients are simply given as the inner product of the function with the basis functions, which greatly simplifies the transform algorithm.

1.2 Why Are Wavelets Useful ?

The properties mentioned in the previous section are important. Most of the data which we encounter in real life is not totally random but has a certain correlation structure. Think for example of audio signals, images, solutions of differential equations, etc. The correlation structure of many of these signals are similar. They have some correlation in space (or time), but the correlation is local. For example, neighbouring pixels in an image are highly correlated but ones that are far from each other are uncorrelated. Similarly, there is some correlation in frequency, but again it's local i.e., around a particular interval.

This motivates approximating these data sets with building blocks that have space and frequency localization as well. Such building blocks will be able to reveal the internal correlation structure of the data sets. This should result in powerful approximation qualities: only a small number of building blocks should already provide an accurate approximation of the data. And hence, these properties of wavelets are extremely useful.

1.3 Applications of Wavelets

A major application of wavelets to technology has been in the area of data compression. The following list indicates the breadth of this application area [2]:

- Audio compression 8 : 1.
- Still-image compression 20 : 1 (BW), 100 : 1 (Color).

- Seismic compression 20 : 1.
- Radiology images 20 : 1.
- Fingerprint images 25 : 1.
- Video compression (color) 140 : 1.

The basic idea in a compression algorithm in all of the above examples is to represent the digitized signal in terms of a wavelet expansion. Using a statistical analysis of the data type involved one carries out a systematic dropping of bits of these wavelet expansion coefficients at specific scales to represent the same signal effectively with fewer bits.

Wavelets have recently become popular in many different scientific fields, including signal processing. Because of the appealing properties mentioned earlier, wavelets appear to be promising signaling waveforms in communications [3]. Motivation for the use of wavelets for waveform coding stems from the fact that the two ideal waveforms often used to benchmark analog pulse shaping performance, namely, the time-limited rectangular pulse and the band-limited sinc pulse, are examples of so-called *scaling functions* and have corresponding wavelets. Thus, wavelet theory appears to have the potential for analog pulse shaping applications.

Other applied fields that are making use of wavelets are: astronomy, acoustics, nuclear engineering, sub-band coding, neurophysiology, music, magnetic resonance imaging, speech discrimination, optics, fractals, turbulence, earthquake-prediction,

radar, human vision, and pure mathematics applications such as solving partial differential equations.

1.4 Motivation and Objective of the Thesis

Around 1985 Ingrid Daubechies started work on wavelet bases and some two years later she made an important mathematical discovery. She put the wavelet theory in proper perspective by showing the intimate relationship between filter banks and wavelets and constructing orthonormal basis functions with finite support that are smooth [4].

Dilations and translations of the mother wavelet, elegantly give rise to *multiresolution analysis*, which was advanced mainly by Mallat [5] and Meyer [6]. The merging of filter banks, wavelets and multiresolution analysis stimulated an enormous amount of research activity in many areas.

Not all filter banks give rise to wavelet bases. Only regular filters do. Cooklev in his Ph.D. thesis [7] investigated and designed a regular filter bank that leads to orthonormal wavelet bases. However, it has been shown in Zarowski [8], that Cooklev's approach has certain limitations which would give rise to irregular (non-smooth) wavelet bases. The main aim of this thesis is to formulate an alternate design algorithm which is more efficient and faster than the ones suggested in [9] and that completely eliminates the problems that appear in Cooklev's theory.

1.5 Outline of the Thesis

This thesis is organized as follows:

Chapter 2, entitled *Orthonormal Wavelet Filters* presents an introduction and some mathematical preliminaries of the concepts of *wavelets* and *multiresolution analysis* and construction of wavelets. It explains the relationship between wavelets and filter banks. Finally, the chapter presents a comprehensive account of Cooklev's theory of half-band filter and wavelet design and it's limitations.

Chapter 3, entitled *New Algorithm for the Design of Half-Band Filters* discusses the approach taken to eliminate the limitations in Cooklev's theory. The highlight of this chapter is the use of the Goldfarb-Idnani dual algorithm to solve the optimization problem and the simulation results that validate it's use, and also demonstrates that the new design algorithm is more efficient and it's implementation time faster than the methods suggested in Zarowski [9].

Chapter 4, entitled *Spectral Factorization and Orthonormal Wavelets* explains the need and presents the theory of spectral factorization of the product filter. The suitability of Bauer's method is explained and it also presents the Interpolatory Graphical Display Algorithm (IGDA), which is an iterative procedure used to construct scaling and wavelet functions. The simulation results demonstrate the validity of the new design algorithm and the choice of Bauer's method for spectral factorization. It also authenticates our claim that the new design algorithm is much superior to Cooklev's method. This chapter also consolidates our claim by comparing the reg-

ularity property and the frequency characteristics of the scaling function created by the new design algorithm, with that of Daubechies and Cooklev's scaling function, respectively.

Chapter 5, entitled *Conclusions and Suggestions for Future Research* summarizes the major contributions made in this thesis and suggests some modifications, new techniques and a few extensions that could be done for future research.

Chapter 2

Orthonormal Wavelet Filters

2.1 Introduction

"If you steal from one author, it's plagiarism;

if you steal from many, it's research"

- Wilson Mizner, *The Legendary Mizners* (1953)

These lines happen to be the spirit of this chapter, as this chapter can be considered to be as a literature review introducing the concept of *wavelets* and *multiresolution analysis* (MRA). It looks into various methods of constructing wavelets and also elucidates the relation between wavelets and filter banks. In particular this chapter explains in detail Cooklev's [7] theory of half-band filter and wavelet design and its limitations.

2.2 What are Wavelets ?

Wavelets are functions that are generated from one single function often called the "mother wavelet", by translations and dilations, and provide a series expansion of functions belonging to $\mathbf{L^2(R)}$, where $\mathbf{R}$ is the set of real numbers. We may regard

$\mathbf{L^2(R)}$ as the space of finite energy analog (i.e., continuous time) signals. We shall let $\mathbf{Z}$ denote integers, and $\mathbf{C}$ denote the complex numbers. If $x(t) \in \mathbf{L^2(R)}$ and $x(t) \in \mathbf{C}$ then $\|x\|^2 = \int_{-\infty}^{\infty} |x(t)|^2 \, dt < \infty$. The name *wavelet* comes from the requirement that the function should have a mean of zero, i.e., $\int \psi(t)dt = 0$, thus, *waving* above and below the time axis. The diminutive connotation of wavelet suggests the function has to be well localized. Wavelet basis functions are localized in time and frequency and hence wavelet analysis is an ideal tool for representing signals that contain discontinuities (in the signal or its derivatives) or for signals that are not stationary. Wavelet analysis is an alternative to Fourier analysis. As with the Fourier transform, the point of wavelets is not the wavelets themselves; they are a means to an end. The goal is to turn the information in a signal into numbers (coefficients), that can be manipulated, stored, transmitted, analyzed, or used to reconstruct the original signal.

2.2.1 Continuous Wavelet Transform

The continuous wavelet transform (CWT) of $g(t)$ with respect to wavelet $\psi(t)$ is defined by

$$WT\{g; a, b\} = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} g(t)\psi^* \left(\frac{t-b}{a}\right) dt \ , \tag{2.1}$$

where, $a \neq 0$ and b are called the *scale* and *translation* parameters, respectively. The asterix superscript denotes complex conjugate as $\psi(t)$ may be complex valued, i.e., $\psi(t) \in \mathbf{C}$. Furthermore, the Fourier transform of the wavelet $\psi(t)$, denoted $\Psi(\omega)$, is

$$\Psi(\omega) = \int_{-\infty}^{\infty} \psi(t) e^{-j\omega t} \, dt \ ,$$

and must satisfy the following admissibility condition:

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{\omega} d\omega < \infty , \tag{2.2}$$

which shows that $\psi(t)$ has to oscillate and decay. This condition guarantees the existence of an inverse transform. These facts are considered in detail in [4].

2.2.2 Multiresolution Analysis

There are two ways to introduce wavelets: one is through the continuous wavelet transform as described earlier, and another is through multiresolution analysis. Here we begin by defining multiresolution analysis, and then point out some connections with the continuous wavelet transform.

The idea of multiresolution analysis is to write L^2-functions $f(x)$ as a limit of successive approximations, each of which is a coarser version of $f(x)$, with more and more details added to it. The successive approximations thus use a different resolution, whence the name multiresolution analysis. To achieve this we seek to expand the given function $f(x)$ in terms of basis functions $\phi(x)$ which can be scaled to give multiple resolutions of the original signal. In order to develop a multilevel representation of a function in $\mathbf{L^2(R)}$ we seek a sequence of embedded subspaces V_j such that

$$\{\emptyset\} \cdots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \cdots \subset \mathbf{L^2(R)} \tag{2.3}$$

with the following properties :

1. $V_j \subset V_{j+1}$ (containment)

2. $v(x) \in V_j \Leftrightarrow v(2x) \in V_{j+1}$ (scaling property)

3. $v(x) \in V_0 \Leftrightarrow v(x+1) \in V_0$ (translation)

4. $\bigcup_{j=-\infty}^{\infty} V_j$ is dense in $\mathbf{L^2(R)}$ (completeness) and $\bigcap_{j=-\infty}^{\infty} V_j = 0$ (uniqueness)

5. A *scaling function* $\phi \in V_0$, with a non-vanishing integral, exists so that the collection $\{\phi(x-l) \mid l \in \mathbf{Z}\}$ is a Riesz basis of V_0 (A set $\{f_k\} \subset V$ is called a *Riesz basis* if every element $s \in V$ of the space can be written as $s = \sum_k c_k f_k$ for some choice of scalers $\{c_k\}$ and if positive constants X and Y exist such that $X\|s\|^2 \leq \sum_k |c_k|^2 \leq Y\|s\|^2$ where $\|\cdot\|$ stands for 2-norm, i.e., $\|x\|^2 = \int_{-\infty}^{\infty} |x(t)|^2\, dt$, clearly, by the definition, the set $\{f_k\}$ is a basis if $\{c_k\}$ are unique for any $s \in V$).

We will use the following terminology: a *level* of a multiresolution analysis is one of the V_j subspaces and one level is *coarser* (respectively, finer) with respect to another whenever the index of the corresponding subspace is smaller (respectively, bigger). An introduction to the concept of multiresolution analysis and its usefulness can be found in [5], [4].

From the above mentioned properties we deduce that if we seek a scaling function $\phi(x) \in V_0$ such that its integer translates $\{\phi(x-k), \mid k \in \mathbf{Z}\}$ form a Riesz basis for the space V_0, then $\{2^{\frac{i}{2}}\phi(2^i x - k) \mid k \in \mathbf{Z}\}$ form a Riesz basis for the space V_i. The detailed argument is lengthy, but may be found in [5]. Since, in particular, the space V_0 lies within the space V_1, we can express any function in V_0 in terms of the

basis of V_1. Consequently, for appropriate h_k

$$\phi(x) = \sqrt{2} \sum_{k=-\infty}^{\infty} h_k \, \phi(2x - k) \ , \tag{2.4a}$$

in which h_k, $k \in \mathbb{Z}$ is a square summable sequence. The construction of dyadic orthonormal wavelets is based on Equation (2.4a) as will be shown in the succeeding sections. For scaling functions supported on interval $[0, N]$

$$\phi(x) = \sqrt{2} \sum_{k=0}^{N} h_k \, \phi(2x - k) \tag{2.4b}$$

where N is odd [4]. It has been shown in [4] that the sequence h_k must be of even length. The sequence h_k must also satisfy the following conditions [4], [10] :

$$\sum h_k = \sqrt{2} \tag{2.5}$$

$$\sum h_k \, h_{k+2m} = \delta(m) \tag{2.6}$$

$$\sum (-1)^k k^m h_k = 0 \ , \forall m = 0, 1, \cdots, L-1 \ , \tag{2.7}$$

where $L \geq 1$ and where $\delta(m)$ denotes a discrete Kronecker delta function. The parameter L is very important. As shown in [4], the larger L is, the smoother the solution $\phi(x)$ to Equation (2.4a) will be. Furthermore, L is equal to the number of vanishing moments of the wavelet corresponding to $\phi(x)$ [4, 10] (L vanishing moments of the wavelet function $\psi(x)$ corresponds to $\int x^k \psi(x) \, dx = 0, \ k = 0, 1, \cdots, L-1$). The functional Equations (2.4a,b) go by several different names: the *refinement equations*, the *dilation equations* or the *two-scale difference equations.*

We can now also define

$$\phi_{j,k}(x) \;=\; 2^{\frac{j}{2}}\phi\left(2^{j}x-k\right)\;, \tag{2.8}$$

in which case $\{\phi_{j,k}(x) \mid k \in \mathbf{Z}\}$ forms a Riesz basis for the space V_j, as noted earlier.

2.2.3 The Wavelet Function

We now investigate the difference between subspaces V_{j-1} and V_j. We define a new subspace W_{j-1} such that it is the orthogonal complement of V_{j-1} in V_j i.e.,

$$V_j \;=\; V_{j-1} \bigoplus W_{j-1} \;\;;\;\; V_{j-1} \perp W_{j-1}\;\;, \tag{2.9}$$

where $\bigoplus$ represents a *direct sum.* If $f(x) \in V_{j-1}$, $g(x) \in W_{j-1}$ then their inner product is

$$\langle f,g\rangle \;=\; \int_{-\infty}^{\infty} f(x)g^{*}(x)\,dx \;=\; 0\;\;.$$

It follows then that the spaces W_j are orthogonal and that

$$\bigoplus_{j\,\in\,Z} W_j \;=\; \mathbf{L^2(R)} \tag{2.10}$$

Now let us introduce a wavelet function $\psi(x)$ such that $\{\psi(x-k) \mid k \in \mathbf{Z}\}$ forms a Riesz basis for the subspace W_0. Then, it turns out that,

$$\psi_{j,k} \;=\; 2^{\frac{j}{2}}\,\psi\left(2^{j}x-k\right)\;,\; k \in \mathbf{Z} \tag{2.11}$$

is a Riesz basis for W_j [5]. If in addition, the set $\{\psi(x-k),\; k \in \mathbf{Z}\}$ forms an orthonormal set, then it follows that $\{\psi_{j,k},\; j,k \in \mathbf{Z}\}$ forms an orthonormal basis for $\mathbf{L^2(R)}$.

Now, since the space W_0 is contained in the space V_1, we can express the wavelet function in terms of the scaling function at the next higher scale [10], i.e.,

$$\psi(x) \;=\; \sqrt{2} \sum_{k=-\infty}^{\infty} g_k \, \phi\,(2x-k) \;\;, \tag{2.12a}$$

and for $\psi(x)$ on interval $[0, N]$

$$\psi(x) \;=\; \sqrt{2} \sum_{k=0}^{N} g_k \, \phi\,(2x-k) \;\;, \tag{2.12b}$$

where for (2.12a)

$$g_k \;=\; (-1)^k h^*_{1-k} \;\;, \tag{2.13a}$$

and for (2.12b)

$$g_k \;=\; (-1)^k h^*_{N-k} \;\;. \tag{2.13b}$$

2.3 The Relation Between Wavelets and Filter Banks

The connection between continuous-time wavelets and the discrete filter banks, was originally investigated by Daubechies [4]. According to Daubechies a 2-band parauni-tary FIR filter bank as shown in Figure 2.1 can be used to generate a multiresolution analysis with compactly supported orthonormal wavelets. Let us define $H(z)$ and $G(z)$ to be the z-transforms of the sequences h_k and g_k, i.e.,

$$H(z) \;\;=\;\; \sum_{k=0}^{N} h_k z^{-k} \tag{2.14}$$

$$G(z) \;\;=\;\; \sum_{k=0}^{N} g_k z^{-k} \;\;. \tag{2.15}$$

The filters $H(z)$ and $G(z)$ are called scaling and wavelet filters, respectively. Equation (2.13b) implies that $H(z)$ and $G(z)$ are quadrature mirror filters (QMF's) (seen in Figure 2.1), i.e., assuming $h_k \in \mathbf{R}$

$$G(z) = -z^{-N}H(-z^{-1}) \ , \tag{2.16}$$

where $N + 1$ is the filter length (N is the degree of the filter). Therefore, only one filter, e.g., low-pass filter $H(z)$ has to be designed. The paraunitary condition [11], [12], [13], is given as

$$P(z) + P(-z) = 2 \ , \tag{2.17}$$

where the *"product filter"* is

$$P(z) = H(z)H(z^{-1}) \ . \tag{2.18}$$

Equation (2.17) indicates that $P(z)$ is a halfband filter and Equation (2.18) shows that $P(e^{j\omega})$ must be nonnegative.

The connection between the paraunitary solutions ($H(z)$ and $G(z)$) and wavelets can be described as follows. Suppose that the analysis stage of the filter bank of Figure 2.1 is iterated on the low pass branch at each step of the decomposition [12], then this generates equivalent band-pass filters of the form [14]

$$G^i(z) = H(z)H(z^2)\cdots H(z^{2^{i-2}})G(z^{2^{i-1}}) \ . \tag{2.19}$$

Letting $i \longrightarrow \infty$ gives the *"mother wavelet"* $\psi(t)$ [4]. That is,

$$\psi(t) = \lim_{i\to\infty} g^i_{\lfloor t2^i \rfloor} \ , \tag{2.20}$$

where g_n^i is the impulse response of $G^i(z)$. In the next section we describe two examples of orthonormal wavelets. This is one possible way to obtain plots of wavelets. Another is via the interpolatory graphical display algorithm (IGDA) considered again in Section 4.4, though only briefly.

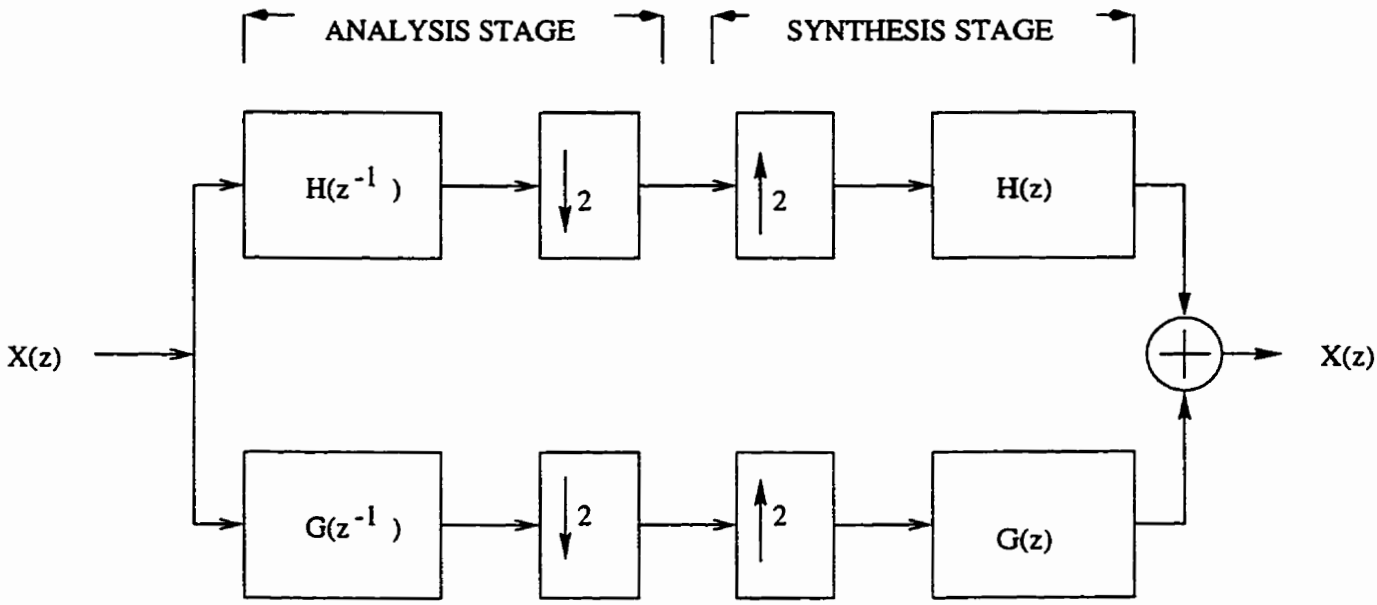


Figure 2.1: Paraunitary two-band FIR filter bank. $H(z)$ and $G(z)$ are half-band low-pass and high-pass filters, respectively.

2.4 Orthonormal Wavelets

Recall that a function $\psi(t) \in \mathbf{L}^2(R)$ is called an orthonormal wavelet if the collection of functions $\psi_{j,k}(t), j, k \in \mathbf{Z}$, is an orthonormal basis of $\mathbf{L^2(R)}$. We now summarize various methods of constructing such $\psi(t)$.

2.4.1 Meyer Wavelets

The Meyer wavelets are orthonormal wavelets defined over the entire set $\mathbf{R}$, i.e., they are not supported on a finite interval. The Fourier transform of the Meyer's scaling function is given by

$$\Phi(\omega) = \int_{-\infty}^{\infty} \phi(t) e^{-j\omega t}\, dt \ .$$

In this case,

$$\Phi(\omega) = \begin{cases} 1 & , |\omega| \geq \frac{2\pi}{3} \\ \cos\left[\frac{\pi}{2} v\left(\frac{3}{2\pi}|\omega| - 1\right)\right] & , \frac{2\pi}{3} \leq |\omega| \leq \frac{4\pi}{3} \\ 0 & , \text{otherwise} \ , \end{cases} \tag{2.21}$$

where the real-valued function $v(x)$ satisfies

$$v(x) = \begin{cases} 0 & , x \leq 0 \\ 1 & , x \geq 1 \end{cases} \ , \tag{2.22}$$

and the symmetry condition

$$v(x) + v(1-x) = 1 \tag{2.23}$$

on the interval $[0, 1]$. A procedure for finding $\Psi(\omega)$ from $\Phi(\omega)$ is in Vetterli and Kovacevic [15].

Because of the definition in Equation (2.21) we can readily show that the Meyer scaling function satisfies

$$\sum_{k=-\infty}^{\infty} |\Phi(\omega + 2\pi k)|^2 = 1 \ . \tag{2.24}$$

Consequently, the set $\{\phi(t-k) \mid k \in \mathbf{Z}\}$ is orthonormal and hence this establishes the orthonormality of the Meyer wavelets. As noted earlier, the Meyer wavelets are not supported on a finite interval, hence they are not compactly supported. Another scaling function that is very much like the Meyer scaling function has been recently proposed by Xia [16]. A recent contribution by Sheikholeslami and Kabal [17], proposes a general family of Nyquist functions of which the raised cosine function is a special case. It must be noted that the Meyer scaling functions are actually a generalization of the square root raised cosine functions [18].

2.4.2 Daubechies Wavelets

Various procedures exist for constructing wavelets with different properties aside from orthogonality alone. The approach used by Daubechies is to introduce a new MRA of $\mathbf{L^2(R)}$ that is generated by compactly supported scaling functions. In [19] a constructive procedure for obtaining the sequence $\{h_k\}_{k=0}^{N}$, with $h_k \in \mathbf{R}$ has been provided. We give a statement of the main result, which is drawn from the summary of Daubechies's work in [4].

Define $p_k = \sqrt{2}h_k$ (and similarly define $q_k = \sqrt{2}g_k$), and let

$$\hat{P}(z) = \frac{1}{2}\sum_{k=0}^{N} p_k z^k \ , \ z \in \mathbf{C} \ .$$

From Daubechies [4]

Theorem 2.1 (Daubechies) *Let $S(z)$ be a Laurent polynomial (i.e., $S(z^{-1})$ is the z-transform of an autocorrelation sequence) satisfying*

$$\max_{|z| = 1} | S(z) | \leq 2^{M-1} \quad (M \geq 1) \tag{2.25}$$

and

$$| S(e^{-j\omega}) |^2 = \sum_{k=0}^{M-1} \binom{M+k-1}{k} \left(\sin\frac{\omega}{2}\right)^{2k} + \left(\sin\frac{\omega}{2}\right)^{2M} T_0\left(\frac{\cos\omega}{2}\right)$$

for some odd polynomial T_0. Then

$$\hat{P}(z) = \left(\frac{1+z}{2}\right)^M S(z) \ .$$

The wavelet and scaling function obtained from this $\hat{P}(z)$ are compactly supported and orthonormal.

Daubechies considered the special case where $T_0 = 0$. Condition (2.25) is satisfied for all $M \geq 1$ in this instance. It turns out that $N \in \{1, 3, 5, 7, \cdots\}$ as $N = 2M - 1$. That is, the sequences h_k, and g_k are of even length. Theorem 2.1 characterizes all orthonormal wavelets supported on an interval.

2.4.3 Cooklev's Theory of Wavelet Design

Cooklev [7] has presented a theory of wavelet design based on the eigenfilter approach to the design of half-band filters. This theory also involved Bernstein polynomial expansions since these made it easy to incorporate regularity into the design of the half-band filter. The incorporation of regularity is vital in wavelet construction since

wavelets are essentially constructed from lowpass filters, and it is desirable to have at least one zero at $z = -1$ in the filter's transfer function. The presence of such zeros is to be seen in the expression $\hat{P}(z)$ of Theorem 2.1. A filter having at least one such zero is said to be *regular*. A zero at this location is sufficient to ensure convergence of the iterative procedures (e.g., the IGDA [20]) used to construct the wavelet function from the lowpass filter coefficients. The approach to half-band filter design in [1] is also very useful in the design of orthogonal and regular QMF filter banks.

Cooklev's method was motivated by another method by Rioul and Duhamel [14]. The method in [14] modifies the Remez exchange and results in equiripple and regular filters. On the other hand, Cooklev's method as mentioned earlier is based on the eigenfilter approach [21]. The advantages of the eigenfilter approach as compared to the Remez exchange are:

- The eigenfilter formulation is numerically efficient and can be used in the orthogonal and biorthogonal cases (see [10]).
- It is more general, since it allows time-domain constraints which cannot be taken care of in the Remez exchange approach.
- Eigenfilter method allows nearly-equiripple designs, if they are necessary.
- The eigenfilter formulation can be extended to the 2-D case [22], while the Remez exchange does not generalize to multiple dimensions.

It must be noted that Cooklev's method can be considered as a technique to evaluate T_0 of Theorem 2.1.

2.5 Cooklev's Theory of Half-Band Filter Design

2.5.1 Introduction

Zarowski [8] has presented a very detailed derivation of the half-band filter design method found in Chapter 3 of [7], and we repeat [8] almost verbatim in this section. A useful modification has been included in [8] to the original procedure in [7] that avoids the computation of eigenvalues and eigenvectors and this appears in Section 2.5.5.

In [7] a least squares approach, similar to eigenfilter design, is employed. It is also seen that *Bernstein polynomials* [1] are central to the theory. They are important in that they make it relatively easy to incorporate regularity constraints into the filter design. The method makes it possible to develop new types of wavelet functions as well. We see that the presentation in Zarowski [8] is more detailed in some respects than that in [7], and also, amongst other things, it points out the fact that the method in [7] does not generally give a unique solution.

What follows now is in essence, the sequence of transformations that we carry out on the product filter $P(z)$ to ensure that it is available to us in a form that makes it easy to use it for our optimization problem. We show how we transform $P(z)$ having real valued coefficients p_k, into $P(e^{j\omega})$ which is a function of the coefficients b_k and also c_k (with the help of a lemma that we use), which are also real valued. We then show that the type of filter that we consider can be a half-band filter, whose

[1] Bernstein polynomials for half-band filters were first considered by others. See Section 4.3 in [23].

spectrum is a function of the real-valued coefficients d_k which in turn are related to the coefficients p_k. We show how the coefficients c_k are related to another set of real-valued coefficients e_k, which finally leads us to a form of the product filter (by now transformed into $P(x)$) which has been transformed to an equation (in terms of the parameters α_k and the Bernstein polynomials), which we can use for our optimization problem such that the energy of the product filter $P(x)$ in the stopband is minimized, and the frequency response is nonnegative, i.e., $P(x) \geq 0$.

