

## INFORMATION TO USERS

The most advanced technology has been used to photograph and reproduce this manuscript from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

**The quality of this reproduction is dependent upon the quality of the copy submitted.** Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.



University Microfilms International  
A Bell & Howell Information Company  
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA  
313/761-4700 500/521-0600



Order Number 1342267

**The wavelet transforms and time-scale analysis of signals**

**Gopinath, Ramesh Ambat, M.S.**

Rice University, 1990

**U·M·I**  
300 N. Zeeb Rd.  
Ann Arbor, MI 48106



RICE UNIVERSITY

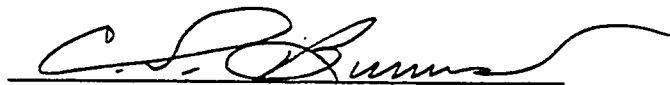
THE WAVELET TRANSFORMS AND  
TIME-SCALE ANALYSIS OF SIGNALS

by

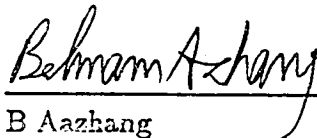
RAMESH A. GOPINATH

A THESIS SUBMITTED  
IN PARTIAL FULFILLMENT OF THE  
REQUIREMENTS FOR THE DEGREE  
MASTER OF SCIENCE

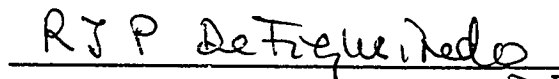
APPROVED, THESIS COMMITTEE:



C S Burrus, Chairman  
Professor of Electrical and Computer  
Engineering



B Aazhang  
Asst. Professor of Electrical and  
Computer Engineering



R J P DeFigueiredo  
Professor of Electrical Engineering and  
Mathematical Sciences

pp F. h. Tittel

Houston, Texas

January, 1990

# THE WAVELET TRANSFORMS AND TIME-SCALE ANALYSIS OF SIGNALS

RAMESH A. GOPINATH

## Abstract

Orthonormal wavelet bases provide an alternative technique for the analysis of non-stationary signals. Unlike the Gabor representation, the basis functions in the wavelet representation all have the same bandwidth on a logarithmic scale.

This thesis develops a general framework for the time-scale analysis of signals. In this context, the ON wavelets form a subclass of DWT wavelets. Efficient algorithms for the computation of the wavelet transforms are also developed.

As an application, we discuss the problem of detection of (wideband) signals subjected to scale-time perturbations. The probable unknown parameters for scale-time perturbed signals are the gain, and the scale and time perturbations. This problem is set in the context of classical composite hypothesis testing with unknown parameters, and depending on what the unknown parameters are, one of the wavelet transforms, developed is shown to naturally lead to a detector.

## Acknowledgments

I take this opportunity to thank my advisor Prof.C.S.Burrus who introducing me to the theory of wavelets, and creating the right environment for me to work in this area.

Special thanks to all at AWARE Inc., where most of this work was done. In particular I would like to thank W.M.Lawton and H.L.Resnikoff. The seminal ideas in this thesis owe their existence to W.M. Lawton. Besides, he was extremely helpful in clarifying most of the mathematical concepts of wavelet theory.

Staying in Cambridge and submitting the thesis in Houston entails a few administrative difficulties and I would like to thank the many people who helped in ironing out these difficulties: specifically Nora Quiocho and Rajat Mukherjee were of great help in this regard.

The funding for this research from AWARE Inc., Cambridge is gratefully acknowledged.

# Contents

Abstract	i
Acknowledgments	iii
List of Illustrations	vii
List of Tables	ix
<b>1 Introduction</b>	<b>1</b>
<b>2 Orthonormal Wavelet Bases - An Example</b>	<b>4</b>
2.1 The $\sin(x)/x$ wavelet basis . . . . .	4
<b>3 The Wavelet Transforms</b>	<b>11</b>
3.1 The Affine Group . . . . .	11
3.2 The Wavelet Transforms . . . . .	13
3.2.1 The Continuous Wavelet Transform . . . . .	14
3.2.2 The Discrete Wavelet Transform . . . . .	20
3.2.3 Properties of the DWT . . . . .	24
3.2.4 The Discrete-Time Wavelet Transform . . . . .	25
3.2.5 Properties of the DTWT . . . . .	27
3.2.6 The Discrete-Scale Wavelet Transform . . . . .	27
3.2.7 Properties of the DSWT . . . . .	29
<b>4 Orthonormal Wavelet Bases</b>	<b>30</b>
4.1 Orthonormal Wavelet Bases . . . . .	30
4.1.1 Multiscale Analysis . . . . .	30



4.1.2	Construction of the Wavelet Basis . . . . .	35
4.1.3	The Haar Example . . . . .	38
4.1.4	Daubechies Wavelets . . . . .	39
4.1.5	The Moments of $\phi$ and $\psi$ . . . . .	42
4.2	Multirate Filter Banks and Conjugate Quadrature Filters . . . . .	48
4.3	Computation with FIR CQF banks . . . . .	53
4.3.1	Lattice structures and CQFs . . . . .	58
4.4	Generalization of ON wavelet bases . . . . .	60
<b>5</b>	<b>Computational Techniques using ON Wavelets</b>	<b>62</b>
5.1	Generation of $\phi$ and $\psi$ . . . . .	62
5.1.1	Interpolation Method . . . . .	62
5.1.2	Infinite Product Method . . . . .	64
5.2	Analysis/Synthesis in Wavelet Bases . . . . .	65
5.2.1	The Projection onto the Finest Scale . . . . .	66
5.2.2	One level Analyzer . . . . .	68
5.2.3	One level Synthesizer . . . . .	71
5.2.4	Modified one level analyzer . . . . .	72
5.2.5	Modified one level synthesizer . . . . .	74
5.2.6	Expansion at any scaling or wavelet level . . . . .	75
5.3	Computation of the DWT . . . . .	75
5.4	The DTWT with respect to a DWT wavelet . . . . .	78
5.5	The DSWT with respect to a DWT wavelet . . . . .	79
5.6	The CWT with respect to a DWT wavelet . . . . .	80
5.7	Computation of the CWT . . . . .	81
5.8	Computation of the DTWT . . . . .	84
5.9	Computation of the DSWT . . . . .	84

<b>6</b>	<b>Detection of Time-Scale Perturbed Signals</b>	<b>85</b>
6.1	Introduction . . . . .	85
6.2	The Problem . . . . .	86
6.3	Detection and Estimation . . . . .	88
6.3.1	$\tau, u$ known . . . . .	90
6.3.2	$\tau$ unknown, $u$ known . . . . .	91
6.3.3	$\alpha$ known, $u$ unknown . . . . .	92
6.3.4	$\tau, u$ unknown . . . . .	92
6.4	Conclusion . . . . .	93
<b>A</b>	<b>Program Listing</b>	<b>94</b>
A.1	The Programs . . . . .	94
A.1.1	$h_0$ the scaling filter - daubcmf.m . . . . .	94
A.1.2	$\phi$ - phi.m . . . . .	94
A.1.3	$\psi$ - psi.m . . . . .	95
A.1.4	The one level Analyzer - latdec.m . . . . .	96
A.1.5	The one level Synthesizer - latrec.m . . . . .	97
A.1.6	The modified Analyzer - mlatdec.m . . . . .	98
A.1.7	The modified Synthesizer - mlatrec.m . . . . .	99
A.1.8	The DWT - dwt . . . . .	100
A.1.9	The DSWT with respect to $\psi$ - dswtpsi . . . . .	101
A.1.10	The DTWT with respect to $\psi$ - dtwtpsi . . . . .	102
A.1.11	The CWT with respect to $\psi$ - cwtpsi . . . . .	103
A.1.12	The CWT - cwt.c . . . . .	104
	<b>Bibliography</b>	<b>110</b>

## Illustrations

2.1	Frequency Channels in STFT . . . . .	5
2.2	Frequency channels in a Wavelet Representation . . . . .	7
2.3	The Sinc Scaling Function . . . . .	9
2.4	The Sinc Wavelet . . . . .	9
3.1	Fourier Transform of a Wavelet at different Scales . . . . .	17
3.2	The STFT Time-frequency Resolution Cells . . . . .	17
3.3	The CWT Time-frequency Resolution Cells . . . . .	18
3.4	Relationship between Frames and Bases . . . . .	21
4.1	The Haar Scaling Function . . . . .	40
4.2	Approximation using Haar basis . . . . .	40
4.3	The Haar Wavelet . . . . .	41
4.4	Scaling function and Wavelet - D6 . . . . .	43
4.5	Scaling function and Wavelet - D8 . . . . .	43
4.6	A Multirate Filter Filter Bank . . . . .	49
4.7	The Mirror Property of QMFs . . . . .	50
4.8	Polyphase Implementation of an Analysis Filter Bank . . . . .	55
4.9	Polyphase Implementation of an Synthesis Filter Bank . . . . .	57
5.1	Generation of $\phi$ using infinite product . . . . .	65
5.2	The One Level DWT analyzer . . . . .	69

5.3	Lattice Analyzer of Two Channel CQF . . . . .	71
5.4	The One Level Synthesizer . . . . .	73
5.5	Modified One Level Analyzer . . . . .	74
5.6	Modified One Level Synthesizer . . . . .	76
6.1	Matched filter Receiver . . . . .	89

## Tables

4.1	The filters associated with Daubechies wavelets . . . . .	44
4.2	The moments of $\phi$ . . . . .	47

# Chapter 1

## Introduction

For many signal processing applications the representation of a signal in the time or the frequency domain is not adequate. One needs to study the frequency content in a signal locally in time. In such cases a mixed time-frequency representation is desirable and sometimes necessary [10]. Typically such time-frequency representations are useful in the analysis of non-stationary or time-varying signals like speech [8], music, seismic signals [19], underwater acoustic signals [13] etc. In the past a number of time-frequency representations like the Short Time Fourier Transform (also referred to as the Gabor Transform when the window used in the STFT is Gaussian), the Wigner-Ville distribution [19], the ambiguity function [19] have been used to gain insight into the signal characteristics of non-stationary signals. Perhaps due to simplicity, the Gabor Transform is the most popular time-frequency representation.

All the different time-frequency representations take a function of one variable, *time*, and map it into a function of two variables, corresponding to *time* and *frequency*. Because of this, there is some redundancy in these representations. Otherwise all functions of two variables would correspond to the time-frequency representation of a function of one variable! This redundancy has a number of characterizations the simplest of which is that the time-bandwidth product, the smallest resolution cell in time-frequency, is greater than  $2\pi$ .

This redundancy enables us, in some cases, to sample the representation at a discrete set of points and yet be able to recover the original signals completely. The ability to discretize the representation also means that one can store and manipulate the representation on a computer.

If localization (with respect to variance as a measure) in time-frequency is the goal, then the Wigner Distribution is the most appropriate [6, 3, 5, 4, 2, 20, 19]. However, since the Wigner Distribution is bilinear, it is difficult to interpret the transform [20]. The Choi-Williams distribution partially solves this by suppressing the bilinear terms [17, 6]. Besides there are no easy inversion formulas for these transforms.

The STFT, while it can be discretized, and easily interpreted, suffers from the ad hoc choice of a window, the choice of which determines the resolution in time-frequency. This inflexibility in time-frequency is undesirable for many classes of time-varying signals [13]. Furthermore the reconstruction of the signal from the STFT, may in some cases involve numerical instabilities [16]. A excellent and exhaustive survey of time-frequency distributions can be found in [6].

This thesis develops the recently discovered wavelet family of functions, (which lead to a time-scale representation) as an alternative to time-frequency analysis. Wavelets are analogous to windows in the STFT, and viewed in terms of time and frequency, they give flexibility in the time and frequency resolution. Most of the published work on wavelets [7] is centered around orthonormal wavelet bases and their properties, the expansion coefficients in terms of this basis and algorithms for their computation etc. ON wavelets in themselves have a number of properties that make the wavelet coefficients an attractive alternative to the Discrete STFT in the representation of signals.

1. The analysis and reconstruction algorithms with ON wavelets are efficient and can be implemented using efficient lattice implementations of Conjugate Mirror Filters(a class of multirate filters that have been receiving considerable attention recently [32]).
2. The wavelet coefficients of a real signal is real
3. The wavelet functions have constant bandwidth on a *log* scale and hence there is some flexibility in time and frequency resolution.

Despite all these attractive features, the wavelet representation (with only ON wavelets) suffers from the lack of translation invariance. The wavelet coefficients of a signal and an arbitrarily delayed version of it are very different. To solve this we introduce the wavelet family of transforms, a generalization of the ON wavelet representation. We define the transforms, give relationships between them, and develop efficient numerical techniques for their computation. The efficient algorithms developed depend on the attractive computational techniques that are associated with ON wavelets. Finally this thesis studies the detection of time-scale perturbed wideband signals submerged in additive white Gaussian noise and gives detectors based on the wavelet transforms developed. whose scale and time origin of occurrence are unknown. The time-scale analysis framework could be used in the detection of *unknown* signals like passive sonar data etc.

The thesis is organized as follows. The second chapter introduces an ON wavelet bases with an example that is illustrative of the techniques involved in wavelet based analysis of signals. The third chapter introduces the wavelet family of transforms and discusses their properties. The fourth chapter discusses orthonormal wavelet bases. The fifth chapter develops efficient computational techniques with wavelets. And the last chapter discusses detection algorithms for signals subject to time-scale perturbations.



## Chapter 2

### Orthonormal Wavelet Bases - An Example

Though the wavelet representation is a time-scale representation, it is very closely related to the Short Time Fourier Transform, which is a familiar tool for analyzing non-stationary signals. In this chapter we introduce an ON wavelet basis, that brings out the essential similarities and differences between the wavelet coefficients and the Discrete STFT coefficients. The time and frequency resolution tradeoff that is permitted in the wavelet representation is also shown.

#### 2.1 The $\sin(x)/x$ wavelet basis

Let  $f(t)$  be a finite energy signal (i.e  $f(t) \in L^2(\mathbf{R})$ ). The Fourier transform of  $f(t)$  is given by

$$\hat{f}(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-i\omega t} dt \quad (2.1)$$

where  $i$  is  $\sqrt{-1}$ . By the uniqueness of the Fourier transform  $\hat{f}$  uniquely determines  $f$ .

Now we will construct a local time-frequency representation. Let the frequency axis be split into a set of contiguous bands,  $\beta_j$ , centered at the integral multiples of  $2\pi$  and each of bandwidth  $2\pi$  radians (i.e 1 Hz) as shown in Fig. 2.1. Clearly the bands are non-overlapping the entire frequency axis is covered by these bands. Indeed

$$\beta_j(\omega) = \Pi\left(\frac{\omega - 2\pi j}{2\pi}\right) \quad (2.2)$$

where  $\Pi(t)$  is defined by

$$\Pi(t) = \begin{cases} 1 & |t| < \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \quad (2.3)$$

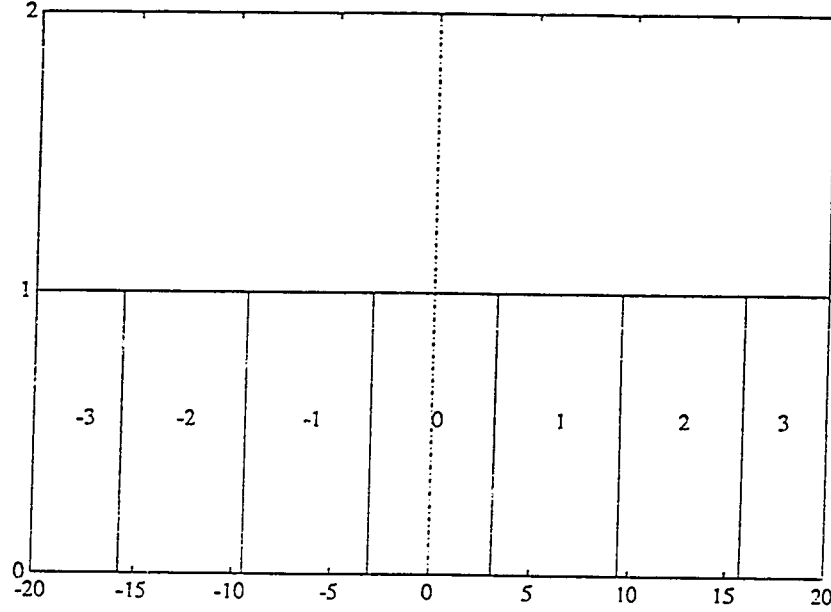


Figure 2.1: Frequency Channels in STFT

Now  $\hat{f}$ , is decomposed into its components,  $\hat{f}_j$  each confined to band  $\beta_j$ . Indeed, we have

$$\hat{f}(\omega) = \sum_{j=-\infty}^{j=+\infty} \hat{f}_j(\omega) \quad (2.4)$$

If  $\hat{f}_j$  is represented as a Fourier series confined to the band  $j$ , it follows that

$$\hat{f}_j(\omega) = \sum_{k=-\infty}^{k=+\infty} a_{j,k} e^{i\omega k} \Pi\left(\frac{\omega - j2\pi}{2\pi}\right) \quad (2.5)$$

Hence  $\hat{f}$  becomes

$$\hat{f}(\omega) = \sum_{j=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} a_{j,k} e^{i\omega k} \Pi\left(\frac{\omega - j2\pi}{2\pi}\right) \quad (2.6)$$

Taking the inverse Fourier transform on both sides

$$f(t) = \sum_{j,k} a_{j,k} \frac{\sin(\pi(t+k))}{\pi(t+k)} e^{i2\pi j t} \quad (2.7)$$

This equation is precisely the defining equation for the Discrete STFT of  $f(t)$  with respect to the window

$$w(t) = \frac{\sin(\pi t)}{\pi t} = \text{sinc}(t) \quad (2.8)$$

The window, a *sinc*, is clearly concentrated around  $t = 0$  in time, and  $\omega = 0$  in frequency. Thus the basis function corresponding to  $j$  and  $k$ , namely  $\frac{\sin(\pi(t-k))}{\pi(t-k)}$ , is concentrated at  $(j, k)$  in time frequency. Furthermore, the basis functions are mutually orthogonal.

The above orthonormal basis for  $L^2(\mathbf{R})$  is obtained by initially dividing the frequency domain into an infinite number of equally spaced bands. The absence of *holes* in frequency enables us to obtain a complete representation for  $f$ . It is conceivable that the frequency domain could be split in many other ways, each leading to an orthonormal basis for  $L^2(\mathbf{R})$ , and each having different time-frequency localization characteristics.

Why would one be interested in doing this? The physiology of early mammalian vision indicates that the retinal image is decomposed spatially-oriented frequency channels each having the same bandwidth on a log scale [30]. That is, the retinal system, splits the visual input into information at different scales. Furthermore, in many naturally occurring classes of signals like transients, it has been observed that the larger the center frequency of the signal, the smaller its duration, and hence greater its bandwidth. Such a frequency behavior has been found to be useful in the modeling of nearly  $\frac{1}{f}$  noise processes [35]

This suggests splitting the frequency domain into channels as shown in Fig. 2.2. As in Eqn. 2.4,  $\hat{f}$  can be represented as a Fourier series in each of the bands, say  $\beta_j$ . However, the different bands have different fundamental periods. It is useful to consider the  $\beta_j$  as the difference between two  $\cap$  functions centered at  $\omega = 0$ , with support-widths  $2^{j+1}\pi$  and  $2^j\pi$  respectively. Let  $\chi_j(\omega)$  be defined by

$$\chi_j(\omega) = \cap\left(\frac{\omega}{2^{j+1}\pi}\right) \quad (2.9)$$

Then the  $j^{th}$  band, say  $\beta_j$  is given by

$$\beta_j(\omega) = \chi_j(\omega) - \chi_{j-1}(\omega)$$

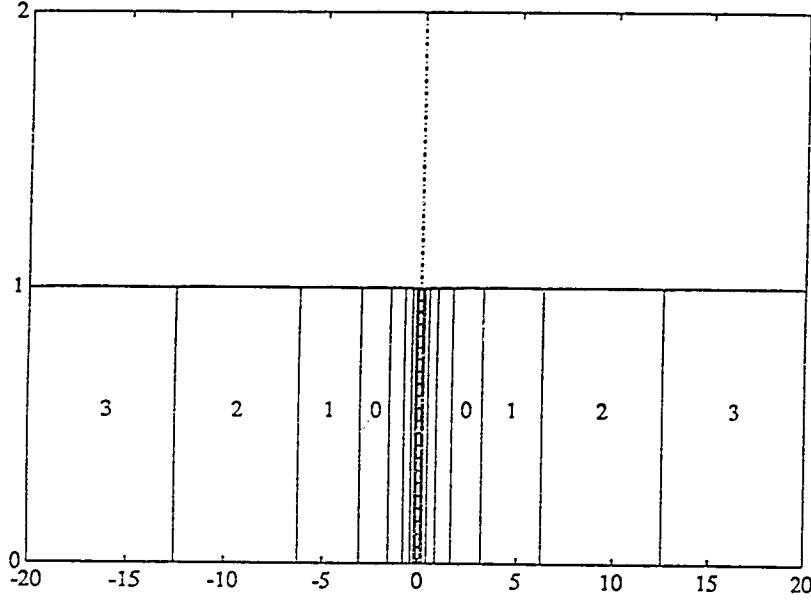


Figure 2.2: Frequency channels in a Wavelet Representation

Now the Fourier transform of  $f$  in band  $\beta_j$  can be written as

$$\hat{f}_j(\omega) = \beta_j(\omega) \sum_k b_{j,k} e^{i\omega k/2^j} \quad (2.10)$$

where  $\{b_{j,k}\}$  are the Fourier coefficients of  $\hat{f}$  restricted to  $\chi_j$ . From the conjugate symmetry of the  $\hat{f}$  (remember  $f$  is real), we see that the  $b_{j,k}$ 's are real. Now by summing over all  $j$  and taking the Fourier transform on both sides we get

$$f(t) = \sum_{j,k} b_{j,k} \left\{ \left[ \frac{\sin(2^j \pi(t - 2^{-j}k))}{\pi t} \right] - \left[ \frac{\sin(2^{j-1} \pi(t - 2^{-j}k))}{\pi t} \right] \right\} \quad (2.11)$$

We have obtained a local representation of  $f(t)$ . Now let us define the following :

$$\phi(t) = \frac{\sin(\pi t)}{\pi t} \quad (2.12)$$

$$\psi(t) = 2\phi(2t) - \phi(t) \quad (2.13)$$

$$\phi_{j,k}(t) = 2^{j/2} \phi(2^j t - k) \quad (2.14)$$

$$\psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k) \quad (2.15)$$

In terms of  $\{\psi_{j,k}\}$  we can express  $f(t)$  as

$$f(t) = \sum_{j,k} a_{j,k} \psi_{j,k} \quad (2.16)$$

Here the  $a_{j,k}$ 's are scaled versions of  $b_{j,k}$ 's. In fact

$$a_{j,k} = \sqrt{2^{-j}} b_{j,k} \quad (2.17)$$

where the factor  $\sqrt{2^{-j}}$  normalizes the expansion functions. Eqn. 2.16 is the *wavelet* expansion of a function  $f(t)$  with respect to the *wavelet*  $\psi$ . The basis functions are translates and dilates of a single function  $\psi$ . In the case of the STFT, the basis functions are translates and modulates of the window.