2.5.2 Some Preliminaries

It is useful to begin with the following Lemma.

Lemma 2.1 • *We may write*

$$cos\,(n\theta) \;=\; \sum_{k\,=\,0}^{n} \beta_{n,k}\; cos^{k}\,\theta \quad , \tag{2.26}$$

where

$$\beta_{m+1,k} \;=\; 2\beta_{m,k-1} \;-\; \beta_{m-1,k} \quad . \tag{2.27}$$

The initial conditions for this recursion are $\beta_{1,0} = 0$, $\beta_{1,1} = 1$, *and* $\beta_{2,0} = -1$, $\beta_{2,1} = 0$, $\beta_{2,2} = 2$.

Proof • The proof is by induction and can be found in Zarowski [8].

An immediate consequence of the lemma is $\beta_{0,0} = 1$ and $\beta_{n,k} = 0\ \forall\ k < 0$ and $k > n$. This result is employed in the theory to follow.

We now consider a Type I FIR filter (i.e., a filter whose impulse response is finite in length [24]) with system function

$$P(z) = \sum_{k=0}^{2N} p_k z^{-k} \quad . \tag{2.28}$$

Being a Type I FIR filter (according to the definitions in Oppenheim and Schafer [24]) N is odd and

$$p_k = p_{2N-k} \tag{2.29}$$

for $k = 0, 1, \ldots, N-1$.

We observe that we may write

$$P(z) = \sum_{k=0}^{2N} p_k z^{-k} = p_N z^{-N} + \sum_{k=0}^{N-1} p_k z^{-k} + \sum_{k=N+1}^{2N} p_k z^{-k}$$

$$= p_N z^{-N} + \sum_{k=0}^{N-1} p_k z^{-k} + \sum_{j=0}^{N-1} p_{2N-j} z^{-(2N-j)} \quad , \tag{2.30}$$

and so

$$P(z) = p_N z^{-N} + \sum_{k=0}^{N-1} p_k [z^{-k} + z^{-(2N-k)}] \quad . \tag{2.31}$$

We may write

$$P(e^{j\omega}) = e^{-j\omega N} \sum_{k=0}^{N} b_k \cos(k\omega) \quad , \tag{2.32}$$

where

$$b_0 = p_N \, , \quad b_r = 2p_{N-r} \quad (r = 1, 2, \ldots, N) \quad . \tag{2.33}$$

From (2.32) we must have, with the aid of Lemma 2.1,

$$P(e^{j\omega}) \;=\; e^{-j\omega N} \sum_{k=0}^{N} c_k \cos^k \omega \quad . \tag{2.34}$$

We use Lemma 2.1 to obtain an expression for $\{c_k\}$ in terms of $\{b_k\}$. Thus,

$$\sum_{k=0}^{N} b_k \cos(k\omega) \;=\; \sum_{k=0}^{N} b_k \left[\sum_{j=0}^{k} \beta_{k,j} \cos^j \omega \right] \tag{2.35}$$

$$= \sum_{k=0}^{N} b_k \left[\sum_{j=0}^{N} \beta_{k,j} \cos^j \omega \right] \tag{2.36}$$

$$= \sum_{j=0}^{N} \left[\sum_{k=0}^{N} b_k \beta_{k,j} \right] \cos^j \omega \tag{2.37}$$

$$= \sum_{j=0}^{N} \underbrace{\left[\sum_{k=j}^{N} b_k \beta_{k,j} \right]}_{=c_j} \cos^j \omega \tag{2.38}$$

which reveals the following upper triangular linear system of equations that relates $\{c_k\}$ to $\{b_k\}$

$$\begin{bmatrix} \beta_{0,0} & \beta_{1,0} & \cdots & \beta_{N-1,0} & \beta_{N,0} \\ 0 & \beta_{1,1} & \cdots & \beta_{N-1,1} & \beta_{N,1} \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & \beta_{N-1,N-1} & \beta_{N,N-1} \\ 0 & 0 & \cdots & 0 & \beta_{N,N} \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \\ b_N \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \\ c_N \end{bmatrix} \quad . \tag{2.39}$$

Let us denote the matrix in (2.39) by B. Since $\beta_{k,k} \neq 0$ for all k this linear system always has a unique solution. In fact, from (2.27)

$$\beta_{k+1,k+1} \;=\; 2\beta_{k,k} \;-\; \beta_{k-1,k+1} \;=\; 2\beta_{k,k} \tag{2.40}$$

so $\beta_{k,k} \;=\; 2^{k-1}$, for $k \;\geq\; 1$.

2.5.3 Half-band Filters

If we now assume $p_{2k+1} = 0$, but that $p_N \neq 0$, then the odd indexed elements of $\{p_k\}$ are forced to zero, except for element p_N which is in the middle of the sequence. For these assumptions we see that from (2.31)

$$P(z) = p_N z^{-N} + \sum_{k=0}^{N-1} p_k [z^{-k} + z^{-(2N-k)}] \tag{2.41}$$

$$= p_N z^{-N} + \sum_{k=0}^{(N-1)/2} p_{2k} [z^{-2k} + z^{-(2N-2k)}] + \sum_{k=1}^{(N-1)/2} p_{2k-1} [z^{-(2k-1)} + z^{-(2N-2k+1)}]$$

$$= p_N z^{-N} + \sum_{k=0}^{(N-1)/2} p_{2k} [z^{-2k} + z^{-(2N-2k)}] \quad (\text{as } p_{2k-1} = 0) \ , \tag{2.42}$$

and so

$$P(-z) = -p_N z^{-N} + \sum_{k=0}^{(N-1)/2} p_{2k} [z^{-2k} + z^{-(2N-2k)}] \tag{2.43}$$

for which we conclude immediately that

$$P(z) - P(-z) = 2p_N z^{-N} \ . \tag{2.44}$$

This is called the *half-band condition*, and a filter that satisfies it is called a *half-band filter*. We have therefore shown that Type I FIR filters can be half-band.

From the preceding we also see that

$$P(z) = z^{-N} \left[p_N + \sum_{k=0}^{(N-1)/2} p_{2k} [z^{N-2k} + z^{-N+2k}] \right] \tag{2.45}$$

so that

$$P(e^{j\omega}) = e^{-j\omega N} \left[p_N + \sum_{k=0}^{(N-1)/2} p_{2k} [e^{j\omega[N-2k]} + e^{-j\omega[N-2k]}] \right]$$

$$= e^{-j\omega N}\left[p_N + 2\sum_{k=0}^{(N-1)/2} p_{2k}\cos((N-2k)\omega)\right]$$
$$= e^{-j\omega N}\left[p_N + 2\sum_{n=0}^{(N-1)/2} p_{N-1-2n}\cos((2n+1)\omega)\right] ,$$

where $(2n = N - 1 - 2k)$ and for which we have

$$P(e^{j\omega}) = e^{-j\omega N}\left[d_{(N+1)/2} + \sum_{n=0}^{(N-1)/2} d_n\cos((2n+1)\omega)\right] , \qquad (2.46)$$

where we have

$$d_{(N+1)/2} = p_N , \quad d_n = 2p_{N-1-2n} \quad (n = 0, 1, \ldots, (N-1)/2) . \qquad (2.47)$$

The half-band condition may be described in a different but equivalent manner. Suppose that the FIR filter is noncausal with system function

$$P(z) = \sum_{k=-N}^{N} p_k z^{-k} , \qquad (2.48)$$

where N is again assumed to be odd, and $p_k = p_{-k}$. Thus, except for noncausality, this filter is Type I as before. As well, we impose condition $p_{2k} = 0$, but $p_0 \neq 0$. It is then easy to show that

$$P(z) + P(-z) = 2p_0 \qquad (2.49)$$

which is an equivalent definition of the half-band condition, i.e., is equivalent to (2.44). Since $-e^{j\omega} = e^{j(\omega+\pi)}$ we also see from (2.49) that

$$|\, P(e^{j\omega}) + P(e^{j(\omega+\pi)}) \,| = \text{constant} . \qquad (2.50)$$

(The filter coefficients in (2.48) are the same as those of (2.28) except for indexing.)

2.5.4 Bernstein Polynomials and Half-band Filter Design

The kth *Bernstein polynomial* of degree N is defined to be

$$b_k^N(x) = \binom{N}{k} x^k (1 - x)^{N-k} \quad . \tag{2.51}$$

It will be useful to recall that from the Binomial theorem

$$(a - x)^k = \sum_{j=0}^{k} (-1)^j \binom{k}{j} x^j a^{k-j} \quad , \tag{2.52}$$

and this gives

$$(1 - x)^k = \sum_{j=0}^{k} (-1)^j \binom{k}{j} x^j \quad . \tag{2.53}$$

Let $x = (1 - \cos\omega)/2$ so that $\cos\omega = 1 - 2x = (1 - x) - x$. Now, recalling (2.34), we can write

$$\sum_{k=0}^{N} c_k \cos^k \omega = \sum_{k=0}^{N} c_k [(1 - x) - x]^k \tag{2.54}$$

$$= \sum_{k=0}^{N} c_k \left[\sum_{j=0}^{k} (-1)^j \binom{k}{j} (2x)^j \right] \quad , \tag{2.55}$$

where the second equality has employed (2.52) (with $a = 1 - x$). For some suitable $\{e_k\}$ we may also write

$$\sum_{k=0}^{N} c_k \cos^k \omega = \sum_{k=0}^{N} e_k \binom{N}{k} x^k (1 - x)^{N-k} = \sum_{k=0}^{N} e_k b_k^N(x) \quad . \tag{2.56}$$

Here $e = [e_0\, e_1 \cdots e_N]^T$, and we define

$$D = \text{diag} \left\{ \binom{N}{0}, \binom{N}{1}, \cdots, \binom{N}{N-1}, \binom{N}{N} \right\} \quad ,$$

and

$$E = \begin{bmatrix} \binom{N}{0} & 0 & \cdots & 0 \\ -\binom{N}{1} & \binom{N-1}{0} & \cdots & 0 \\ \binom{N}{2} & -\binom{N-1}{1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ -\binom{N}{N} & \binom{N-1}{N-1} & \cdots & \binom{N-N}{N-N} \end{bmatrix} .$$

Again, letting $c = [c_0\ c_1\ \cdots\ c_N]^T$, and defining

$$\Lambda = \text{diag}\,\{2^0\ ,\ -2^1\ ,\ 2^2\ ,\ \ldots\ ,(-1)^k 2^k,\ \ldots\ ,(-1)^N 2^N\}\quad ,$$

and

$$C = \begin{bmatrix} \binom{0}{0} & \binom{1}{0} & \binom{2}{0} & \cdots & \binom{N}{0} \\ 0 & \binom{1}{1} & \binom{2}{1} & \cdots & \binom{N}{1} \\ 0 & 0 & \binom{2}{2} & \cdots & \binom{N}{2} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \binom{N}{N} \end{bmatrix}$$

it can be shown that

$$\Lambda C c = E D e\quad . \tag{2.57}$$

At this point we have well-defined matrix or linear transformations between all of the different expressions for $P(z)$ and/or $P(e^{j\omega})$.

From (2.48) with the given constraints

$$P(z) = p_0 + \sum_{k=1}^{N} p_k[z^{-k} + z^k]\quad , \tag{2.58}$$

and so

$$P(e^{j\omega}) \;=\; p_0 \;+\; 2\sum_{k=1}^{N} p_k \cos \omega k \quad . \tag{2.59}$$

Thus, for suitable $\{c_k\}$

$$P(e^{j\omega}) \;=\; \sum_{k=0}^{N} c_k \cos^k \omega \quad . \tag{2.60}$$

From (2.56) we may therefore define

$$P(x) \;=\; \sum_{k=0}^{N} e_k b_k^N(x) \;=\; \sum_{k=0}^{N} c_k \cos^k \omega \quad . \tag{2.61}$$

As a result of this we may write

$$\mid\; P(e^{j\omega}) \;+\; P(e^{j(\omega+\pi)}) \;\mid = \mid\; \sum_{k=0}^{N} c_k \cos^k \omega \;+\; \sum_{k=0}^{N} c_k \cos^k(\omega+\pi) \;\mid$$

$$= \mid\; \sum_{k=0}^{N} c_k \cos^k \omega \;+\; \sum_{k=0}^{N} (-1)^k c_k \cos^k \omega \;\mid \quad . \tag{2.62}$$

Now via (2.55), and $\cos \omega \;=\; 1 \;-\; 2x$ we have

$$P(x) \;=\; \sum_{k=0}^{N} c_k[(1 \;-\; x) \;-\; x]^k \;=\; \sum_{k=0}^{N} c_k \cos^k \omega \quad , \tag{2.63}$$

and

$$P(1 \;-\; x) \;=\; \sum_{k=0}^{N} c_k[x \;-\; (1 \;-\; x)]^k$$

$$= \sum_{k=0}^{N} (-1)^k c_k \cos^k \omega \tag{2.64}$$

so that (2.62) becomes

$$\mid\; P(e^{j\omega}) \;+\; P(e^{j(\omega+\pi)}) \;\mid = \mid\; P(x) \;+\; P(1 \;-\; x) \;\mid = \text{constant} \quad . \tag{2.65}$$

Thus,

$$P(x) + P(1 - x) = 1 \tag{2.66}$$

satisfies (2.50), and so is an equivalent half-band condition.

We shall now show that if

$$e_k = \begin{cases} 1 & , \ k < L \\ 1 - \alpha_{k-L+1} & , \ L \leq k \leq (N-1)/2 \\ \alpha_{N+1-L-k} & , \ (N+1)/2 \leq k \leq N - L \\ 0 & , \ N+1-L \leq k \leq N \end{cases} \tag{2.67}$$

then the condition in (2.66) is met. Clearly, for this to be well-defined, we must have $L \leq (N-1)/2$. We shall also see that for (2.67) $P(x)$ has a zero of order L at $x = 1$. This may be used to impose a certain regularity on the half-band filter (i.e., zeros at $z = -1$ in the lowpass filter leading to wavelets).

We have

$$P(x) = \sum_{k=0}^{N} e_k b_k^N(x) \ . \tag{2.68}$$

Using (2.67) it can be shown that

$$P(x) = \sum_{k=0}^{(N-1)/2} b_k^N(x) + \sum_{j=1}^{(N+1)/2-L} \alpha_j \binom{N}{j+L-1} [x^{N+1-L-j}(1-x)^{L+j-1} - x^{j+L-1}(1-x)^{N+1-L-j}] \ ,$$

and so

$$\begin{aligned} P(x) + P(1-x) &= \sum_{k=0}^{N} \binom{N}{k} x^k (1-x)^{N-k} \\ &= \sum_{k=0}^{N} b_k^N(x) = 1 \ , \end{aligned} \tag{2.69}$$

(Proof of which is provided in Zarowski [8] where the last equality can be argued from probability theory ($b_k^N(x)$ is a binomial pdf (Papoulis [25]); use the binomial theorem)). Thus, we have shown that if (2.67) holds then (2.66) holds.

In the previous paragraph we have shown that

$$P(x) = \sum_{k=0}^{(N-1)/2} \binom{N}{k} x^k(1-x)^{N-k} + \sum_{k=1}^{(N+1)/2-L} \alpha_k \binom{N}{k+L-1} [x^{N+1-L-k}(1-x)^{L+k-1} - x^{k+L-1}(1-x)^{N+1-L-k}] . \quad (2.70)$$

By inspection of this expression we see that $P(x)$ under the condition of (2.67) has a zero of order L at $x = 1$.

2.5.5 A Least Squares Approach

The half-band filter $P(z)$ has frequency response denoted by $P(e^{j\omega})$, for which we normally consider $\omega \in [0, \pi]$. Recalling that $x = (1 - \cos\omega)/2$, this interval maps to $x \in [0, 1]$. Thus, we consider $P(x)$ for $x \in [0, 1]$. As a half-band filter is lowpass, it has a passband $[0, \omega_p]$, and a stopband $[\omega_s, \pi]$, where $\omega_p < \omega_s$. Thus, for $P(x)$ the passband is $[0, x_p]$, and the stopband is $[x_s, 1]$.

Refer to Equation (2.70). Define the polynomials

$$v_0(x) = \sum_{k=0}^{(N-1)/2} \binom{N}{k} x^k(1-x)^{N-k} , \quad (2.71)$$

and

$$v_k(x) = \binom{N}{k+L-1} [x^{N+1-L-k}(1-x)^{L+k-1} - x^{k+L-1}(1-x)^{N+1-L-k}] , \quad (2.72)$$

where $k = 1, 2, \ldots, (N+1)/2-L$. Defining $\alpha_0 = 1$, and $a = [\alpha_0\ \alpha_1\ \alpha_2\ \cdots\ \alpha_{(N+1)/2-L}]^T$, and $v(x) = [v_0(x)\ v_1(x)\ v_2(x)\ \cdots\ v_{(N+1)/2-L}(x)]^T$ then

$$P(x) = a^T v(x) \quad . \tag{2.73}$$

One way to design $P(z)$ is to select vector a such that the energy of the filter in the stopband is minimized. This energy may be defined to be

$$E_s(a) = \int_{x_s}^{1} [a^T v(x)][a^T v(x)]^T \, dx \tag{2.74}$$

$$= a^T \underbrace{\left[\int_{x_s}^{1} v(x) v^T(x) \, dx\right]}_{= R} a \quad . \tag{2.75}$$

In this expression $R = [r_{j,k}]_{j,k = 0,1, \ldots, (N+1)/2-L}$, and $R = R^T$. Of course

$$r_{j,k} = \int_{x_s}^{1} v_j(x) v_k(x) \, dx \quad . \tag{2.76}$$

One approach, considered in [7], minimizes (2.75) subject to the constraint that $a^T a = 1$. We thus select the eigenvector of R corresponding to the smallest eigenvalue of R ($R > 0$, i.e., R is positive definite), and normalize it so the first element is unity (to satisfy $\alpha_0 = 1$). This is the desired value for a.

However, there is another possibility, apparently not considered in [7]. Since $\alpha_0 = 1$ we may partition a as $a = [1\ \alpha^T]^T$, where $\alpha = [\alpha_1\ \cdots\ \alpha_{(N+1)/2-L}]^T$. Similarly

$$R = \begin{bmatrix} r_{0,0} & r^T \\ r & \mathcal{R} \end{bmatrix} \quad , \tag{2.77}$$

where $r = [r_{1,0}\ r_{2,0}\ \cdots\ r_{(N+1)/2-L,0}]^T$. Thus, the stopband energy expression can be rewritten as

$$E_s(\alpha) = [1\ \alpha^T]\begin{bmatrix} r_{0,0} & r^T \\ r & \mathcal{R} \end{bmatrix}\begin{bmatrix} 1 \\ \alpha \end{bmatrix} = r_{0,0} + 2\alpha^T r + \alpha^T \mathcal{R}\alpha \quad . \tag{2.78}$$

Since R is positive definite, $\mathcal{R}$ will be as well. Thus, (2.78) can be rewritten as (upon completing the square)

$$E_s(\alpha) = [\alpha + \mathcal{R}^{-1}r]^T\mathcal{R}[\alpha + \mathcal{R}^{-1}r] + r_{0,0} - r^T\mathcal{R}^{-1}r \quad . \tag{2.79}$$

The optimum choice of α, which we shall denote by $\hat{\alpha}$, therefore satisfies

$$\mathcal{R}\hat{\alpha} = -r \quad . \tag{2.80}$$

Clearly, this choice minimizes the stopband energy. This approach to designing the filter is easier (or computationally more efficient) than the eigenproblem approach since solving a linear system of equations is typically simpler than solving an eigenproblem.

2.5.6 Half-band Filters With Nonnegative Frequency Response

If half-band filters are to be employed in the construction of orthonormal wavelets it is necessary to create half-band filters with a nonnegative frequency response. Recall that $P(x)$ is real-valued, so we therefore want $P(x) \geq 0$ for all $x \in [0,1]$. To obtain half-band filters with this property consider the following approach which is based on that as suggested in [7].

Suppose we solve (2.80). In this case the optimum choice for a is $\hat{a} = [1\ \hat{\alpha}^T]^T$. Find the $x = x_{min}$ such that $P(x)$ is minimized for $a = \hat{a}$, i.e.,

$$P(x_{min}) = \hat{a}^T v(x_{min}) = \delta_{min} < 0 \quad . \tag{2.81}$$

Let $\hat{\delta}_{min} = a^T v(x_{min})$. Now define a new stopband energy function

$$E_s'(a) = \int_{x_s}^{1} [\hat{\delta}_{min} - a^T v(x)][\hat{\delta}_{min} - a^T v(x)]\, dx \tag{2.82}$$

$$= a^T \underbrace{\left[\int_{x_s}^{1} [v(x_{min}) - v(x)][v(x_{min}) - v(x)]^T\, dx\right]}_{=Q} a \quad . \tag{2.83}$$

It is clear that matrix Q replaces R in (2.75). At this point a to minimize $E_s'(a)$ in (2.83) may be found using the procedure in Subsection 2.5.5 earlier.

From (2.83)

$$Q = \int_{x_s}^{1} [v(x_{min}) - v(x)][v(x_{min}) - v(x)]^T\, dx =$$

$$\int_{x_s}^{1} \left[v(x_{min})v^T(x_{min}) - v(x)v^T(x_{min}) - [v(x)v^T(x_{min})]^T + v(x)v^T(x)\right] dx \,. \tag{2.84}$$

Clearly, the last term corresponds to matrix R in (2.75). Similarly to (2.77) we may partition Q according to

$$Q = \begin{bmatrix} q_{0,0} & q^T \\ q & \mathcal{Q} \end{bmatrix} \quad . \tag{2.85}$$

Thus, the optimum *new* choice for α is $\hat{\alpha}$ satisfying

$$\mathcal{Q}\hat{\alpha} = -q \quad . \tag{2.86}$$

This is similar to (2.80). Note that in [7] (see p. 46) it is remarked that the number of elements in $\hat{\alpha}$ of (2.86) must be an even number. This assertion will be challenged later on.

From (2.84) we see that

$$\int_{x_s}^{1} v(x_{min})v^T(x_{min})\,dx = (1-x_s)\begin{bmatrix} v_0(x_{min}) \\ v_1(x_{min}) \\ \vdots \\ v_{(N+1)/2-L}(x_{min}) \end{bmatrix} [v_0(x_{min})\ v_1(x_{min})\ \cdots\ v_{(N+1)/2-L}(x_{min})]\ ,$$

$$\int_{x_s}^{1} v(x)v^T(x)\,dx \ = \ R \quad ,$$

and

$$\int_{x_s}^{1} v(x)v^T(x_{min})\,dx = \begin{bmatrix} \int_{x_s}^{1} v_0(x)\,dx \\ \int_{x_s}^{1} v_1(x)\,dx \\ \vdots \\ \int_{x_s}^{1} v_{(N+1)/2-L}\,dx \end{bmatrix} [v_0(x_{min})\ v_1(x_{min})\ \cdots\ v_{(N+1)/2-L}(x_{min})]\ . \qquad (2.87)$$

It is important to note that the above procedure may need to be iterated, and that there is no known proof it will converge [8]. This fact is not at all clear from reading [7]. In fact, it will be shown later that this procedure is not very satisfactory.

2.6 A DFT/FFT Approach

The direct approach to finding p_k given e_k was defined in Subsection 2.5. However, this procedure involves inverting the matrices B and C. For $N > 15$ (approximately) the condition numbers of these matrices rises rapidly. Hence this procedure is not recommended except for small N.

Zarowski [8] shows the implementation using an alternative DFT/FFT-based approach (i.e., solving a Discrete Fourier Transform using Fast Fourier Transform algorithms [24]). The idea is similar to the use of the DFT in obtaining the impulse response of an equiripple FIR filter obtained from the Parks-McClellan algorithm

using the polynomial representation of the filter's frequency response. This idea was noted in the last paragraph of Section 7.6.3 (p. 478) of Oppenheim and Schafer [24]. From the expression for $P(x)$ in Equation (2.56)

$$\begin{aligned} P(e^{j\omega}) &= e^{-j\omega N} \sum_{k=0}^{N} e_k \begin{pmatrix} N \\ k \end{pmatrix} \left[\frac{1 - \cos\omega}{2}\right]^k \left[\frac{1 + \cos\omega}{2}\right]^{N-k} \\ &= \sum_{n=0}^{2N} p_n e^{-j\omega n} \quad , \end{aligned} \tag{2.88}$$

where $\cos\omega = 1 - 2x$. Now we define $P_r = P(e^{j\omega})$ for $\omega = \frac{2\pi}{2N+1}r$, where $r = 0, 1, \ldots, 2N$. Taking the inverse DFT of the sequence $\{P_r\}$ (via an FFT algorithm) will give $\{p_k\}$. This turns out to be a more numerically reliable method of getting p_k than the direct method for both large N, and large L.

2.7 Limitations of Cooklev's Design Method

Cooklev's approach to the design of half-band filters with a nonnegative frequency response in Section 2.5 was shown to have two significant difficulties in Zarowski [8] as explained below.

2.7.1 Zero Splitting

As shown in Zarowski [8], Cooklev's theory of design for half-band filters via Bernstein polynomial expansions suffers from the problem of the splitting of the desired multiple zero at $z = -1$ into simple zeros if care is not exercised in its implementation. This will likely cause problems in the spectral factorization stage which is necessary in the construction of wavelets based on this approach. A similar problem of zero splitting has also been observed in the case of Daubechies polynomials [26]. Actually, if some

or all the coefficients of a polynomial are known only to a specified accuracy - as is ordinarily the case in scientific computing - the concept of multiple zeros become meaningless : An arbitrary small change of the coefficients leads to the disintegration of an m-fold zero into a dense cluster of m distinct zeros [27].

Figure 2.2 shows a typical plot of filter zeros for $P(z)$ obtained via the direct method and via the DFT/FFT method. We see that both of the filter designs do not posses the multiple zero at $z = -1$. This multiple zero splits into several simple zeros in the vicinity of $z = -1$. However the splitting is less severe in the DFT/FFT method.

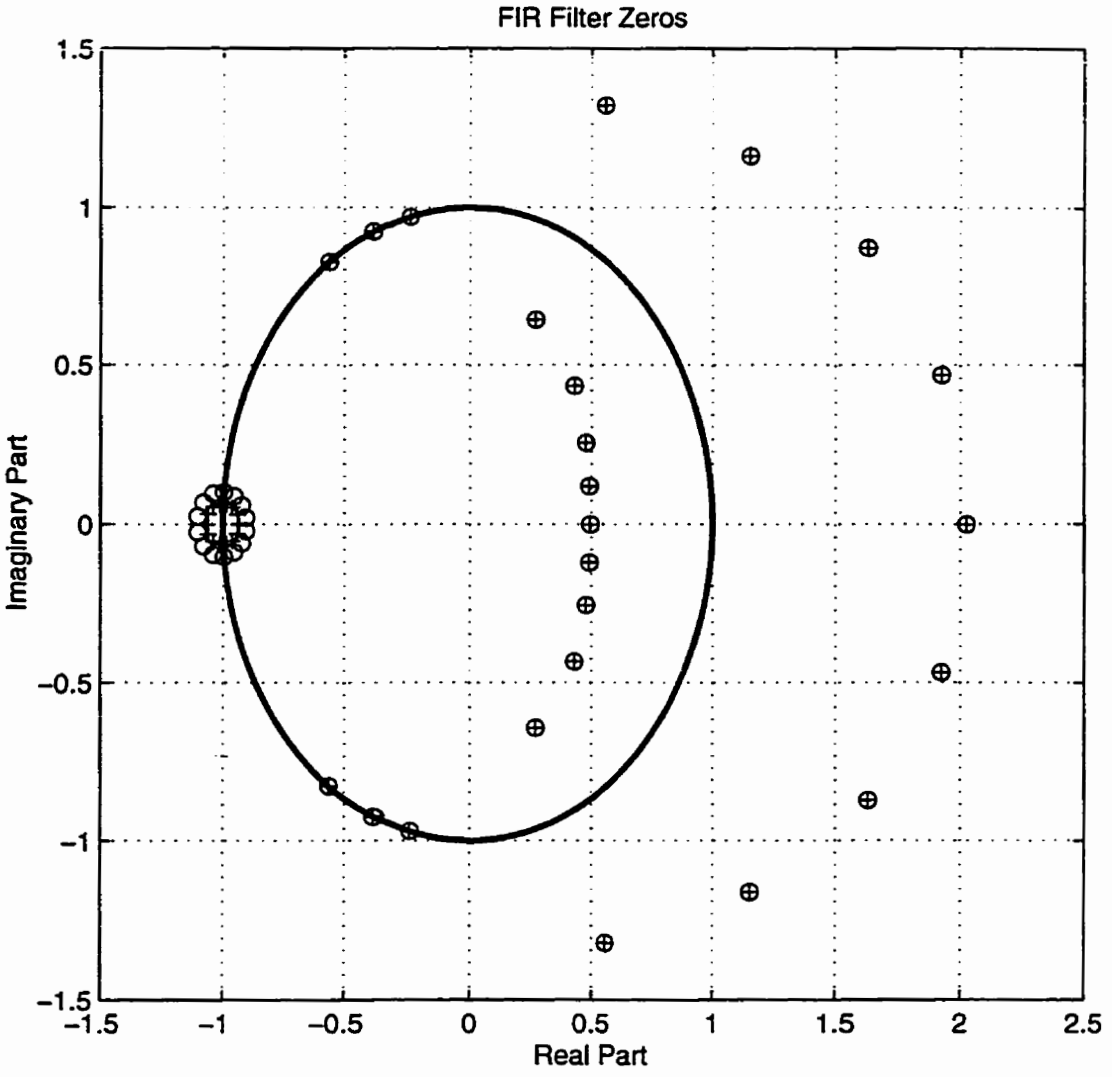


Figure 2.2: Plot of the zeros of a half-band filter for $x_s = 0.6$, $N = 19$, and $L = 7$. The circles are the zeros for the filter using the matrix inverse or direct approach (Section 2.5) while the plus signs are the zeros for the filter using the DFT/FFT method (Section 2.6).

2.7.2 Non-Convergence of Frequency Response

It was noted in [8] that the procedure for half-band filters with a nonnegative frequency response considered in Cooklev [7] may need to be iterated, and that convergence is not guaranteed. By this we mean that the stopband energy is not minimized, as evidenced by the local minima in the stopband not touching the frequency axis. Clearly, not being able to be sure of convergence is a major problem since this leads to sub-optimal results which leads to irregular wavelets as will be shown in the next chapter.

The plot in Figure 2.3 shows a typical magnitude response of the half-band filter with $x_s = 0.6$, $N = 17$, and $L = 5$ designed using Cooklev's method of Section 2.5. We notice that the frequency response fails to converge. This is evident from the failure of the two local minima in the stopband to touch the frequency axis.

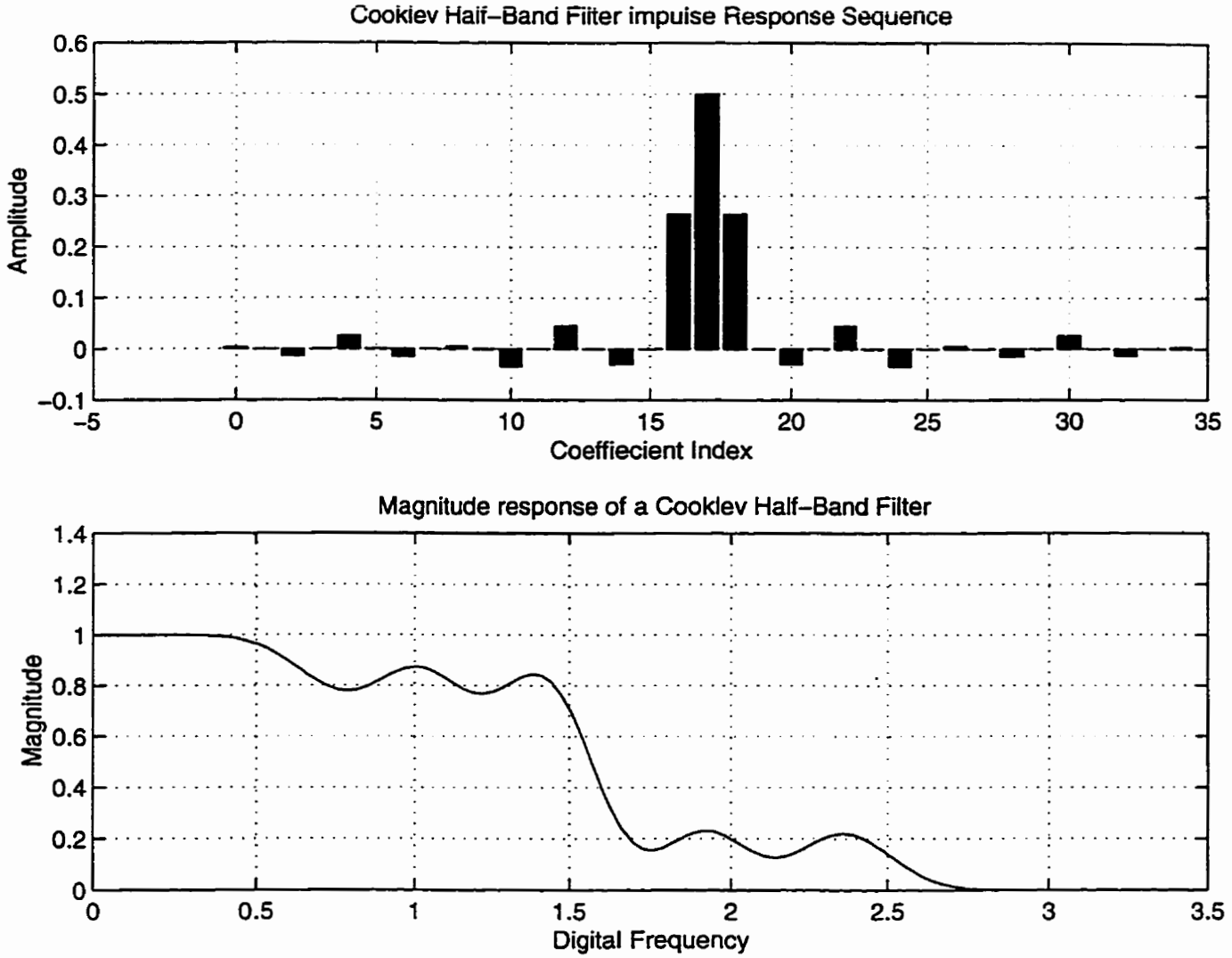


Figure 2.3: The magnitude response of the half-band filter for $x_s = 0.6$, $N = 19$, and $L = 7$ designed using the matrix inverse method (Section 2.5).

Chapter 3

New Algorithm for the Design of Half-Band Filters

3.1 Introduction

In this chapter we discuss the approach we take to eliminate the limitations in Cooklev's theory of half-band filter and wavelet design as was alluded to in Chapter 2. We first discuss the elimination of the zero-splitting problem and then introduce a new design algorithm that completely eliminates the non-convergence problem of the magnitude response of Cooklev's half-band filter. We then present some important simulation results that show the validity of the new design algorithm and it's superiority over the other methods including Cooklev's [7] and Zarowski's [8].

3.2 Elimination of Zero-splitting

As shown in Zarowski [8], Cooklev's theory of design for half-band filters via Bernstein polynomial expansions suffers from the problem of the splitting of the desired multiple zero at $z = -1$ into simple zeros if care is not exercised in its implementation. This is likely to cause problems in the spectral factorization stage, which is necessary in

the construction of wavelet functions based on this approach. MATLAB's mroots(·) function (version 5.x of MATLAB) can prevent the splitting, but only "up to a point," and is not capable of preventing the splitting from arising in the first place. It therefore only masks but does not solve the underlying problem, and so is not satisfactory in this respect.

Zarowski [8] observed that the transformation matrices denoted by Equation (2.39) and Equation (2.57) are in fact ill-conditioned for large N. This could be a probable cause for the splitting of the multiple zero at $z = -1$. A similar ill-conditioning problem has been successfully overcome in [28], by using the Chebyshev polynomials which are orthogonal. Motivated by this we derived Chebyshev polynomial expressions (see Appendix A) to orthogonalize the Bernstein polynomials which have been used in [7] in the design of the half-band filter. However, this did not yield the expected result.

It is observed that, for low-order polynomials, commonly available subroutine packages for root-finding work quite well. For higher order filters, the burden on the root-finding program can be considerably reduced by taking advantage of the fact that the locations of all the unit circle double zeros of the product filter are known *a priori*, i.e., they correspond to the stopband zeros of the frequency response. Hence, we present here a very simple solution to the zero-splitting problem that involves factoring out the offending factor of $(1 - x)^L$ from the expression for $P(x)$ in Equation (2.71) (a similar technique has been used in [14]). We then compute the transfer function corresponding to the factor that remains. This is a numerically

well-behaved process because this factor usually only consists of a z-polynomial with simple zeros (or low multiplicity multiple zeros). The process is equivalent to factoring $(1 + z^{-1})^{2L}$ out from $P(z)$, which is the desired half-band filter system function. The multiple zero at $z = -1$ of order $2L$ can be "put back later on" if desired.

3.2.1 Factoring out $(1 + z^{-1})^{2L}$

We may restate Equation (2.71) for convenience here as

$$P(x) = \sum_{k=0}^{(N-1)/2} \binom{N}{k} x^k (1-x)^{N-k} +$$

$$\sum_{k=1}^{(N+1)/2-L} \alpha_k \binom{N}{k+L-1} [x^{N+1-L-k}(1-x)^{L+k-1} - x^{k+L-1}(1-x)^{N+1-L-k}] \; . \tag{3.1}$$

We observe that in term no. 1 factor $(1 - x)^k$ has k in the range of $(N+1)/2$ to N, while term no. 2 has it in the range L to $(N-1)/2$, and term no. 3 has it in the range of $(N+1)/2$ to $N-L$. We recall that $L \leq (N-1)/2$. Suppose that

$$P(x) = (1-x)^L G(x) \tag{3.2}$$

then from (3.1)

$$G(x) = \sum_{k=0}^{(N-1)/2} \binom{N}{k} x^k (1-x)^{N-L-k} +$$

$$\sum_{k=1}^{(N+1)/2-L} \alpha_k \binom{N}{k+L-1} [x^{N+1-L-k}(1-x)^{k-1} - x^{k+L-1}(1-x)^{N+1-2L-k}]. \tag{3.3}$$

Now we recall that $x = (1 - \cos\omega)/2$, and if we use analytic continuation (i.e., replace $e^{j\omega}$ with z) then since $\cos\omega = (e^{j\omega} + e^{-j\omega})/2$

$$\frac{z + z^{-1}}{2} \text{ replaces } \frac{e^{j\omega} + e^{-j\omega}}{2} \; ,$$

and so from (3.2)

$$P(z) = z^{-N}\left[\frac{1+\frac{z+z^{-1}}{2}}{2}\right]^{L} G\left(\frac{1-\frac{z+z^{-1}}{2}}{2}\right) \quad , \tag{3.4}$$

where the factor z^{-N} is included to make the impulse response sequence that gives $P(z)$ into a causal sequence. This factor corresponds to $e^{-j\omega N}$ in (9.1) of [1]. We may rewrite (3.4) as

$$P(z) = \frac{1}{2^{2L}} z^{-(N-L)}(1+z^{-1})^{2L}\, G\left(\frac{-z+2-z^{-1}}{4}\right) \quad . \tag{3.5}$$

From this we see that it is possible to find the zeros of $G(z)$ independently from those of $P(z)$ and put the multiple zero at $z = -1$ back afterwards. Note that the degree of $P(z)$ is $2N$ while the degree of $G(z)$ is $2(N-L)$.

This solves our problem.

Routine makeh3.m implements the above procedure. Routine check.m compares the output of this routine with that provided by makeh1.m; these routines can be found in Appendix B . It does this by producing a plot of the zeros of both filter designs. Typical output appears in Fig. 3.1, and we see that our problem is truly solved.

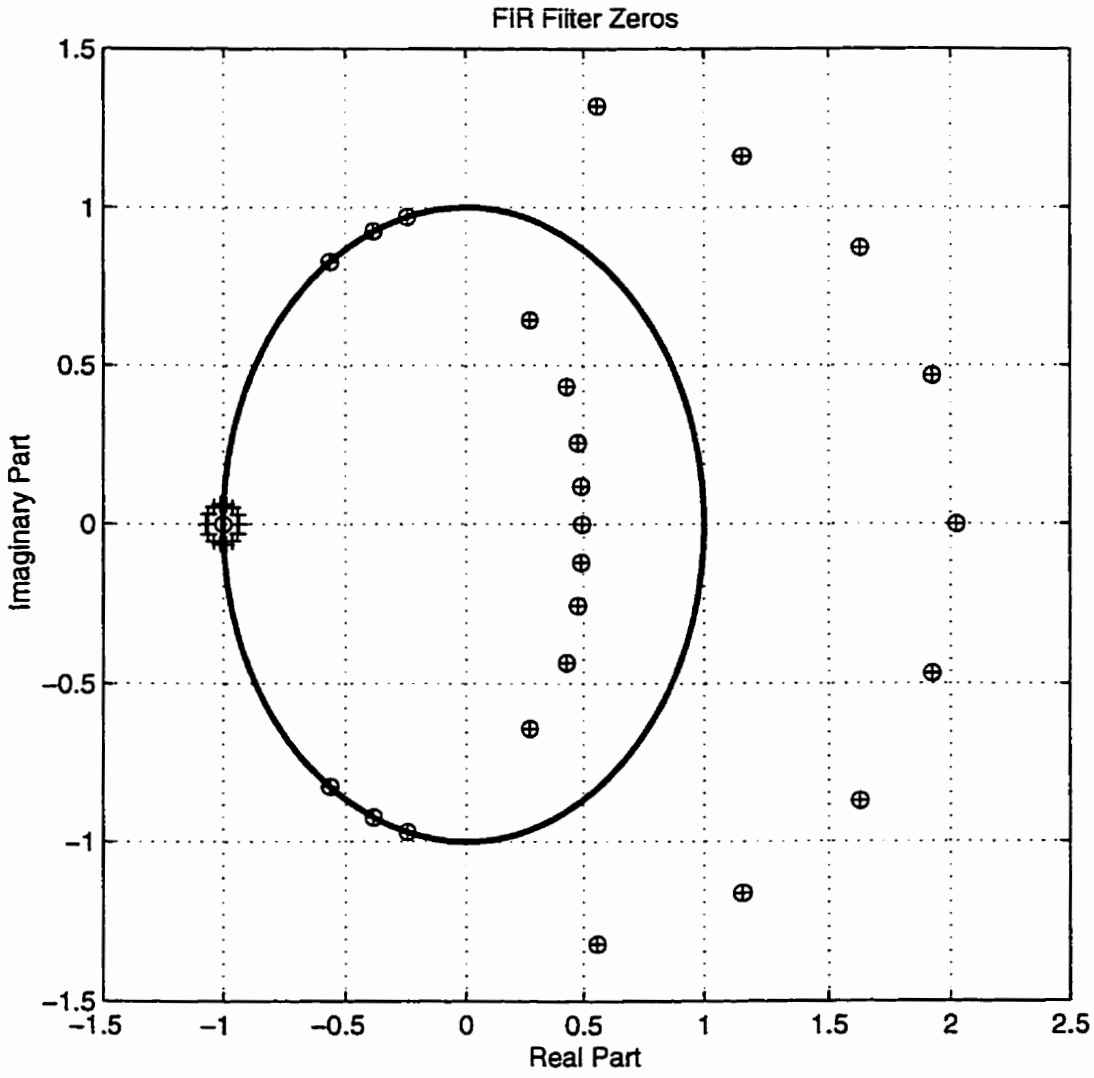


Figure 3.1: Typical output from check.m in Appendix C. The parameters are $x_s = 0.6$, $N = 17$, and $L = 8$. The plusses are the zeros of the half-band filter using the DFT/FFT method given in Chapter 2, while the circles are the zeros of the half-band filter given by the procedure in this Section.

3.3 New Design Algorithm

To eliminate the problem of non-convergence of the magnitude response of Cooklev's half-band filter we rephrase the original optimization problem and use the Goldfarb-Idnani (GI) Dual Algorithm [29] to solve it. The GI-algorithm is particularly useful since it has an excellent reputation for efficiency [30] and also it has been successfully used in the design of FIR filters before [31].

3.3.1 The Optimization Problem

We restate the optimization problem as given in Equation (2.77) of Chapter 2. Recall that

$$P(x) \;=\; a^T v(x) \tag{3.6}$$

for which we only let $x \;\in\; [0,1]$, i.e., x is confined to the unit interval. Equation (3.6) is the frequency response of the half-band filter, and for us we want $P(x) \;\geq\; 0$ for all $x \;\in\; [0,1]$. This necessitates finding the proper vector a. However, we also want to minimize energy in the stopband $[x_s, 1]$ (see Equation (2.75)). From (2.78) in Chapter 2, this energy is given by

$$f(\alpha) \;=\; \alpha^T \mathcal{R} \alpha \;+\; 2\alpha^T r \;+\; r_{0,0} \quad , \tag{3.7}$$

where $a \;=\; [1 \;\; \alpha^T]^T$, and the remaining quantities in (3.7) are defined in Section V of [2]. We have what is commonly called a Quadratic Programming (QP) problem with a linear inequality constraint $P(x) \;\geq\; 0$.

The optimization occurs over the elements of the vector $\alpha = [\alpha_1\ \alpha_2\ \cdots\ \alpha_K]^T$, (and we define $\alpha_0 = 1$) which has $K = (N+1)/2 - L$ elements. The Goldfarb-Idnani (GI) algorithm expects the problem to be phrased as

$$\min_{\alpha} \quad f(\alpha) = \frac{1}{2}\alpha^T(2\mathcal{R})\alpha + 2r^T\alpha + r_{0,0} \tag{3.8}$$

subject to the inequality constraint

$$S(\alpha) = C^T\alpha - b \geq 0 \ . \tag{3.9}$$

Equation (3.9) is explained below : The value of x is evaluated in a manner similar to the one given in [32]. We let

$$x = x_k = y + \frac{1-y}{M}k \tag{3.10}$$

where $k = 0, 1, \cdots, M-1$, $0.5 \leq y < 1$ and $M =$ the number of sample points.

Now let us express for all k the scalar $\hat{S}(\alpha) = P(x_k, \alpha) = a^T v(x_k) = v(x_k)^T a$ in matrix-vector form as

$$\underbrace{\begin{bmatrix} P(x_0) \\ P(x_1) \\ \vdots \\ P(x_{M-1}) \end{bmatrix}}_{=S} = \underbrace{\begin{bmatrix} v_0(x_0) & v_1(x_0) & \cdots & v_K(x_0) \\ v_0(x_1) & v_1(x_1) & \cdots & v_K(x_1) \\ \vdots & \vdots & & \vdots \\ v_0(x_{M-1}) & v_1(x_{M-1}) & \cdots & v_K(x_{M-1}) \end{bmatrix}}_{=V} \underbrace{\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_K \end{bmatrix}}_{=a} \tag{3.11}$$

where $P(x, \alpha) = P(x)$ is a change in notation to reflect the dependency of P on both scalar x, and vector α. The R.H.S of Equation (3.11) can further be written as (using the partition property)

$$\underbrace{\begin{bmatrix} v_0(x_0) \\ v_0(x_1) \\ \vdots \\ v_0(x_{M-1}) \end{bmatrix}}_{=-b} + \underbrace{\begin{bmatrix} v_1(x_0) & v_2(x_0) & \cdots & v_K(x_0) \\ v_1(x_1) & v_2(x_1) & \cdots & v_K(x_1) \\ \vdots & \vdots & & \vdots \\ v_1(x_{M-1}) & v_2(x_{M-1}) & \cdots & v_K(x_{M-1}) \end{bmatrix}}_{=C^T} \underbrace{\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_K \end{bmatrix}}_{=a} \tag{3.12}$$

which is the same form as Equation (3.9).

3.3.2 Justification Of The Use Of The GI-Algorithm

Adams and Sullivan [31] state that both the minimax (MM) and the least-squares (LS) optimality criteria used in the design of digital filters can be viewed as special cases in the class of peak-constrained least-squares (PCLS) optimization problems. In PCLS optimization problems, the peak error is constrained while the total squared error is minimized. Figure 3.2 shows the trade-off between the total squared error and the peak error. The best solutions for most practical applications are in the knee of the trade-off curve. The LS and MM are at the end-points (as shown in Figure 3.2), where the slopes are the most extreme. Therefore the LS and MM solutions are the two special cases of PCLS solutions that have the worst performance trade-off.

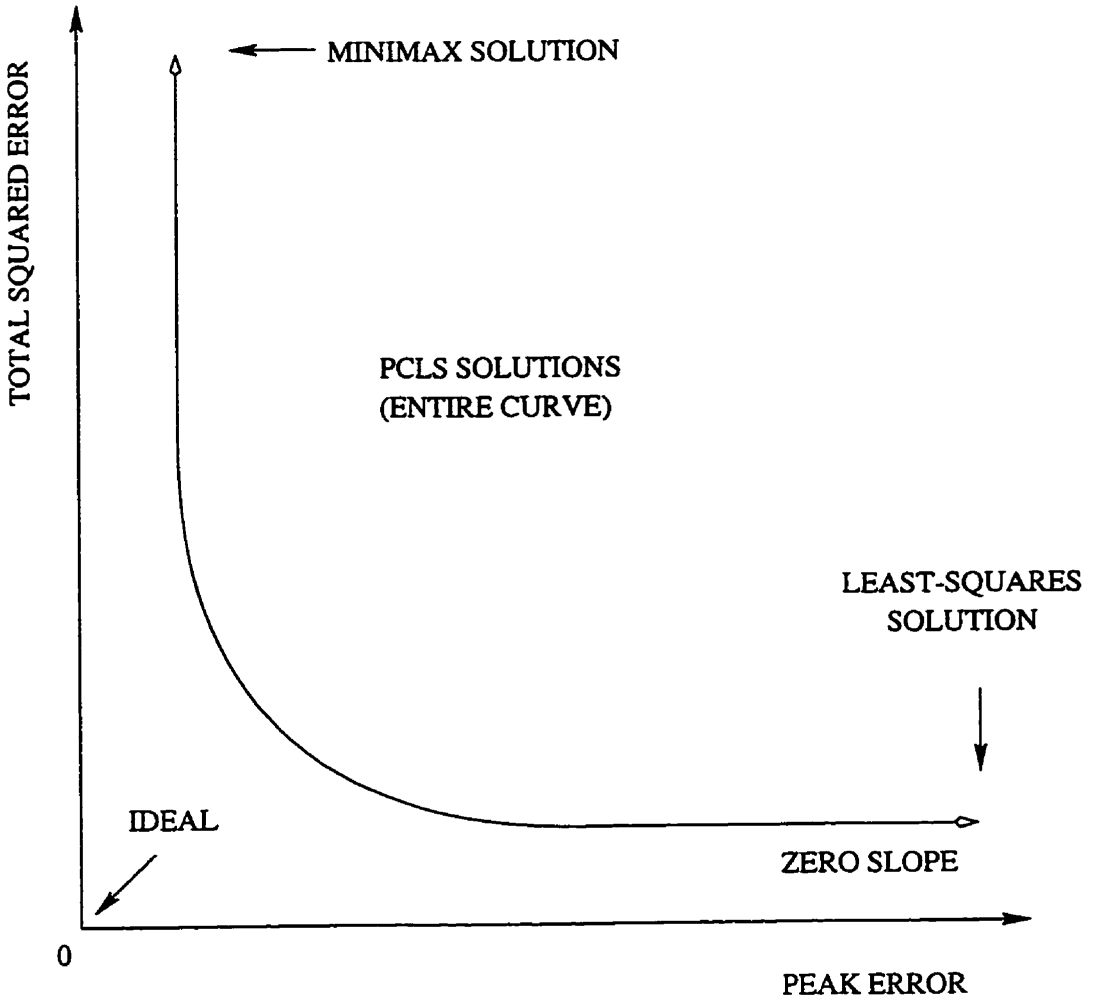


Figure 3.2. Tradeoff between total squared error and peak error.

Starting from the LS solution, a very large reduction in the peak error can be obtained at the expense of a very small increase in the total squared error. Starting from the MM solution, a very large reduction in the total squared error can be obtained at the expense of a very small increase in the peak error. Therefore as mentioned in [32]-[33], LS and MM are inherently inefficient. The primary advantage of the PCLS optimization is the ability to control the trade-off between peak error and total squared error. Second, in most practical applications, it is important for the designer to have the ability to specify inequality constraints on the gains at the band edge frequencies.

Noting these advantages, we believe that in the design of the required half-band filter, instead of using the minimax criterion as was done by Rioul and Duhamel [14], or using the least-squares method used by Cooklev [7], we could use the PCLS optimization to achieve a more efficient design. In [32]-[33], a strategy for PCLS based on the theory of the "multiple exchange algorithm", has been suggested. Most constrained algorithms use a single exchange of active constraints from one iteration to the next. Single exchange algorithms are appropriate for solving general constrained least-squares (CLS) problems, where the constraints are arbitrary. Unfortunately, single exchange algorithms converge very slowly. If a CLS problem includes peak-error constraints on a smooth function, then multiple exchanges improve the rate of convergence. In [34, 35] it has been proven that the generalized multiple exchange algorithm is guaranteed to converge to a unique optimal solution of any feasible positive definite quadratic programming problem. In [32] it was proposed to combine

the multiple exchange and the GI-algorithm to exploit the convergence property of the latter. Also, the GI-algorithm does not require primal feasibility until the last iteration is completed, which makes it more efficient to be combined with the multiple exchange algorithm since most quadratic programming algorithms require primal feasibility at the beginning and end of each iteration.

Since the GI-algorithm forms the core of the method suggested by Adams and Sullivan in [31], we demonstrate the use of the GI-algorithm in conjunction with the matrix inverse problem as suggested by Zarowski [8], in the design of the half-band filter with non-negative frequency response. We observe that this method is far better than those suggested in [9], since this algorithm converges both quickly and accurately.

3.3.3 The Goldfarb-Idnani (GI) Algorithm

We now outline the GI-algorithm. There are certain errors (typographical and omissions) in the algorithm as presented in [5], which have been corrected in this outline. This dual algorithm is of the active set type and is both efficient and numerically stable.