Notice the wavelet  $\psi(t)$  can be written as the difference between a function  $\phi(t)$  and  $\phi(2t)$ . Such a function  $\phi$  that is related to the wavelet is called the scaling function, and it plays an important role in the construction of, and analysis in orthonormal wavelet bases.

In the specific ON wavelet basis developed,  $\psi$  and  $\phi$  are ideal continuous time bandpass and lowpass filters respectively. The functions  $\psi(t)$  and  $\phi(t)$  satisfy the following equations.

$$\phi(t) = 2 \sum_k \frac{\sin \pi k/2}{\pi k} \phi(2t - k) \quad (2.18)$$

and

$$\psi(t) = 2 \sum_k (-1)^k \frac{\sin \pi(k+1)/2}{\pi(k+1)} \phi(2t - k) \quad (2.19)$$

Fig. 2.3 and Fig. 2.4 shows the scaling function and the wavelet respectively. Let  $V_j$  denote the space of all functions spanned by  $\{\phi_{j,k}\}$  for fixed  $j$ . Then  $V_j$  is the space of all real functions that are band-limited to  $2^j\pi$ . Let  $W_j$  denote the space of all functions spanned by  $\{\psi_{j,k}\}$  for fixed  $j$ . Then  $W_j$  is the set of all real functions that are bandlimited to  $[2^j\pi, 2^{j+1}\pi]$ . Therefore Eqn. 2.18 implies  $V_j \subset V_{j+1}$  and Eqn. 2.19 implies that  $W_j \subset V_{j+1}$ .

Furthermore the construction of  $\psi_{j,k}$  implies that  $\{\psi_{j,k}\}$  is an orthonormal basis for  $L^2$ , and that, for fixed  $j$ ,  $\{\psi_{j,k}\}$  and  $\{\phi_{j,k}\}$  are mutually orthonormal systems.

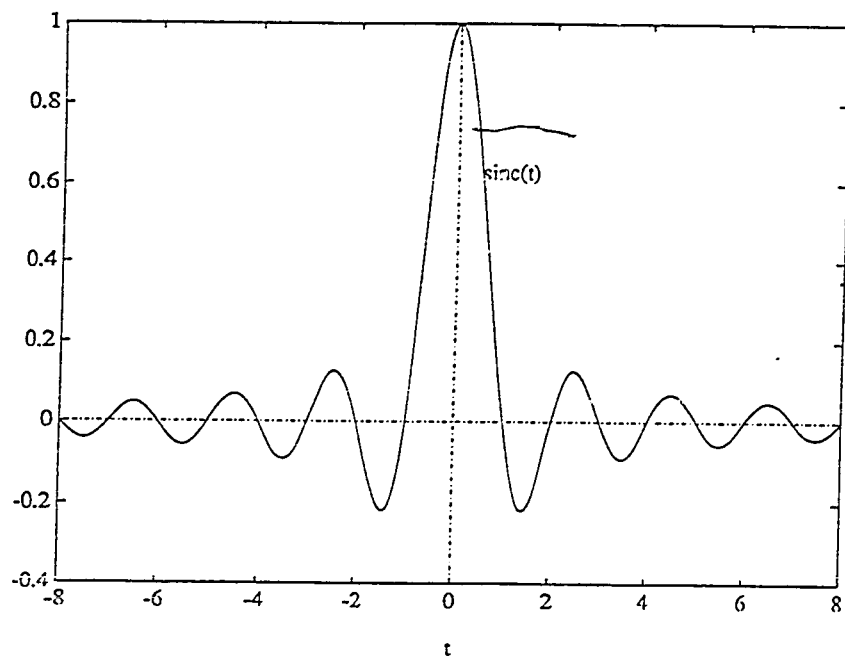


Figure 2.3: The Sinc Scaling Function

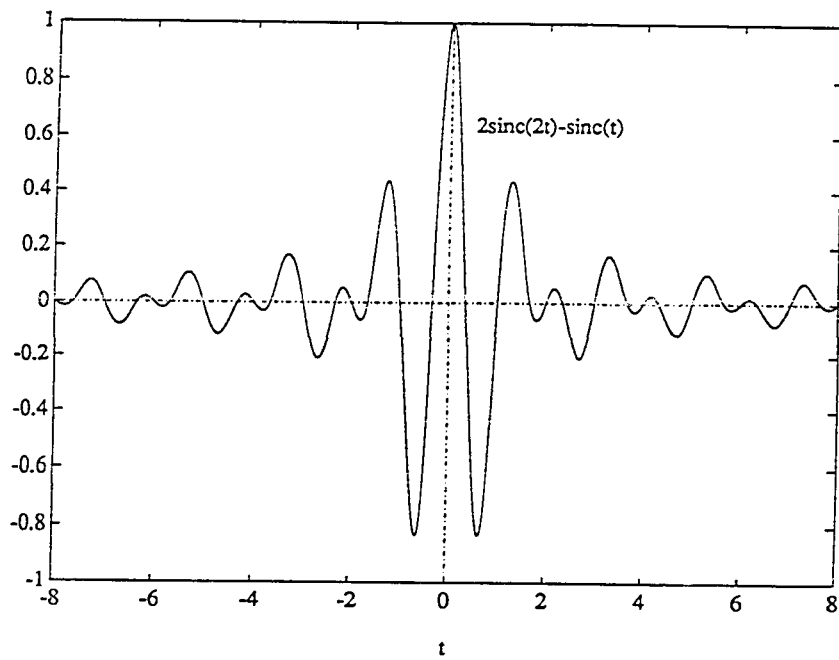


Figure 2.4: The Sinc Wavelet

This example has most of the features of the orthonormal wavelet basis that will be used in this thesis in the analysis of transient signals. Notice that the supports of  $\psi$  and  $\phi$  introduced here are of infinite support. Later we will see compactly supported wavelet and scaling functions.

## Chapter 3

### The Wavelet Transforms

The *wavelet transforms* are the transformations of a finite energy signal into a time-scale representation that is related to a certain function called the *wavelet*. The two parameters of the transform, namely scale and time, could be both continuous, both discrete or mixed; that is, the one could be continuous and the other discrete. Corresponding to these four cases we identify four wavelet transforms. This is somewhat analogous to the continuous and discrete Short Time Fourier Transforms (STFTs). Chapter 2 introduced an ON wavelet basis, where the wavelet function was a difference of sines. This chapter introduces the various wavelet transforms, characterizes the wavelets that give rise to these transforms and discusses their properties. Since the goal is to introduce time-scale analysis as an alternative to conventional time-frequency analysis, whenever possible the similarities and differences between the STFT and the wavelet representation will be mentioned.

#### 3.1 The Affine Group

The window function in the STFT is operated on by translations and modulations (by the complex exponential) to generate the STFT basis. Most of the properties of the STFT can be directly linked to the nature of the underlying operations of translation and modulation, which form the Weil-Heisenberg group.

In an analogous way the wavelet representation is obtained by the action of translations and dilations on the basic wavelet function. The group of translations and dilations is called the affine group. The affine group will be parametrized by  $u$  (corresponding to scale) and  $\tau$  (corresponding to time).



**Definition 3.1.1** The affine group  $G_0$  is made up of elements of the form  $(u, \tau)$ ,  $u, \tau \in \mathbf{R}$ , where the group operation is defined by

$$(u_1, \tau_1)(u_2, \tau_2) = (u_1 + u_2, 2^{u_2}\tau_1 + \tau_2) \quad (3.1)$$

It can easily be verified that the set  $G_0$ , with the group operation defined as above forms a non-commutative group with identity element  $(0, 0)$ . The inverse of an element  $u, \tau$  in  $G_0$  is given by

$$(u, \tau)^{-1} = (-u, -2^{-u}\tau) \quad (3.2)$$

since

$$(u, \tau)(-u, -2^{-u}\tau) = (u - u, 2^{-u}\tau - 2^{-u}\tau) = (0, 0)$$

Let  $G_1, G_2$  and  $G_3$  be respectively, the subsets of  $G$  obtained by restricting  $u, \tau$  or both to take on only integral values.

$$G_1 = \{(j, \tau) | \tau \in \mathbf{R}, j \in \mathbf{Z}\} \quad (3.3)$$

$$G_2 = \{(u, k) | u \in \mathbf{R}, k \in \mathbf{Z}\} \quad (3.4)$$

$$G_3 = \{(j, k) | j, k \in \mathbf{Z}\} \quad (3.5)$$

The wavelet transforms developed will be parametrized by one of the four sets  $G_0$  through  $G_3$ . Let  $L(u, \tau) : L^2(\mathbf{R}) \mapsto L^2(\mathbf{R})$ , and  $R(u, \alpha) : L^2(\mathbf{R}) \mapsto L^2(\mathbf{R})$  be defined by

$$L(u, \tau)f(t) = 2^{u/2}f(2^u t - \tau) \quad (3.6)$$

$$R(u, \alpha)f(t) = 2^{u/2}f(2^u(t - \alpha)) \quad (3.7)$$

It can be verified that both  $L(u, \tau)$  and  $R(u, \alpha)$  act unitarily on  $L^2(\mathbf{R})$ . Moreover,  $L$  and  $R$  are related by the equation

$$L(u, \tau) = R(u, 2^{-u}\tau) \quad (3.8)$$

Consider the action of  $L(u_1, \tau_1)$  on  $L(u_2, \tau_2)f$ .

$$\begin{aligned}
 L(u_1, \tau_1)L(u_2, \tau_2)f(t) &= L(u_1, \tau_1)2^{u_2/2}f(2^{u_2}t - \tau_2) \\
 &= 2^{u_1/2}2^{u_2/2}f(2^{u_2}(2^{u_1}t - \tau_1) - \tau_2) \\
 &= L(u_1 + u_2, 2^{u_2}\tau_1 + \tau_2)f(t) \\
 &= L((u_1, \tau_1)(u_2, \tau_2))f(t)
 \end{aligned}$$

Thus  $L(u, \tau)$  is the affine group acting as an operator on the space  $L^2(\mathbf{R})$ . The operators  $L(u, \tau)$  and  $R(u, \alpha)$  are very useful in the development of the wavelet transforms and hence we note relevant properties.

Let  $f, g \in L^2(\mathbf{R})$ . Let  $f_T = f(t - T)$  and  $f_S = 2^{S/2}f(2^S t)$ . We have

**1. Translation property:**

$$L(u, \tau)f_T = L(u, \tau + T)f \quad (3.9)$$

$$R(u, \alpha)f_T = R(u, \alpha + 2^{-u}T) \quad (3.10)$$

**2. Scaling Property:**

$$L(u, \tau)f_S = L(u + S, 2^S \tau)f \quad (3.11)$$

$$R(u, \alpha)f_S = L(u + S, \alpha)f \quad (3.12)$$

**3. The Unitariness Property:**

$$\langle f, L(u, \tau)g \rangle = \langle L(u, \tau)f, g \rangle = \langle R(-u, -\tau)f, g \rangle \quad (3.13)$$

These properties are direct consequences of the definitions.

### 3.2 The Wavelet Transforms

The term *wavelet* as mentioned earlier has been used loosely in the literature. Here we give precise definitions and try to motivate the restrictions imposed on a function

to make it a wavelet. In the function domain the index is  $\mathbf{R}$  (usually time), while in the wavelet transform domain the index is the affine group  $G_0$  or one of its subsets  $G_1, G_2$  or  $G_3$ . The transform should be linear and energy preserving. That is the wavelet transform should be a linear operator from the space  $L^2(\mathbf{R})$  onto the space  $L^2(\mathbf{R} \times \mathbf{R})$  that is a multiple of an isometry.

### 3.2.1 The Continuous Wavelet Transform

As mentioned earlier the wavelet is analogous to the window in the STFT. But unlike in the STFT case, where the window could be any function in  $L^2(\mathbf{R})$ , wavelets are characterized by a certain restriction referred to as the admissibility condition. Group-theoretic arguments for the need for this condition in the case of the CWT can be found in [15]. Here we will impose some desirable properties on the transform and the admissibility condition will arise as a natural consequence of this. We begin by defining the CWT wavelet.

**Definition 3.2.1** A CWT wavelet (or simply wavelet), is a function  $w \in L^2(\mathbf{R})$ , such that for all  $f \in L^2(\mathbf{R})$  we have the following identity

$$f(t) = \frac{1}{c_w} \int_{\mathbf{R}^3} f(y) 2^u w(2^u t - \tau) w(2^u y - \tau) d\tau du dy \quad (3.14)$$

where  $c_w$  is a constant depending only on  $w$ .

The above condition on the CWT wavelet is not easy to work with. We would like a simpler characterization. The reason for incorporating such a complicated definition is that for other characterizations we have to derive this as a result and we wish to avoid this, though it is actually a matter of taste.

If we define  $R_w(t)$  to be the autocorrelation function of  $w$ , then Eqn. 3.14, on changing the order of integration becomes

$$f(t) = \frac{1}{c_w} \int_{\mathbf{R}} dy \int_{\mathbf{R}} du f(y) 2^u R_w(2^u(t - y))$$

which automatically implies that

$$\int 2^u R_w(2^u t) du = c_w \delta(t) \quad (3.15)$$

Taking the Fourier Transform on both sides it can be shown that

$$\int_{\mathbf{R}} \frac{|\hat{w}(\omega)|^2}{|\omega|} d\omega = c_w \log 2 \quad (3.16)$$

It is also clear that if  $w$  satisfies Eqn. 3.15 then Eqn. 3.14 is automatically satisfied.

Hence we have proved the following lemma

**Lemma 3.2.1** Let function  $w \in L^2(\mathbf{R})$ . Let  $R_w(t)$  be the autocorrelation of  $w$ . Then the following statements are equivalent

1.  $w$  is a CWT wavelet

2.

$$\int_{\mathbf{R}} \frac{|\hat{w}(\omega)|^2}{|\omega|} d\omega = c_w \log 2 < \infty$$

3.

$$\int 2^u R(2^u t) du = c_w \delta(t)$$

This condition imposed on  $w$ , usually in the form (2) above, is referred to as the admissibility condition in the wavelet literature. The admissibility condition forces  $\hat{w}(0)$  (in all practical cases) to be 0. Thus loosely speaking a wavelet is a bandpass filter. Moreover every bandpass filter is a wavelet.

In terms of a CWT wavelet the CWT is defined as follows,

**Definition 3.2.2** Let  $f \in L^2(\mathbf{R})$ . The CWT or the  $(u, \tau)$  transform of  $f$  with respect to a CWT wavelet  $w$  is given by

$$W(u, \tau, f, w) = \int f(t) 2^{u/2} w(2^u t - \tau) = \langle f, L(u, \tau) w \rangle \quad (3.17)$$

For comparison, we give the equation for the STFT

$$STFT(\omega, \tau, f, w) = \int f(t) w^*(t - \tau) e^{-i\omega(t-\tau)} dt \quad (3.18)$$

Here  $w$  is *any* finite energy window.

While the latter gives the decomposition of a signal into a set of equal bandwidth filters sweeping over all frequency, the former gives the decomposition of a signal into a continuous parameter ( $u$ ) set of constant  $Q$  (or equal bandwidth on a logarithmic scale) bandpass channels. The constancy of bandwidth on a logarithmic scale can be seen from the following equation.

$$\hat{w}_u(\omega) = \frac{1}{\sqrt{2^u}} \hat{w}\left(\frac{\omega}{2^u}\right) \quad (3.19)$$

where  $w_u(t) = w(2^u t)$ . Fig. 3.1 shows the magnitude spectrum of the Fourier Transform of a typical wavelet function for three different values of  $u$  at three consecutive integers. The roles played by the parameters is also different. The time parameter  $\tau$  in the STFT refers to the actual time instant in the signal, while the parameter  $\tau$  in the CWT refers to time instant  $2^{-u}\tau$ . In other words, the time parameter in the wavelet representation measure time using a yardstick that is local in scale. This makes sense, for if the frequency scale in which a signal resides is coarse or fine, then the time scale should be appropriately big or small. This is the primary reason for the efficiency of the wavelet transform domain as a representation of the signal relative to the STFT. This fact also suggests that a multirate system would be involved in a wavelet representation.

Fig. 3.2 and Fig. 3.3 show the fundamental difference between the STFT and the CWT. There is an obvious time-frequency resolution trade-off. To understand how the CWT spans the time-frequency plane, the notions of resolution in time and resolution in frequency must be defined. Let  $\sigma_t$  and  $\sigma_\omega$  be the standard deviations in the time and the frequency domain. That is,

$$\sigma_t^2 = \frac{\int (t - \bar{t})^2 |w(t)|^2 dt}{\int |w(t)|^2 dt} \quad (3.20)$$

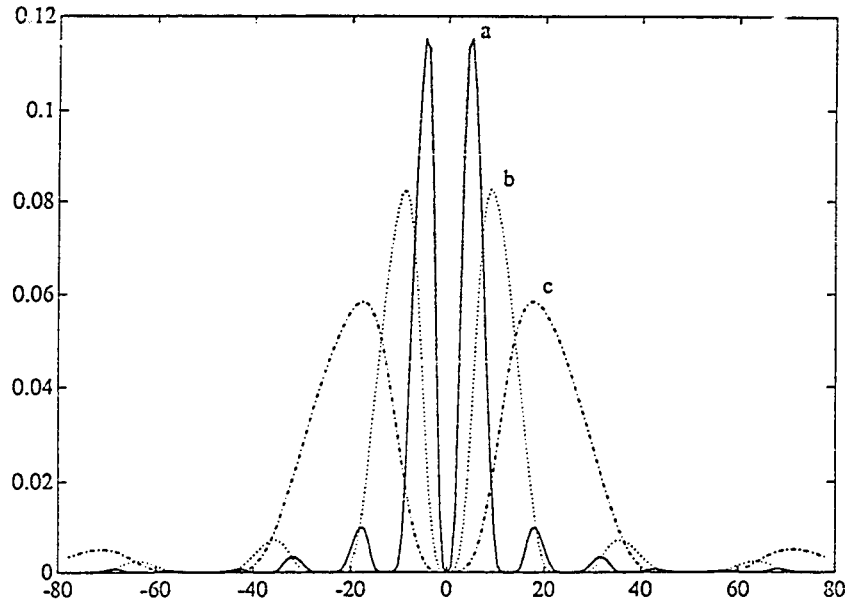


Figure 3.1: Fourier Transform of a Wavelet at different scales.  $a$ ,  $b$  and  $c$  are the Fourier Transform of the same signal  $w(t)$  at three scales at a spacing of an octave

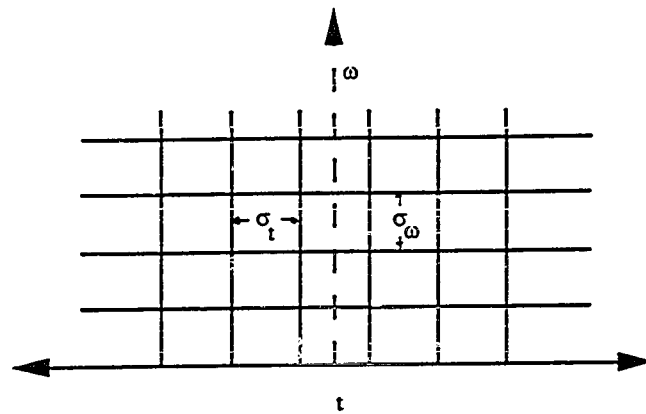


Figure 3.2: The STFT Time-frequency Resolution Cells

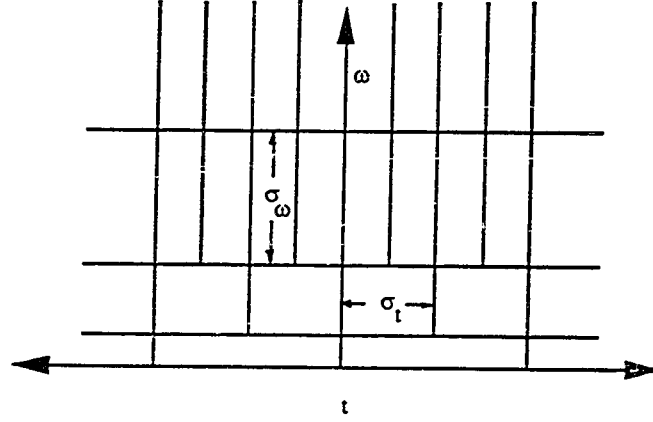


Figure 3.3: The CWT Time-frequency Resolution Cells

and

$$\sigma_\omega^2 = \frac{\int (\omega - \bar{\omega})^2 |\hat{w}(\omega)|^2 d\omega}{\int |\hat{w}(\omega)|^2 d\omega} \quad (3.21)$$

where

$$\bar{t} = \frac{\int t |w(t)|^2 dt}{\int |w(t)|^2 dt} \quad (3.22)$$

$$\bar{\omega} = \frac{\int \omega |\hat{w}(\omega)|^2 d\omega}{\int |\hat{w}(\omega)|^2 d\omega} \quad (3.23)$$

Let  $w(t)$  be centered at  $0, \bar{\omega}$  in time-frequency. Then  $w_u(t - 2^{-u}\tau)$  is centred at  $0, 2^u\bar{\omega}$  with standard deviations  $\sigma_t/2^u$  and  $2^u\sigma_\omega$ , in time and frequency respectively.