The GI-algorithm is concerned with the strictly convex (positive definite) quadratic programming problem,

$$\min_{x} \quad f(x) \;=\; \frac{1}{2}x^T G x \;+\; a^T x \;+\; f_0 \tag{3.13a}$$

subject to the inequality constraint

$$S(x) \;=\; C^T\, x \;-\; b \;\geq\; 0 \quad , \tag{3.13b}$$

where x and a are n-vectors, G is a n x n symmetric positive definite matrix, C is a n x m matrix, b is a m-vector, and the superscript T denotes transpose.

As already mentioned, the dual algorithm is of the active set type. By active set we mean a subset of the m constraints in Equation (3.13b) that are satisfied as equalities by the current estimate of x of the solution to Equation (3.13a). We shall use W to denote the set $\{1, 2, \cdots, m\}$ of indices of the constraints (3.13b) and $A \subseteq W$ to denote the indices of the active set.

We define a subproblem $P(J)$ to be the Quadratic Programming Problem (QPP) with the objective function 3.13a subject only to the subset of constraints (3.13b) indexed by $J \subset W$. For example $P(\emptyset)$, where $\emptyset$ denotes the empty set, is the problem of finding the unconstrained minimum of (3.13a).

If the solution x of the subproblem $P(J)$ lies on some linearly independent active set constraints indexed by $A \subseteq J$ we call (x, A) a solution-(S) pair. Clearly, if (x, A) is an S-pair for subproblem $P(J)$ it is also an S-pair for the sub-problem $P(A)$.

In order to describe the algorithm, it is necessary to introduce some notation. The matrix of normal vectors of the constraints in the active set indexed by A will be denoted by $\tilde{N}$ (i.e. $\tilde{N}$ is a subset of the coefficients of x in the rows of $S(x)$ in Equation (3.13b), and the cardinality of A will be denoted by q. When the columns of $\tilde{N}$ are linearly independent one can define the operators

$$N^* = (\tilde{N}^T G^{-1} \tilde{N})^{-1} \tilde{N}^T G^{-1} \tag{3.14a}$$

and

$$H = G^{-1}(I - \tilde{N}N^*). \tag{3.14b}$$

3.3.3.1 Dual Algorithm

The algorithm given below conforms to the dual approach and it's details are as follows:

- **Step 0 :** *Find the unconstrained minimum :*

$$x \leftarrow -G^{-1}a,\ f \leftarrow \frac{1}{2}a^T x,\ H \leftarrow G^{-1},\ \tilde{N} \leftarrow \emptyset,\ A \leftarrow \emptyset,\ q \leftarrow 0$$

- **Step 1 :** *Choose a violated constraint, if any :*

  Compute $S_j(x)$ (the row j of Equation (3.13b)), for all $j \in W \setminus A$. If $V = \{j \in W \setminus A \mid S_j(x) < 0\} = \emptyset$, STOP, the current solution x is both feasible and optimal;

  otherwise, choose $p \in V$ and set $n^+ \leftarrow n$ and $u^+ \leftarrow \begin{bmatrix} u \\ 0 \end{bmatrix}$. If $q = 0$, set $u \leftarrow 0$.

- **Step 2 :** *Check for feasibility and determine a new S-pair :*

  (a) *Determine step direction*

  Compute $z = Hn^+$ (the step direction in the primal space) and if $q > 0$, $r = N^* n^+$ (the negative of the step direction in the dual space).

  (b) *Compute step length*

(i) *Partial step length* t_1 (maximum step in dual space without violating dual feasibility). If $r \leq 0$ (i.e.,all elements in vector r are non-positive) or $q = 0$, set $t_1 \leftarrow \infty$, otherwise set

$$t_1 \leftarrow \min_{\substack{r_j > 0 \\ j = 1, \cdots, q}} \left(\frac{u_j^+}{r_j} \right) = \frac{u_l^+}{r_l},$$

where u_j^+ is the jth element of the vector of Lagrange multipliers.

In Step 2(c) below, element $k \in W$ corresponds to the lth element in A.

(ii) *Full step length* t_2 (minimum step in the primal space such that the pth constraint becomes feasible).

If $|z| = 0$, set $t_2 \leftarrow \infty$ otherwise, set $t_2 \leftarrow \frac{-S_p(x)}{z^T n^+}$.

(iii) *Step length* t

Set $t \leftarrow min(t_1, t_2)$.

(c) *Determine new S-pair and take a step*

(i) *No step in primal or dual space* If $t = \infty$, STOP, subproblem and hence Quadratic Programming Problem (QPP) are infeasible.

(ii) *Step in dual space* If $t_2 = \infty$, then $u^+ \leftarrow u^+ + t \begin{bmatrix} -r \\ 1 \end{bmatrix}$, drop constraint k, i.e. set $A \leftarrow A \setminus \{k\}$, $q = q - 1$, drop the lth element of u^+, drop lth column of $\tilde{N}$, update H and N^* using Equation (3.14), and goto Step 2(a).

(iii) *Step in primal and dual space*

Set $x \leftarrow x + tz$, $u^+ \leftarrow u^+ + t\begin{bmatrix} -r \\ 1 \end{bmatrix}$. !! If $t = t_2$ (full set) set $u \leftarrow u^+$, add constraint p; i.e., set $A \leftarrow A \cup \{p\}$, $q = q + 1$, add new constraint to $\tilde{N}$. Goto Step 1.

If $t = t_1$ (partial step) drop constraint k, i.e., set $A \leftarrow A \setminus \{k\}$, $q = q - 1$, drop the lth element of u^+, drop lth column of $\tilde{N}$, update H and N^* using Equation (3.14), and goto Step 2(a).

Some slight modifications have been made to the above algorithm to make it more efficient and to account for round-off errors. They are :

- Instead of choosing any violated constraint as given in Step 1, we choose the most violated constraint, by selecting the most negative value of S from Equation (3.13b). This not only reduces the number of iterations required for convergence, but it is also a good strategy to help prevent numerical instabilities as stated by Goldfarb in [36].

- We have introduced a small perturbation parameter E in the evaluation of $S(x)$ as given in Equation (3.13b), which now becomes :

$$S(\alpha) = C^T\alpha - b - E \geq 0 \Rightarrow C^T\alpha \geq b + E\,.$$

  This has been done to shift the frequency response by an extremely small value E above zero to ensure convergence in certain cases which will be dealt with in the next section.

- Convergence is achieved when all the elements in the set S as given by Equation (3.13b) are greater than or equal to zero while minimizing $f(x)$ in Equation (3.13a). However there may be certain round-off errors which we account for by adding an extremely small tolerance parameter.

3.4 Simulation Results

We now give various examples showing certain successful implementations of our design algorithm and showing that this method is more efficient and faster than the other existing methods [7], [9].

3.4.1 Example No.1

We first show an example that was quoted in [9] for the specifications $x_s = 0.5$, $N = 7$, $L = 1$, $M = 9$, $y = 0.64$ and $E = 0.001$, where M and x are as defined by Equation (3.10), and E is the perturbation parameter. For this example it is clear that $K = 3$ so the optimization is with respect to three parameters. This converges in 8 iterations only, which is a big improvement over the methods given in [9], since for these specifications the MATLAB optimization toolbox simulation resulted in failure to converge. Figure 3.3 shows the result produced by the GI-algorithm. The parameters a and b seen on top of the figure define the stop-band region $[a, b) = [0.5, 1)$, and so $a = x_s$.

In Appendix C is the MATLAB code that implements the GI-algorithm using the routine GI_Algo.m and plots the figures in this section using the routine H_to_plot.m

and Hxplot.m. The supporting routines are in Appendix B, these consist of fact.m, binco.m, u.m, maker.m, makeR.m v0.m, vk.m (which have been taken from [8]) and H.m.

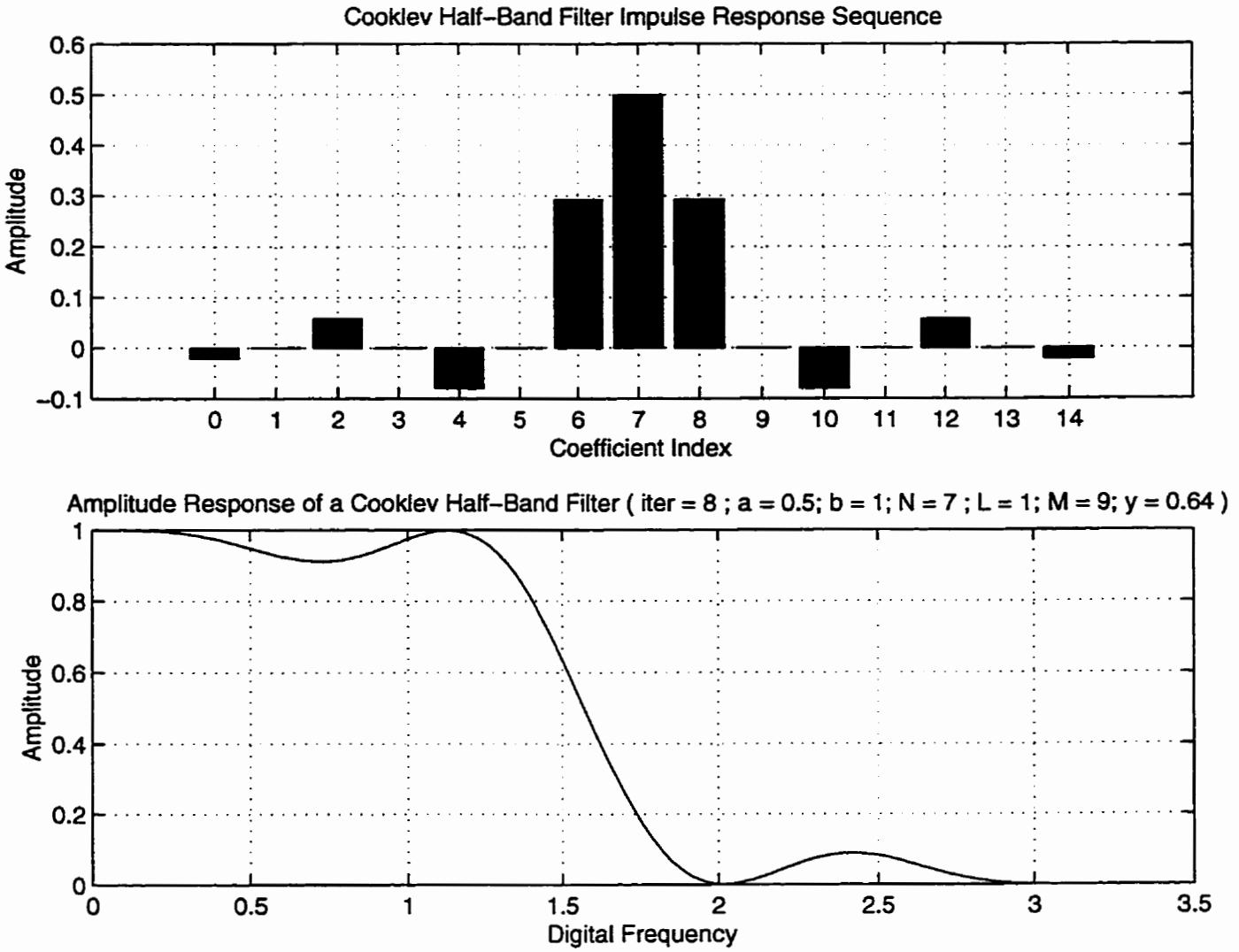


Figure 3.3: Magnitude response plot for a half-band filter produced for the specifications, $x_s = 0.5$, $N = 7$, and $L = 1$.

3.4.2 Example No.2

Figure 3.4 shows an example again that was quoted in [9], for the specifications, $x_s = 0.6$, $N = 7$, $L = 2$, $M = 11$, $y = 0.5$ and $E = 0$. The new algorithm results in convergence in 2 iterations, as compared to the POCs algorithm [9] that resulted in convergence in 75 iterations.

The impulse response sequence of both the filters, one designed using the POCs algorithm (taken from [9]) and the other designed by the new method proposed is tabulated as follows :

k	p_k POCs solution after 75 iterations	p_k GI-Algorithm solution after 2 iterations
0	-0.0164	-0.0152
1	0	0
2	0.0499	0.0476
3	0	0
4	-0.0828	-0.0830
5	0	0
6	0.2993	0.3006
7	0.5000	0.5000
8	0.2993	0.3006
9	0	0
10	-0.0828	-0.0830
11	0	0
12	0.0499	0.0476
13	0	0
14	-0.0164	-0.0152

The impulse response sequence of the filter designed using the POCs solution was found using MATLAB routines that have been described in Appendix D of the report [4], whereas the impulse response sequence of the filter designed using the Goldfarb-Idnani (GI) algorithm was found using the MATLAB routine GI_Algo.m.

This example shows that the POCs method is too slow to converge, hence not very efficient. It must be noted that the above tabulated comparison is risky in the sense that the POCs was not implemented very efficiently in [4] and hence the comparison will not be considered fair.

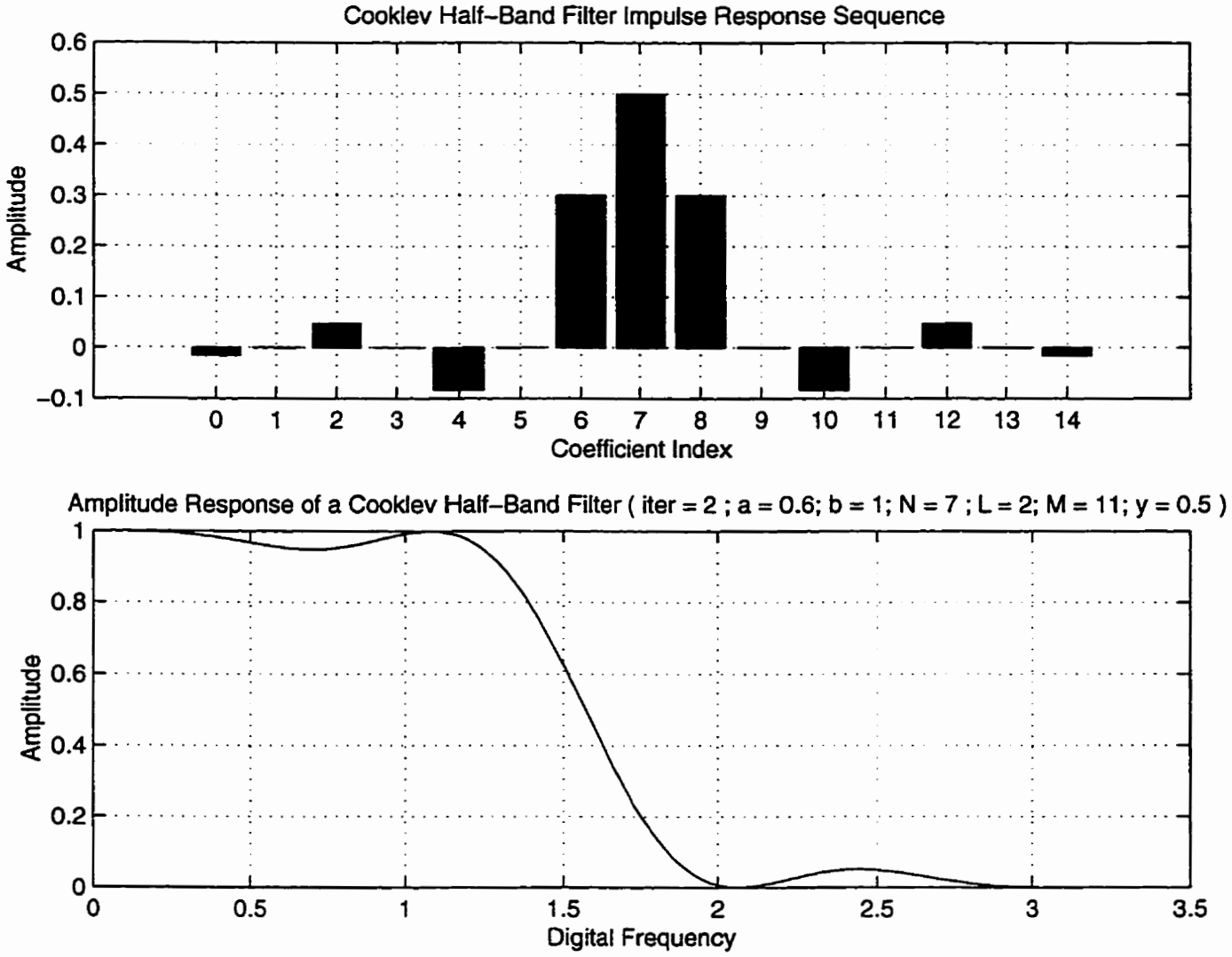


Figure 3.4: Magnitude response plot for a half-band filter produced for the specifications, $x_s = 0.5$, $N = 7$, and $L = 2$.

3.4.3 Example No.3

The plot in Figure 3.5 below illustrates a typical magnitude response for the specifications $x_s = 0.5$, $N = 35$, $L = 16$, $M = 10$, $y = 0.5$ and $E = 0$. Again, for this example it is clear that $K = 2$ so the optimization is with respect to two parameters. This converges in only 3 iterations, which is a big improvement in terms of speed of convergence, compared to the methods given in [9].

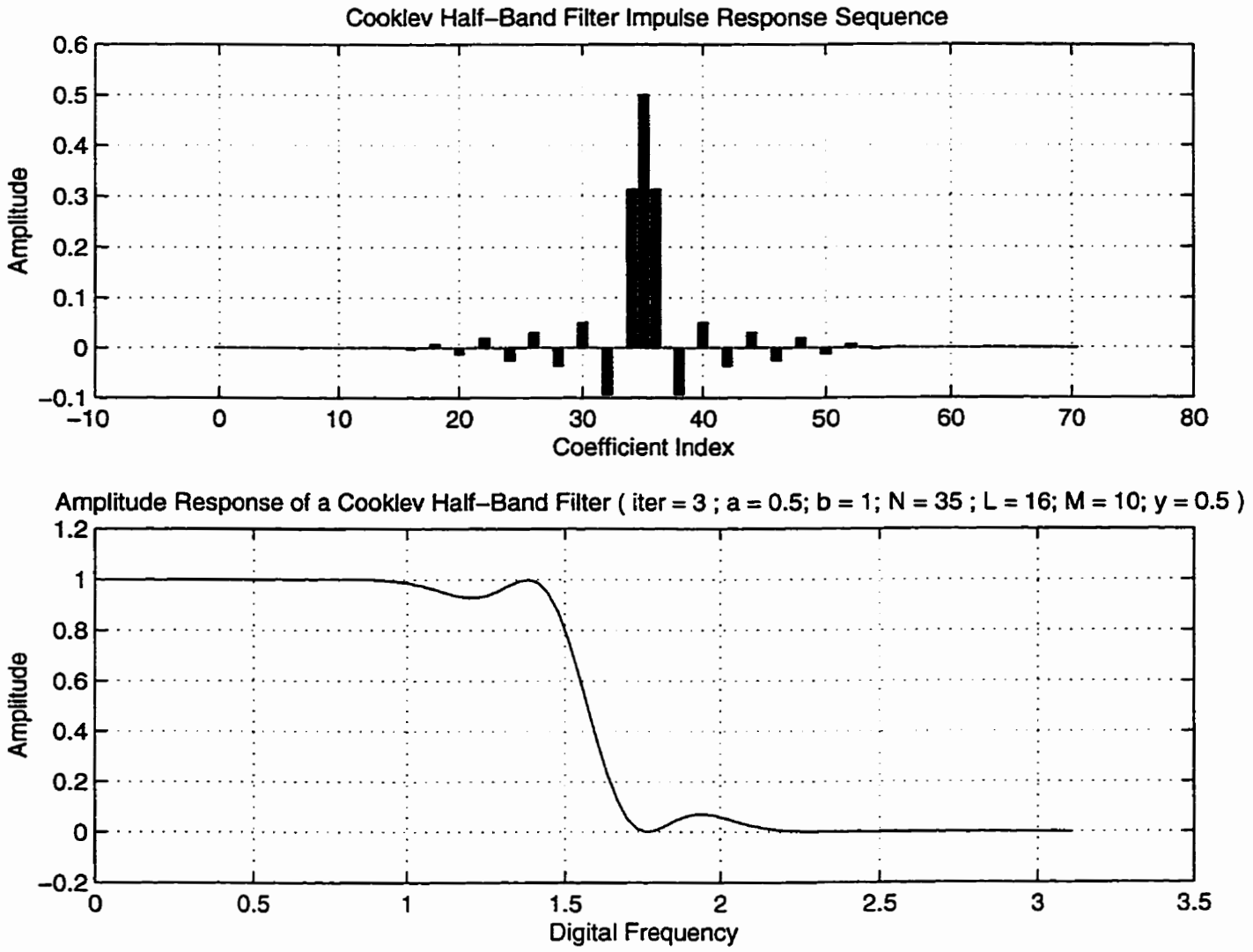


Figure 3.5: Magnitude response plot for a half-band filter produced for the specifications, $x_s = 0.5$, $N = 35$, and $L = 16$.

Figure 3.6 is a plot of the zeros of the half-band filter of Example No.3. The figure shows suitable double transmission zeros as would be appropriate for spectral factorization which will be discussed in the next section. The zero-plots in this section have been plotted using the MATLAB routines zeroplot.m and H_to_plot.m (Appendix C).

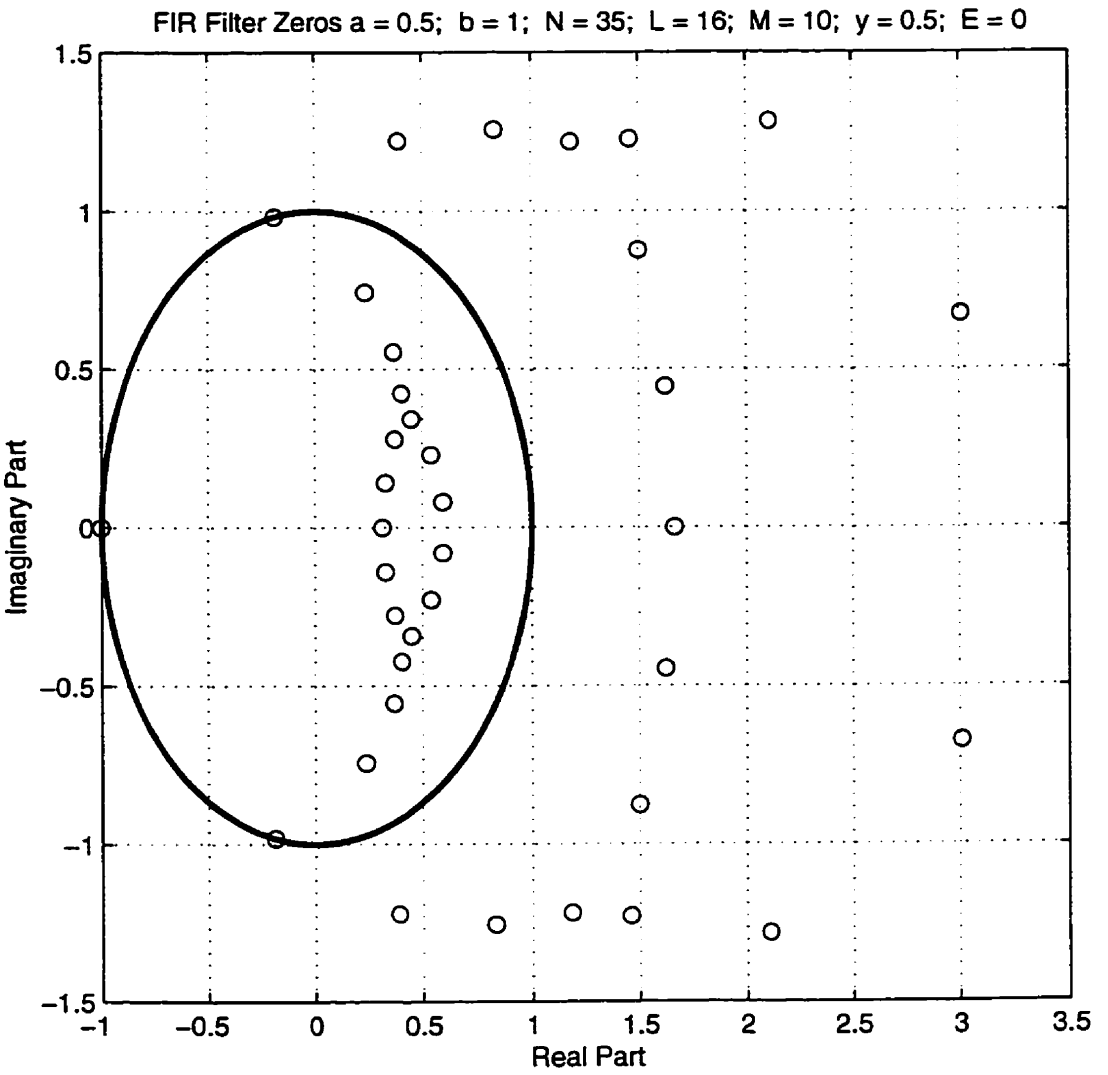


Figure 3.6: Zero plot for the half-band filter (Example No.3) produced by the proposed new algorithm for the specifications $x_s = 0.5$, $N = 35$, $L = 16$, $M = 10$, $y = 0.5$, and $E = 0$,where M and y are as defined by Equation (2.4), and E is the tolerance parameter.

3.4.4 Example No.4

Figure 3.7 shows another example with the specifications $x_s = 0.5$, $N = 23$, $L = 2$, $M = 10$, $y = 0.5$ and $E = 0$. For this example it is clear that $K = 10$ so the optimization is with respect to ten parameters. This converges in 11 iterations. Examples 3 and 4 both show that our design method can be used for high-order filters. A comprehensive list of the specifications for which half-band filters with non-negative frequency response were successfully created for all cases up to $N = 25$ can be found in Appendix D.