### Properties of the CWT

**Inverse Transform:** From the defining equation we already have that the inverse is the adjoint and hence the inverse transform is given by the equation

$$f(t) = \frac{1}{c_w} \int_{\mathbf{R}} du \int_{\mathbf{R}} d\tau \sqrt{2^u} w(2^u t - \tau) W(u, \tau, f, w) \quad (3.24)$$

**Energy Preserving Property:** For every  $f \in L^2(\mathbf{R})$ ,  $W(u, \tau, f, w)$  satisfies

$$\int_{\mathbf{R}} du \int_{\mathbf{R}} d\tau |W(u, \tau, f, w)|^2 = c_w \int_{\mathbf{R}} |f(t)|^2 dt \quad (3.25)$$

where  $c_w$  is given by

$$c_w = \frac{1}{\log 2} \int_{\mathbf{R}} \frac{|\hat{w}(\omega)|^2}{|\omega|} d\omega \quad (3.26)$$

This can be first verified for  $f(t) = w(t)$  and then for  $f(t) = w(2^u t - \tau)$ , for arbitrary constants  $u$  and  $\tau$ . Then the relationship is seen to hold for any linear combination of functions of the type  $w(2^u t - \tau)$ . By continuity this holds for all functions in  $L^2(\mathbf{R})$ .

A direct proof that brings out the need for the admissibility condition is as follows,

$$\begin{aligned} LHS &= \int_{\mathbf{R}} du \int d\tau \int dx \int dt f(t) \sqrt{2^u} w(2^u t - \tau) f(x) \sqrt{2^u} w(2^u x - \tau) \\ &= \int du 2^u \int dx \int dt \frac{1}{2\pi} [\int d\omega |\hat{w}(\omega)|^2 e^{i\omega 2^u (t-x)}] \\ &= \frac{1}{2\pi} \int ds \int d\omega |\hat{w}(\omega)|^2 |\hat{f}(\omega)|^2 \\ &= \frac{1}{\log 2} \int d\omega \frac{|\hat{w}(\omega)|^2}{|\omega|} \int dt |f(t)|^2 \\ &= c_w \int dt |f(t)|^2 \end{aligned}$$

The last step requires treating the cases where  $\omega$  is negative and positive separately.

**Redundancy of the CWT:** The redundancy of the CWT, as in the case of the STFT, is expressed in the following reproducing kernel equation.

**Theorem 3.2.1** Let the wavelet kernel  $K$  given by

$$K(u_1, \tau_1, u_2, \tau_2) = \int_{\mathbf{R}} \sqrt{2^{u_1+u_2}} w(2^{u_1} t - \tau_1) w(2^{u_2} t - \tau_2) dt$$

Then  $K$  satisfies the following equation.

$$W(u_1, \tau_1, f, w) = \int_{\mathbf{R}} du_2 \int_{\mathbf{R}} d\tau_2 W(u_2, \tau_2, w, f) K(u_1, \tau_1, u_2, \tau_2) \quad (3.27)$$

For time-scale analysis the most important properties are what happens to the CWT of a signal that is either translated or dilated.

**Translation of  $f$ :** Let  $f_T(t) = f(t - T)$ . Then,

$$W(u, \tau, f_T, w) = W(u, \tau + 2^u T, f, w) \quad (3.28)$$



Indeed we have

$$W(u, \tau, f_T, w) = \langle f_T, L(u, \tau)w \rangle$$

Using Eqn. 3.13

$$W(u, \tau, f_T, w) = \langle R(-u, -\tau)f, w \rangle$$

which becomes

$$W(u, \tau, f_T, w) = \langle R(-u, -\tau + 2^u T), w \rangle$$

Using Eqn. 3.13 again we obtain the desired result.

**Scaling of  $f$ :** Let  $f_S(t) = 2^{S/2} f(2^S t)$ . Then,

$$W(u, \tau, f_S, w) = W(u - S, \tau, f, w) \quad (3.29)$$

This is obtained by invoking Eqn. 3.13 and Eqn. 3.12.

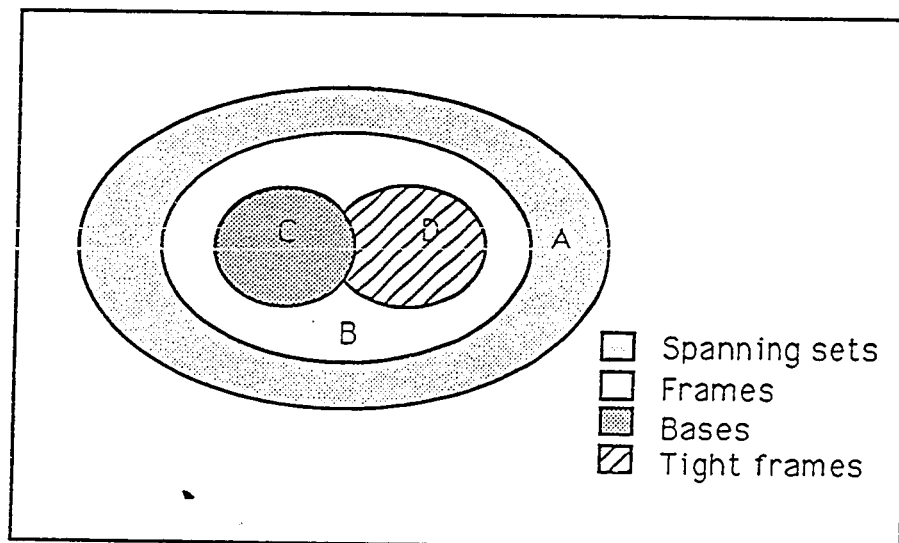
### 3.2.2 The Discrete Wavelet Transform

The CWT wavelet under mild conditions can be used to obtain discretizations that gives rise to stable inverses. For such a function  $w$ , the set of all functions  $\{L(j, k)w, j, k \in \mathbf{Z}\}$ , give rise to what is called a *frame*. Frames are generalizations of bases. In fact a frame is a family  $\{\phi_j(t) | j \in \mathbf{Z}\}$  of functions such that any function can be projection onto it and expanded back again in terms of a dual family  $\{\tilde{\phi}_j\}$  called the dual frame. A frame and a dual frame are similar to a basis and its bi-orthogonal basis. Self-dual frames, that are analogous to orthogonal bases are called tight frames. The essential difference between a frame and a basis is that no function in the latter can be in the closed linear span of its complement. The relationship between frames, bases etc. are shown in Fig. 3.4. The DWT wavelet defined here will give rise to a tight frame.

We begin by defining a DWT wavelet.

**Definition 3.2.3** A function  $w \in L^2(\mathbf{R})$  is called DWT wavelet if for all  $f \in L^2(\mathbf{R})$  it satisfies the equation

$$f(t) = \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f(y) w(2^j y - k) w(2^j t - k) \right] \quad (3.30)$$



**Figure 3.4: Frames and Bases:**  $A$  is the set of all spanning sets,  $B$  is the set of all frames,  $C$  is the set of all bases and  $D$  is the set of all tight frames.  $C \cap D$  is the set of all ON bases.

where  $c_w$  is a constant that depends only on  $w$ .

There is a close relationship between the DWT wavelets and QMFs that will be discussed later.

We now show that the DWT wavelet is characterized by its discrete autocorrelation. Let  $r_w(t)$  denote the discrete autocorrelation of  $w$ . Then by changing the order of summation in Eqn. 3.30 we obtain

$$f(t) = \frac{1}{c_w} \int dy f(y) 2^j r(2^j(t-y))$$

which implies

$$\sum_j 2^j r(2^j t) = c_w \delta(t) \quad (3.31)$$

Taking the Fourier Transform on both sides we have

$$\sum_k \hat{w}(\omega + 2\pi k) \hat{w}(2\pi k) = c_w \quad (3.32)$$

Hence we have the following lemma.

**Lemma 3.2.2** Let  $w \in L^2(\mathbf{R})$ . Let  $r_w(t)$  denote the discrete autocorrelation of  $w$ , i.e

$$r(t) = \sum_k w(t)w(t+k)$$

Then the following conditions are equivalent

1.  $w$  is a DWT wavelet.

2.

$$\sum_j 2^j r(2^j t) = c_w \delta(t)$$

3.

$$\sum_k \hat{w}(\omega + 2\pi k) \hat{w}(2\pi k) = c_w$$

There is a relationship between the sets of CWT and DWT wavelets. To see this consider an arbitrary DWT wavelet  $w$ . First consider the effect of translation on the expansion. Let  $f_x(t) = f(t-x)$ . Then applying Eqn. 3.30 to  $f_x$  we get

$$f_x(t) = \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f_x(y) w(2^j y - k) w(2^j t - k) \right] \quad (3.33)$$

or equivalently,

$$f(t) = \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f(y) w(2^j(y-x) - k) w(2^j(t-x) - k) \right] \quad (3.34)$$

Notice that the left hand side is independent of  $x$ . Therefore by averaging over all time we get

$$f(t) = \lim_{X \rightarrow \infty} \frac{1}{2^X} \int dx \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f(y) w(2^j(y-x) - k) w(2^j(t-x) - k) \right] \quad (3.35)$$

On simplification this equation becomes

$$f(t) = \frac{1}{c_w} \sum_j 2^j R_w(2^j(t-y)) f(y) dy \quad (3.36)$$

where  $R_w$  is the continuous autocorrelation of  $w$ .

Now consider the effect of scaling on the expansion. Define  $f_z(t) = f(2^z t)$ .

$$f(t) = \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f(y) 2^z w(2^{j+z}(t-x) - k) w(2^{j+z}(y-x) - k) \right] \quad (3.37)$$

Notice again that the left hand side is independent of  $z$ . Hence by integrating over  $z$  from 0 to 1, we obtain

$$f(t) = \frac{1}{c_w} \int 2^u r_w(2^u(t-y)) du \quad (3.38)$$

Now combining the effect of scale and time on the expansion we get,

$$f(t) = \frac{1}{c_w} \sum_{j,k} 2^j \left[ \int dy f(y) 2^z w(2^z 2^j(t-x) - k) w(2^z 2^j(y-x) - k) \right] \quad (3.39)$$

Again the LHS is independent of  $x$  and  $z$ . Now, averaging over time and then over scale as before, it can be shown that

$$f(t) = \frac{1}{c_w} \int f(y) 2^u R_w(2^u(t-y)) dy \quad (3.40)$$

Thus  $w$  is a CWT wavelet too!!

We note these results as a lemma.

**Lemma 3.2.3** Let  $w$  be a DWT wavelet. Let  $R_w(t)$  and  $r_w(t)$  be its continuous and discrete autocorrelations respectively. Then,

1.

$$\int R_w(2^u t) 2^u du = c_w \delta(t) \quad (3.41)$$

2.

$$\int du 2^u r_w(2^u t) dt = c_w \delta(t) \quad (3.42)$$

3.

$$\sum_j 2^j R_w(2^j t) = c_w \delta(t) \quad (3.43)$$

Now the DWT is defined as follows

**Definition 3.2.4** The  $j, k$  transform (or the Discrete Wavelet Transform) is given by the equation

$$W(j, k, f, w) = \int f(t) 2^{j/2} w(2^j t - k) dt = \langle f, L(j, k)w \rangle \quad (3.44)$$

This transform is analogous to the Discrete Short Time Fourier Transform. The one big difference is that the DWT wavelet gives rise to a tight frame, while the DSTFT gives rise to frame (not necessarily tight), and hence the inverse transform has to be taken with respect to the dual frame in the case of the latter.

### 3.2.3 Properties of the DWT

**Inverse Transform:** The inverse DWT is given by the equation,

$$f(t) = \frac{1}{c_w} \sum_{j,k} W(j, k, f, w) 2^{j/2} w(2^j t - k) \quad (3.45)$$

Direct computation of the forward and inverse transforms is not efficient. Practically always, associated with the DWT wavelet, is a unique function called the *scaling function*, the scaling relationship of which will be used to compute the transform and the inverse efficiently.

**Energy Preserving Property:** For all  $f \in L^2(\mathbf{R})$ ,  $W(j, k, f, w)$  satisfies the equation,

$$\sum_{j,k} |W(j, k, f, w)|^2 = c_w \int_{\mathbf{R}} |f(t)|^2 dt \quad (3.46)$$

This is straightforward to verify.

**Scaling of  $f$ :** As can be expected, unlike the CWT, for arbitrary scaling of  $f$ , the DWT is not related in a simple fashion to the DWT of  $f$ , even though both contain all the information about the waveform  $f$ . But for scaling integral powers of 2, we have the following. Let  $f_l = 2^{l/2} f(2^l t)$ , where  $l$  is an integer. Then,

$$W(j, k, f_l, w) = W(j - l, k, f, w) \quad (3.47)$$

The result follows directly from the fact that the DWT wavelet is in particular a CWT wavelet, and has to satisfy the corresponding scaling relationship.

**Translation of  $f$ :** The DWT lacks translation invariance. The DWT of  $f$  and the DWT of  $f(t - T)$  are quite different. Moreover, the energy of the DWT of  $f(t - T)$  at a certain scale  $j$  may be quite different from the corresponding energy of  $f$ . This is disappointing since without explicitly specifying the time origin one cannot speak of a signal as *residing* in a particular scale or set of scales. However, for each scale the DWT is periodically time varying, and hence, could be used in the study of many statistical signals that are inherently periodically time varying.

### 3.2.4 The Discrete-Time Wavelet Transform

**Definition 3.2.5** A function  $w \in L^2(\mathbf{R})$  is called a DTWT wavelet if for all  $f \in L^2(\mathbf{R})$  we have

$$f(t) = \sum_k \frac{1}{c_w} \int du \left[ \int dy 2^u w(2^u y - k) f(y) \right] w(2^u t - k) \quad (3.48)$$

The DTWT is characterized by the discrete autocorrelation.

**Lemma 3.2.4** The following statements are equivalent

1.  $w$  is a DTWT wavelet
2. The discrete autocorrelation  $r_w$  satisfies

$$\int du 2^u r_w(2^u(t)) = c_w \delta(t) \quad (3.49)$$

3.

$$\sum_k \frac{\hat{w}(2\pi k) \hat{w}(\omega + 2\pi k)}{|\omega|} = c_w \quad (3.50)$$

From Lemma. 3.2.3 and Lemma. 3.2.4 it is clear that every DWT wavelet is a DTWT wavelet. It can also be shown that every DTWT wavelet is a CWT wavelet.

Consider an arbitrary DTWT wavelet  $w$ , with corresponding constant  $c_w$ . Then for the function  $f_x(t) = f(t - x)$ , Eqn. 3.48 implies

$$f_x(t) = \frac{1}{c_w} \sum_k \int du \left[ \int dy 2^u w(2^u y - k) f_x(y) \right] w(2^u t - k) \quad (3.51)$$

which on a change of variables becomes

$$f(t) = \frac{1}{c_w} \sum_k \int du \left[ \int dy 2^u w(2^u y - 2^u x - k) f(y) \right] w(2^u t - 2^u x - k) \quad (3.52)$$

The left hand side of the above equation is independent of  $x$ . Thus by averaging over  $x$  on both sides,

$$f(t) = \frac{1}{c_w} \lim_{X \rightarrow \infty} \frac{1}{2^X} \int dx \sum_k \int du \left[ \int dy 2^u w(2^u y - 2^u x - k) f(y) \right] w(2^u t - 2^u x - k) \quad (3.53)$$

The above equation after a few simplifications can be shown to reduce to

$$f(t) = \frac{1}{c_w} \int R_w(t - y) f(y) dy \quad (3.54)$$

Thus a DTWT wavelet is, in particular a CWT wavelet.

We have proved the result,

**Lemma 3.2.5** Let  $w(t)$  be a DTWT wavelet. Let  $R_w(t)$  denote its continuous autocorrelation. Then,

$$\int 2^u R(2^u t) du = c_w \delta(t) \quad (3.55)$$

That is every DTWT wavelet is a CWT wavelet.

In terms of the DTWT wavelet we define the DTWT as follows

**Definition 3.2.6** The  $u, k$  (or the Discrete-Time Wavelet Transform) is given by the equation

$$W(u, k, f, w) = \int f(t) 2^{u/2} w(2^u t - k) dt = \langle f, L(u, k)w \rangle \quad (3.56)$$

### 3.2.5 Properties of the DTWT

**Inverse Property:** The inverse DTWT transform is given by the equation

$$f(t) = \frac{1}{c_w} \int_{\mathbf{R}} du \sum_k W(u, k, f, w) 2^{u/2} w(2^u t - k) \quad (3.57)$$

**Energy Preserving Property:** The DTWT is an isometry upto a constant,

**Theorem 3.2.2** For all  $f \in L^2(\mathbf{R})$ ,  $W(u, k, f, w)$  satisfies

$$\int du \sum_k |W(u, k, f, w)|^2 = c_w \int |f(t)|^2 dt \quad (3.58)$$

**Scaling of  $f$ :** Since the scale parameter is continuous and since the DTWT wavelet is also a CWT wavelet we have,

$$W(u, k, f_S, w) = W(u - S, k, f, w) \quad (3.59)$$

where  $f_S$  is  $2^{S/2} f(2^S t)$ . Thus scaling of a function leads to a mere shift in the transform in the scale dimension.

**Translation of  $f$ :** Since the time parameter is discrete, as in the case of the DWT the transform is not translation invariant. This transform would be applicable when the signal origin is known precisely.

### 3.2.6 The Discrete-Scale Wavelet Transform

**Definition 3.2.7** A DSWT wavelet is defined to be any function  $w \in L^2(\mathbf{R})$ , such that,  $\forall f \in L^2(\mathbf{R})$  we have,

$$f(t) = \frac{1}{c_w} \sum_j 2^j \int d\alpha \left[ \int dy f(y) w(2^j y - \tau) \right] w(2^j t - \tau) \quad (3.60)$$

for some constant  $c_w$  that depends only on  $w$ .

The following lemma gives equivalent characterizations of a DSWT wavelet.

**Lemma 3.2.6** The following conditions are equivalent



1.  $w$  is a DSWT wavelet
2. The autocorrelation  $R_w$ , of  $w$ , satisfies the equation

$$\sum_j 2^j R_w(2^j(t)) = c_w \delta(t) \quad (3.61)$$

- 3.

$$\sum_j |\hat{w}(2^j \omega)|^2 = c_w \quad (3.62)$$

From Lemma. 3.2.3 and the lemma above it is clear that every DWT wavelet is a DSWT wavelet. Now we proceed to show that every DSWT wavelet is a CWT wavelet.