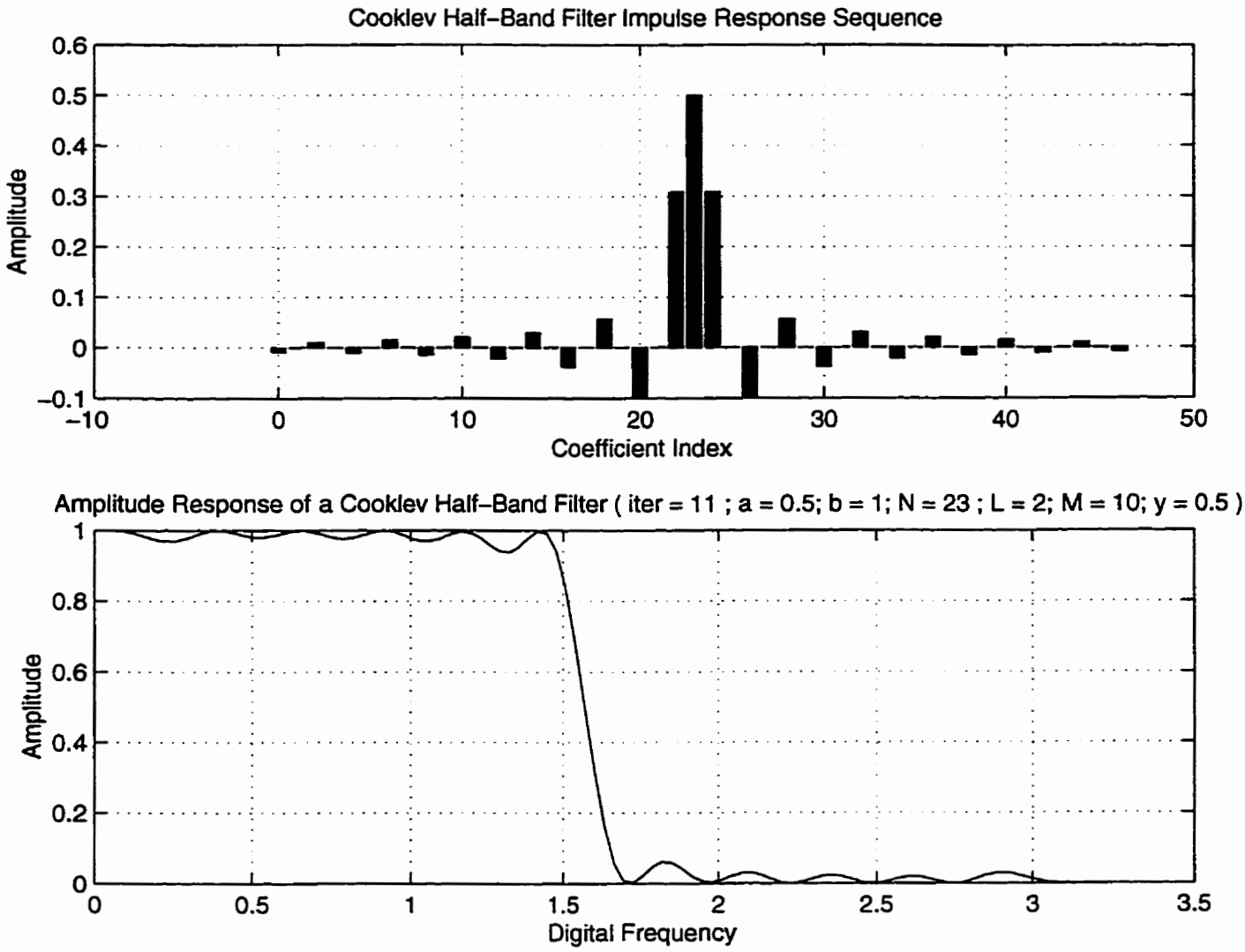


Figure 3.7: Magnitude response plot for a half-band filter produced for the specifications $x_s = 0.5$, $N = 23$, and $L = 2$.

Figure 3.8 is a plot of the zeros of the half-band filter of Example No.4. This figure too shows suitable double transmission zeros on the unit circle as would be appropriate for spectral factorization.

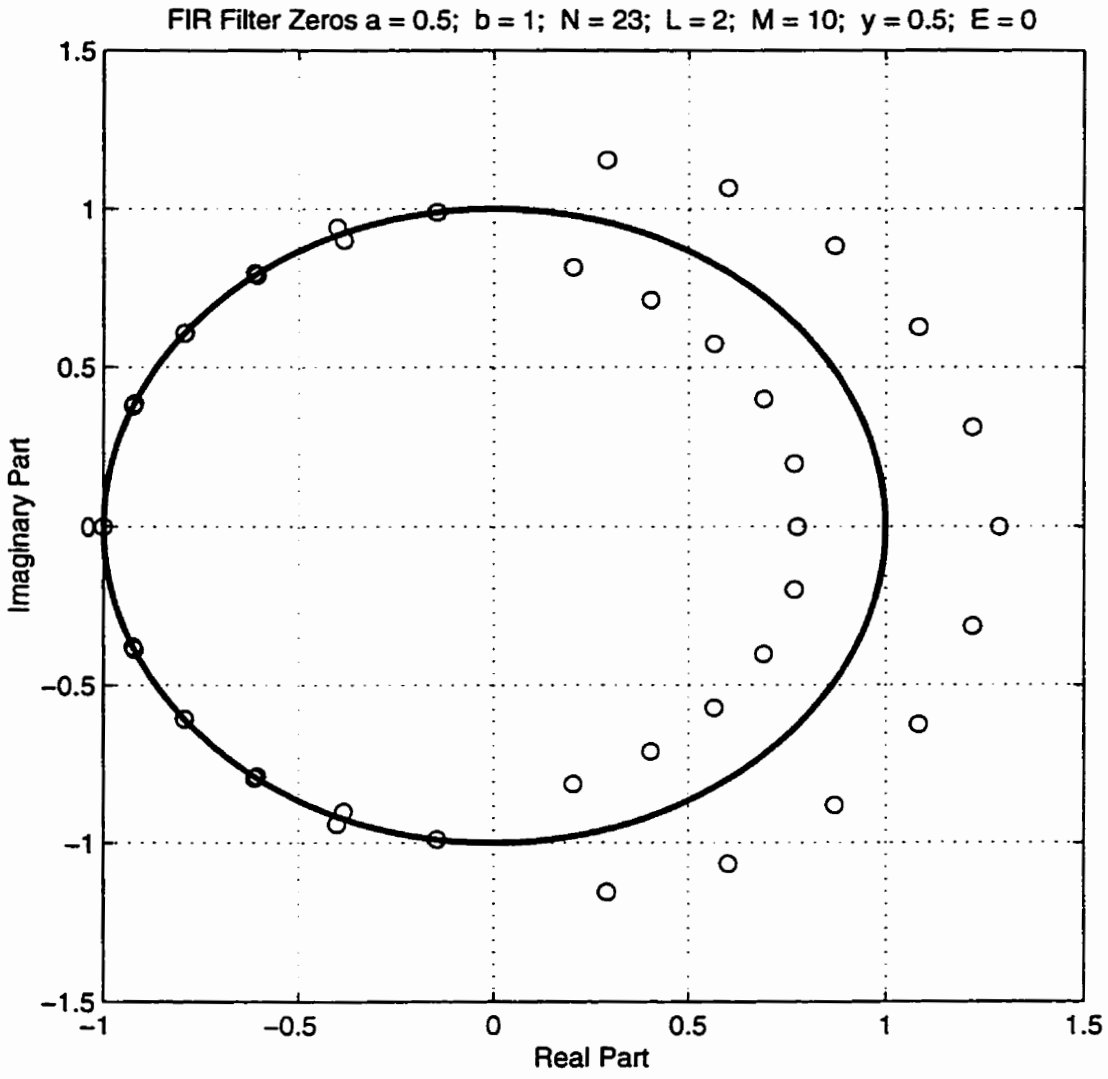


Figure 3.8: Zero plot for the half-band filter (Example No.4) produced by the proposed new algorithm for the specifications $x_s = 0.5$, $N = 23$, $L = 2$, M = 10, y = 0.5 and $E = 0$.

3.4.5 Example No.5

We now show an example where it is observed that in the case when the optimization takes place with respect to an odd number of parameters, if the input specifications are correctly chosen then we can succeed in getting *two additional* zeros at $z = -1$. Figure 3.9 shows the magnitude response of a half-band filter having the following specifications $x_s = 0.5$, $N = 3$, $L = 1$, $M = 11$, $y = 0.5$ and $E = 0.00068175$.

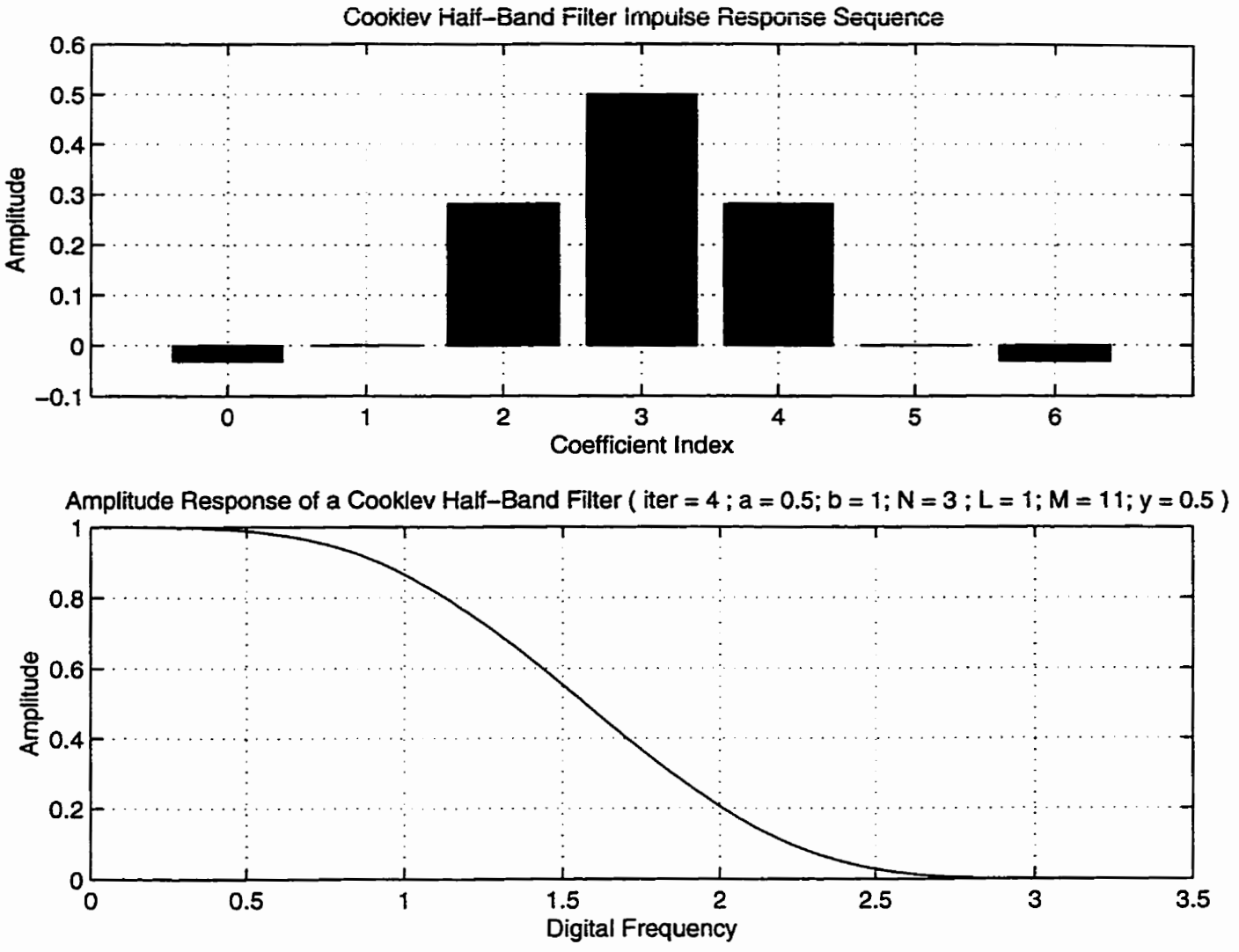


Figure 3.9: Magnitude response plot for a half-band filter produced for the specifications $x_s = 0.5$, $N = 3$, and $L = 1$.

Figure 3.10 is a plot of the zeros of the half-band filter of Example No.5. We observe the presence of two additional zeros at $z = -1$ in this case. The presence of this additional pair of zeros is of importance and will be discussed in the succeeding sections.

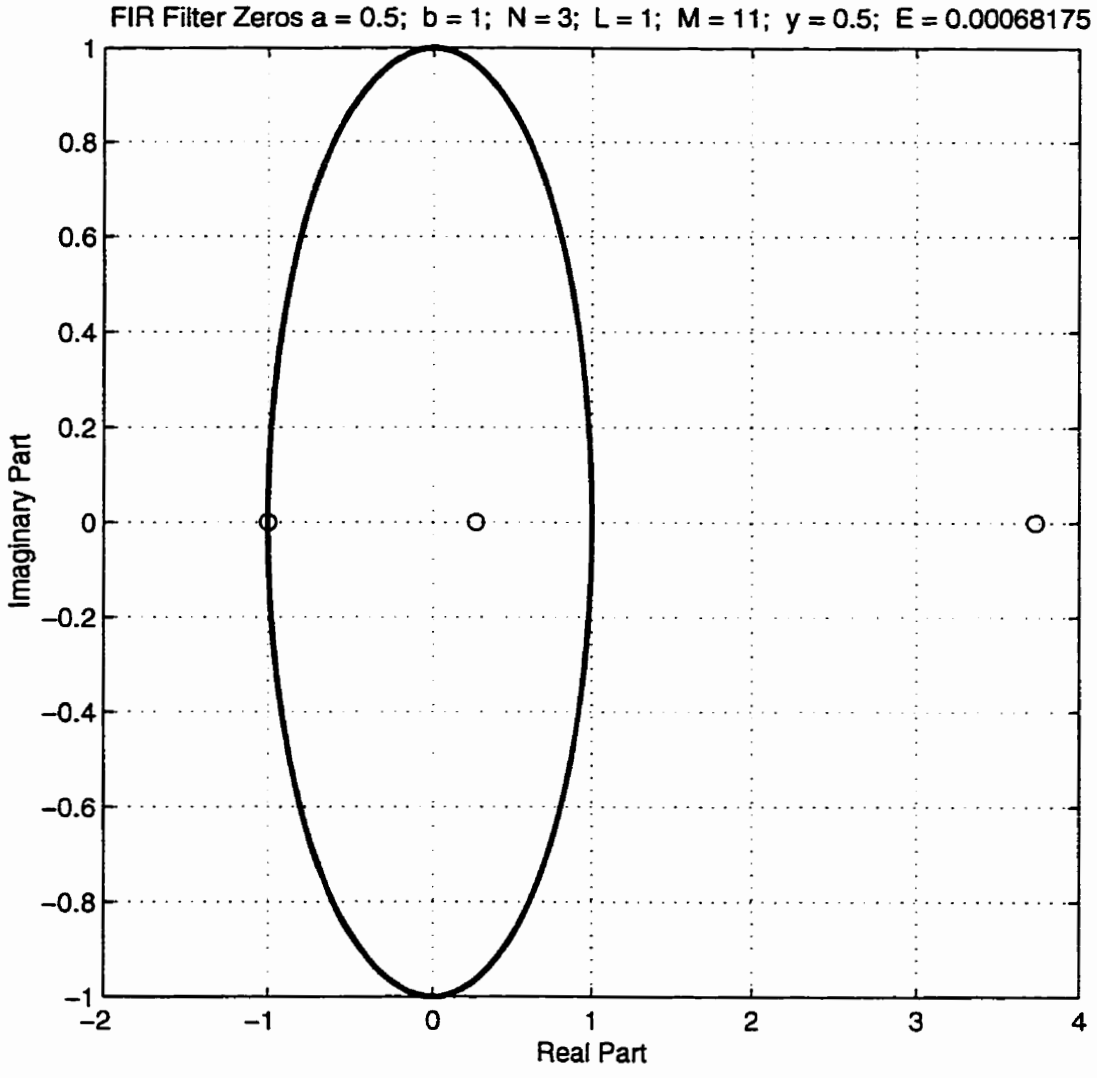


Figure 3.10: Zero plot for the half-band filter (Example No.5) produced by the proposed new algorithm for the specifications $x_s = 0.5$, $N = 3$, $L = 1$, M = 11,$y = 0.5$ and $E = 0.00068175$.

3.4.6 Some observations

In all of the examples in this section, we notice that the GI-algorithm seems to work extremely well for our purposes. It is more efficient in the sense that it is more accurate and it converges very quickly and hence it's computation time is much less than the methods proposed in [9]. We also observe the following :

- As noted in [7],the optimization process occurs with respect to the elements of the vector $\alpha = [\alpha_1\ \alpha_2\ \cdots\ \alpha_K]^T$, where $K = (N+1)/2 - L$. We recall that the half-band filter that results will have $2N + 1$ (N is odd) impulse response coefficients, and it's system function $P(z)$ will have $2L$ zeros at $z = -1$. A large L implies a high regularity. For a solution to the spectral factorization problem to exist, it is stated in [7] (p. 46) that K must be an even number. Using the new design algorithm we notice a phenomenon that is inconsistent with what is stated in [7]. We notice that a solution to the spectral factorization problem exists for all K. In fact in some cases we even manage to get an additional pair of zeros at $z = -1$. And hence the claim made in [7] and a similar claim made in [14] are both inaccurate since they maintain that the technique used in designing the half-band filter works only when the number of coefficients of K are even.

- When K is even, i.e., when the number of elements in the vector α are even, then the number of alternations in the frequency response in the stop band (i.e., the number of times the frequency response in the stop band changes from zero

to a positive value and vice-versa) is exactly equal to K'. When K' is odd, the number of alternations in the frequency response in the stop band is exactly equal to $K' - 1$.

Chapter 4

Spectral Factorization and Orthonormal Wavelets

4.1 Introduction

In Chapter 3 we have discussed a new design algorithm for a half-band filter. To obtain the low pass filter that parametrizes a wavelet, essentially one must spectrally factorize an appropriately designed half-band filter. The basic theory is summarized as follows in Cooklev [1]:

Theorem 4.1 (Cooklev) *[7]* • *To design a two-channel perfect reconstruction (PR) filter bank it is necessary and sufficient (i) to find a $P(z)$ satisfying Equation (2.17), and (ii) factor it as $P(z) = H_0(z)G_0(z)$.*

Proof• The proof of this theorem has been discussed in Herley and Vetterli [37].•

In this theorem filter $H_0(z)$ is low pass. For orthonormal wavelets, $P(z)$ must have a nonnegative frequency response [1]. This is also needed for orthonormal filter banks. More specifically, we wish to find $H(z)$ such that $P(z) = H(z)H(z^{-1})$. The

[1]The necessity of this should be apparent from considering the function $R(x)$ in Chui [19], pp. 229-230.

Féjer-Riesz theorem (see [10], p. 157) guarantees the existence of the low pass factor $H(z)$. The theory in Chapter 3 shows how to find $P(z)$, and we see that $H(z)$ is a spectral factor of $P(z)$.

In this chapter we summarize different methods of spectral factorization and determine the most suitable one. Having found the spectral factor we then discuss an iterative procedure to construct orthonormal wavelets and present some simulation results, which again authenticates our claim that the new design algorithm is superior and more efficient than the methods that were used before in [7], [9]. We substantiate our claims by comparing the regularity and the frequency response of the scaling function constructed using the new design algorithm with that of the widely used Daubechies scaling functions. We also compare the frequency response of the scaling functions obtained by the new design algorithm with the ones designed by Cooklev's original design method.

4.2 Spectral factorization for the Design of Two-channel Orthonormal Filter Banks

Theorem 4.1 illustrates that the design of a two-channel orthonormal filter-bank consists of essentially two steps : obtaining $P(z) = H(z)H(z^{-1})$ (which we call the product filter) which is the analytic continuation of a nonnegative magnitude response function of a half-band filter on the unit circle, and then finding $H(z)$ by spectral factorization. In general, no solution exists in closed form. The spectral factor is not unique, and we can find all possible solutions by finding the zeros of $P(z)$

and grouping them appropriately. We are interested in the minimum phase spectral factor, since it is unique. We now describe some commonly used spectral factorization methods. It must be noted that we are dealing with half-band filters having only real coefficients.

4.2.1 Spectral Factorization by Completely Factoring a Polynomial

The most straightforward method of spectral factorization is to perform a complete factorization of the polynomial. The advantages are :

- Complete factorization of a polynomial works very well for polynomials of low order.
- Any spectral factor (not only the minimum phase one, i.e., having no zeros outside the unit circle) can be found.

The disadvantage of this method, however, is that for higher order polynomials, this is slow and/or numerically unreliable. While the zeros of the polynomial can be found, the numerical error can be quite significant, especially when the impulse response coefficients are computed from these zeros.

4.2.2 Spectral Factorization Using the Cepstrum

This is a method for finding the unique minimum-phase spectral factor without finding the zeros of the polynomial. The technique uses some fundamental properties of the complex cepstrum of a sequence, which is the inverse z-transform of the natural logarithm of the z-transform. The idea is to convert multiplication $P(z) = H(z)H(z^{-1})$

into addition. Formally, $\log\left(\sum p(n)z^{-n}\right)$ (where $p(n)$ is the impulse response of the product filter $P(z)$) is easily separated into positive and negative powers of z. The easy separation into $\log H(z^{-1}) + \log H(z)$ is the key advantage of this method. Let the sequence $l(n)$ be the complex cepstrum of $p(n)$ [10]. The series for $L(z)$ converges in an annulus of the complex plane. However, if a root of $P(z)$ is present on the unit circle, then $L(z) = \log P(z)$ will be infinite at that root and the series would not converge. Hence, this method fails in the case when zeros are present on the unit circle.

4.3 Bauer's Spectral Factorization and it's Suitability

There are significant practical difficulties in computing the spectral factorization of $P(z)$, especially when $P(z)$ has zeros on the unit circle, which is a frequent occurrence. Half-band filters with a non-negative frequency response generally have multiple zeros on the unit circle, other than those imposed at $z = -1$.

An efficient matrix spectral factorization algorithm is necessary that can handle zeros on the unit circle and also one that is sufficiently reliable for polynomials having a high degree (i.e., filter's having large support). We observe from the various examples provided in the previous chapter that we do have zeros on the unit circle. Some of the widely used spectral factorization algorithms mentioned in the previous section cannot handle such zeros. Cooklev [7] suggested the use of the Bauer-type factorization of Youla and Kazanjian [38], because it can handle zeros on the unit

circle. We therefore use the same algorithm. An outline of the Bauer factorization has been given in Cooklev [39] and is briefly described here. It must be noted here that the Bauer method is slower (though perhaps it can be sped up using fast Toeplitz factorizers) and less accurate as compared to the other popularly used algorithms.

The Bauer method is based on Cholesky factorization of a banded Toeplitz correlation matrix. The nonzero elements in the rows of the Cholesky factor converge to the minimum phase solution. We restate the problem once again:
Given $P(e^{j\omega}) \geq 0$ for all real ω; find a polynomial $H(z)$ that satisfies $P(z) = H(z)H(z^{-1})$. This is a problem of spectral factorization of scalar polynomials that are nonnegative on the unit circle.

It is clear that if the polynomial $P(z)$ is Hermitian (i.e.,$P(z) = P^H(z)$ where $P^H(z)$ is the transpose conjugate of $P(z)$ obtained by conjugating every element of $P^T(z)$, the transpose of $P(z)$), then the Toeplitz matrices

$$T_m = \begin{pmatrix} p(0) & p(1) & \cdots & p(m) \\ p(-1) & p(0) & \cdots & p(m-1) \\ \vdots & \vdots & \vdots & \vdots \\ p(-m) & p(-m+1) & \cdots & p(0) \end{pmatrix} \tag{4.1}$$

of respective sizes $(m+1)$ x $(m+1)$, where $m > N$, are Hermitian and $p(k) = p^*(-k)$. The choice of m influences the numerical precision and should be as large as possible. For polynomials with real coefficients (as in our case) T_m is a real symmetric matrix. The matrices T_m are also nonnegative definite.

The coefficients of the minimum phase factor can be determined by means of the following two-step procedure:

- **Step 1 :** For every $m > N$ calculate the unique Cholesky factorization

$$T_m = L_m^H L_m \tag{4.2}$$

where

$$L_m = \begin{pmatrix} l_{00}^m & 0 & \cdots & 0 \\ l_{10}^m & l_{11}^m & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ l_{m0}^m & l_{m1}^m & \cdots & l_{mm}^m \end{pmatrix} \tag{4.3}$$

is lower triangular with positive diagonal entries.

- **Step 2 :** For every fixed r and k, $r \geq k \geq 0$,

$$\lim_{m \to \infty} l_{r0}^m = h(r) \tag{4.4}$$

These two steps have been presented in [39], and represent the algorithm of Bauer.

4.4 The Interpolatory Graphical Display Algorithm (IGDA)

We have enumerated the basic steps leading to the design of a two-channel orthonormal filter-bank in the previous section. The main objective of this is to design orthonormal wavelets. There are two main approaches to computing the wavelet and scaling functions. One is an iterative procedure sometimes called the cascade algorithm [40], and the other is the Interpolatory Graphical Display Algorithm (IGDA) as given in [19] and [20], which is also called dyadic expansion [40]. An advantage [41] of the IGDA over the cascade algorithm is that, in principle, it gives the exact answer in a finite number of steps, though only on dyadic points, i.e., $2^{-j}\mathbf{Z} = \{k/2^j \mid k \in \mathbf{Z}\}$,

where $\mathbf{Z}$ is the set of integers, and $j \geq 0$. Noting this advantage we use the IGDA to construct the wavelet function from the low pass filter coefficients $h(n)$ of $H(z)$ that are obtained after spectral factorizing the product filter.

4.5 Simulation Results

Now using Bauer's method for spectrally factorizing the product filter $P(z)$ and using the IGDA, we present some simulation results which will confirm our claim that the new design algorithm produces better results than Cooklev's approach and we will also compare the regularity and the frequency response characteristics of the new scaling functions with the well known Daubechies scaling functions [4]. The first example illustrates a comparison of the scaling and wavelet function constructed using Cooklev's original method [7] with their counterparts constructed using the new design method.

4.5.1 Example No.1

Figures 4.1 and 4.2 are the plots of the scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter having the specifications $x_S = 0.5$, $N = 17$ and $L = 7$ using Cooklev's design and the new design algorithm, respectively. We observe that the scaling and wavelet function obtained using the new design algorithm is "*visually smoother*" than the one obtained using Cooklev's method, this demonstrates that unlike Cooklev's method the new design algorithm leads to the convergence of the magnitude response of

the product filter (as seen in the previous chapter), which in turn is responsible for the construction of smoother scaling and wavelet functions. Hence, the new design algorithm is superior to Cooklev's approach.

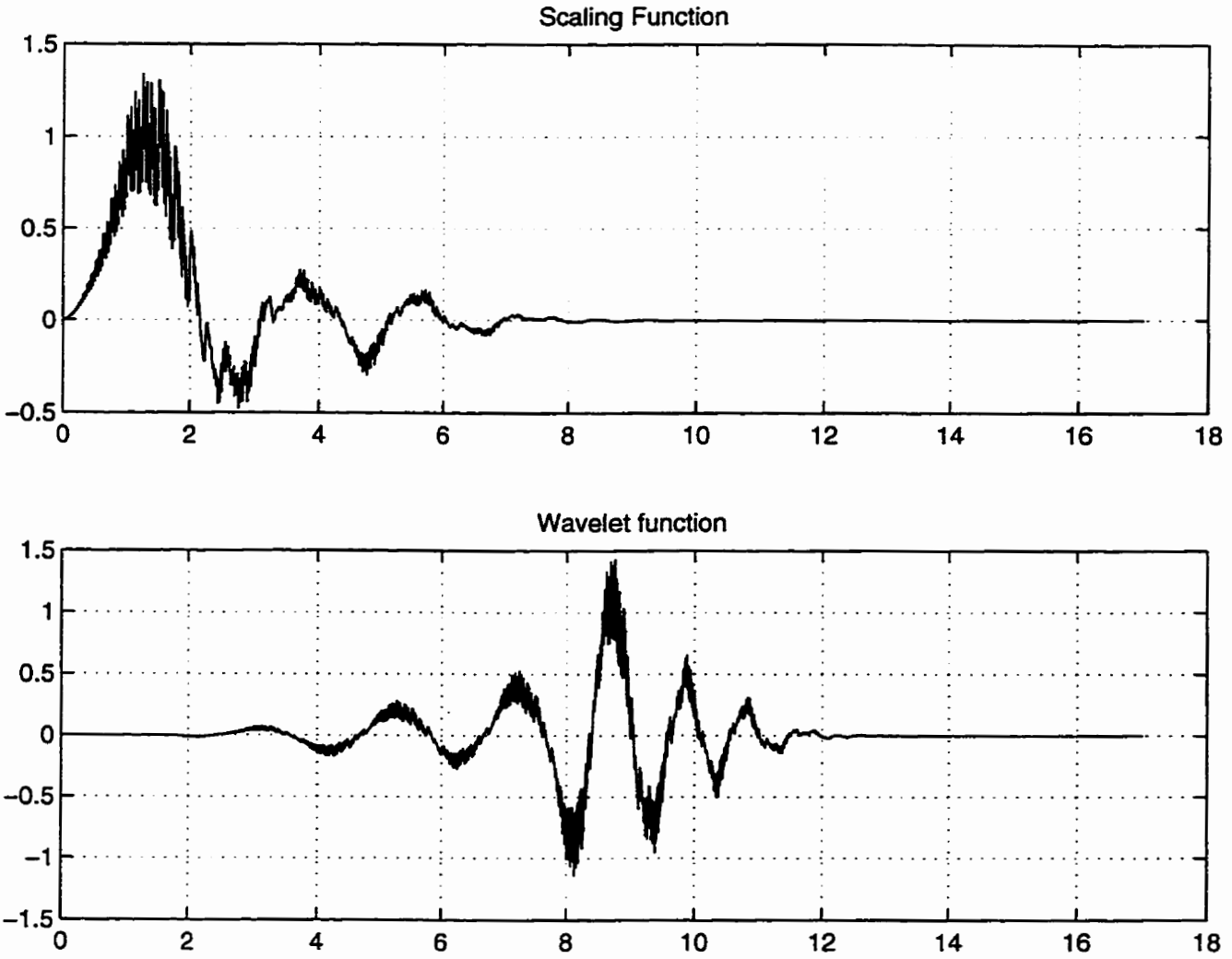


Figure 4.1: Scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter using Cooklev's method, having the specifications $x_s = 0.5$, $N = 17$, and $L = 7$.