Consider an arbitrary DSWT wavelet  $w(t)$  with corresponding constant  $c_w$ . Consider the effect of scaling on the DTWT expansion of  $f_z(t) = f(2^z t)$ . Indeed we have,

$$f_z(t) = \frac{1}{c_w} \sum_j 2^j \int d\alpha \left[ \int dy f_z(y) w(2^j y - \tau) \right] w(2^j t - \tau) \quad (3.63)$$

which becomes,

$$f(t) = \frac{1}{c_w} \sum_j 2^{j+z} \int d\alpha \left[ \int dy f(y) w(2^{j+z} y - \tau) \right] w(2^{j+z} t - \tau) \quad (3.64)$$

The left hand side is independent of  $z$ . By averaging over  $z$  from 0 to 1, we obtain,

$$f(t) = \int_0^1 dz \frac{1}{c_w} \sum_j 2^{j+z} \int d\alpha \left[ \int dy f(y) w(2^{j+z} y - \tau) \right] w(2^{j+z} t - \tau) \quad (3.65)$$

which on simplification becomes

$$f(t) = \frac{1}{c_w} \int 2^u R_w(2^u(t - y)) f(y) dy \quad (3.66)$$

Thus every DSWT wavelet is in particular a CWT wavelet.

Now we define the DSWT as follows

**Definition 3.2.8** The  $j, \tau$  (or the Discrete-Scale Wavelet Transform) is given by the equation

$$W(j, \tau, f, w) = \int f(t) 2^{j/2} w(2^j t - \tau) dt = \langle L(j, \tau) w, f \rangle \quad (3.67)$$

### 3.2.7 Properties of the DSWT

**Inverse Property:** From the defining equation we have the inverse transform given by

$$f(t) = \frac{1}{c_w} \int d\tau \sum_j W(j, \tau, f, w) 2^{j/2} w(2^j t - \tau) \quad (3.68)$$

**Energy Preserving Property:**

$$\int d\tau \sum_j |W(j, \tau, f, w)|^2 = \int dt |f(t)|^2 \quad (3.69)$$

**Translation of  $f$ :** Since the DSWT wavelet is in particular a CWT wavelet, we have

$$W(j, \tau, f_T, w) = W(j, \tau - 2^j T, f, w) \quad (3.70)$$

**Scaling of  $f$ :** Clearly scaling in octaves only gives a simple relationship between the corresponding DSWTs. We have

$$W(j, \tau, f_l, w) = W(j - l, \tau, f, w) \quad (3.71)$$

We summarize the relationships between the four different wavelets introduced in the form of a theorem

**Theorem 3.2.3** Let  $D_0, D_1, D_2$  and  $D_3$  be the subsets of  $L^2(\mathbf{R})$  that are the set of all CWT, DTWT, DSWT and DWT wavelets respectively. Then we have the following proper inclusion relationships between the transforms.

$$D_1 \subset D_0 \quad (3.72)$$

$$D_2 \subset D_0 \quad (3.73)$$

$$D_3 \subset D_1 \cap D_2 \quad (3.74)$$

## Chapter 4

### Orthonormal Wavelet Bases

The previous chapter defined the CWT, DWT, DTWT and DSWT wavelets and the corresponding transforms. The CWT wavelet has to satisfy an *admissibility condition* that makes it an isometry (upto a constant) of  $L^2(\mathbf{R})$  onto  $L^2(\mathbf{R} \times \mathbf{R})$ . The orbit of the DWT wavelet under the action of the discrete subset  $G_1$  of the affine group must form a tight frame. Tight frames differ from orthonormal bases in that they are overcomplete sets. Chapter 2 introduced the *sinc* wavelet, which is a DWT wavelet which gave rise to an ON wavelet basis that had infinite support. This chapter discusses the recently discovered ON wavelets that have compact support. Computations involving compactly supported ON wavelets can be efficiently done using Conjugate Quadrature Filters. Recently it has been proved that every FIR CQF that satisfies a certain DC summation condition, is associated with a DWT wavelet and hence gives rise to a tight frame [24].

#### 4.1 Orthonormal Wavelet Bases

The multiscale analysis framework of Mallat and Meyer [26] gives the best framework for understanding ON wavelets.

##### 4.1.1 Multiscale Analysis

Multiscale analysis is a framework for approximating a signal by a sequence of smoothed versions of the signal. The smoothing operation is accomplished by convolving with a function called the **scaling function**. More precisely,

**Definition 4.1.1** A multiscale analysis consists of a sequence  $\{V_j \mid j \in \mathbb{Z}\}$  of closed subspaces of  $L^2(\mathbb{R})$  that satisfy the following conditions

**1. Containment Property**

$$\dots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \dots \quad (4.1)$$

**2. Completeness Property**

$$\bigcap_{j \in \mathbb{Z}} V_j = \{0\}; \quad \overline{\bigcup_{j \in \mathbb{Z}} V_j} = L^2(\mathbb{R}) \quad (4.2)$$

**3. Scaling Property**

$$f(\cdot) \in V_j \Leftrightarrow f(2\cdot) \in V_{j+1} \quad (4.3)$$

for any function  $f \in L^2(\mathbb{R})$ .

**4. The Basis/Frame Property** There exists  $\phi \in V_0$  such that  $\forall j \in \mathbb{Z}$ , the set  $\{\phi_{j,k} = 2^{-j/2} \phi(2^j t - k) \mid \forall k \in \mathbb{Z}\}$  spans  $V_j$ .

Let  $W_j$  denote the orthogonal complement of  $V_j$  in  $V_{j+1}$ . Let us denote by  $P_j$  and  $Q_j$ , the projection operators from  $L^2(\mathbb{R})$  onto  $V_j$  and  $W_j$  respectively. Clearly the containment and completeness properties imply the existence of  $W_j$  and hence the projection operators are well defined. The completeness property also ensures that  $\lim_{j \rightarrow \infty} P_j f = f$  for any signal  $f \in L^2(\mathbb{R})$ , where the limit is taken in the  $L^2$  norm. The containment property implies that  $P_j f$ , for successively increasing  $j$ , leads to successively better approximations of  $f$ . That is, given  $P_j f$ ,  $P_{j-1} f$  is completely determined. The scaling property ensures that the approximations reside on different scales. The scaling property is the crucial property that leads to the ON wavelet basis.

If the  $\phi_{j,k}$ , for fixed  $k$ , form a basis for  $V_j$ , then by a Gram orthonormalization process in the Hilbert space  $L^2(\mathbb{R})$ , a function  $\tilde{\phi}$  is constructed from  $\phi$ , as in [25], so that the  $\tilde{\phi}_{j,k}$  forms an orthonormal basis for  $L^2$ . Therefore, henceforth  $\{\phi_{j,k}\}$  is assumed to be an orthonormal system.

### Properties of $\phi$

Now  $\phi(t) \in V_0 \subset V_1$ , and  $V_1 = \text{Span}\{\phi(2t - k)\}$ . Therefore we have the fundamental scaling equation

$$\phi(t) = 2 \sum_k h_0(k) \phi(2t - k) \quad (4.4)$$

for some sequence  $h_0$ .

This functional relationship (called a two-scale difference equation) for general functions  $\phi$  that do not necessarily give rise to a multiscale analysis has been studied [9, 18]. Conditions for the existence and uniqueness of  $L^1(\mathbf{R})$  solutions of this functional equation are well known. By assumption the  $\phi$  in a multiscale analysis is in  $L^2(\mathbf{R})$ . Furthermore, if  $\phi$  is compactly supported, then by considering the restriction of  $L^2(\mathbf{R})$  to functions that are square integrable in the support of  $\phi$ , which is of finite measure,  $\phi$  is in  $L^1(\mathbf{R})$  too.

The fundamental scaling equation immediately implies

$$\phi_{j,k} = \sqrt{2} \sum_l h_0(l) \phi_{j+1,2k+l} \quad (4.5)$$

thus relating scaling functions that span  $V_j$  and  $V_{j+1}$ .

Taking the Fourier Transform on both sides of Eqn. 4.4,

$$\hat{\phi}(\omega) = \sum_k \hat{\phi}\left(\frac{\omega}{2}\right) e^{-i\omega k/2} h_0(k)$$

Let  $H_0(e^{i\omega})$  denote the Fourier Transform of the sequence  $\{h_0(k)\}$ , i.e.,

$$H_0(e^{i\omega}) = \sum_k e^{-i\omega k} h_0(k)$$

Then

$$\hat{\phi}(\omega) = \hat{\phi}(\omega/2) H_0(e^{i\omega/2}) \quad (4.6)$$

which becomes the following infinite product equation for  $\hat{\phi}$

$$\hat{\phi}(\omega) = \hat{\phi}(0) \prod_{j=1}^{j=\infty} H_0(e^{i\omega/2^j}) \quad (4.7)$$

The sequence  $h_0$  thus uniquely determines the multiscale analysis framework. If  $h_0 \in l^2(\mathbf{Z})$ ,  $\phi(t)$  is a unique  $L^2(\mathbf{R})$  distribution. The sequence  $h_0$  is characterized by the CQF property. To see this the following lemmas are necessary.

**Lemma 4.1.1** A set of functions  $\phi(\cdot - k)$  form an orthonormal family iff their Fourier transforms satisfy

$$\sum_k |\hat{\phi}(\omega + 2\pi k)|^2 = 1 \quad (4.8)$$

**Lemma 4.1.2** A function  $\phi(\cdot)$  is orthogonal to a set of functions  $\{\psi(\cdot - k)\}$  iff

$$\sum_k \hat{\phi}(\omega) \hat{\psi}^*(\omega - 2\pi k) = 0 \quad (4.9)$$

The above equations are merely the Fourier Transforms of the defining conditions of orthonormality.

Since  $\{\phi(t - k)\}$  forms an orthonormal system, Lemma. 4.1.1 implies

Eqn. 4.7 can now be invoked and with some simplification becomes

$$|H_0(e^{i\omega})|^2 + |H_0(e^{i(\omega+\pi)})|^2 = 1 \quad (4.10)$$

This is the equation characterizing the low pass filter in two channel maximally decimated perfect reconstruction CQF [31]. Thus a multiscale analysis gives rise to a CQF. Moreover the sum of the coefficients  $h_0$  is 1. The converse is also true. Every CQF, that satisfies the summation condition gives rise to a multiscale analysis (where the basis property is replaced by the frame property). Thus the sequence and hence the multiscale analysis is completely characterized by the CQF.

#### Lowpass property of $\phi$

Integrating both sides of the fundamental scaling equation, we get the following conditions on  $h_0$ ,

$$\sum_k h_0(k) = 1 \quad (4.11)$$

or what amounts to the same

$$H_0(1) = 1 \quad (4.12)$$

which implies on substitution into Eqn. 4.7

$$\hat{\phi}(0) = 1 \quad (4.13)$$

Thus the scaling function is a low pass filter. Moreover the sequence is also a lowpass filter in the sense of non-zero DC.

### Compactness Property

The sequence  $h_0$  can be of finite or infinite length. If it is of finite length  $N$ , and Eqn. 4.4 is of the form

$$\phi(t) = \sum_0^{N-1} 2h_0(k)\phi(2t - k) \quad (4.14)$$

it is seen that  $\phi$  is of compact support, with support in  $(0, N - 1)$ . This has only got to do with the nature of the two-scale difference equation and has nothing to do with the fact that  $\phi$  gives rise to a multiscale analysis framework. Conversely, if  $\phi$  is compactly supported, by taking the inner products with  $\phi(2t - k)$ , it is clear that  $h_0$  is of compact support.

**Lemma 4.1.3** The scaling function  $\phi$  is compactly supported if and only if the sequence  $h_0$  is of finite length. Moreover if the sequence is supported in  $(0, N - 1)$ , then  $\phi$  is supported in  $(0, N - 1)$  and vice-versa.

**Smoothness Property** For typical engineering applications, the function  $\phi$  must usually be reasonably smooth. This restriction should be manifested in the decay of the Fourier Transform  $\hat{\phi}$ . Now this in turn must manifest itself in some properties on the sequence  $h_0$ . Decay in the Fourier Transform can be achieved if the Fourier Transform of the sequence has a factor of the form  $(1 + e^{-j\omega})^N$  for some integer  $N$ . This introduces a  $\omega^{-N}$  factor in the Fourier Transform of  $\phi$ . This idea was used by I.Daubechies to prove the following sufficiency condition on the  $h_0$  for the smoothness of  $\phi$ .

**Theorem 4.1.1** Let the Fourier Transform  $H_0(e^{i\omega})$  be factorizable into the form

$$H_0(e^{i\omega}) = P(e^{i\omega})(1 + e^{i\omega})^D \quad (4.15)$$

for some trigonometric polynomial  $P(e^{i\omega})$ . Furthermore, let  $P(e^{i\omega})$  satisfy the following equation

$$\sup_{\omega \in \mathbf{R}} \left| \prod_{k=0}^{l-1} P(e^{i\omega/2^k}) \right| < 2^{l(D-m-1)} \quad (4.16)$$

for some  $l > 1$ , then  $h_0$  gives rise to an  $L^1$  scaling function that is  $m$  times continuously differentiable.

Imposing these restrictions on the  $h_0$  that characterizes a multiscale analysis I. Daubechies constructed a family of scaling functions that are compactly supported and have degree of regularity that increases approximately linearly with the support of  $\phi$ .

#### 4.1.2 Construction of the Wavelet Basis

The ON wavelet in Chapter 2 was obtained by taking the difference of the lowpass scaling function (the sinc function) at two different scales. A somewhat similar procedure can be applied in the case of a multiscale analysis to obtain an ON wavelet.

The space  $W_0$  is given by  $V_1 \ominus V_0$ . Now  $\phi(t-k) \in V_0$  and  $\phi(2t-k) \in V_1$ . Since  $V_0 = \text{Span}\{\phi(t-k)\}$  and  $V_1 = \text{Span}\{\phi(2t-k)\}$ , it is reasonable to expect the existence of a function  $\psi$ , such that  $W_0 = \text{Span}\{\psi(t-k)\}$ . This is indeed true and can be proved by group representation arguments. This function  $\psi$  is the wavelet associated with the multiscale analysis. Clearly, by the scaling property,  $\text{Span}\{\psi(2^j t - k)\} = W_j$ . Now  $L^2(\mathbf{R})$  can be decomposed as

$$L^2(\mathbf{R}) = \cdots \oplus W_{-j} \oplus W_{-j+1} \oplus \cdots \oplus W_0 \cdots \oplus W_{j-1} \oplus W_j \oplus \cdots \quad (4.17)$$

which implies that  $L^2(\mathbf{R}) = \text{span}\{2^{j/2}\psi(2^j t - k)\}$ . The set  $\{\psi_{j,k} = 2^{j/2}\psi(2^j t - k)\}$  is the wavelet basis associated with the multiscale analysis.



Since  $W_0 \subset V_1$ , there exists a sequence  $h_1$  such that

$$\psi(t) = \sum_k h_1(k) \phi(2t - k) \quad (4.18)$$

This is the fundamental wavelet equation, the dual of the fundamental scaling equation. The sequence  $h_1$  is uniquely determined by the sequence  $h_0$ .

Taking the Fourier Transform on both sides of Eqn. 4.18

$$\hat{\psi}(\omega) = \sum_k \hat{\psi}\left(\frac{\omega}{2}\right) e^{-i\omega k/2} h_1(k)$$

Let  $H_1(e^{i\omega})$  be the Fourier Transform of  $h_1$ , i.e

$$\bar{H}_1(e^{i\omega}) = \sum_k e^{-i\omega k} h_1(k)$$

Then

$$\hat{\phi}(\omega) = \hat{\phi}(\omega/2) H_1(e^{i\omega/2}) \quad (4.19)$$

which implies

$$\hat{\psi}(\omega) = \hat{\phi}(0) H_1(i\omega/2) \prod_{j=2}^{\infty} H_0(e^{i\omega/2^j}) \quad (4.20)$$

This is the infinite product equation for  $\hat{\psi}$ .

Since  $\{\psi(t - k)\}$  forms an ON basis for  $W_0$ , using Lemma 4.1.1,

$$|H_1(e^{i\omega})|^2 + |H_1(e^{i(\omega+\pi)})|^2 = 1 \quad (4.21)$$

Thus  $h_1$  is also a CQF. In fact it is the dual of  $h_0$  in the two channel CQF bank. Indeed, Lemma 4.1.2 implies

$$H_0(e^{i\omega}) H_1^*(e^{i\omega}) + H_0(e^{i(\omega+\pi)}) H_1^*(e^{i(\omega+\pi)}) = 0 \quad (4.22)$$

Equations 4.10, 4.21, and 4.22 are precisely the conditions to be satisfied by the two filters in a two channel maximally decimated power-complementary Conjugate Quadrature Filter Bank. In this case  $h_0$  uniquely specifies  $h_1$  and vice versa. Borrowing from the CQF literature  $h_1$  is given by

$$h_1(n) = (-1)^n h_0(-n + 1) \quad (4.23)$$

Let  $Wf(j, k)$  and  $Sf(j, k)$  denote the inner products of  $f$  with  $\psi_{j,k}$  and  $\phi_{j,k}$  respectively.  $Wf(j, k)$  and  $Sf(j, k)$  are called the wavelet and scaling coefficients of  $f$  respectively. Then,

$$P_j f = \sum_k Sf(j, k) \phi_{j,k}$$

and

$$Q_j f = \sum_k Wf(j, k) \psi_{j,k}$$

where  $Q_j$  denotes the projection operator onto  $W_j$ .

Since any  $\phi$  satisfies Eqn. 4.4 for some  $h_0$ , and since  $h_0$  uniquely determines  $\phi$ , a probable approach to generate multiscale analysis is to start with a suitable  $h_0$  and try to construct a multiscale analysis. As mentioned earlier, the problem of existence and uniqueness of  $L^1$  solutions of two-scale difference equations has been studied [9, 18]. It turns out that if  $h_0$  satisfies the DC summation condition,

$$\sum h_0(k) = 1 \tag{4.24}$$

then there can be at most one function  $\phi$  that satisfies the scaling property.

Since orthonormal multiscale analyses result in  $h_0$ 's that are CQF's a pertinent question is whether all CQF's give rise to an orthonormal wavelet basis. The answer is in the negative [7] Despite this, all CQF's do give rise to a tight wavelet frames [24]. Furthermore, if some regularity conditions are imposed on  $h_0$ , then it does indeed give rise to an ON wavelet basis. This is precisely how compactly supported wavelet bases were constructed.

### Properties of the ON wavelet

The properties of the ON wavelets are direct consequences of the corresponding properties of the scaling function and the CQF condition relating them.

#### Bandpass Property

Integrating both sides of the wavelet equation and remembering that  $\phi(\hat{0}) = 1$  we get

the following conditions on  $h_1$ ,

$$\sum_k h_1(k) = 0 \quad (4.25)$$

or what amounts to the same

$$H_1(1) = 0 \quad (4.26)$$

which implies on substitution into Eqn. 4.7

$$\hat{\psi}(0) = 0 \quad (4.27)$$

Thus the wavelet is a bandpass filter.

### Compactness Property

The sequence  $h_1$  is infinite if the sequence  $h_0$  is infinite. Assuming that  $h_0$  is compactly supported in  $[0, N - 1]$  it follows that

$$\psi(t) = \sum_{-N+1}^1 2h_1(k)\phi(2t - k) \quad (4.28)$$

**Lemma 4.1.4** The wavelet is compactly supported iff the scaling function is compactly supported. Moreover, if the scaling function is supported in  $[0, N - 1]$ , the wavelet  $\psi$  is supported in  $[1 - N/2, N/2]$

**Smoothness Property** It is clear that if the scaling function is  $m$  times differentiable then the wavelet is also  $m$  times differentiable.

### 4.1.3 The Haar Example

We already saw an orthonormal wavelet basis in Chapter. 2. Here we give the classical Haar basis in order to illustrate multiscale analysis.

Let  $V_j$  be the space of piecewise constant functions defined by

$$V_j = \{f \in L^2(\mathbf{R}) : f \text{ is constant on } [2^{-j}k, 2^{-j}(k+1)), \forall k \in \mathbf{Z}\} \quad (4.29)$$

Then the sequence of spaces  $V_j$  satisfies all the properties of a multiscale analysis. The *containment property* is obvious since functions that are constants on intervals of the type  $[2^{-j}k, 2^{-j}(k+1)]$  are clearly constant on intervals of the type

$[2^{-(j+1)}k, 2^{-(j+1)}(k+1)]$ . The *completeness property* and the *scaling property* are also fairly obvious. As for the *frame property* the function  $\phi$  given by

$$\phi(t) = \begin{cases} 1 & 0 \leq t < 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.30)$$

and its integral translates span  $V_0$ . Hence  $\{\phi_{j,k}\}$  defined in the usual way forms a frame. In fact they form an orthonormal basis for  $L^2(\mathbb{R})$ . The orthonormality is obvious because for fixed  $j$ ,  $\phi_{j,k_1}$  and  $\phi_{j,k_2}$  overlap if and only if  $k_1 = k_2$ .