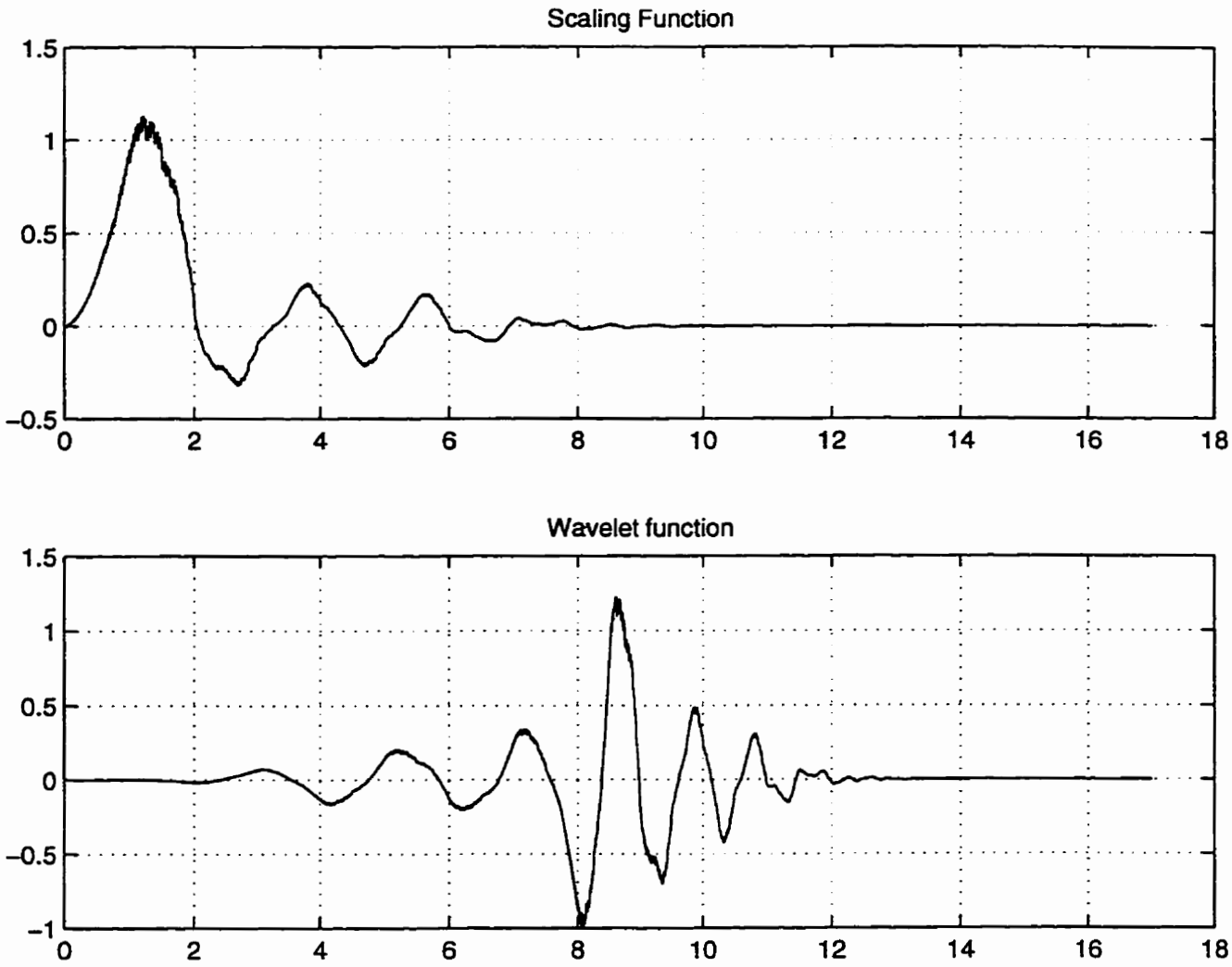


Figure 4.2: Scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter using the new design algorithm, having the specifications $x_s = 0.5$, $N = 17$, L = 7, $M = 11$, $y = .5$ and $E = 0$.

4.5.2 Example No.2

This example shows that the new design algorithm is also suitable for filters having large support since it leads to smooth scaling and wavelet functions. Figure 4.3 is the plot of the scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter having the specifications $x_s = 0.5$, $N = 35$ and $L = 16$ (Example No. 3 of Chapter 3). This example authenticates the claim in [4] that the larger the support length the smoother the wavelet.

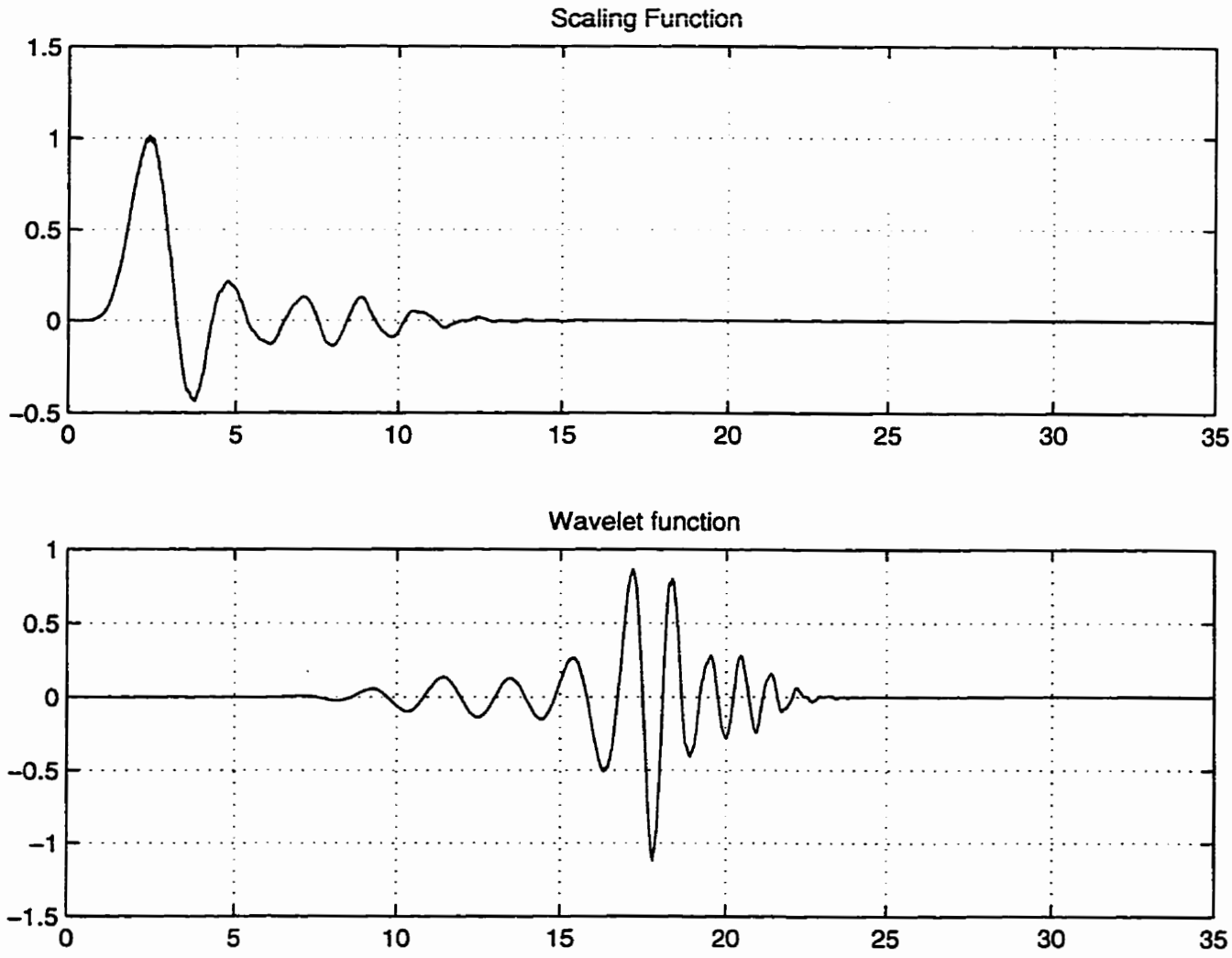


Figure 4.3: Scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter using the new design algorithm, having the specifications $x_s = 0.5$, $N = 35$, $L = 16$, $M = 10$, $y = .5$ and $E = 0$.

4.5.3 Example No.3

Figure 4.4 is a plot of the scaling and wavelet functions derived by spectrally factorizing the product filter of Example No.5 of Chapter 3. This example is particularly interesting since it looks exactly the same as the 4-tap Daubechies scaling and wavelet functions. It is also observed that as the size m of the Toeplitz matrix representation of the product filter $P(z)$ given by Equation (4.1) (used in the spectral factorizer) increases, the new scaling and wavelet function tends to the Daubechies scaling and wavelet functions. In this case the size of the matrix T_m is 300. It must be noted here that this is an example in which there is an extra pair of zeros at $z = -1$ and hence it has similarity to it's Daubechies counterpart (since not only is the support length the same, but the number of zeros at $z = -1$ is also the same).

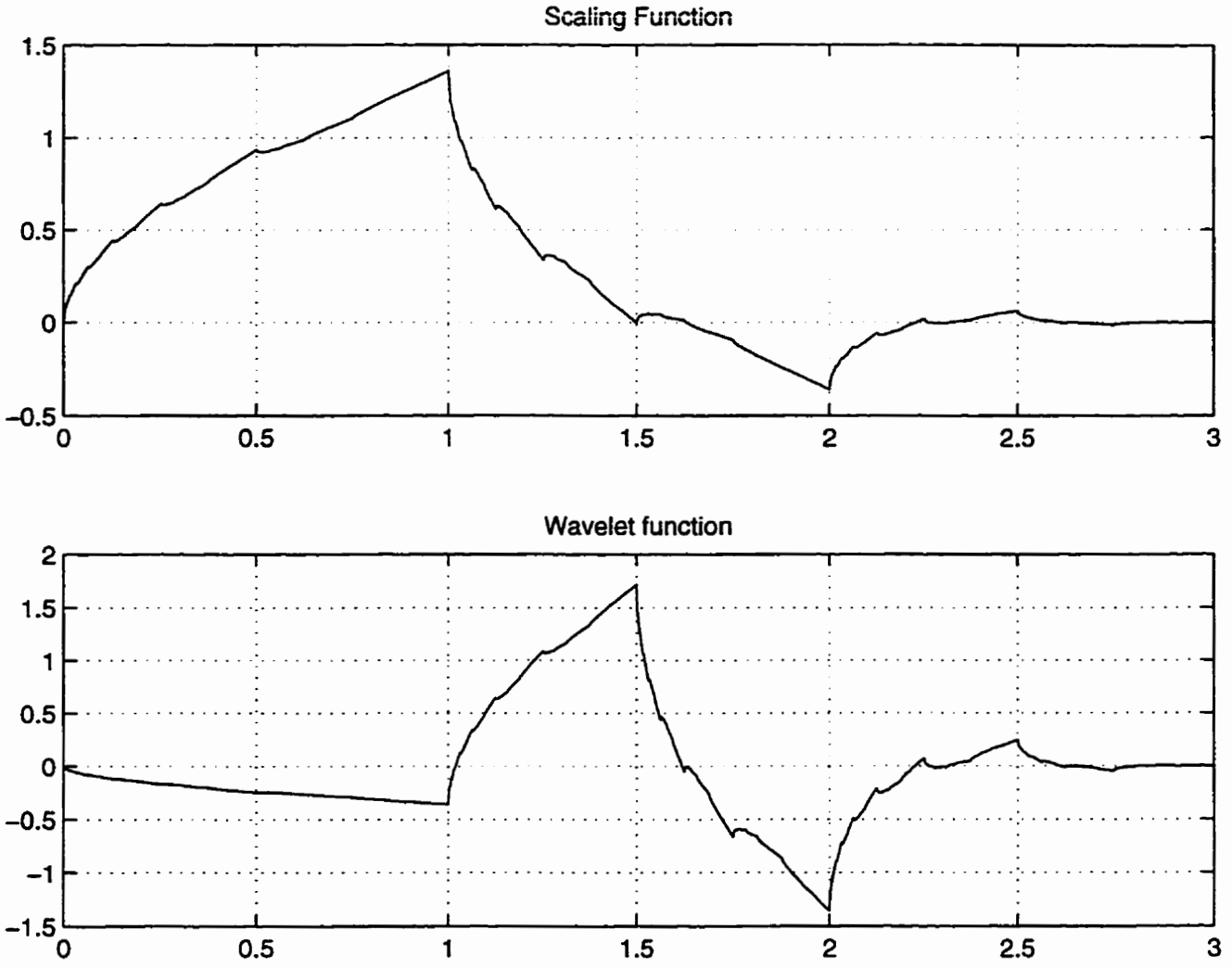


Figure 4.4: Scaling and wavelet functions constructed from the low-pass filter derived by spectrally factorizing the product filter using the new design algorithm, having the specifications $x_s = 0.5$, $N = 3$, $L = 1$, $M = 11$, $y = .5$ and $E = 0.00068175$.

4.6 Regularity

In this section we introduce an important property relating to the wavelet function. The continuous time measure of wavelet smoothness is regularity. Since the wavelet is determined from the scaling function by means of high pass filter taps [4], it is the smoothness of the scaling function (infinitely iterated low pass filter) which determines the smoothness of the overall wavelet system. Smooth wavelet bases are deemed to be important for several applications, and particularly for image compression where the goal is to limit spurious artifacts due to non-smooth basis functions in the presence of quantization of the individual subbands.

In Chapter 2, Section 2.7.2, we claimed that the non-convergence problem of the magnitude response of the half-band filter would lead to irregular wavelets, which has been shown in Section 4.5.1. In this section we compare the regularity of the wavelets constructed using the Daubechies method [4], to the new method that we have designed.

Daubechies [4] showed that the L-vanishing moment construction led to scaling functions of arbitrary differentiability, if L is taken large enough (L is the number of zeros at $z = -1$). Another refined tool for measuring the regularity (differentability) of the scaling function is the Sobolev notion of differentiability [42], [43].

Given a real number s, the Sobolev space $\mathcal{H}^s$ is defined by

$$\mathcal{H}^s = \{\hat{\phi}(\omega) \mid \int_{\mathbf{R}} |\hat{\phi}(\omega)|^2 (1 + |\omega|^2)^s \, d\omega < \infty\} , \tag{4.5}$$

where $\hat{\phi}(\omega)$ is the Fourier transform of the scaling function. (Note that $\hat{\phi}(\omega) =$

$\Phi(\omega)$.) It has been shown [42], [43] and [44] that the Sobolev smoothness of the scaling function is determined by the maximum eigenvalue of a finite dimensional linear operator T associated with the coefficient sequence $h(n)$ of the low pass filter (i.e., $H(z)$ factor of the product filter $P(z)$).

As described in [42], [43] and [10], the explicit formula of the Sobolev smoothness $s(\phi)$ of the scaling function ϕ is given by

$$s(\phi) = -log_4(|\lambda_{max}(T)|) \ , \tag{4.6}$$

where $T = (\downarrow 2)2HH^T$ and $\lambda_{max}(T)$ is the maximum eigenvalue of T excluding $\lambda = 1, \cdots \left(\frac{1}{2}\right)^{2L-1}$.

Now using Equation (4.6) we compare the regularity of the wavelet system based on both the design methods mentioned above. The results are tabulated for a few examples with specifications as follows:

Filter length	Daubechies design $s(\phi)$	New design $s(\phi)$
4 tap	1	1
6 tap	1.415	1.42
8 tap	1.775	1.7809
10 tap	2.096	2.0535
12 tap	2.388	2.3613
14 tap	2.658	2.4780

The following points are worth observing:

- Equation (4.5) is of importance since it signifies that the factor $(1 + |\omega|^2)^s$ grows when ω goes to $\pm\infty$, so for the integral to be finite, $|\hat{\phi}(\omega)|^2$ must be decreasing sufficiently rapidly with $|\omega|$. This also tells us about the smoothness of $\phi(t)$: If the integral in Equation (4.5) is finite then by Parseval's relation

the sth derivative of $\phi(t)$ is an $\mathbf{L}^2$-function. Relating this back to ordinary derivatives, there is a theorem called the Sobolev embedding theorem that says if the sth derivative of $\phi(t)$ is in $\mathbf{L}^2(\mathbf{R})$, then $\phi(t)$ is $\lfloor(s-\frac{1}{2})\rfloor$ times differentiable (differentiability implying smoothness).

- The Sobolev regularity of the scaling functions constructed using the new design in the 4-tap, the 6-tap and the 8-tap (where tap refers to the filter coefficients) cases is equal to or greater than those obtained using the Daubechies construction method. The presence of an additional pair of zeros at $z = -1$, as noted in Section V, is the main cause of this.

- Even though the half-band filter is designed in such a way that it will always have one zero less at $z = -1$ as compared to the Daubechies filters (except in certain cases where additional zeros can be obtained when optimization takes place with respect to an odd number of parameters as seen earlier), we notice that the scaling function obtained using the new design algorithm is quite regular.

- The size of the Toeplitz matrix T_m given in Equation (4.1), representing the product filters used in the above cases for the spectral factorization (using the Bauer method) was $m = 300$.

4.7 Additional Observations

In this section we make some observations regarding the frequency spectrum of the scaling functions obtained by using the GI-algorithm and compare it with the well-

known Daubechies scaling functions [4] and to the scaling functions obtained using Cooklev's algorithm. In communications, especially in waveform design, it is important to have functions that have spectra with steep roll-off (narrow transition band), and have low side lobes, to avoid adjacent channel interference.

It is interesting to compare the frequency response of the Daubechies 4-tap scaling function with that of the 4-tap scaling function obtained using the new algorithm, because both the scaling functions not only appear to be very alike, they also have the same Sobolev regularity. Figure 4.5 shows this comparison. In this case, we observe that the two frequency responses overlap perfectly. The Fourier transform of the scaling function $\phi(t)$ was computed using

$$\hat{\phi}(\omega) = \prod_{k=1}^{\infty} H(e^{\frac{j\omega}{2^k}}) \ , \tag{4.7}$$

where

$$H(e^{j\omega}) = \frac{1}{2} \sum_{k=0}^{N} h(k) e^{-j\omega k} \ . \tag{4.8}$$

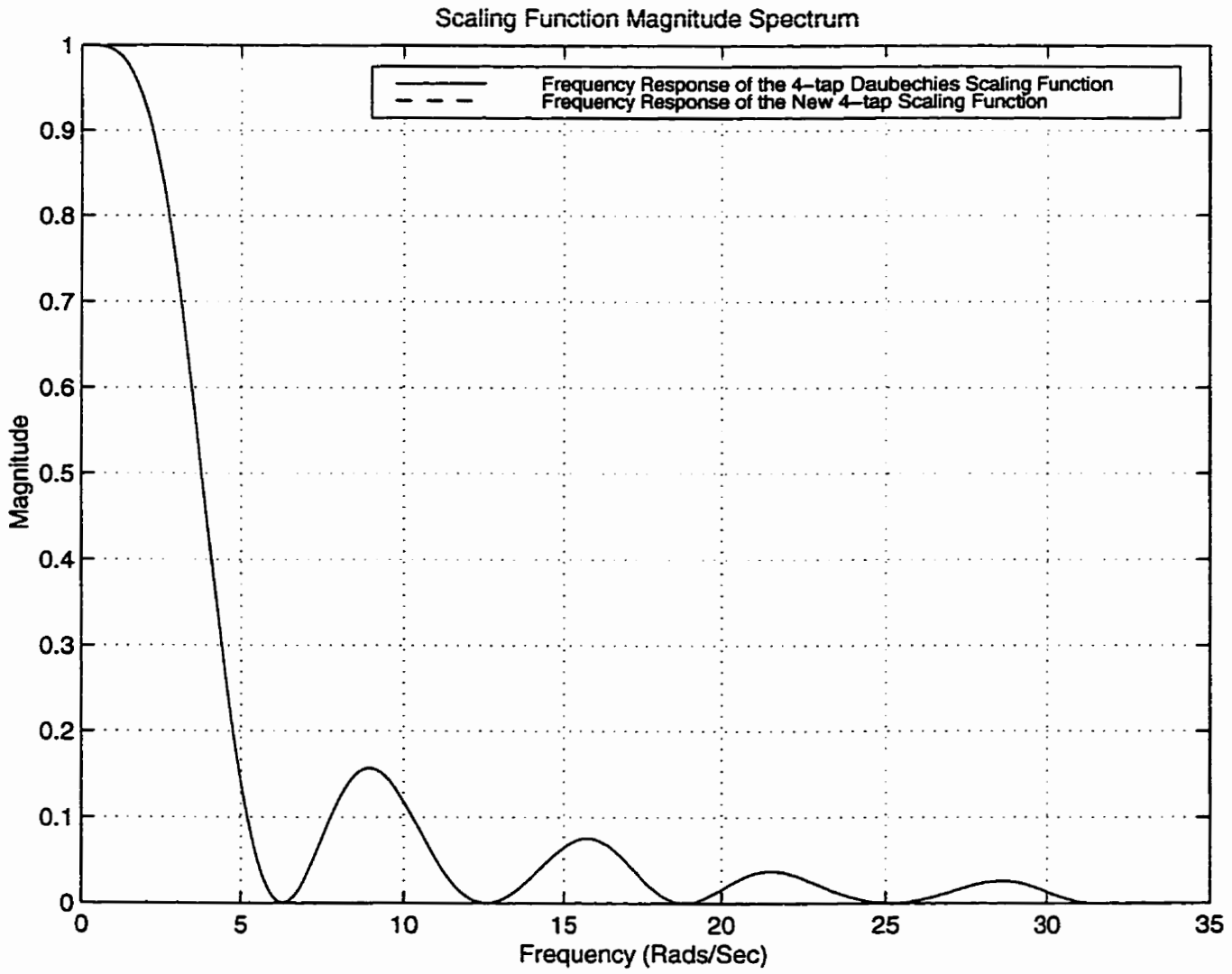


Figure 4.5: Comparison between the spectrum of the scaling function of the Daubechies 4-tap scaling function and the 4-tap scaling function obtained using the new design algorithm.

Figure 4.6 shows the comparison between the spectra of the 8-tap scaling function obtained using Cooklev's design [7] and the scaling function obtained using the GI-algorithm. Here we notice that the spectrum obtained using the GI-algorithm is superior to that obtained using Cooklev's method in terms of having much lower side lobes. This figure is evidence of the efficiency of the new design algorithm over Cooklev's method.

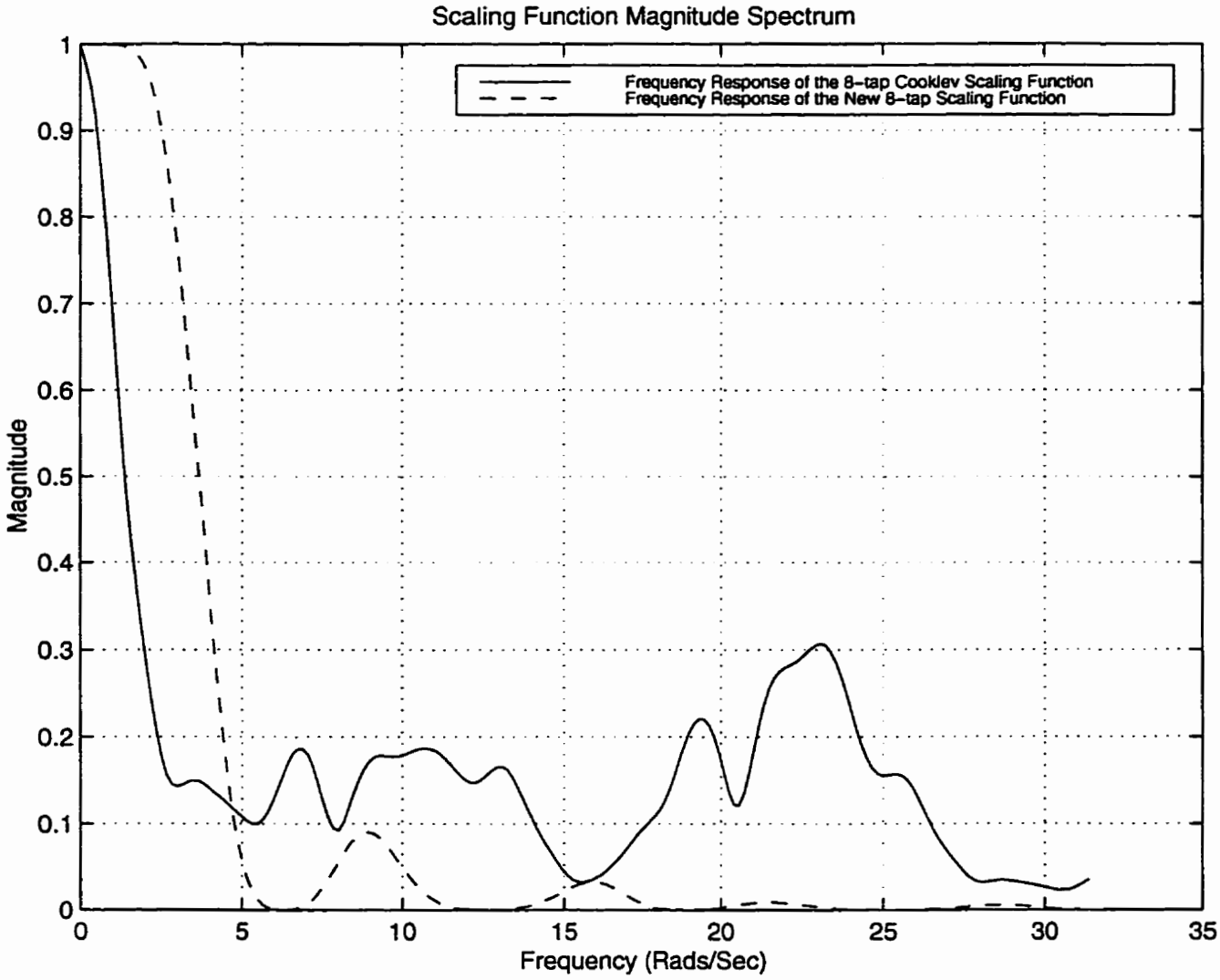


Figure 4.6: Comparison between the spectrum of the 8-tap scaling function obtained using Cooklev's design and that obtained using the new design algorithm.

4.8 Conclusions

In this section we observed that the scaling and wavelet functions constructed using the new design algorithm are *smoother* than those constructed by Cooklev's method for the same specifications. An important point to be noted here is that using the new algorithm it is possible in certain cases, when optimization takes place with respect to an odd number of parameters, to achieve an additional pair of zeros at $z = -1$. As a result of this phenomenon we actually obtain scaling functions that not only are more regular than those of it's Daubechies counterparts, but they also have similar frequency responses.

Chapter 5

Conclusions and Suggestions for Future Research

5.1 Introduction

> " **In reasearch, the horizon recedes as we advance, and is no nearer at 60 than it was at 20. As the power of endurance weakens with age, the urgency of pursuit grows more intense...and research is always incomplete** "
>
> - Mark Pattison (1875)

This thesis presented an algorithm for the design of half-band filters which was then used to construct orthonormal wavelets and it completely eliminated the limitations in Cooklev's method.

In this chapter we first summarize the thesis, and then along with our conclusions we provide some suggestions for future research.

5.2 Summary and Conclusions

We first introduced a general overview of the concepts of *wavelets* and *multiresolution analysis* in Chapter 2. We presented a detailed outline of Cooklev's half-band filter design method via Bernstein polynomial expansions. We also stated the limitations of Cooklev's method, namely *zero splitting* and *non-convergence* of the magnitude response of the *product filter.*

The most important contribution of our theory is presented in Chapter 3. As mentioned earlier, the main aim of this thesis was to eliminate the limitations that existed in Cooklev's method. We completely eliminate the *zero splitting* problem by factoring out the zeros at $z = -1$, and the problem of *non-convergence* of the magnitude response of the *product filter* was eliminated by using the Goldfarb-Idnani (GI) Dual Algorithm for the optimization problem. We observed from the various simulation results provided that not only does the GI-algorithm guarantee convergence of the magnitude response of the *product filter*, but it also leads to another contribution of this thesis: the rebuttal of the claims made in [7] and [14], that the technique used in designing half-band filters to construct orthonormal wavelets works only when the optimization process occurs with respect to even number of coefficients (i.e., even K). We observed that the half-band filters designed using the GI-algorithm also worked for an odd number of coefficients of (i.e., odd K).

Having eliminated the limitations we proceeded to construct the scaling and wavelet functions as described in Chapter 4 using Bauer's spectral factorizer, and

the IGDA was used to plot solutions to the two-scale equations that yielded the scaling and wavelet functions. From the simulation results provided in this chapter it seems clear as stated in [45], that many smooth wavelets are similar in shape. It is observed that the scaling and wavelets functions look similar to their Daubechies counterparts both in the time and frequency domains. We also observed that not only are the new scaling and wavelets functions more regular than those obtained by Cooklev's method, but also in some cases they are more regular than their Daubechies counterparts.

5.3 Suggestions for Future Work

" Every solution breeds new problems "

- Arthur Bloch

Even though we have been successful in obtaining a more efficient design method than Cooklev's [7], or Zarowski's [9], there still remain some loop holes to be closed. We suggest the following steps that could be taken in the hope of achieving a better design method :

- The Multiple Exchange (ME) approach [31] should be adopted. Apparently, since it also uses the K-T conditions to check for optimality, this should be a useful strategy. This would also convert the problem into a PCLS optimization problem, and therefore the advantages of the PCLS design as described in [31] could be used.

- The numerically stable method of QR decomposition for the GI-algorithm [29] could be implemented in the hope of making the algorithm even more numerically robust.

Not only could the algorithm be modified as explained above, but we could also explore other methods of comparing the performance of the new design algorithm, with that of Daubechies. Cooklev in his Ph.D thesis [7] mentions that his filter bank outperformed the filters of Daubechies (in a number of computer simulations in image coding), in terms of the S/N ratio, assuming the same compression ratio. The same tests could be conducted to compare the new design algorithm with Cooklev's and Daubechies method.

To construct symmetric wavelet functions, a future research direction may be to extend the usage of the new algorithm to design *biorthogonal* filter banks. Biorthogonal filters allow linear phase and lead to symmetric scaling and wavelet functions. In this thesis we have considered filter banks having real-valued coefficients only. It would be interesting to extend the new design to the complex coefficient case, which can also lead to symmetric wavelet functions (see Lawton [46]).

Another interesting avenue of research could be to extend this theory to M-band and also multidimensional filter banks. Linear-phase M-band wavelets are useful in image coding [47], whereas multidimensional filter banks and wavelets are extremely useful when 2-D and 3-D still images and video signals are coded.

Lastly, further work needs to be done in developing a better spectral factorizer for the case where there are zeros on the unit circle. It must be noted here that the Bauer

method is slower (though perhaps it could be sped up using fast Toeplitz factorizers) and less accurate as compared to other popularly used algorithms, except for the fact that most other methods do not cope well with zeros on the unit circle.

Appendix A

Chebyshev Polynomial Expressions to Orthogonalize the Bernstein Polynomials

According to the orthogonality relation of Chebyshev polynomials of the first kind

$$\int_{-1}^{1} T_n(x)T_m(x)\left(1-x^2\right)^{-\frac{1}{2}}dx \;=\; \delta_{n,m}h_n \qquad (A.1)$$

where $h_0 = \pi$ and $h_n = \frac{\pi}{2}$ $(n \neq 0)$. Now let the Bernstein polynomial given in Equation (2.51) in Chapter 2 be expressed as a series expansion in terms of Chebyshev polynomials according to

$$b_k^N(x) \;=\; \sum_{n=0}^{N} c_n T_n(cos\,\omega) \qquad (A.2)$$

where also for $x \in [0,1]$ we define

$$cos\,\omega \;=\; 1-2x \;\;.$$

Therefore,

$$b_k^N(x) \;=\; \sum_{n=0}^{N} c_n T_n(1-2x) \;\;. \qquad (A.3)$$

Using $y = 1 - 2x$, multiplying by Chebyshev polynomials, and using their orthogonality property relations gives

$$\binom{N}{k} \int_{-1}^{1} \left(\frac{1-y}{2}\right)^{k} \left(\frac{1+y}{2}\right)^{N-k} T_n(y) \left(1-y^2\right)^{-\frac{1}{2}} \, dy = c_n \int_{-1}^{1} T_n(y) T_n(y) (1-y^2)^{-\frac{1}{2}} \, dy \tag{A.4}$$

so that

$$\binom{N}{k} 2^{-N} \int_{-1}^{1} (1-y)^{k-\frac{1}{2}} (1+y)^{N-k-\frac{1}{2}} T_n(y) dy = c_n h_n \tag{A.5}$$

and therefore

$$c_n h_n \binom{N}{k}^{-1} 2^N = \int_{-1}^{1} (1-y)^{k-\frac{1}{2}} (1+y)^{N-k-\frac{1}{2}} T_n(y) \, dy \tag{A.6}$$

Now using Gauss' hypergeometric series we have

$$T_n(x) = \sum_{p=0}^{n} \frac{(-n)_p (n)_p}{p! (\frac{1}{2})_p} 2^{-p} (1-x)^p \tag{A.7}$$

where

$$(a)_p = a(a+1)(a+2)\cdots(a+p-1) = \frac{\Gamma(a+p)}{\Gamma(a)} \quad . \tag{A.8}$$

Therefore (A.6) becomes

$$c_n h_n \binom{N}{k}^{-1} 2^N = \sum_{p=0}^{n} \frac{(-n)_p (n)_p 2^{-p}}{p! (\frac{1}{2})_p} \int_{-1}^{1} (1-y)^{p+k-\frac{1}{2}} (1+y)^{N-k-\frac{1}{2}} dy \quad . \tag{A.9}$$

Now using the beta integral in Equation (3.196.3) of [48]

$$\int_b^a (t-b)^{x-1} (a-t)^{y-1} dt = (a-b)^{x+y-1} B(x,y) \Rightarrow \int_{-1}^{1} (1+t)^{x-1} (1-t)^{y-1} dt = 2^{x+y-1} B(x,y)$$

Now, we may define y according to

$$p + k - \frac{1}{2} = p + k + \frac{1}{2} - 1 \Rightarrow y = p + k + \frac{1}{2}$$

and similarly, we may let

$$x \;=\; N - k + \frac{1}{2} \;.$$

Therefore,

$$\int_{-1}^{1}(1+t)^{x-1}(1-t)^{y-1}dt \;=\; 2^{N-k+\frac{1}{2}+p+k+\frac{1}{2}-1}B(x,y) \;=\; 2^{N+p}B(N-k+\frac{1}{2},p+k+\frac{1}{2}) \;.$$

According to Equation (6.12.1) of [49]

$$B(p,q) \;=\; \frac{\Gamma(p)\,\Gamma(q)}{\Gamma(p+q)} \qquad (A.10)$$

so that

$$\int_{-1}^{1}(1+t)^{x-1}(1-t)^{y-1}dt \;=\; 2^{N+p}\frac{\Gamma(N-k+\frac{1}{2})\Gamma(p+k+\frac{1}{2})}{\Gamma(N+p+1)} \;. \qquad (A.11)$$

Also from Equation (A.8),

$$\left(k+\frac{1}{2}\right)_p \;=\; \frac{\Gamma\left(p+k+\frac{1}{2}\right)}{\Gamma\left(k+\frac{1}{2}\right)} \;. \qquad (A.12)$$

Using (A.11) and (A.12) in Equation (A.9)

$$\sum_{p=0}^{n}\frac{(-n)_p(n)_p}{p!\left(\frac{1}{2}\right)_p}\,2^N\,\frac{\Gamma(p+k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{\Gamma(N+p+1)}$$

$$= 2^N\sum_{p=0}^{n}\frac{(-n)_p(n)_p\left(k+\frac{1}{2}\right)_p}{p!\left(\frac{1}{2}\right)_p(N+1)_p}\,\frac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{\Gamma(N+1)} \qquad (A.13)$$

$$= 2^N\sum_{p=0}^{n}\frac{(-n)_p(n)_p\left(k+\frac{1}{2}\right)_p}{p!\left(\frac{1}{2}\right)_p(N+1)_p}\,\frac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{N!} \qquad (A.14)$$

Therefore, from Equation (2.1.1.2) of [50]

$$= {}_3F_2(-n,n,k+\frac{1}{2};\frac{1}{2},N+1)\,2^N\frac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{N!} \;. \qquad (A.15)$$

Using (A.15) in (A.9) we get

$$c_n h_n \binom{N}{k}^{-1} 2^N = \frac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{N!} {}_3F_2(-n,n,k+\frac{1}{2};\frac{1}{2},N+1)\, 2^N \quad (A.16)$$

$$\Rightarrow c_n = \frac{1}{h_n}\binom{N}{k}\frac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{N!} {}_3F_2(-n,n,k+\frac{1}{2};\frac{1}{2},N+1)\,. \quad (A.17)$$

Therefore,

$$c_n = \tfrac{1}{h_n}\tfrac{N!}{k!(N-k)!}\tfrac{\Gamma(k+\frac{1}{2})\Gamma(N-k+\frac{1}{2})}{N!}\, {}_3F_2\left(-n,n,k+\tfrac{1}{2};\tfrac{1}{2},N+1\right)\,. \quad (A.18)$$

From Equation (2.25) of [51]

$$\Gamma\left(k+\frac{1}{2}\right) = \frac{(2k)!}{2^{2k}\,k!}\sqrt{\pi}\,. \quad (A.19)$$

Using this in (A.18) we get

$$c_n = \frac{\pi}{h_n}\frac{\frac{(2k!)}{2^{2k}\,k!}\frac{[2(N-k)]!}{2^{2(N-k)}\,(N-k)!}}{k!\,(N-k)!}\, {}_3F_2\left(-n,n,k+\frac{1}{2};\frac{1}{2},N+1\right),$$

or

$$c_n = \frac{\pi}{h_n}\frac{(2k)!\,(2(N-k))!}{2^{2N}\,(k!)^2\,[(N-k)!]^2}\, {}_3F_2\left(-n,n,k+\frac{1}{2};\frac{1}{2},N+1\right). \quad (A.20)$$

Hence, (A.3) becomes

$$b_k^N(x) = \textstyle\sum_{n=0}^{N} \frac{\pi}{h_n}\frac{(2k)!\,(2(N-k))!}{2^{2N}\,(k!)^2\,[(N-k)!]^2}\, {}_3F_2\left(-n,n,k+\frac{1}{2};\frac{1}{2},N+1\right)\, T_n(1-2x). \quad (A.21)$$

Appendix B

Matlab Routines Implementing Supporting Functions

The following routines are the functions that are called by the main programs. The equation numbers cited in the in-line comments of the routines contained in this appendix refer to equations in Chapter 2 and 3 of this thesis.

```
%
%                fact.m
%
% This function computes n ! (i.e., n-factorial).

function m = fact(n)

if n > 0
  j = 1;
  for i = 1:n
    j = j*i;
    end
  m = j;
else
  m = 1;
end
%
%                  binco.m
```

```
%
% This function returns binomial coefficient
% "n choose m".

function b = binco(n,m)
if (m >= 0) & ( (n-m) >= 0 )
  b = fact(n)/( fact(m)*fact(n - m) );
else
  b = 0;
end
%
%                          u.m
%
% This routine computes certain integrals needed by
% Cooklev's theory of half-band filter design.
%
% The integrals are used to compute the elements of a
% certain matrix and a certain vector that ultimately
% give the filter coefficients.
%
% The integral computed is:
%                b    m       n
%         y = int   x (1 - x)   dx
%                a
% We have a <= b, and m,n >= 0.

function y = u(a,b,m,n)

y = 0;
for k = 0:n
  term1 = ((b^(m+1))*((1 - b)^k));
  term2 = ((a^(m+1))*((1 - a)^k));
  y = term1 - term2 + k*y;
  y = y/(k + m + 1);
```

```
end;



%
%                              makeR.m
%
% This routine makes the matrix R (calligraphic font) in (2.81)
% of the document "Cooklev's Theory of Half-Band Filter Design."
%
% Note that 0 < a < b = 1, and that 0 <= L <= (N-1)/2.
%

function R = makeR(a,b,N,L)

for k = 1:(N+1)/2 - L
  for j = k:(N+1)/2 - L
    term1 = u(a,b,2*(N+1-L)-j-k,2*(L-1)+k+j);
    term2 = u(a,b,N+j-k,N+k-j);
    term3 = u(a,b,N+k-j,N+j-k);
    term4 = u(a,b,2*(L-1)+k+j,2*(N+1-L)-j-k);
    R(k,j) = term1 - term2 - term3 + term4;
    R(k,j) = binco(N,j+L-1)*binco(N,k+L-1)*R(k,j);
    R(j,k) = R(k,j);
    end;
  end;
%
%                              maker.m
%
% This routine makes the vector r in (2.81)
% of the document "Cooklev's Theory of Half-Band Filter Design."
%
% Note that 0 < a < b = 1, and that 0 <= L <= (N-1)/2.
%
```

```
function r = maker(a,b,N,L)

for k = 1:(N+1)/2-L
  r(k) = 0;
  for i = 0:(N-1)/2
    term1 = u(a,b,N+i+1-L-k,N+L+k-i-1);
    term2 = u(a,b,L+k+i-1,2*N+1-L-k-i);
    r(k) = r(k) + binco(N,i)*binco(N,k+L-1)*(term1 - term2);
    end;
  end;
%
%                          maker00.m
%
% This routine makes the matrix element r00 in (2.78)
% of the document "Cooklev's Theory of Half-Band Filter Design."
%
% This involves the use of Equation (2.73).
%
% Note that 0 < a < b = 1.
%

function r00 = maker00(a,b,N)

r00 = 0;
for k = 0:(N-1)/2
  for j = 0:(N-1)/2
    r00 = r00 + binco(N,j)*binco(N,k)*u(a,b,j+k,2*N-j-k);
    end;
  end;
%
%                            v0.m
%
% This routine implements the function in Equation (2.72)
```

```
% of "Cooklev's Theory of Half-Band Filter Design."
%
% NOTE: N is odd.
%

function y = v0(x,N)

y = 0;
for k = 0:(N-1)/2
  y = binco(N,k)*(x^k)*((1 - x)^(N-k)) + y;
  end;
%
%                          vk.m
%
% This routine implements the function in Equation (2.73)
% of "Cooklev's Theory of Half-Band Filter Design."
%
% NOTE: N is odd, and 0 <= L <= (N-1)/2.  As well, we have
%       k = 1,2, ... , (N+1)/2 - L .
%

function y = vk(x,k,N,L)

term1 = (x^(N+1-L-k))*((1-x)^(L+k-1));
term2 = (x^(k+L-1))*((1-x)^(N+1-L-k));
y = binco(N,k+L-1)*(term1 - term2);

%
%                          H.m
%
% This routine evaluates the matrix V in Equation (2.5)
% of this report.
%
% NOTE: N is odd, and 0 <= L <= (N-1)/2.
```

```
%

function Vx = H(x,N,L)

vk2 = [];

for k = 0:(N+1)/2 - L
  if k == 0
    v02 =   v01(x,N);
  end;
  if k > 0
    vk2 =  [vk2 vk1(x,k,N,L)];
  end;
end;
Vx = [v02 vk2];
```