Now for an arbitrary function in  $L^2$ ,  $P_j f$ , the projection of  $f$  onto  $V_j$ , is simply the step approximation of the function corresponding to steps of width  $2^{-j}$ . Clearly with decreasing step-size, i.e. with increasing  $j$ , the approximation gets better and better as is seen in Fig. 4.3.

In the case of the Haar system the unique filter corresponding to the multiscale analysis is given by

$$h_0(k) = \begin{cases} 1/2 & \text{if } k = 0, 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.31)$$

From Eqn. 4.23 the corresponding  $h_1$ , the filter corresponding to the wavelet is obtained as

$$h_1(k) = \begin{cases} 1/2 & \text{if } k = 0 \\ -1/2 & \text{if } k = 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.32)$$

The corresponding wavelet is given by

$$\psi(t) = \begin{cases} 1 & 0 \leq t < 1/2 \\ -1 & 1/2 \leq t < 1 \\ 0 & \text{otherwise} \end{cases} \quad (4.33)$$

#### 4.1.4 Daubechies Wavelets

Compactly supported wavelets of arbitrary degree of smoothness were constructed by Daubechies by constructing the corresponding CQF using Theorem. 4.1.1 quite

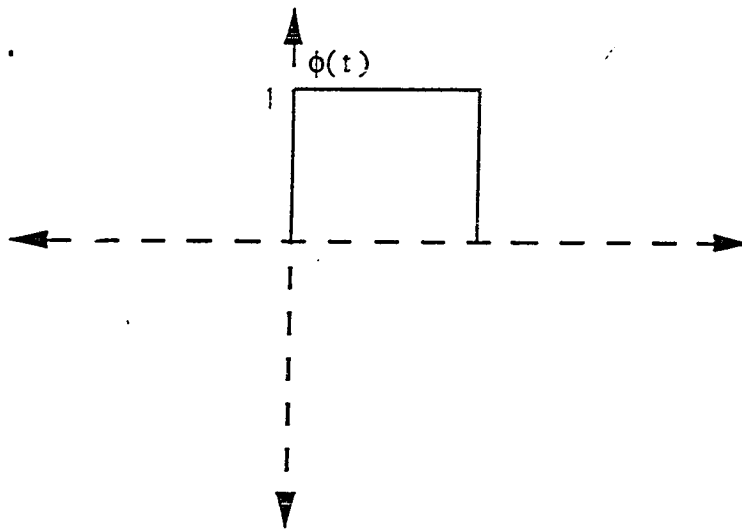


Figure 4.1: The Haar Scaling Function

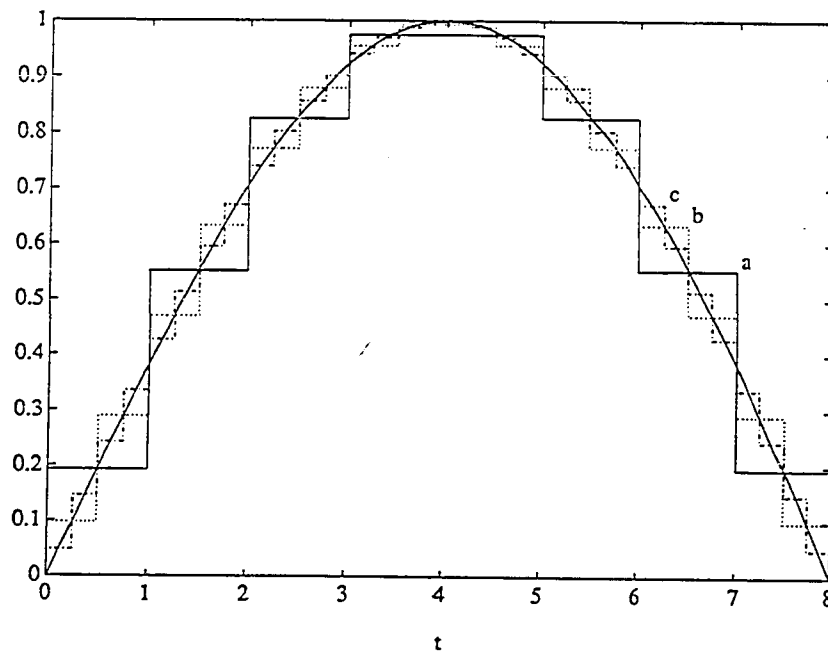


Figure 4.2: Approximation using Haar basis

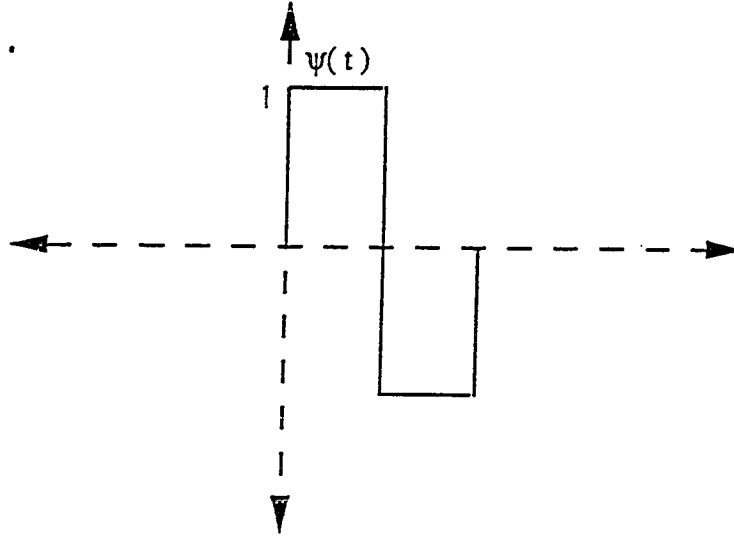


Figure 4.3: The Haar Wavelet

complicated and the interested reader can read the paper. The important result is the following theorem.

**Theorem 4.1.2** Any CQF  $h_0$  with the DC summation condition is necessarily of the form

$$H_0(\omega) = P(e^{i\omega})[(1 + e^{-j\omega})/2]^{N/2}$$

for some integer  $N \geq 2$ , where the polynomial  $P$  satisfies the following equation

$$|P(e^{i\omega})|^2 = \left[ \sum_{k=0}^{N/2-1} \binom{N/2-1+k}{k} \sin^{2k}\left(\frac{\omega}{2}\right) + \sin^N\left(\frac{\omega}{2}\right) R\left(\frac{\cos(\omega)}{2}\right) \right] \quad (4.34)$$

where  $R$  is a polynomial such that

$$R(x) = -R(1-x) \quad (4.35)$$

Furthermore,  $h_0$  gives rise to a multiscale analysis associated with a tight frame (i.e a DWT wavelet). Moreover, if  $R$  satisfies a certain growth conditions that  $h_0$  leads to a multiscale analysis associated with an ON wavelet basis. The support width of both the  $\phi$  and  $\psi$  associated with the CQF is  $N - 1$  if  $R = 0$ . For fixed  $N$ , exactly  $N/2 - 1$  of the moments of the associated  $\psi$  vanish.

This includes all the CQF's discussed in the engineering literature where  $K$  is typically 1 and  $R$  is arbitrary. Thus the resulting  $\phi$  may be a distribution. Specifically by letting  $R = 0$  we get a whole class of CQF's and corresponding wavelet bases. We refer to them as the Daubechies wavelet bases and the corresponding wavelets the Daubechies wavelets.

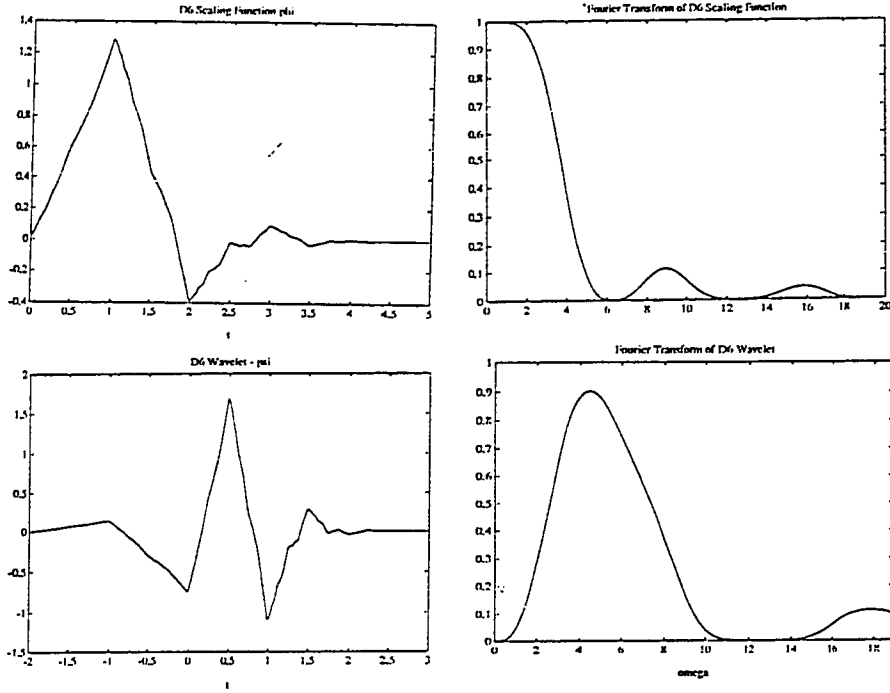
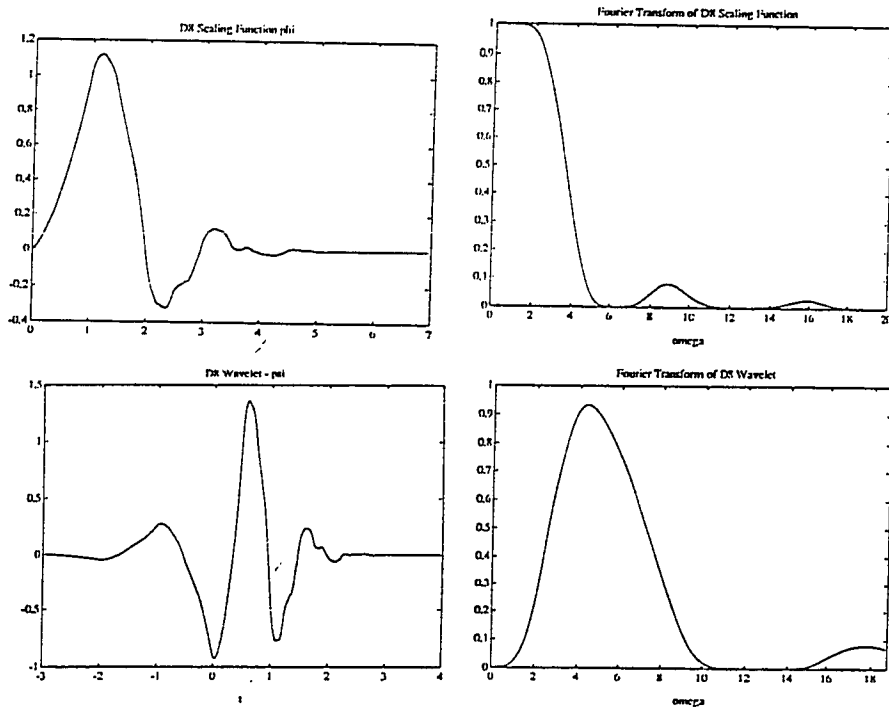
Figs. 4.4 and 4.5 show the Daubechies scaling functions, the Daubechies wavelets and their Fourier Transforms for  $N = 6$ , and  $N = 8$ . Notice the increase in smoothness of both  $\phi$  and  $\psi$  with increasing  $N$ . Also note that the support of these functions increases with  $N$ , as does the Fourier Transform decay more rapidly for increasing  $N$ .

#### 4.1.5 The Moments of $\phi$ and $\psi$

The moments of the Daubechies scaling function and wavelet have a number of interesting properties. The first observation is that the moments of  $\phi$  are completely characterized by the discrete moments of the sequence  $h_0$ . This means that the moments of  $\phi$  can be calculated exactly and efficiently without taking recourse to any numerical integration. This property depends only on the fact that the scaling function satisfies a two-scale difference equation. To see this, consider an arbitrary function  $f(t)$  that satisfies the following two-scale difference equation

$$f(t) = 2 \sum_k h(k)g(2t - k) \quad (4.36)$$

for some function  $g$  and some sequence  $h$ .

Figure 4.4:  $\phi$ ,  $\psi$ ,  $\hat{\phi}$  and  $\hat{\psi}$  for  $N=6$ Figure 4.5:  $\phi$ ,  $\psi$ ,  $\hat{\phi}$  and  $\hat{\psi}$  for  $N=8$



**Table 4.1: The filters associated with Daubechies wavelets**

$N$	$n$	$h_0^N(n)$	$k_n^N$	$\beta$
4	0	3.4150635e-01	1.7320508e+00	1.2940952e-01
	1	5.9150635e-01	3.7320508e+00	
	2	1.5849365e-01		
	3	-9.1506351e-02		
6	0	2.3523360e-01	2.4254972e+00	3.5226292e-02
	1	5.7055846e-01	-5.4609640e-01	
	2	3.2518250e-01	-9.4438141e+00	
	3	-9.5467208e-02		
	4	-6.0416104e-02		
	5	2.4908750e-02		
8	0	1.6290171e-01	3.1029315e+00	-1.0597402e-02
	1	5.0547286e-01	-8.1093134e-01	
	2	4.4610007e-01	2.5929388e-01	
	3	-1.9787513e-02	2.1739085e+01	
	4	-1.3225358e-01		
	5	2.1808150e-02		
	6	2.3251801e-02		
	7	-7.4934947e-03		
10	0	1.1320949e-01	3.7715192e+00	-3.3357253e-03
	1	4.2697177e-01	-1.0639434e+00	
	2	5.1216347e-01	4.2482928e-01	
	3	9.7883481e-02	-1.3318455e-01	
	4	-1.7132836e-01	-4.7996278e+01	
	5	-2.2800566e-02		
	6	5.4851329e-02		
	7	-4.4134001e-03		
	8	-8.8959351e-03		
	9	2.3587140e-03		

Let  $m_k$ ,  $\mu_k$  and  $d_k$  denote the  $k$ th moments of  $f$ ,  $g$  and  $h$  respectively, where the  $d_k$  is the discrete moment of the sequence  $h$ . We have

$$m_k = \int t^k f(t) dt \quad (4.37)$$

and

$$d_k = \sum_n n^k h(n) \quad (4.38)$$

Now substituting for  $f(t)$  from Eqn. 4.36 in term of  $g(2t - k)$  and rearranging, the  $k$ th moment of  $f$  can be expressed in terms of the discrete moments of  $h$  and the moments of  $g$  up to  $k$  as follows,

$$m_k = \frac{1}{2^k} \sum_{i=0}^{i=k} \binom{k}{i} d_i \mu_{k-i} \quad (4.39)$$

In particular when  $f = g$ ,  $\mu_k = m_k$  and hence the continuous moments of  $f$  are completely characterized by the discrete moments of  $h$ . From now on we shall assume that  $f = g$ .

The above equations hold, even if  $f$  is shifted by a certain amount. Using this fact, if  $f$  is shifted to its center of mass, then its first moment is zero. Now if the moments of  $f$  are such that  $m_i = m_1^i$ ,  $1 \leq i \leq k$ , then by shifting to the center of mass all moments upto order  $k$  vanish. This property will prove useful in the approximation of functions in wavelet bases later on. This property can be characterized in terms of the discrete moments which are more amenable to manipulation.

**Theorem 4.1.3** For a given  $f$  as above, we have  $m_i = m_1^i$ ,  $1 \leq i \leq k$ , if and only if for all  $i$ ,  $1 \leq i \leq k$ ,  $d_k = d_1^k$ . Moreover  $m_i = d_i$ ,  $1 \leq i \leq k$ .

This theorem can be proved by induction using Eqn. 4.39 To prove some property of the moments of  $f$ , it can be first converted into into a corresponding property for the discrete moments of  $h$ . The discrete moments of  $h$  is given in terms of  $z$  transform as follows

**Lemma 4.1.5** The discrete moments of  $f$  are given by the equation

$$d_i = \left( z \frac{d}{dz} \right)^i H(z)|_{z=1} \quad (4.40)$$

where  $H(z)$  is the  $z$ -transform of  $h$  defined by

$$H(z) = \sum_k h(k) z^{-k}$$

In particular if  $g_1$  and  $g_2$  denote the derivatives of  $H(z)$  evaluated at  $z = 1$ , then  $d_1 = g_1$  and  $d_2 = g_1 + g_2$  and hence we have  $m_1^2 = m_2$ , if and only if  $g_2 = g_1(1 + g_1)$ .

So far  $f$  has been any function that satisfies the two-scale difference equation. What if  $f$  is either  $\phi$  or  $\psi$ ?  $\phi$  satisfies Eqn. 4.36 with  $f = \phi = g$  and  $h = h_0$  while  $\psi$  satisfies the same equation with  $f = \psi$ ,  $g = \phi$  and  $h = h_1$ . The one difficulty with deducing the properties of  $\phi$  and  $\psi$  is that  $\phi$  is characterized by the *magnitude* of the Fourier Transform of  $h_0$  on the unit circle. Hence the only way to obtain  $\phi$  or  $\psi$  or  $h_0$  is numerically by a spectral factorization of the  $|H_0(e^{i\omega})|$ . Yet, from the behavior of  $|H_0(e^{i\omega})|$  in a neighborhood of  $\omega = 0$ , one can deduce certain relationships between the derivatives of  $H_0$  and hence the discrete moments of  $h_0$  and consequently the moments of  $\phi$ . Indeed by differentiating Eqn. 4.34 all derivatives upto order  $K$  are found to be zero. In particular the second derivative is zero. This implies that  $g_2 = g_1(1 + g_1)$  and hence for  $\phi$ ,  $m_1^2 = m_2$ . Since this result is useful later on we note it as a theorem.

**Theorem 4.1.4** For the Daubechies scaling functions  $\phi$ , for  $N \geq 4$ , (i.e excluding the Haar case), we always have that  $m_1^2 = m_2$ .

Again from Eqn. 4.34, using the form of  $h_1$ , it can be shown that the discrete moments of  $h_1$  up to  $N/2$  is zero, and hence the corresponding continuous moments of  $\psi$  are also zero. Explicitly we have

$$\sum_k (2n)^k h_0 = \sum_k (2n+1)^k h_0 (2n+1) \quad (4.41)$$

for  $k \leq N/2 - 1$ .

**Table 4.2: The moments of  $\phi$** 

$N$	$k$	$m_k$	$d_k$
4	0	1.0000000e+00	1.0000000e+00
	1	6.3397460e-01	6.3397460e-01
	2	4.0192379e-01	4.0192379e-01
	3	1.3109156e-01	-6.1121593e-01
	4	-3.0219333e-01	-4.2846097e+00
	5	-1.0658728e+00	-1.6572740e+01
6	0	1.0000000e+00	1.0000000e+00
	1	8.1740117e-01	8.1740117e-01
	2	6.6814467e-01	6.6814467e-01
	3	4.4546004e-01	-1.5863308e-01
	4	1.1722635e-01	-1.8579194e+00
	5	-4.6651091e-02	3.7516197e+00
8	0	1.0000000e+00	1.0000000e+00
	1	1.0053932e+00	1.0053932e+00
	2	1.0108155e+00	1.0108155e+00
	3	9.0736037e-01	2.5392023e-01
	4	5.8377181e-01	-2.0440853e+00
	5	6.3077524e-02	-2.4420547e+00
10	0	1.0000000e+00	1.0000000e+00
	1	1.1939080e+00	1.1939080e+00
	2	1.4254164e+00	1.4254164e+00
	3	1.5802598e+00	8.5092254e-01
	4	1.4513041e+00	-2.0317424e+00
	5	8.1371053e-01	-5.9644946e+00

## 4.2 Multirate Filter Banks and Conjugate Quadrature Filters

Conjugate Quadrature Filter banks are very useful and have been extensively used in sub-band coding and in trans-multiplexers [34, 32, 31, 27, 11]. Conjugate Quadrature Filters are also known as Quadrature Mirror Filters, a term that has been used loosely to refer to any set of filters in a multirate filter bank. A multirate filter bank is two sets of filters that are arranged as shown in Fig. 4.6. The input signal  $x$ , is split into  $M$  channels using an analysis filter bank of  $M$  filters  $h_0$  through  $h_{M-1}$ . The signal in each channel is then decimated by a factor of  $N$ . This constitutes the analysis bank. The signals in each of the  $M$  channels are then interpolated by a factor of  $N$ , by interlacing adjacent samples with  $N - 1$  zeros, and then passed through a set of filters  $g_0$  through  $g_{M-1}$ . The output of all the filters are added together to get the output signal. This constitutes the synthesis bank. Thus the analysis and synthesis banks of a filter bank together take an input signal  $x$ , and produce an output signal  $y$ . By transposing the analysis and synthesis banks, the filter bank (FB) can also be considered to be one that takes in  $M$  signals and outputs  $M$  signals. The former structure is called the subband coder structure since it is used in subband coders and the latter structure is called the transmultiplexer structure since it is useful in transmultiplexers (FDM to TDM and TDM to FDM converters). The goal in a FB design is to design *good* analysis and synthesis filters, where the *goodness* will be determined by the application. The reason for considering FB structures is that the decimation and interpolation operations make the FB structures computationally efficient. Depending on whether  $M$  is less than, equal to, or greater than  $N$ , the QMF bank is called under-decimated, critically (maximally) decimated, or over-decimated. filter bank.