Appendix C

Matlab Routines Implementing Main Programs

```
%*****************************************************
%
% Implementation of the Goldfarb-Idnani Algorithm
% based on the paper
% " A Numerically Stable Dual Method for Solving
%   Strictly Convex Quadratic Programs "
% Mathematical Programming 27 (1983) pp. 1-33
%
%  GI_Algo.m
%
%*****************************************************

% G = n X n positive definite matrix
% a = n vector
% C = n x m matrix
% b = m vector
% E = perturbation parameter (only used sometimes to
%     assist in convergence to the Magnitude response
%     in most cases it's value is zero).

% AUTHOR : Sanjay C.Verma (Queen's Univerity)
% ------------------------------------------
```

```
%

function [x,iter] = GI_Algo(G,a,C,b,E)

%*****************************************************
%
% Step 0 : Find the unconstrained minimum
%
%*****************************************************

G1 = inv(G);
x = -G1*a;
H = G1;
A = [];
q = 0;

% initialize variables for iterative loop

iter = 1;
add = 0;
del = 0;


% Evaluate constraint function

S = (C'*x - b)';

% # of constraints in the constraint set

K = size(S,2);

%*****************************************************
%
% Step 1 : Choose a violated constraint if any
```

```
%
%*******************************************************

while min(S) < -10e-10

  if add == 1 | iter == 1

    % Reinitialize the add variable

    add = 0;

    % To evaluate the MOST violated constraint
    % ----------------------------------------

    % (i) First evaluate the set V such that it
    %     covers all constraints that do not
    %     belong to the active set A

    w = [];
    for vcon = 1:K
      if isempty(A) ~= 1
        if A ~= vcon
          w = [w vcon];
        end;
      else
        w = [w vcon];
      end;
    end;

    % (ii) Now find the MOST violated constraint
    %      ie: find the most negative value of S

    wsize = size(w,2);
    S2 = [];
```

```
  for vcon1 = 1:wsize
    S2 = [S2 S(:,w(vcon1))];
  end;

  constraint = min(S2);

  % (iii) Once the most violated constraint has
  %       been selected, assign this constraint
  %       number to "p".

  for vcon2 = 1:K
    if S(vcon2) == constraint & A ~= vcon2
      p = vcon2;
    end;
  end;

  % Select the pth column of the matrix C

  nplus = C(:,p);

  if q == 0
    u = 0;
    uplus = 0;
  else
    uplus = [u;0];
  end;
end; % End of the condition (if add == 1 | iter == 1)

%*************************************************************
%
% Step 2 : Check for feasibility and determine a new S-pair
%
%*************************************************************
```

```
if del == 1 | iter >= 1

  % reinitialize the del variable

  del = 0;

  %-------------------------------------
  % Step 2(a) : Determine step direction
  %-------------------------------------

  if isempty(A) == 1
    H = G1;
    z = H*nplus;
    r = 0;
  else
    Ns = N'*G1*N;
    Nstar = inv(Ns)*N'*G1;
    N2 = N*Nstar;
    c1 = size(N2,2);
    H = G1*(eye(c1) - N2);
    z = H*nplus;
    r = Nstar*nplus;
  end;

  %---------------------------------
  % Step 2(b) : Compute step length
  %---------------------------------

  % (i) Partial step length t1
  % realmax is equivalent to infinity

  if r <= 0 | q == 0
    t1 = realmax;
  else
```

```
 ur = [];
 for j = 1:q
   if r(j) > 0
     ur = [ur uplus(j)./r(j)];
   end;
 end;
 t1 = min(ur);

 %-------------------------------------------------
 % selecting the constraint that needs to be dropped
 % if a case arises
 %-------------------------------------------------

 cu = size(ur,2);
 for cdrop = 1:cu
   if ur(cdrop) == t1
     k = cdrop;
     break;
   end;
 end;
end;

% (ii) Full step t2

if abs(z) == 0
  t2 = realmax;
else
  t2 = -S(p)/(z'*nplus);
end;

% (iii) Step length t

t = min(t1,t2);
```

```
%-------------------------------------------
% Step 2(c) : Determine new S-pair & take a step
%-------------------------------------------

%-----------------------------------
% (i) No step in primal or dual space
%-----------------------------------

if t == realmax
  OOPs = 1
  break;
end;

%------------------------
% (ii) Step in dual space
%------------------------

if t2 == realmax
  if r == 0
    uplus = uplus + t;
  else
    uplus = uplus + t*[-r;1];
  end;


  %------------------
  % DROP CONSTRAINT k
  %------------------

  %----------------------------------------
  % Update A : Drop the kth constraint from A
  %----------------------------------------

  A1 = [];
```

```
for dropA = 1:q
  if (dropA) ~= k
    A1 = [A1 A(dropA)];
  end;
end;
A = A1

%-------------------
% Decrease q by one
%-------------------

q = q - 1;

%------------------------------
% Drop the kth element of uplus
%------------------------------

u1 = [];
cu1 = size(uplus,1);
for dropu = 1:cu1
  if dropu ~= k
    u1 = [u1;uplus(dropu)];
  end;
end;
uplus = u1;

%--------------------------------------------
% Update matrix N ie: drop the kth constraint
%--------------------------------------------

Ndrop = [];
%A2 = sort(A);
for Nd = 1:q
  Ndrop = [Ndrop C(:,A(Nd))];
```

```
  end;
  N = Ndrop;

  %----------------
  % GOTO Step 2(a)
  %----------------

  del = 1;
end; % end of (if t2 == realmax)

%-------------------------------------
% (iii) Step in primal and dual space
%-------------------------------------

if del ~= 1

  % Evaluate the x variable

  x = x + t*z;

  %-------------------------------------------------------
  % Evaluate the tolerance parameter to shift the impulse
  % response slightly above zero so that there is no
  % splitting of double zeros on the unit circle.
  %-------------------------------------------------------

  ep = size(b,1);
  Etol = [];
  for e = 1:ep
    Etol = [Etol;E];
  end;

  % Use this tolerance parameter to evaluate matrix S
```

```
S = (C'*x - b-Etol)';

%-----------------------------------------------------
% If ALL the elements in S are greater than ZERO then
% EUREKA !!!! :) CONVERGENCE has been achieved. STOP.
%-----------------------------------------------------

if min(S) > -10e-9
  EUREKA = 1
  break;
end;

% Update iteration counter

iter = iter + 1;

%-----------------------------------------------------
% Break out of the loop if the iteration count exceeds
% pre-determined limits
%-----------------------------------------------------

if iter == 100
  disp('Iteration Limit Exceeded....User should try using different ...
  INPUT parameters');
  break;
end;

% Update uplus

if r == 0
  uplus = uplus + t;
else
  uplus = uplus + t*[-r;1];
end;
```

```
%--------------
% Full Step t2
%--------------

if t == t2
  u = uplus;

  % Add constraint p
  % ----------------

  %----------
  % Update A
  %----------

  A = [A p];

  %----------
  % Update q
  %----------

  q = q + 1;

  %----------
  % Update N
  %----------

  Nadd = [];
  for Na = 1:q
    Nadd = [Nadd C(:,A(Na))];
  end;
  N = Nadd;

  %-------------
```

```
  % Goto Step 1
  %-------------

  add = 1;
end; % end of (if t == t2)

%-----------------
% Partial Step t1
%-----------------

if t == t1

  %-------------------
  % DROP CONSTRAINT k
  %-------------------

  %-----------------------------------------
  % Update A : Drop the kth constraint from A
  %-----------------------------------------

  A1 = [];
  for dropA = 1:q
    if (dropA) ~= k
      A1 = [A1 A(dropA)];
    end;
  end;
  A = A1;

  %-------------------
  % Decrease q by one
  %-------------------

  q = q - 1;
```

```
        %------------------------------
        % Drop the kth element of uplus
        %------------------------------

        u1 = [];
        cu1 = size(uplus,1);
        for dropu = 1:cu1
          if dropu ~= k
            u1 = [u1;uplus(dropu)];
          end;
        end;
        uplus = u1;

        %--------------------------------------------
        % Update matrix N ie: drop the kth constraint
        %--------------------------------------------

        Ndrop = [];
        for Nd = 1:q
          Ndrop = [Ndrop C(:,A(Nd))];
        end;
        N = Ndrop;

        %---------------
        % GOTO Step 2(a)
        %---------------

        del = 1;
      end;   % end of (if t == t1)
    end; % end of (if del ~= 1)
  end;       % end of (if del == 1 | iter >= 1)
end;     % end of (while min(S) < -10e-10)
```

```
%*************************************************************
% H_to_plot.m
%
% This routine creates the half-band filter impulse
% response sequence h.
%
% Note that 0 < a < b = 1, and that 0 <= L <= (N-1)/2.
%*************************************************************

function h = H_to_plot(a,b,N,L,M,x,E)

% Create the alpha parameter vector (Equation (2.81))
x1 = x;
M1 = M;
E1 = E;
b1 = b;
R = 0;
r = 0;
x = 0;
N1a = [];
Tha = [];

R = 2*makeR(a,b,N,L);
r = 2*maker(a,b,N,L);
r00 = maker00(a,b,N);

% to evaluate Equation (3.12).

C2 = [];
M = M1*N;
```

```
for k = 0:(M-1)
  x = x1 + (1 - x1)/(M)*k;
  Vx = H(x,N,L);
  C2 = [C2;Vx];
end;
sizeC = size(C2,2);
b2 = -C2(:,1);
C1 = C2(:,[2:sizeC]);
C2 = C1'

[alpha,iter] = GI_Algo(R,r.',C2,b2,E1)
no_of_iterations = iter;

    % Create the e parameter vector (Equation (2.67))


if L > 0
 for k = 0:L-1
   e(k+1) = 1;
 end;
 for k = L:(N-1)/2
   e(k+1) = 1 - alpha(k-L+1);
 end;
 for k = (N+1)/2:N-L
   e(k+1) = alpha(N+1-L-k);
 end;
 for k = N+1-L:N
   e(k+1) = 0;
 end;
else
 for k = L:(N-1)/2
   e(k+1) = 1 - alpha(k-L+1);
 end;
 for k = (N+1)/2:N-L
```

```
   e(k+1) = alpha(N+1-L-k);
 end;
end

% Compute the frequency response of the Half-Band filter

deltaomega = (2*pi)/(2*N+1);

for r = 0:(2*N)

  omega = r*deltaomega;

  H(r+1) = exp(-j*omega*N)*e(1)*binco(N,0)*((1+cos(omega))/(2))^N;
  for k = 1:N
    H(r+1) = H(r+1) + exp(-j*omega*N)*e(k+1)*binco(N,k)* ...
             ((1-cos(omega))/(2))^k*((1+cos(omega))/(2))^(N-k);
  end;

end;

% Use the inverse FFT to compute to compute the half-band filter
% impulse response sequence

h= real(ifft(H)).';

% when using this program to plot the magnitude response
% uncomment the next two lines, otherwise when using it for
% plotting the zero-plot or using it with the IGDA comment
% the next two lines.

h2 = [h; N; L; iter;a;b1;M1;x1]

h = h2;
```

```
%************************************************************************
% Hxplot.m
%
% This routine plots the half-band filter frequency response
% for the filter parameters obtained with the aid of the
% GI-Algorithm  (ie: GI_Algo.m)
%
% The filter impulse response sequence is also plotted
%************************************************************************

function Hxplot(h)
h1 = size(h,1)
h3 = [];
h4 = [];
for h2 = 1:(h1)
  if h2 < (h1-6)
    h3 = [h3;h(h2)];
  else
    h4 = [h4;h(h2)];
  end;
end;

h = h3;
N2 = num2str(h4(1));
L2 = num2str(h4(2));
iter2 = num2str(h4(3));
a2 = num2str(h4(4));
b2 = num2str(h4(5));
M2 = num2str(h4(6));
x2 = num2str(h4(7));


subplot(211),bar(0:length(h)-1,h),grid
xlabel('Coefficient Index')
```

```
ylabel('Amplitude')
title('Cooklev Half-Band Filter Impulse Response Sequence')

L = 100;
a = [1];
[H,w] = freqz(h,a,L);
N = (length(h) - 1)/2;
domega = pi/L;
for k = 0:L-1
  Hx(k+1) = exp(j*domega*k*N)*H(k+1);
end;

subplot(212), plot(w,real(Hx)),grid
xlabel('Digital Frequency')
ylabel('Amplitude')
title(['Amplitude Response of a Cooklev Half-Band Filter ', '...
( iter = ',iter2,' ; a = ',a2,'; b = ',b2,'; N = ',N2,' ; L = '...
,L2, '; M = ',M2,'; x = ',x2,' )']))


%****************************************************************
% This routine produces the zero-plots shown in Chapters 2 and 3
% of this thesis.
%
% zeroplot.m
%
% This routine plots the zeros of the filters with
% impulse response sequences h, and g.
% These sequences are generated by makeh1.m and
% Hoptfft.m, respectively.
%****************************************************************

function zeroplot(a,b,N,L,M,x,E)
```

```
g = H_to_fact(a,b,N,L,M,x,E);

rg = roots(g);

 % Add the multiple zeros of order 2L at z = -1 to
 % the zeros of the FIR filter g.

rg = [rg ; -1*ones(2*L,1)];

clf

theta = 0:.005:2*pi;
plot(cos(theta),sin(theta),'.')
hold
for k = 1:length(rg)
  plot(real(rg(k)),imag(rg(k)),'o')%,real(rg(k)),imag(rg(k)),'+')
end;
grid
xlabel(' Real Part ')
ylabel(' Imaginary Part ')

N2 = num2str(N);
L2 = num2str(L);
a2 = num2str(a);
b2 = num2str(b);
M2 = num2str(M);
x2 = num2str(x);
E2 = num2str(E);

title([' FIR Filter Zeros ','a = ',a2,';  b = ',b2,';  N = ',N2,'; ...
      L = ',L2,';  M = ',M2,';  x = ',x2,';  E = ',E2])
axis('square')

%*************************************************************
```

```
% H-to-fact.m
%
% This routine creates the half-band filter impulse
% response sequence h with the multiple zeros at z = -1
% factorized out (i.e., with the multiple zeros) to avoid
% zero splitting.
%
% In what follows the equation numbers cited refer to
% equations in the document "Cooklev's Theory of Half-Band
% Filter design."
%
% Note that 0 < a < b = 1, and that 0 <= L <= (N-1)/2.
%************************************************************

function h = H_to_fact(a,b,N,L,M,x,E)

% Create the alpha parameter vector (Equation (2.81))

x1 = x;
M1 = M;
E1 = E;
b1 = b;
R = 0;
r = 0;
x = 0;
N1a = □;
Tha = □;
R = 2*makeR(a,b,N,L);
r = 2*maker(a,b,N,L);
r00 = maker00(a,b,N);
C2 = □;
M = M1*N;
for k = 0:(M-1)
  x = x1 + 1/(2*M)*k;
```

```
  Vx = H(x,N,L);
  C2 = [C2;Vx];
end;
sizeC = size(C2,2);
b2 = -C2(:,1);
C1 = C2(:,[2:sizeC]);
C2 = C1'

[alpha,iter] = GI_Algo(R,r.',C2,b2,E1)%,r00);
no_of_iterations = iter;

    % Create the e parameter vector (Equation (2.67))


if L > 0
 for k = 0:L-1
   e(k+1) = 1;
 end;
 for k = L:(N-1)/2
   e(k+1) = 1 - alpha(k-L+1);
 end;
 for k = (N+1)/2:N-L
   e(k+1) = alpha(N+1-L-k);
 end;
 for k = N+1-L:N
   e(k+1) = 0;
 end;
else
 for k = L:(N-1)/2
   e(k+1) = 1 - alpha(k-L+1);
 end;
 for k = (N+1)/2:N-L
   e(k+1) = alpha(N+1-L-k);
 end;
```

```
end

% Compute the frequency response of the Half-Band filter with multiple
% zeros factored out to avoid zero splitting.

 deltaomega = (2*pi)/(2*(N-L)+1);

for r = 0:2*(N-L)

  omega = r*deltaomega;
  H(r+1) = 0;
  for k = 0:(N-1)/2
    H(r+1) = H(r+1) + exp(-j*omega*(N-L))*binco(N,k)* ...
    ((1-cos(omega))/(2))^k*((1+cos(omega))/2)^(N-L-k);
  end;
  for k = 1:((N+1)/2-L)
    H(r+1) = H(r+1) + exp(-j*omega*(N-L))*alpha(k)* ...
             binco(N,k+L-1)*(((1-cos(omega))/(2))^(N+1-L-k) ...
             *((1+cos(omega))/(2))^(k-1)-((1-cos(omega))/(2))^(k+L-1)* ...
             ((1+cos(omega))/(2))^(N+1-2*L-k));
  end;

end;

% Use the inverse FFT to compute to compute the half-band filter
% impulse response sequence

h= real(ifft(H)).';
```

```
%*********************************************************
%
% specfact.m
%
%*********************************************************
%
%  This program implements the spectral factorization
% of a polynomial using the Bauer method which
% is based on the Cholesky factorization of a banded
% Toeplitz correlation matrix.
%
% This program has been taken from the report "On the
% design and implementation of filter bank trees for
% multiple access communications", by Dr.Todor Cooklev.
%
%*********************************************************

% ndi is the dimension of each matrix pi
% specify also the size of the matrix (about 10 times
% the size of the filter

function y = specfact(p,m)

len = length(p);
lenh = (len-1)/2;

%
% p must be a half-band filter with positive frequency
% response
%

ndi = 1;
dimt = ndi*(m+1);
```

```
% t is of dimension ndi(m+1) X ndi(m+1)

t = sparse(1:ndi:dimt,1:ndi:dimt,p(lenh+1),dimt,dimt);

for n = 1:lenh
  t = t + sparse(1:ndi:dimt-n,n+1:ndi:dimt,p(lenh+n+1),dimt,dimt);
end;

t1 = triu(t,1);
t = t + t1';

u(dimt,dimt) = sqrt(t(dimt,dimt));
u(dimt-1,dimt) = t(dimt-1,dimt)/u(dimt,dimt);
u(dimt-1,dimt-1) = sqrt(t(dimt-1,dimt-1)-(u(dimt-1,dimt))^2);

for j = dimt-2:-1:1

  u(j,dimt) = t(j,dimt)/u(dimt,dimt);

  for k = dimt-1:-1:j+1

    abra = 0;

    for m = k+1:dimt
      abra = abra+u(j,m)*u(k,m);
    end;

    u(j,k) = (1/u(k,k))*(t(j,k)-abra);

  end;

  abra = 0;
```

```
  for m = j+1:dimt
    abra = abra+u(j,m)*u(j,m);
  end;

  u(j,j) = sqrt(t(j,j)-abra);

end;

% UL factorization of a positive definite matrix
% the matrix u i supper-triangular; t = u. u^t; the first column
% of u^t contains the matrix coefficients of the spectral factor.

size(u)

auu = u(1,1:(len+1)/2);
y = full(auu);
su = 0.0;

for n = 1:length(y)
  su = su + y(n);
end;

y = y/su;

%end;

%************************************************************
%% Implementation of IGDA Algorithm based on outline in
%% "Notes on Orthogonal Wavelets and Wavelet Packets ",
%% by Dr. C. J. Zarowski.
%************************************************************

% p=coefficients of low pass filer of QMF bank
```

```
qw = (specfact(Hoptfft4(.5,1,17,1,10,.5,0.001),40));
as = sum(qw);

% since for Orthonormal wavelets one of the requirements is
% that all the filter coefficients of the product filter
% should add up to be equal to 2, hence we multiply the coefficients
% by 2 to normalize it.

p=2*qw;

N = size(p,2)-1;
J=8;  % J = resolution

size_p = size(p);
p_length = size_p(2);

% compute q vector

for k=0:N
  q(k+1) = (-1)^(N-k-1)*p(N-k+1);
end

for j = 1:(N-1)
  for m = 1:(N-1)
    index = 2*j - m + 1;
    if (index > p_length | index < 1)
      M(j,m) = 0;
    else
      M(j,m) = p(index);
    end
  end
end

% solve for eigen
```

```
[b,d] = eig(M);
[lambda,K] = sort(real(diag(d)));
num = size(lambda);
pos = 1;
flag = 1;
while ((lambda(pos) > 1.001 | lambda(pos) < 0.999) & flag ==1)
  pos = pos+1;
  if (pos > num(1))
    flag = 0;
    pos = pos -1;
  end
end
b = b(:,K);
m = b(:,pos);

% enforce partition of unity property

alpha = sum(m);
phi = (m ./ alpha)';

phi = [0 phi 0];
a=[1];
for j=0:J-1
 aa=zeros(1,2*size(a,2));
 for i=1:size(a,2)
  aa((i-1)*2+1)=a(i);
 end;
 aa=aa(1:size(aa,2)-1);
 a=conv(p,aa);

end;
```

```
% convolve a and phi on the integers to get
% the scaling function

phi_out=conv(a,phi);
subplot(2,1,1);axis1=0:size(phi_out,2)-1;plot(axis1/(2^J),phi_out);
grid;
title('Scaling Function');

% now upsample q by 2^J (q has same length N+1 as p)

for i = 0:N
  qq(2^J*i+1) = q(i+1);
end

% convolve q with scaling function to get wavelet

wavelet = conv( phi_out,qq );


subplot(2,1,2);axis2=0:size(wavelet,2)-1;plot(axis2/2^(J+1),wavelet);
grid;
title('Wavelet function');
```

Appendix D

A Comprehensive List for Half-band Filter Specifications for N up to 25.

N	L	M	y	Iterations	E	Convergence
3	1	11	.5	4	0.00068175	X
5	1	10	.5	3	0	X
5	2	10	.5196	7	0	X
7	1	9	.64	8	0.001	X
7	2	11	.5	3	0	X
7	3	9	.6	5	0	X
9	1	11	.5	7	0	X
9	2	11	.65	17	0.00005	X
9	3	10	.5	5	0	X
9	4	9	.53	12	0	X
11	1	11	.53	18	0.00153	X
11	2	10	.5	6	0	X
11	3	10	.5	23	0.0001	X
11	4	10	.5	4	0	X
11	5	10	.61	21	9.44e-09	X
13	1	10	.5	15	0	X
13	2	10	.5	22	5.65e-05	X
13	3	10	.5	6	0	X
13	4	10	.55	34	5e-07	X
13	5	10	.5	3	0	X

N	L	M	y	Iterations	E	Convergence
13	6	10	.52	19	9.905e-09	X
15	1	11	.52	24	0.00204	X
15	2	10	.5	18	0	X
15	3	11	.52	33	6.297e-07	X
15	4	10	.5	8	0	X
15	5	11	.65	30	9.487e-09	X
15	6	10	.5	7	0	X
15	7	11	.55	12	0	X
17	1	11	.5	20		X
17	2	11	.5	28	7.8e-05	X
17	3	11	.5	17	0	X
17	4	11	.5	39	8.86e-09	X
17	5	11	.5	10	0	X
17	6	11	.63	34	8.89e-09	X
17	7	11	.5	4	0	X
17	8	11	.75	18	9.157e-09	X
19	1	11	.51	32	0.00275	X
19	2	11	.55	16	0	X
19	3	11	.51	39	1.8e-06	X
19	4	11	.5	15	0	X
19	5	11	.51	56	5e-08	X
19	6	10	.5	10	0	X
19	7	12	.63	30	9.985e-09	X
19	8	10	.5	5	0	X
19	9	11	.69	5	0	X
21	1	10	.51	13	0	X
21	2	10	.56	31	0.000108	X
21	3	10	.55	31	0	X
21	4	10	.57	43	1.515e-08	X
21	5	10	.55	18	0	X
21	6	10	.57	48	9.22e-09	X
21	7	10	.55	9	0	X
21	8	10	.54	13	0	X
21	9	10	.55	4	0	X
21	10	10	.73	6	0	X
23	1	10	.57	36	0	X
23	2	10	.5	11	0	X
23	3	10	.57	38	1.75e-06	X

N	L	M	y	Iterations	E	Convergence
23	4	10	.5	30	0	X
23	5	10	.57	55	9.7e-09	X
23	6	10	.55	23	0	X
23	7	10	.58	64	1e-08	X
23	8	10	.55	10	0	X
23	9	9	.56	44	9.68e-09	X
23	10	10	.55	5	0	X
23	11	11	.5	14	0	X
25	1	10	.5	14		X
25	2					
25	3	10	.509	9	0	X
25	4	10	.5	59	9e-08	X
25	5	10	.5	13	0	X
25	6	10	.5	66	8.7e-09	X
25	7	10	.5	12		X
25	8	10	.5	61	1e-08	X
25	9	10	.5	12	0	X
25	10	10	.61	36	9.35e-09	X
25	11	10	.5	6	0	X
25	12	10	.56	10	0	X

References

[1] Amara Graps. An introduction to wavelets. *IEEE Computational Sciences and Engineering*, 12(2):50–61, 1995.

[2] Jr R. O. Wells. Recent advances in wavelet technology, 1994. http://www.cml.rice.edu/94.html.

[3] S. Rao P. Ghandhi and R. S. Pappu. Wavelets for waveform coding of digital symbols. *IEEE Transactions on Signal Processing*, 45(9):2387–2390, 1997.

[4] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. Pure Applied Mathematics*, 41:909–996, 1988.

[5] S. Mallat. A theory of multiresolution signal decomposition : The wavelet representation. *IEEE Trans. Pattern Anal. Machine Intell.*, 11:674–693, 1989.

[6] Y. Meyer. *Wavelets and Operators.* Cambridge University Press, Paris, 1992.

[7] T. Cooklev. *Regular Perfect-Reconstruction Filter Banks and Wavelet Bases.* PhD thesis, Tokyo Institute of Technology, Japan, 1995.

[8] C. J. Zarowski. Cooklev's theory of half-band filter design. Technical report, Queen's University at Kingston, 1997.

[9] C. J. Zarowski. Design of half-band filters with a nonnegative frequency response. Technical report, Queen's University at Kingston, June 1997.

[10] G.Strang and T.Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, Wellesley, Massachusetts, 1996.

[11] F. Mintzer. Filters for distortion-free two-band multirate filter banks. *IEEE Trans. on Acoustics, Speech and Signal Proc.*, ASSP-33:626–630, June 1985.

[12] M. J. T Smith and T. P. Barnwell. Exact reconstruction techniques for tree-structured subband coders. *IEEE Trans. ASSP.*, 38:1446–1456, 1990.

[13] P. P. Vaidyanathan. *Multirate Systems and Filter banks.* Prentice Hall, 1993.

[14] O.Rioul and P.Duhamel. A remez exchange algorithm for orthonormal wavelets. *IEEE Transactions on on Circuits and Systems*, 41:550–560, 1994.

[15] Martin Vetterli and Jelena Kovacevic. *Wavelets and Subband Coding.* Prentice-Hall, 1995.

[16] X. G. Xia. A family of pulse-shaping filters with isi-free matched and unmatched filter properties. *IEEE Trans. on Comm.*, 45:1157–1158, 1997.

[17] N. Sheikholeslami and P. Kabal. A family of nyquist filters based on generalized raised-cosine spectra. In *Proceedings of the 19th Biennial Symposium on Communications*, pages 131–135, Kingston, Ontario, May 31 to June 3 1998.

[18] W. W. Jones. *A Unified Approach to Orthogonally Multiplexed Communication Using Wavelet Bases and Digital Filter Banks.* PhD thesis, Ohio University, 1994.

[19] C.K.Chui. *An Introduction to Wavelets.* Academic Press, Boston, Massachusetts, 1992.

[20] C. J. Zarowski. Notes on orthogonal wavelets and wavelet packets. Technical report, Queen's University at Kingston, September 1996.

[21] P. P. Viadyanathan and T. Q. Nguyen. Eigenfilters: A new approach to least-squares fir filter design and applications including nyquist filters. *IEEE Transactions on Circuits and Systems*, 34(1):11–23, 1987.

[22] T. Yoshida T. Cooklev, A. Nishihara and M. Sablatash. Regular linear-phase multidimensional fir filter banks and wavelet bases. In *Proc. IEEE Int. Conf. Acoust. Speech and Signal Processing*, Detroit, MI, 1995.

[23] A. N. Akansu and R. A. Hadad. *Multiresolution Signal Decomposition: Transforms, Subbands, Wavelets.* Academic Press, 1992.

[24] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing.* Prentice-Hall, 1989.

[25] A. Papoulis. *Probability, Random Variabless, and Stochastic Processes.* McGraw-Hill, New York, 2nd edition, 1984.

[26] J. H. Shen and G. Strang. Asymptotic analysis of daubechies polynomials. In *Proceedings of the American Mathematical Society*, volume 124, pages 3819–3833, 1996.

[27] V. Hribernig and H. J. Stetter. Detection and validation of clusters of polynomial zeros. *Journal of Symbolic Computation*, 24:667–681, 1997.

[28] Wim Sweldens and Robert Piessens. Quadrature formulae and asymptotic error expansions for wavelet approximations of smooth functions. *SIAM Journal of Numerical Analysis*, 31(4), August 1994.

[29] D.Goldfarb and A.Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27:1–33, 1983.

[30] M. J. D. Powell. On the quadratic programming algorithm of goldfarb and idanani. *Mathematical Programming Study*, 25:46–61, 1985.

[31] John W.Adams and James L.Sullivan. Peak-constrained least-squares optimization. *IEEE Transactions on Signal Processing*, 46(2):306–320, 1998.

[32] John W.Adams. A new optimal window. *Mathematical Programming*, 39(8):1753–1769, 1991.

[33] John W.Adams. Fir digital filters with least-squares stopbands subject to peak-gain constraints. *IEEE Transactions on on Circuits and Systems*, 39:376–388, 1991.

[34] E.A.Gauer John W.Adams, James L.Sullivan and B.Tucker. Digital filters with peak-constrained weighted least-squared error. In *Proceedings of IEEE Conference on Signals, Systems and Computers*, pages 256–260, October 1992.

[35] R.Hashemi John W.Adams, James L.Sullivan and C.Ghadimi. New approaches to constrained optimization of digital filters. In *Proceedings of IEEE International Symposium on Circuits and Systems*, pages 80–83, May 1993.

[36] D.Goldfarb. Numerically stable approaches to linearly constrained optimization. In M.J.D.Powell, editor, *Non-linear Optimization*, pages ??–?? Academic Press, New York, 1981.

[37] C. Herley and M. Vetterli. Linear phase wavelets : Theory and design. In *Proc. IEEE ICASSP*, pages 2017–2020, Toronto, Canada, 1991.

[38] D.C.Youla and N.N. Kazanjian. Bauer-type factorization of positive matrices and the theory of matrix polynomials orthogonal on the unit circle. *IEEE Transactions on Circuits and Systems*, 25:57–69, 1978.

[39] T. Cooklev. On the design and implementation of filter bank trees for multiple access communications. Technical report, Communications Research Center at Ottawa, 1996.

[40] C. Burrus and R. Gopinath. Introduction to wavelets and wavelet transforms. In *Proc. ICASSP'93*, April 1993.

[41] C. J. Zarowski. An approach to initializing the wavelet packet transform. *IEEE Signal Proc. Letters*, 4:132–134, May 1997.

[42] J. Shapiro P. Heller and R.O. Wells. Optimally smooth symmetric quadrature mirror filters for image coding, 1995. ftp://ftp.aware.com/pub/papers/heller.

[43] Timo Eirola. Sobolev characterization of solutions of dilation equations. *SIAM Journal of Mathematical Analysis*, 23(4):1015–1030, 1992.

[44] Lars Villemoes. Energy moments in time and frequency for two-scale difference equation solutions and wavelets. *SIAM Journal of Mathematical Analysis*, 23(6):1519–1543, 1992.

[45] M. Oslick S. Maslakovic, I. R. Linscott and J. D. Twicken. Smooth orthonormal wavelet libraries: Design and application. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, May 1998.

[46] W. Lawton. Applications of complex valued wavelet transformations to subband decomposition. *IEEE Trans. on Signal Processing.*, 41:3566–3568, Dec. 1993.

[47] H. Singh P. Heller, T. Nguyen and W. Carey. Linear-phase m-band wavelets with application to image coding, 1995. http://tweed.ee.cornell.edu/ knox.

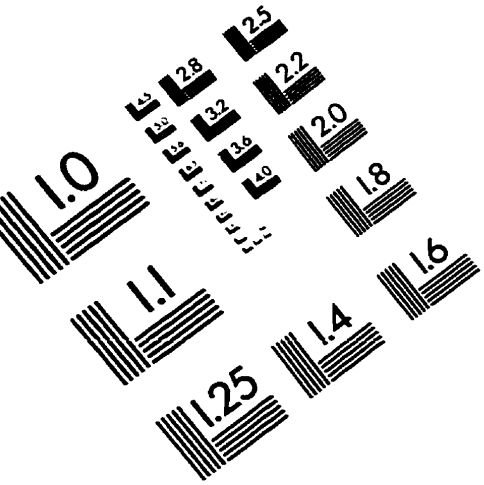
[48] I.S. Gradshteyn and I.M. Ryzhik. *Table of Integrals, Series and Products.* Academic Press Inc., Boca Raton, Florida, 1980.

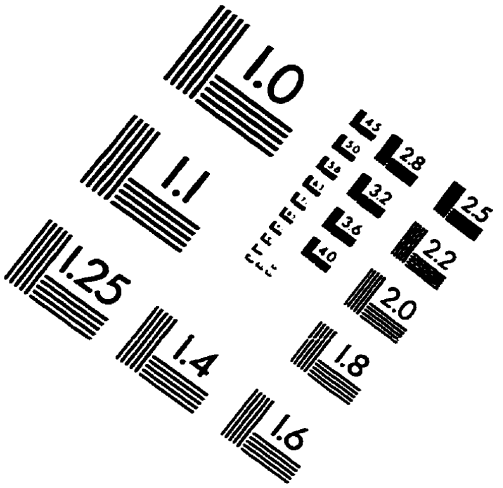
[49] W. H. Beyer. *CRC Standard Mathematical Tables.* CRC Press, Boca Raton, Florida, 1991.

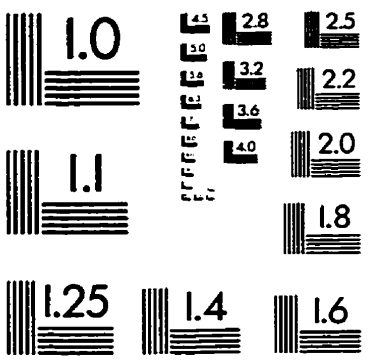
[50] F.Oberhettinger A Erdelyi, W.Magnus and F.G.Triconir. *Higher Transcendental Functions.* McGraw Hill Book Company Inc., 1953.

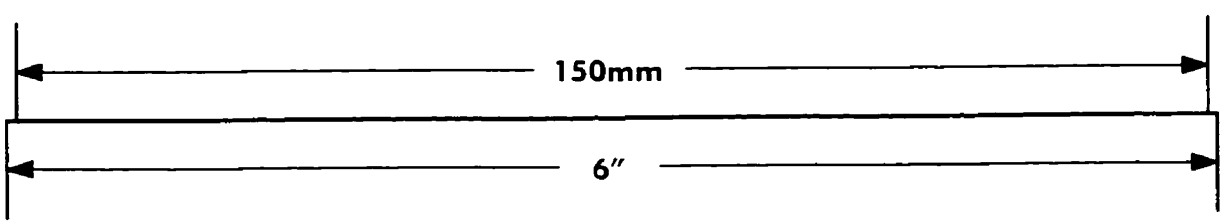
[51] Larry C. Andrews. *Special Functions for Engineers and Mathematicians.* Macmillan, New York, 1985.

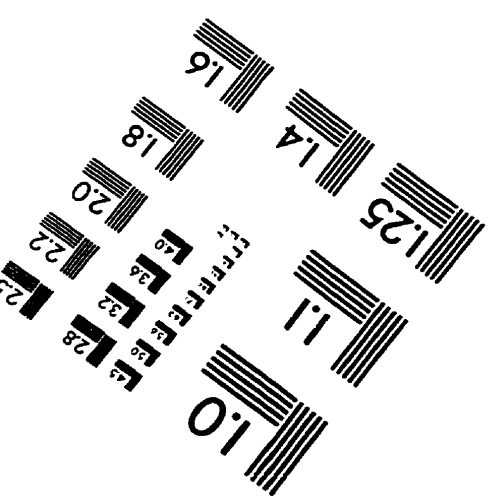
IMAGE EVALUATION
TEST TARGET (QA-3)

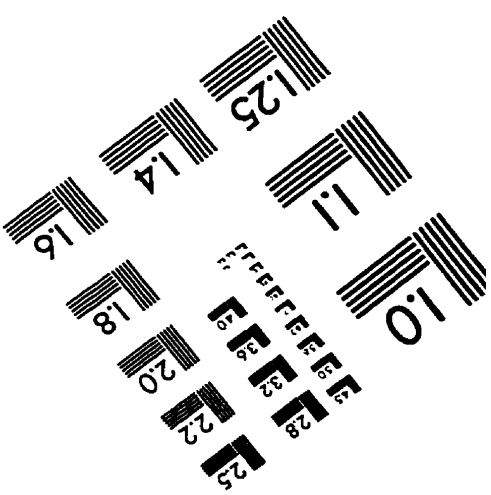












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989