In the remainder, only the subband coder structure is discussed since everything carries over to the transmultiplexer structure naturally.

Historically, two channel FBs were considered where the filters  $h_0$  and  $h_1$  were lowpass and highpass respectively, and the frequency response of the former was the

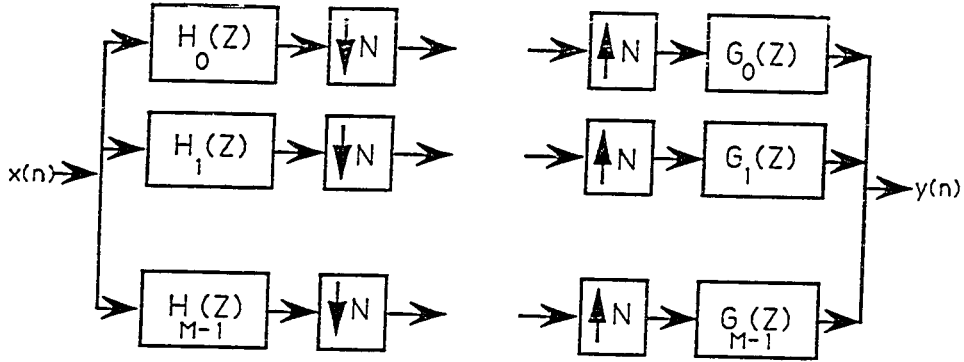


Figure 4.6: A Multirate Filter Bank

mirror image of the latter around  $\pi/2$  as shown in Fig. 4.7. Hence the filters were called Quadrature Mirror Filters (QMFs). In most of the Filter Bank literature, the analysis and synthesis filters, independent of whether they satisfy the QMF property or not, are referred to as QMFs.

The first step in the evolution of FBs was the design two channel critically decimated FBs such that  $x$  and  $y$  were related by a simple convolution despite the multirate nature of the system which introduces aliasing terms ([12]). Thus the FB behaved from input to output exactly like an ordinary filter, with an amplitude and phase distortion. Soon it was realized that not only could the aliasing terms be removed, but exact reconstruction could be achieved. That is, the impulse response of the entire FB is a delayed discrete delta function (upto a constant) [31]. Such FBs will be called Perfect Reconstruction Filter banks (PRFBs). In [31], the PR property was achieved by enforcing a certain *unitariness* property on the analysis and synthesis

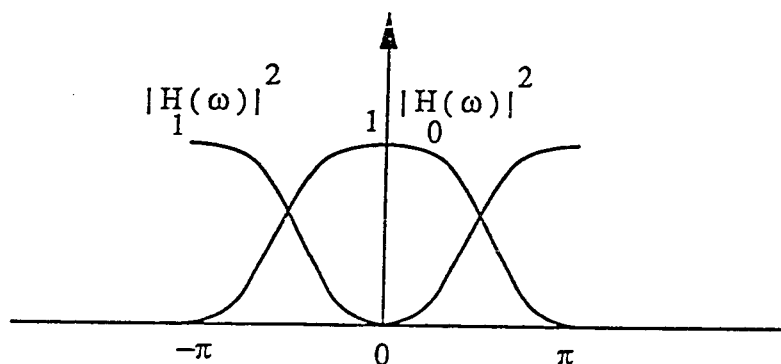


Figure 4.7: The Mirror Property of QMFs

filters. More precisely, the filter  $h_0$  had to satisfy,

$$|H_0(e^{i\omega})|^2 + |H_0(e^{i(\omega+\pi)})|^2 = 1 \quad (4.42)$$

This property of the analysis filters was related to the losslessness property of the *scattering* matrix in classical network theory by P.P.Vaidyanathan [32], enabling the use of various factorization theorems for unitary (scattering) matrices in [1] to be reinterpreted in terms of  $z$  transform matrices, leading to explicit parametrizations of such PRFBs, which are called Conjugate Quadrature Filters (CQFs). Simultaneously it was realized that the analysis and synthesis filters in two channel PRFBs could be made linear phase [27] provided the unitariness property was given up. Such filters are called linear phase PR filters (LPPRFs). It was also realized that if the number of channels is greater than 2 then linear phase can be achieved without giving up the unitariness condition. It is not clear why linear phase is always important in a PRFB, since anyway the input  $x$  and output  $y$  are identical, (to wit, related by a linear phase filter). It is well known that the symmetry of linear phase filters halves

the computation of direct convolutions with those filters. Since a filter bank can in itself be implemented efficiently as will be seen shortly, it is not clear, at least for most applications, whether designing linear phase PRFBs is really necessary. Nonetheless, LPPRFs are closely related to the construction of bi-orthogonal wavelet bases [33], just as CQFs are related to ON wavelet bases and tight frames.

Let  $h_0, h_1, \dots, h_{M-1}$  be the analysis filters, and let  $g_0, g_1, \dots, g_{M-1}$  be the synthesis filters in an  $M$ -channel PRFB. Let the modulated analysis filter matrix be defined as,

$$H_m(z) = \begin{bmatrix} H_0(z) & H_0(Wz) & \dots & H_0(W^{N-1}z) \\ H_1(z) & H_1(Wz) & \dots & H_1(W^{N-1}z) \\ \vdots & \vdots & \dots & \vdots \\ H_{M-1}(z) & H_{M-1}(Wz) & \dots & H_{M-1}(W^{N-1}z) \end{bmatrix} \quad (4.43)$$

where  $W = e^{i2\pi/N}$  and  $H_i(z)$  is the  $z$  transform of  $h_i$ . Let

$$x_m(z) = (X(z), X(zW), \dots, X(zW^{N-1}))^T \quad (4.44)$$

be the modulated input vector, and

$$g(z) = (G_0(z), G_1(z), \dots, G_{M-1}(z))^T \quad (4.45)$$

be the output filter vector, where  $G_i(z)$  is the  $z$  transform of  $g_i$ . Let  $c_i$  be the output sequences of the analysis bank. Let  $c(z)$  be the vector of  $z$  transforms of the outputs of the analysis bank. That is

$$c(z) = (C_0(z), C_1(z), \dots, C_{M-1}(z))^T \quad (4.46)$$

with  $C_i(z)$  being the  $z$  transform of  $c_i$ . Then it is easily seen that,

$$c(z) = \frac{1}{N} H_m(z^{1/N}) x_m(z^{1/N}) \quad (4.47)$$

The output  $Y(z)$  is related to  $c(z)$  by the synthesis bank as

$$Y(z) = g^T(z) c(z^N) \quad (4.48)$$



which implies

$$Y(z) = 1/N(g(z))^T H_m(z) x_m(z) \quad (4.49)$$

For perfect reconstruction  $Y(z)$  must be related to  $X(z)$  which is the first component of  $x_m(z)$ , by just a delay. Therefore,

$$(H_m(z))^T g(z) = (z^{-k}, 0, \dots, 0)^T \quad (4.50)$$

for some integer  $k$ . Thus once an arbitrary analysis bank of filters  $\{h_i\}$  is chosen it is clear how to obtain the synthesis bank, i.e., by inverting Eqn. 4.50. Even assuming that this  $z$  matrix inversion can be carried out, the filters  $g(z)$  may not be stable. One way to avoid instability problems in a filter is to make it FIR. If  $g(z)$  is ensured to have polynomial entries then all the synthesis filters would be FIR. It is clear that this can happen if and only if  $\det(H_m(z))$  is of the form  $z^j$  for some integer  $j$ . If the modulated analysis matrix is unitary on the unit circle ( $z = e^{i\omega}$ ), then trivially it follows that PR is achieved without any instability problems. This is precisely the unitariness condition referred to earlier. Notice that all we need is for  $H_m(z)$  to be left unitary, i.e

$$H_m(z^{-1}) H_m^T(z) = \mathbf{I} \quad (4.51)$$

**Definition 4.2.1** A CQF bank is a PRFB where the modulated filter matrix corresponding to the analysis filters is left unitary.

The corresponding modulated synthesis filter matrix will also be left unitary. In the critically sampled case, the synthesis filters are merely the time reversed versions of the analysis filters. That is,

$$g(z) = z^{-k} h(z^{-1}) \quad (4.52)$$

In the case of orthonormal wavelets with a scaling factor of 2, the filters  $h_0$  and  $h_1$  satisfy the orthogonality conditions (Eqn. 4.22 and Eqn. 4.21) which may be written in the form

$$H_m(z) H_m^T(z^{-1}) = 1 \quad (4.53)$$

where

$$H_m(z) = \begin{bmatrix} H_0(z) & H_0(Wz) \\ H_1(z) & H_1(Wz) \end{bmatrix} \quad (4.54)$$

$W = e^{i2\pi/2} = -1$  in the above equations. Now it is clear that the modulated filter matrix associated with  $h_0$  and  $h_1$  is a CQF. Notice that the orthonormality properties of the wavelets manifests itself as a condition on the associated modulated filter matrix that makes the sequences  $h_0$  and  $h_1$  have the CQF property. This might give one the impression that the orthonormality of the wavelets and the CQF property are synonymous. This is true only in the critically decimated case. If  $M \neq N$  then the orthonormality condition implies that  $H_m$  is right unitary while the CQF condition implies that it is right unitary. This can also be understood from the dimensionality of  $H_m$ . If  $M > N$  then orthonormality cannot be achieved because there are *too many channels*. On the other hand if  $M < N$  then the channels have been overdecimated and there is no way to reconstruct the signal, and hence no way to satisfy the CQF condition.

### 4.3 Computation with FIR CQF banks

For the general FIR CQF bank the most efficient way to implement the analysis and synthesis banks is to use polyphase filter structures [29]. The polyphase implementation has an associated polyphase filter matrix that is related to the modulated filter matrix of the previous section by a Discrete Fourier Transform as will be seen shortly.

The analysis bank has to implement  $M$  equations of the form

$$c_i(n) = \sum_k h_i(k)x(Nn - k) \quad (4.55)$$

Let  $H_i(z)$ ,  $X(z)$ , and  $C_i(z)$  denote the  $z$  transforms of  $h_i$ ,  $x$ , and  $c_i$  respectively. Then,

$$H_i(z) = \sum_{k=0}^{N-1} z^{-k} H_{i,k}(z^N) \quad (4.56)$$

where,

$$H_{i,k}(z) = \sum_n h(k + Nn)z^{-n} \quad (4.57)$$

and

$$X(z) = \sum_{k=0}^{N-1} z^{-k} X_k(z^N) \quad (4.58)$$

where,

$$X_k(z) = \sum_n x(k + Nn)z^{-n} \quad (4.59)$$

The functions  $H_{i,k}$  and  $X_k$  and the corresponding sequences are called the polyphase components of the filter  $h_i$  and the input  $x$  respectively. Then  $C_i(z)$  is given by

$$C_i(z) = X_0(z)H_{i,0}(z) + \sum_{k=1}^{N-1} X_k(z)H_{i,N-k}(z) \quad (4.60)$$

This is the polyphase implementation of Eqn. 4.55. Fig. 4.8 shows the block diagram of the polyphase implementation of a filter.

Notice that only the output values that are not being thrown away by the decimation operation is being computed. This is the reason why the polyphase implementation is computationally efficient compared to the direct implementation.

Now for the entire analysis bank, define the polyphase analysis filter matrix as follows,

$$H_p(z) = \begin{bmatrix} H_{0,0}(z) & H_{0,1}(z) & \cdots & H_{0,N-1}(z) \\ H_{1,0}(z) & H_{1,1}(z) & \cdots & H_{1,N-1}(z) \\ \vdots & \vdots & \cdots & \vdots \\ H_{M-1,1}(z) & H_{M-1,1}(z) & \cdots & H_{M-1,N-1}(z) \end{bmatrix} \quad (4.61)$$

Let  $J$  be the matrix defined by,

$$J = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix} \quad (4.62)$$

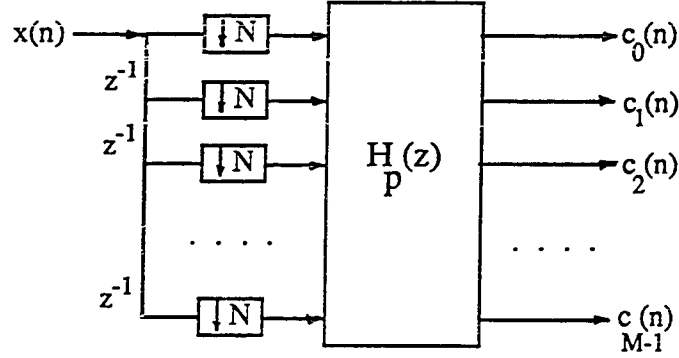


Figure 4.8: Polyphase Implementation of an Analysis Filter Bank

and let the polyphase input vector  $x_p$  be defined by,

$$x_p(z) = (X_0(z), X_1(z), \dots, X_{N-1}(z))^T \quad (4.63)$$

Then the analysis bank equation can be written as

$$c(z) = H_p(z) J x_p(z) \quad (4.64)$$

If the length of the analysis filters are each  $K_h$  and the length of the input is  $K_x$  then, the direct computation of the analysis bank requires  $M(K_x + K_h - 1)K_h$  multiplications, while the polyphase structure requires  $MN \lfloor \frac{K_h}{N} \rfloor (\lfloor \frac{K_h}{N} \rfloor + \lfloor \frac{K_x}{N} \rfloor - 1)$  multiplications. The computational complexity of the polyphase implementation is lesser than that of the direct implementation by a factor of  $N$ .

Now consider the synthesis bank that must implement the equation

$$y(n) = \sum_{i=0}^{M-1} c_i(k) g_i(n - Nk) \quad (4.65)$$

Equivalently, in terms of  $z$  transforms,

$$Y(z) = g^T(z)c(z^N). \quad (4.66)$$

Let the polyphase output vector  $y_p$  be given by,

$$y_p(z) = (Y_0(z), Y_1(z), \dots, Y_{M-1}(z))^T \quad (4.67)$$

and the polyphase synthesis matrix by

$$G_p(z) = \begin{bmatrix} G_{0,0}(z) & G_{0,1}(z) & \cdots & G_{0,N-1}(z) \\ G_{1,0}(z) & G_{1,1}(z) & \cdots & G_{1,N-1}(z) \\ \vdots & \vdots & \cdots & \vdots \\ G_{M-1,0}(z) & G_{M-1,1}(z) & \cdots & G_{M-1,N-1}(z) \end{bmatrix} \quad (4.68)$$

Then,

$$y_p(z) = G_p^T(z)c(z) \quad (4.69)$$

This is the polyphase implementation of the synthesis bank. This equation does not do any of the trivial computations in a direct implementation of the synthesis bank.  $y(n)$  is obtained by interlacing the sequences corresponding to the components of  $y_p(z)$ . Fig. 4.9 shows the polyphase structure for the synthesis bank. The polyphase synthesis bank is superior to the direct implementation approximately by a factor of  $N$ .

The analysis and synthesis banks can be characterized by either the polyphase representation or the modulated representation. The modulated representation is the more natural representation, while the polyphase representation is computationally more efficient. The two representations are related as follows: If  $x_m$  is a modulated vector, then it is related to  $x_p$  the corresponding polyphase vector as

$$x_m(z) = DFT(N) \text{diag}\{z^{-1}, z^{-2}, \dots, z^{-N+1}\} x_p(z^N) \quad (4.70)$$

where  $DFT(N)$  is the length  $N$ , DFT matrix. From Eqns. 4.47 and 4.64,

$$H_p(z)Jx_p(z) = \frac{1}{N}H_m(z^{1/N})x_m(z^{1/N}) \quad (4.71)$$

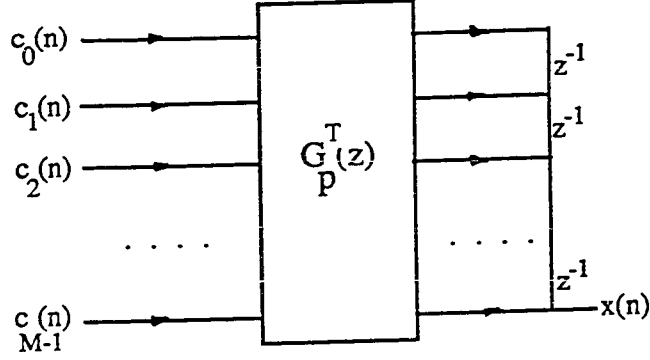


Figure 4.9: Polyphase Implementation of an Synthesis Filter Bank

which implies,

$$H_m(z) = DFT(N) \text{diag}\{z^{-1}, z^{-2}, \dots, z^{-N-1}\} H_p(z^N) \quad (4.72)$$

A similar relationship holds for the corresponding synthesis matrices too. From the last equation, it is clear that the unitariness of the matrix  $H_m$  is equivalent to the unitariness of the matrix  $H_p$ . Thus the analysis filters form a CQF iff  $H_p$  is unitary. This characterization of CQFs is superior to the characterization in terms of  $H_m$ . This is because a filter coefficient (of any filter) in the analysis bank affects exactly one and only element in  $H_p$ , while it occurs in all the rows of a column in  $H_m(z)$ .  $H_p$  is a general unitary matrix in  $z$  while  $H_m$  is not. Hence all the beautiful theorems on the factorizations of unitary polynomial matrices in [1] can be applied directly to  $H_p$  to get a set of independent *lattice* parameters for the CQF bank.

### 4.3.1 Lattice structures and CQFs

Every FB has a polyphase analysis and synthesis structures associated with it, independent of whether the filters form a CQF or not. But when the analysis filters do form a CQF there are some special lattice structures associated with the CQFs. This is related to the unitariness of  $H_p$ . Lattice structures for analysis and synthesis in  $M$  channel critically decimated CQFs are well understood [11]. They were developed to characterize the CQFs with an independent set of parameters, so that the design of CQFs is facilitated. Essentially the lattice structures depend on the factorization of unitary polynomial matrices. The filters arising naturally in the theory of ON wavelets are CQFs. Therefore we proceed to review lattice structures for analysis and synthesis in two channel critically decimated CQF banks.

Consider the analyzer part of a two-channel critically decimated CQF bank. The input  $x$  goes through the analyzer to give sequences  $c_0$  and  $c_1$ . Both  $c_0$  and  $c_1$  depend linearly on  $x$ . Furthermore, if  $x$  is delayed by 2 units,  $c_0$  and  $c_1$  are delayed by 1 unit. Thus the analyzer can be considered to be a 1 input 2 output periodically time-varying system with period 2. Now consider the polyphase components  $x_0$  and  $x_1$  of the input. Both  $c_0$  and  $c_1$  depend linearly on  $x_0$  and  $x_1$  and furthermore, the dependence is shift invariant. That is, a delay of 1 unit in  $x_0$  leads to a delay in both  $c_0$  and  $c_1$ , when  $x_1$  is zero. Thus, the analyzer can also be considered to be a 2 input ( $x_0$  and  $x_1$ ), 2 output linear shift-invariant system. Thus there would be a  $2 \times 2$   $z$  transform matrix relating  $C_0(z)$  and  $C_1(z)$  with  $X_0(z)$  and  $X_1(z)$ .

Since the analyzer filters form a CQF, the energy of input  $x$  gets distributed between  $c_0$  and  $c_1$ . Planar rotations and shifts are the only unitary shift-invariant operations for a two input 2 output system of degree 1. Thus it is reasonable to expect that the most general 2 input 2 output unitary shift invariant system can be implemented as a cascade of planar rotation and shift blocks.

In fact  $c_0$  and  $c_1$  are related to  $x_0$  and  $x_1$  by a cascade of rotation and shift blocks as

$$\begin{bmatrix} C_0(z) \\ C_1(z) \end{bmatrix} = \begin{bmatrix} \cos \theta_{N/2-1} & \sin \theta_{N/2-1} \\ -\sin \theta_{N/2-1} & \cos \theta_{N/2-1} \end{bmatrix} \prod_{i=0}^{N/2-2} \begin{bmatrix} 1 & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} \cos \theta_i & \sin \theta_i \\ -\sin \theta_i & \cos \theta_i \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix} \quad (4.73)$$

Here  $N$  is the length of filter  $h_0$ . Since this implements the filters  $h_0$  and  $h_1$ , the parameters  $\theta_i$  are related to the filter  $h_0$ . The angles  $\theta_i$  could also be parametrized in terms of  $\sin \theta_i$ ,  $\cos \theta_i$  or  $\tan \theta_i$ . The above equation represents the lattice implementation of the analysis bank. The structure above is called the normalized lattice structure to distinguish it from the de-normalized structure where the tangent of the angle is used to parametrize the lattice. In the remainder the tangent parametrization is used since it involves the least amount of computation as is seen below. Let

$$\beta = \prod_{i=0}^{N/2-1} \cos(\theta_i) \quad (4.74)$$

and let

$$k_i = \tan \theta_i \quad (4.75)$$

Then we have,

$$\begin{bmatrix} C_0(z) \\ C_1(z) \end{bmatrix} = \beta \begin{bmatrix} 1 & k_{N/2-1} \\ -k_{N/2-1} & 1 \end{bmatrix} \prod_{i=0}^{N/2-2} \begin{bmatrix} 1 & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & k_i \\ -k_i & 1 \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix} \quad (4.76)$$

The next step is to obtain the  $k$  parameters from the filters. Let  $H_0^N(z)$  and  $H_1^N(z)$  be the two analysis filters. Here  $N$  denotes the length of the filters, or equivalently,  $N-1$  is the degree of the  $z$  transforms. Then by a suitable planar rotation, the  $H_0^N$  and  $H_1^N$  are transformed into another CQF pair,  $H_0^{N-2}$  and  $H_1^{N-2}$  of degree two less. Repeating this process of extraction of rotations until the degree of the resulting CQF becomes zero we get all the rotation angles. Define,

$$k_{N/2-1} = -h_0^N(N-1)/h_1^N(N-1) \quad (4.77)$$



Then it can be shown that

$$\begin{bmatrix} H_0^{N-2}(z) \\ z^{-2}H_1^{N-2}(z) \end{bmatrix} = \frac{1}{1+k_{N/2-1}^2} \begin{bmatrix} 1 & k_{N/2-1} \\ -k_{N/2-1} & 1 \end{bmatrix} \begin{bmatrix} H_0^N(z) \\ z^{-2}H_1^N(z) \end{bmatrix} \quad (4.78)$$

where  $H_0^{N-2}(z)$  and  $H_1^{N-2}(z)$  form a CQF pair of length  $N-2$ . The parameters extracted are precisely the tangent parameters.

As for the synthesis bank  $x$  has to be obtained from  $c_0$  and  $c_1$ . The equation for the  $c_i$ s in terms of the  $x_i$ s can be directly inverted to obtain the  $x_i$ 's from which gives  $x$ . It is seen that the synthesis bank is identical in structure to the analysis bank. Moreover, in software, the same program may be used for both the analysis and synthesis. The synthesis bank equation is given by,

$$\begin{bmatrix} C_1(z) \\ C_0(z) \end{bmatrix} = \beta \begin{bmatrix} 1 & k_{N/2-1} \\ -k_{N/2-1} & 1 \end{bmatrix} \prod_{i=0}^{N/2-2} \begin{bmatrix} 1 & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} 1 & k_i \\ -k_i & 1 \end{bmatrix} \begin{bmatrix} X_1(z) \\ X_0(z) \end{bmatrix} \quad (4.79)$$

The synthesis bank can also be implemented as a lattice. Programs *htok.m*, and *ktoh.m* in the appendix A, respectively convert the filter  $h_0$  to the  $k$  parameters and vice versa.

#### 4.4 Generalization of ON wavelet bases

The theory of ON wavelets can be generalized naturally in two directions. Firstly the scale factor 2 in the fundamental scaling equation could be replaced by any integer  $M$ . Secondly, one could obtain ON wavelet bases for  $L^2(\mathbb{R}^n)$ .

Consider a function  $\phi$  and a multiscale analysis where the scaling property is replaced with a factor  $M$ . That is,

$$\phi(t) = M \sum_k h_0(k) \phi(Mt - k) \quad (4.80)$$

As in the case where scale factor was 2, there is a sequence of embedded closed vector spaces  $\{V_j\}$ . Define

$$\phi_{j,k} = \sqrt{M^j} \phi(M^j t - k) \quad (4.81)$$

Then  $V_j$  is spanned by  $\{\phi_{j,k}\}$  for fixed  $j$ . Now it is reasonable to expect that there exist functions  $\psi_1, \psi_2, \dots, \psi_{M-1}$  such that their integral translates span the orthogonal complement of  $V_0$  in  $V_1$ , though no one has come up with a proof so far. Moreover the spaces  $W_j^0, W_j^1, \dots, W_j^{M-1}$  would be such that

$$V_j = V_{j-1} \oplus W_{j-1}^0 \oplus W_{j-1}^1 \oplus \dots \oplus W_{j-1}^{M-1} \quad (4.82)$$

for all  $j$ . Thus we have filters  $h_1, h_2, \dots, h_{M-1}$ . If the wavelet basis is orthonormal then the bank of filters  $h_0$  through  $h_{M-1}$  should form an  $M$ -channel maximally decimated CQF. Again most of the properties for the case  $M = 2$  carry over to orthonormal wavelet bases arbitrary  $M$ .

As for multidimensional wavelet bases, one could directly obtain them as a tensor product of one dimensional wavelets. A more general approach would be to consider a multiscale analysis framework with  $\mathbf{R}$  replaced by  $\mathbf{R}^n$ . In this case the scaling factor has to be replaced by a scaling matrix  $M$ , whose entries are real but whose determinant is some integer  $m$ , which gives the number of wavelets plus one (since it includes the scaling function too). Both these generalizations have been considered and constructed in [28]. Ref. [23] also gives some facts about these generalizations.

## Chapter 5

### Computational Techniques using ON Wavelets

In the remainder of the thesis the only DWT wavelets considered are the Daubechies wavelets. The wavelets are parametrized by an integer  $N$  and the wavelet is uniquely specified by the minimal phase condition. All this is clearly described in [7]. In this chapter we give a brief summary of the computational techniques associated with ON wavelets. The relevant MATLAB programs may be found in the appendix. Though the wavelet based analysis of a signal involves analog processing, it turns out that one can do very well by processing only in the discrete domain.

#### 5.1 Generation of $\phi$ and $\psi$

Both  $\phi$  and  $\psi$  are generated from the corresponding filters  $h_0$  and  $h_1$ . There are two techniques to generate the scaling functions and the wavelets. Both techniques compute the values of  $\phi$  and  $\psi$  at the dyadic rationals. We will assume that  $\phi$  and  $\psi$  are continuous. The first method is best understood as a time-domain technique, while the second is best understood as a frequency-domain technique.

##### 5.1.1 Interpolation Method

The values of  $\phi$  and  $\psi$  are obtained initially at the integers (remember that by assumption  $\phi$  and  $\psi$  are continuous). Using Eqn. 4.4 the values of  $\phi$  at half integers are obtained. This process is continued recursively to obtain  $\phi$  at all the dyadic rationals. To obtain the values of  $\phi$  and  $\psi$  at the integers, from the scaling equation one obtains an eigen-equation for the values of  $\phi$  at the integers. Let  $\Phi(t)$  be a vector of length

$N$  defined by

$$\Phi_j(t) = \phi(t + j); 0 \leq j < N \quad (5.1)$$

Since  $\phi$  is supported in  $[0, N - 1]$ , it is clear that  $\phi(t)$  is completely determined by  $\Phi(t)$  for  $t \in [0, 1]$ . Now define two  $N \times N$  matrices  $M_0$  and  $M_1$  by

$$\begin{aligned} (M_0)_{ij} &= 2h_0(2i - j) \\ (M_1)_{ij} &= 2h_0(2i - j + 1) \end{aligned} \quad (5.2)$$

Then Eqn. 4.4 becomes

$$\Phi(t) = \begin{cases} M_0 \Phi(2t) & \text{if } 0 \leq t \leq 1/2 \\ M_1 \Phi(2t - 1) & \text{if } 1/2 \leq t \leq 1 \end{cases} \quad (5.3)$$

In particular when  $t = 0$  we have that

$$\Phi(0) = M_0 \Phi(0)$$

Thus the value of  $\phi$  at the integers can be obtained by solving this eigenvalue problem.  $\Phi$  is simply the eigenvector of  $M_0$  corresponding to an eigenvalue of 1. In fact if  $t = 1$  we have

$$\Phi(1) = M_1 \Phi(1)$$

Thus we can again obtain the value of  $\phi$  at the integers by solving this equation. If  $\phi$  is continuous it is completely specified by its values at the dyadic rationals (i.e real numbers that admit a representation of the form

$$\sum_{i=0}^{N-1} t_i 2^i$$

for finite  $M$  and  $N$  and  $t_i \in \{0, 1\}$ ). Let  $t$  be a dyadic rational in  $[0, 1]$ . Then  $t$  can be written as  $.t_1 t_2 t_3 \dots t_K$ . Then from Eqn. 5.3 one easily obtains

$$\Phi(.t_1 t_2 \dots t_K) = M_{t_1} M_{t_2} \dots M_{t_K} \Phi(0) \quad (5.4)$$

Thus  $\phi$  at the dyadic rationals can be obtained. The same analysis can be done for  $\psi$ . Define matrices  $N_0$  and  $N_1$  by

$$\begin{aligned}(N_0)_{ij} &= 2h_1(2i - j) \\ (N_1)_{ij} &= 2h_1(2i - j + 1)\end{aligned}\tag{5.5}$$

Then we have

$$\Psi(.t_1 t_2 \dots t_K) = \tilde{N}_{t_1} \tilde{M}_{t_2} \dots \tilde{M}_{t_K} \Phi(0)\tag{5.6}$$

Though this notation suggests that matrix products are involved in the computation, actually once the values at the integers are obtained (by solving the system of linear equations) the rest is merely going through the synthesis equations of the CQF bank corresponding to the wavelet and can be done very efficiently. This technique for generating  $\phi$  and  $\psi$  was first used by H.Resnikoff and later used by I.Daubechies to study the differentiability properties of the wavelets.

### 5.1.2 Infinite Product Method

This method is a frequency domain technique and computes only an approximation to  $\phi$  and  $\psi$ . The Fourier Transform of  $\phi$ , can be written as an infinite product. The infinite product in the frequency domain becomes an infinite convolution in the time domain. The infinite convolution is directly implemented. This is illustrated in Fig 5.1. A very good approximation is obtained by terminating the infinite product after about 8 or 9 stages. This technique for generating the  $\phi$  and  $\psi$  was first used by I.Daubechies.

In the first method the values of  $\phi$  at the integers are first obtained exactly. Hence the values at the dyadic rationals are also obtained exactly within limits of numerical errors. In the second method since an infinite product is truncated only an approximation of  $\phi$  is obtained. In practise the error in generating using this method is very small after about 8 iterations are done. The big advantage with the second method is that there is no need to solve a system of linear equations. Both

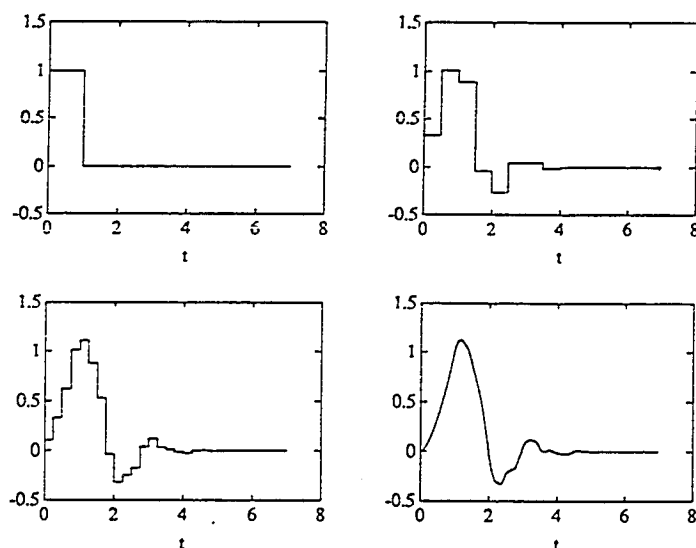


Figure 5.1: Generation of  $\phi$  using infinite product

techniques are identical in their implementation in that the interpolation starts with the values at the integers and goes through a process while the infinite convolution method starts off with a delta and goes through the same process. Eventually they both converge to the same values.

## 5.2 Analysis/Synthesis in Wavelet Bases

The wavelet expansion of a function  $f$  can be written as

$$f(t) = \sum_{j,k} W(j, k, f) \psi_{j,k}$$

On a computer since one cannot represent infinite sequences this form cannot be used. If  $f$  is compactly supported, which is a realistic assumption, and if  $\psi$  is compactly supported, then the index  $k$  can take only a finite number of values. As for the index  $j$  since the wavelets are bandpass filters, to represent any function with DC  $j$  begins at  $-\infty$ . To avoid this one has use the scaling function too in the expansion for  $f$ . Again,

since the signal energy will usually be very small for sufficiently large frequency, it is realistic to assume that the wavelet coefficients of a signal are 0 for all levels greater than  $j_1$ . Thus one has to expand  $f$  as

$$f(t) = \sum_k S(j_0, k, f) \phi_{j_0, k}(t) + \sum_k \sum_{j=j_0}^{j_1} W(j, k, f) \psi_{j, k}(t) \quad (5.7)$$

How do we obtain the coefficients in the above equation? Since the coefficients are merely the inner-product of  $f$  with the corresponding scaling function, one method is to directly evaluate these inner products by numerical integration. Besides, usually a signal is specified in terms of its samples, and hence to completely specify the signal, one has to interpolate between the samples. Both integration and interpolation, introduce errors. Since, as already mentioned, practically all signals have a certain *finest scale of interest*, we can represent the signal completely in terms of scaling functions at that scale. We now discuss how to project a signal onto the finest scale of interest. More precisely we discuss how to compute  $Sf(j_1 + 1, k)$  since this completely determines all the coefficients in Eqn. 5.7

### 5.2.1 The Projection onto the Finest Scale

We know that

$$\int \phi_{j, k}(t) dt = 1 \quad (5.8)$$

With increase in  $j$  the support of  $\phi_{j, k}$  decreases. Hence for sufficiently large  $j$ , say  $j_1 + 1$ , the functions  $\phi_{j, k}$  can be considered to be the Dirac delta measure. Then the *scaling coefficients* at the level  $j_1 + 1$  are given by the samples of the functions as follows.

$$Sf(j_1 + 1, k, f) = f(2^{-j_1 - 1} k) \quad (5.9)$$

Notice that since the support of  $\phi$  is in  $(0, N - 1)$ , vanishing at both endpoints, it makes more sense to sample  $f$ , where the function is concentrated, i.e at its maximum

value or at its center of mass. That is if  $m_1$  is given by

$$m_1 = \int dt t \phi(t) \quad (5.10)$$

then sampling  $f$  on the lattice  $\{2^{-j_1-1}(k + m_1) \mid k \in \mathbb{Z}\}$ , would give a good approximation.

$$Sf(j_1 + 1, k) = f(2^{-j_1-1}(k + m_1)) \quad (5.11)$$

In the initial experiments with D wavelets this approach was used and found to give *very good* approximations even for moderate values of  $j_1$ . We shall see the reason for this later.

The second approach is to sample  $f$  uniformly at the rate of  $2^{j_1+1}$  and use any of the standard interpolation techniques, like piecewise Lagrangian, Hermite, or variations thereof, spline interpolation etc. and integrate by quadrature. Since by Eqn. 4.39 we know that the moments of  $\phi$  can be computed *exactly*, integration by quadrature reduces to some form of convolution and hence is greatly simplified.

The third method is to use the cartographers' technique of approximating the function  $f$  locally (by a polynomial in our case!) for finding the scaling coefficients at the finest level required and go through the analyzer. Notice that this is quite different from interpolating with some set of basis functions and integrating as in the previous paragraph. Arguments in favor of this technique can be given intuitively. Besides, computationally this technique is the best and has been studied extensively [21]. From the point of view of accuracy also this works pretty well. This can be seen as follows. Let us say we want to obtain  $S(j_1 + 1, 0)$ . In fact, we have from a Taylor series expansion of  $f$  around 0

$$f(t) = f(0) + tf'(0) + \frac{t^2}{2!}f^{(2)}(0) + \frac{t^3}{3!}f^{(3)}(\theta) \quad (5.12)$$

where  $\theta \in (0, t)$ . Now integrating with  $\phi$  shifted to the center of mass, we have

$$\int dt f(t) \phi(t + m_1) = f(0) + \frac{m_3 - m_1^3}{3!} f^{(3)}(\theta) \quad (5.13)$$



Thus the samples of  $f$  themselves give a third order approximation to the coefficients  $S(0, k, f)$ . This automatically implies that the samples of  $f$  at  $S(2^{-j_1-1}(k + m_1), f)$  give a third order approximation of  $Sf(j_1 + 1, k, f)$ . In other words by increasing the sampling by a factor of 2, we decrease the approximation error by a factor of 8!. This is precisely the reason why the samples themselves gave excellent numerical results even for moderate values of  $j_1$ . This result was first noticed in numerical investigations with wavelets by W.M. Lawton [21].

As an aside, we mention that for all the programs in the appendix the projection is done by merely taking the samples. Hence this operation is assumed to have zero computational cost.

### 5.2.2 One level Analyzer

We know that  $W_j \oplus V_j = V_{j+1}$ . Thus  $S(j, k, f)$  and  $W(j, k, f)$  can be obtained from  $S(j + 1, k, f)$ . From Eqn 4.4

$$\phi_{j,k}(t) = \sqrt{2} \sum_l h_0(l) \phi_{j+1,k}(t - 2^{-1}(l - 2k)) \quad (5.14)$$

Now projecting  $f$  onto the functions above we get

$$S(j, k, f) = \sqrt{2} \sum_l h_0(l - 2k) S(j + 1, l, f) \quad (5.15)$$

Similarly from Eqn 4.18 we obtain

$$W(j, k, f) = \sqrt{2} \sum_l h_1(l - 2k) S(j + 1, l, f) \quad (5.16)$$

These are precisely the equations for the analyzer part of a two-channel CQF bank as shown in Fig. 5.2. The obvious method to implement the analyzer is to simply filter the sequence  $S(j + 1, \cdot, f)$  through  $z^{-(N-1)}H_0(z^{-1})$  and  $z^{-(N-1)}H_1(z^{-1})$  and take every other sample. This could either be implemented as a polyphase structure or as a lattice structure. The advantages of a lattice structure are plenty.

1. The basic building blocks for the structure are simple rotation and shifts.

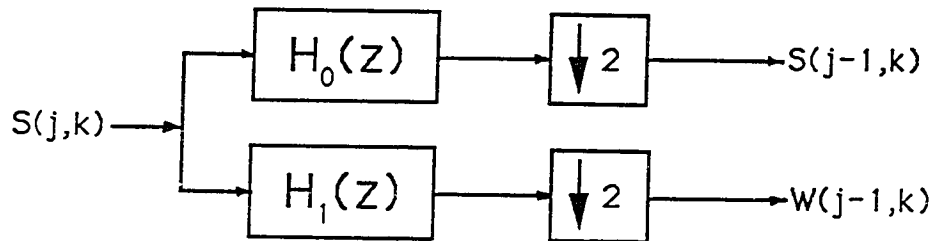


Figure 5.2: The One Level DWT analyzer

2. The structure is simple and regular and modular, and hence amenable to VLSI implementation. With fixed point implementation a number of tricks can also be used to increase efficiency [14].
3. Since both rotations and shifts are unitary, the condition number of the mapping from input to output is 1, and hence the error properties of this implementation are very good.
4. Again because the mapping is unitary, the inverse transform is merely the transpose and hence the inverse transform can be computed using the same program as the forward transform.
5. The lattice structure uses the minimal number of parameters to characterize a CQF.

Since we are not using an arbitrary CQF, but a specific CQF, that has to satisfy the regularity conditions, we might expect that the incorporation of this will further

reduce the computation [21]. This is indeed true, and if the number of multiplies is all that is critical, then this can be accomplished. But in the process the CQF properties of the filters cannot be exploited. Besides the structure is not very regular and the number of adds get out of hand (analogous to the Winograd Fourier Transform in the theory of Fast Discrete Fourier Transform algorithms).

Since the Daubechies CQFs for a fixed even  $N$  are unique, ( upto the degree of freedom in a spectral factorization and assuming the polynomial  $R$  in the Daubechies' characterization of CQFs is zero), and since in a lattice characterization of a CQF there are  $N/2$  independent parameters, the remaining  $N/2$  degrees of freedom (in the Daubechies CQFs), manifest themselves as some non-linear relationships between the angles in the lattice parametrization. These  $N/2$  conditions arise from the  $N/2 - 1$  conditions in Eqn. 4.41 and from the summation condition all on the filter  $h_0$ . For example the latter condition expresses itself in the form of the sum of the angles in the lattice parametrization being  $-45^\circ$ .

Practically the most irksome thing about wavelet-based signal analysis is that one has to keep in mind at all times the time origin of the sequences that denote the scaling or the wavelet coefficients at each level. This problem can be alleviated in some cases by using a periodization of the wavelets, and assuming that the signal being analyzed is periodic.

### Complexity of the One Level Analyzer

Let the number of filter coefficients be  $N$  (i.e the number of lattice parameters is  $N/2$ ). The number of additions in the lattice for a pair of output points is  $N$ , while the number of multiplies is  $N + 2$  (assuming de-normalized implementation) as can be seen from Fig. 5.3. Let  $K$  be the number of input samples. Then number of output samples in the next coarser level for both the scaling and the wavelet levels are both  $\lceil \frac{K+N-1}{2} \rceil$ , where  $N$  is the length of the filter  $h_0$ . The total number of adds  $A$ , and

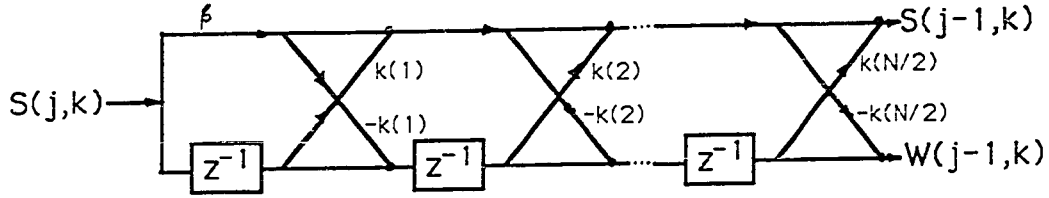


Figure 5.3: Lattice Analyzer of Two Channel CQF

number of multiplies  $M$  are therefore given by,

$$\frac{N}{2}K < A = N\left\lceil \frac{K+1}{2} \right\rceil + \frac{N^2 - 2N}{4} < \frac{N}{2}(K+N) \quad (5.17)$$

and

$$\left(\frac{N}{2} + 1\right)K < M = (N+2)\left\lceil \frac{K+1}{2} \right\rceil + \frac{N^2 - 2N}{4} < \left(\frac{N}{2} + 1\right)(K+N) \quad (5.18)$$

The total complexity per input sample is approximately  $N$ .

### 5.2.3 One level Synthesizer

Given  $S(j, k, f)$  and  $W(j, k, f)$ ,  $S(j+1, k, f)$  is obtained as follows. Actually since the one level analyzer is exactly the implementation of the analysis part of a perfect reconstruction CQF bank, one would expect the one-level synthesizer to be the corresponding synthesis filter bank. Clearly  $P_{j+1}f = P_jf + Q_jf$ . We have

$$F_{j+1}f = \sum_k S(j, k, f)\phi_{j,k} + W(j, k, f)\psi_{j,k} \quad (5.19)$$

and

$$S(j+1, k, f) = \langle \psi_{j+1, k}, P_{j+1} f \rangle \quad (5.20)$$

By substitution we get

$$S(j+1, k, f) = \sqrt{2} \left[ \sum_l h_0(l-2k) S(j, l, f) + h_1(l-2k) W(j, l, f) \right] \quad (5.21)$$

which is precisely the equations governing the synthesis bank of a CQF bank. Again this is best implemented using a lattice structure.

### Complexity of the one level synthesizer

The lattice implementation of the one level synthesizer is shown in Fig. 5.4. Notice that both the analyzer and the synthesizer can be implemented by the same program. Let  $K$  denote the number of the scaling and the wavelet coefficients. Then the number of additions and multiplications are given by the following formulas

$$M = (N+2)K + \frac{N^2 - 2N}{4} \quad (5.22)$$

$$A = NK + \frac{N^2 - 2N}{4} \quad (5.23)$$

Once again the complexity per input point is approximately  $N$  (if each input in both the scaling and the wavelet sections are counted separately). The number of output points is given by  $2K + N - 1$ .

#### 5.2.4 Modified one level analyzer

Later on we will have need to implement the following equation

$$c_0(k) = \sum_l h_0(l) x(2k + Ml) \quad (5.24)$$

and

$$c_1(k) = \sum_l h_1(l) x(2k + Ml) \quad (5.25)$$

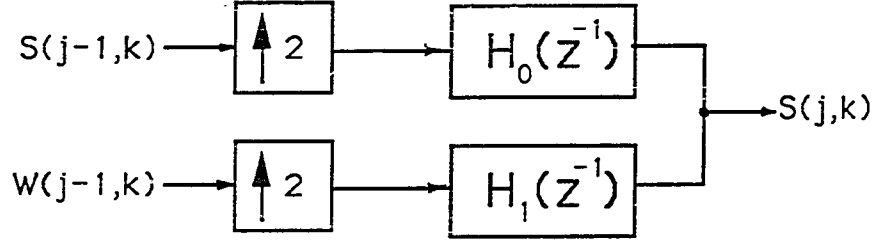


Figure 5.4: The One Level Synthesizer

where  $h_0$  and  $h_1$  are our familiar Conjugate Mirror Filters. Taking the  $z$ -transforms on both sides we get

$$C_0(z) = \frac{1}{2}[X(z)H_0(z^M) + X(-z)H_0((-z)^M)] \quad (5.26)$$

$$C_1(z) = \frac{1}{2}[X(z)H_1(z^M) + X(-z)H_1((-z)^M)] \quad (5.27)$$

Fig. 5.5 shows the block diagrams corresponding to these equations. From the  $z$ -transforms we can see that the equations can be implemented in a lattice using a modified analyzer.

### Complexity of the Modified Analyzer

For an input of  $K$  samples, the number of multiplies and adds for the modified one level analyzer can be shown to be

$$M = (N + 2) \left\lceil \frac{K}{2} \right\rceil + \frac{MN^2}{4} \quad (5.28)$$

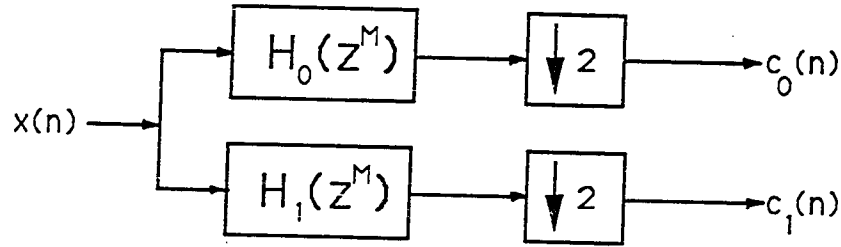


Figure 5.5: Modified One Level Analyzer

$$A = N \left\lceil \frac{K}{2} \right\rceil + \frac{MN^2}{4} \quad (5.29)$$

As in the one level analyzer a few more adds can be saved at the cost of coding complexity which would lower the speed of a program in MATLAB due to its interpretive nature. Overall the complexity per input sample is approximately  $N + 1$  as in the case of the one level analyzer.

### 5.2.5 Modified one level synthesizer

We have no need to implement the modified one level synthesizer. Yet for the sake of completeness we give the equations and the structure for the modified one level synthesizer. Here we are interested in implementing the following equation

$$x(n) = \sum_k h_0(k)c_0(Ml - 2k) + \sum_k h_1(k)c_1(Ml - 2k) \quad (5.30)$$

This can also be implemented as a lattice structure.

This structure is merely the transpose of the corresponding structure for the modified analyzer. It is interesting to note that the synthesizer leads back to the original sequence if and only if  $M$  is odd.

### 5.2.6 Expansion at any scaling or wavelet level

Let us assume that we have to compute the values of a function  $f(t)$  at  $2^{-i}n$  given  $S(j, k, f)$ . Then,

$$f(2^{-i}n) = \sum_k S(j, k, f) 2^{j/2} \phi(2^{j-i}n - k) \quad (5.31)$$

Three cases may be identified depending on whether  $i$  is greater than, equal to or less than  $j$ .

When  $i = j$ , the required samples of  $f(t)$  are obtained by convolving  $S(j, k, f)$  with  $\phi(n)$ , the samples of  $\phi$  at the integers.

When  $i < j$ , the required samples are obtained by convolving the samples of  $\phi$  at the integers with  $S(j, k, f)$  and decimating by the factor  $2^{j-i}$ . An efficient way to implement this would be as a polyphase structure.

When  $i > j$ , the required samples are obtained by first upsampling  $S(j, k, f)$  by a factor of  $2^{i-j}$  and then convolving it with  $\phi(n)$ . This can also be implemented as a polyphase structure.

An alternative method to obtain  $f(t)$  would be to go through the synthesizer enough times until the samples of the function  $f(t)$  by its scaling coefficients at a sufficiently fine level.

## 5.3 Computation of the DWT

We have already seen the basic building blocks in the computation of a DWT and inverse DWT.

1. Projection onto a scaling level
2. The one-level analyzer



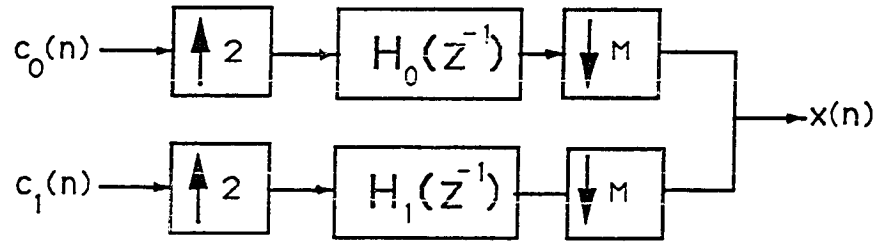


Figure 5.6: Modified One Level Synthesizer

### 3. The one level synthesizer

### 4. Expansion at a scaling or wavelet level

The most efficient way to compute the DWT is find the projection of the function onto  $V_{j_1+1}$ , where  $j_1$  is the finest scale of interest, and then use the analyzer recursively on the scaling coefficients at coarser levels to find the wavelet coefficients at the coarsest level of interest.

Computationally the cost is split between the initial projection onto the finest level, and the analysis. As mentioned earlier we will always assume that the cost of projection is zero. The analyzer and synthesizer can be used to go down and up scaling levels.

Assuming the analyzer is used  $j_1 - j_0 + 1$  times, and that the number of scaling coefficients at the scale  $j_1 + 1$  is  $K$  we compute the number of additions and multiplications as follows. Let  $K_i$  be the number output scaling coefficients after  $i$  analyzer

runs. Then we have

$$K_i = \left\lceil \frac{K_{(i-1)} + N - 1}{2} \right\rceil \quad (5.32)$$

And the number of additions and multiplications in the  $i$ th run are given respectively by

$$\frac{N}{2}K_i < A_i = N \left\lceil \frac{K_i + 1}{2} \right\rceil + \frac{N^2 - 2N}{4} < \frac{N}{2}(K_i + N) \quad (5.33)$$

$$\left(\frac{N}{2} + 1\right)K_i < M_i = (N + 2) \left\lceil \frac{K_i + 1}{2} \right\rceil + \frac{N^2 - 2N}{4} < \left(\frac{N}{2} + 1\right)(K_i + N) \quad (5.34)$$

Solving this for the total number of additions and multiplications we obtain the following inequalities

$$\left(\frac{N}{2} + 1\right)(2(K - N + 1)(2^n - 1)/2^n + n(N - 1)) < M < \left(\frac{N}{2} + 1\right)(2(K - N - 1)(2^n - 1)/2^n + n(2N + 1)) \quad (5.35)$$

and

$$\frac{N}{2}(2(K - N + 1)(2^n - 1)/2^n + n(N - 1)) < A < \frac{N}{2}(2(K - N - 1)(2^n - 1)/2^n + n(2N + 1)) \quad (5.36)$$

As a rough estimate the number of additions is nearly  $NK$  and the number of multiplications roughly  $(N + 2)K$ . Thus the entire DWT computation takes almost as much time as convolving the input with the filter  $h_0$ !! This is property of CQFs that makes wavelet bases analysis of signals so attractive computationally. One can reasonably expect that almost all major applications of wavelets will be centered around the DWT wavelets or more generally CQFs. In some sense wavelets have facilitated looking at a class of CQFs that have excellent properties when the signals being studied behave like polynomials locally.

Since the computation of the DSWT, DTWT and the CWT with respect to arbitrary DSWT, DTWT and CWT wavelets depends on the computation of the corresponding transforms with respect to a DWT wavelet describe how to compute them next.

## 5.4 The DTWT with respect to a DWT wavelet

Let  $\psi$  be a DWT wavelet. Let  $\phi$  be the associated scaling function. We wish to compute  $W(u, k, f)$ . Let us assume that scale is sampled at a rate of  $L$  samples per octave. Then our goal is to compute

$$W(j/L, k, f) = \int f(t) 2^{j/2L} \psi(2^{j/L}t - k) dt \quad (5.37)$$

From Eqn. 4.5 we have already noted that,

$$\phi_{j,k} = \sqrt{2} \sum_l h_0(l) \phi_{j+1, 2k+l} \quad (5.38)$$

This equation is true not only for integral values of  $j$  and  $k$ , but also for arbitrary real values. This can be seen directly from the fundamental scaling equation. Indeed we have,

$$\phi_{u,k} = \sqrt{2} \sum_l h_0(l) \phi_{u+1, 2k+l} \quad (5.39)$$

and

$$\psi_{u,k} = \sqrt{2} \sum_k h_1(k) \phi_{u+1, 2\tau+k} \quad (5.40)$$

This means that there is no preferred origin in scale. The vector spaces  $\{V_j\}$  in the multiscale analysis framework could have been replaced by  $\{V_{j+u}\}$  for arbitrary  $u$ , where  $V_{j+u}$  is understood to be the appropriately interpolated vector space.

Now by taking the inner product of  $f(t)$  with both sides of Eqn. 5.39 and Eqn. 5.40 we get

$$S(u, k, f) = \sum_l h_0(l) S(u+1, 2k+l, f) \quad (5.41)$$

and

$$W(u, k, f) = \sum_l h_1(l) S(u+1, 2k+l, f) \quad (5.42)$$

Thus once the Discrete Time Scaling Transform for scale  $u$  is known the DTST and DTWT for all scales,  $(\{u-j \mid j \in \mathbb{N}\})$  can be obtained. Indeed, the DWT with respect to  $\psi$  is merely the samples of the DTWT at integers in scale and we have already seen how to compute it in the previous section.

Now let us assume that we are given the DTST at the  $L$  finest scales of interest, which are separated by  $1/L$  octave each respectively. Then by going through the one level analyzer of the DWT for each of these DTSTs we have effectively computed the DTWT. Indeed we have the following equations

$$S(\frac{j}{L}, k, f) = \sqrt{2} \sum_l h_0(l) S(\frac{j+L}{L}, 2k+l, f) \quad (5.43)$$

$$W(\frac{j}{L}, k, f) = \sqrt{2} \sum_l h_1(l) S(\frac{j+L}{L}, 2k+l, f) \quad (5.44)$$

As for the computational complexity, note that one has to compute the projection of  $f$  onto the  $L$  coarsest scales. The rest is the run through the DWT algorithm  $L$  times. So approximately the computation is  $L$  times the complexity of the *DWT*.

## 5.5 The DSWT with respect to a DWT wavelet

Here we wish to compute  $W(j, \tau, f)$ . Assuming we sample time at the rate of  $M$  Hz, we have to compute

$$W(j, k/M, f) = \int f(t) 2^{j/2} \psi(2^j t - k/M) dt \quad (5.45)$$

for integers  $j$  and  $k$ . From the fundamental scaling equation we have

$$\phi_{j,\tau} = \sqrt{2} \sum_k h_0(k) \phi_{j+1, 2\tau+k} \quad (5.46)$$

and

$$\psi_{j,\tau} = \sqrt{2} \sum_k h_1(k) \phi_{j+1, 2\tau+k} \quad (5.47)$$

Now by taking the inner product of  $f(t)$  with both sides of Eqn. 5.46 and Eqn. 5.47 we get

$$S(j, \tau, f) = \sqrt{2} \sum_l h_0(l) S(j+1, 2\tau+l, f) \quad (5.48)$$

and

$$W(j, \tau, f) = \sqrt{2} \sum_l h_1(l) S(j+1, 2\tau+l, f) \quad (5.49)$$

Thus once the Discrete Scale Scaling Transform for scale for the finest scale  $j_1$  is known the DTST and DTWT for all scales,  $(\{j_1 - j \mid j \in \mathbf{N}\})$  can be obtained. Letting  $\tau = k/M$ , we obtain the following two equations

$$S(j, k/M, f) = \sqrt{2} \sum_l h_0(l) S(j+1, 2k/M + l, f) \quad (5.50)$$

and

$$W(j, k/M, f) = \sqrt{2} \sum_l h_1(l) S(j+1, 2k/M + l, f) \quad (5.51)$$

These equations can be implemented by the modified analyzer discussed earlier in this chapter.

The computational complexity of this algorithm is divided as always between the initial projection onto the DSWT at the finest scale, and the run through the modified analyzer. Typically we assume that  $f(2^{-j_1-1}[k + m_1])$  is given, where  $j_1 + 1$  is the finest scale of interest. But now we need to find  $S(j_1 + 1, 2^{-j_1-1}k/M, f)$ . If we can sample at  $M$  times the usual rate then the samples may themselves be taken as the corresponding projection. Else some form of interpolation must be used to obtain the necessary DSWT at the finest level. Approximately the number of additions and multiplications increases by a factor of  $M$  over the computation of the DWT because of the fact that the modified analyzer is being used and hence the complexity is about  $M$  times the complexity of the DWT as should be expected.

## 5.6 The CWT with respect to a DWT wavelet

We wish to compute  $W(u, \tau, f)$ . The CWT is basically the combination of the DTWT and the DSWT given the underlying wavelet is a DWT wavelet. Let scale be sampled at the rate of  $L$  samples per octave, and time be sampled at the rate of  $M$  samples per octave. Hence for all integers  $j$  and  $k$  we want

$$W(j/L, k/M, f) = \int f(t) 2^{j/2L} \psi(2^{j/L}t - k/M) dt \quad (5.52)$$

for integers  $j$  and  $k$ . From the fundamental scaling equation we have

$$\phi_{u,\tau} = \sqrt{2} \sum_k h_0(k) \phi_{u+1,2\tau+k} \quad (5.53)$$

and

$$\psi_{u,\tau} = \sqrt{2} \sum_k h_1(k) \phi_{u+1,2\tau+k} \quad (5.54)$$

Now by taking the inner product of  $f(t)$  with both sides of Eqn. 5.53 and Eqn. 5.54 we get

$$S(u, \tau, f) = \sqrt{2} \sum_l h_0(l) S(u+1, 2\tau+l, f) \quad (5.55)$$

and

$$W(u, \tau, \psi, f) = \sqrt{2} \sum_l h_1(l) S(u+1, 2\tau+l, f) \quad (5.56)$$

Letting  $u = j/L$  and  $\tau = k/M$ , we have

$$S(j/L, k/M, f) = \sqrt{2} \sum_l h_0(l) S(j/L+1, 2k/M+l, f) \quad (5.57)$$

and

$$W(j/L, k/M, f) = \sqrt{2} \sum_l h_1(l) S(j/L+1, 2k/M+l, f) \quad (5.58)$$

Hence once the CST at the finest scale is known, one can find the CWT and CST at all scales coarser by an integral multiple of an octave exactly as in the DSWT by using the modified analyzer. If one starts with the CST at  $L$  fine scales each separated by  $1/L$ th an octave, then the entire CWT can be computed.

As for the computational complexity, approximately the number of additions and multiplications increases by a factor of  $LM$  over the computation of the DWT because of the fact that the modified analyzer is being used  $L$  times and the complexity of the modified analyzer is  $M$  times the complexity of the one level analyzer.

## 5.7 Computation of the CWT

If scale is assumed to be sampled at  $L$  samples per octave  $\tau$  at every integral multiple of  $1/M$ , then for integers  $j$  and  $k$ , and for a given function  $f$ , and CWT wavelet  $w$ ,

we have to compute

$$W(j/L, k/M, f, w) = \int dt f(t) 2^{j/2L} w(2^{j/L}t - k/M) \quad (5.59)$$

It is clear from the equation that direct computation using numerical integration is time-consuming. Besides errors are introduced in the transform not only by the numerical integration process, but also by the interpolation of  $f(t)$  from its samples, the scaling of the  $w(t)$  etc. We attempt to solve these by using the CWT with respect to an DWT wavelet, preferably a Daubechies wavelet of small support,  $\psi$ , as an auxiliary computation.

Since  $\psi_{j,k}$  forms an orthonormal basis we can expand both  $f$  and  $w$  in terms of  $\psi_{j,k}$ .

$$f(t) = \sum_{j,k} W(j, k, f) L(j, k) \psi \quad (5.60)$$

and

$$w(t) = \sum_{j,k} W(j, k, w) L(j, k) \psi \quad (5.61)$$

The CWT is given by

$$W(u, \tau, f) = \langle f, L(u, \tau) \psi \rangle \quad (5.62)$$

Substituting for  $f$  and  $w$  from Eqn. 5.60 and Eqn. 5.61 we have

$$W(u, \tau, f, w) = \sum_{j,k,j',k'} W(j', k', f) W(j, k, w) \langle L(j', k') \psi, L(u, \tau) L(j, k) \psi \rangle \quad (5.63)$$

But from Eqn. 3.13 we get

$$\langle L(j', k') \psi, L(u, \tau) L(j, k) \psi \rangle = \langle \psi, U((j', k')^{-1}) U(u, \tau) U(j, k) \psi \rangle \quad (5.64)$$

which on simplifying becomes

$$\langle L(j', k') \psi, L(u, \tau) L(j, k) \psi \rangle = \langle \psi, U(u + j - j', -2^{u+j-j'} k' + 2^j \tau + k) \psi \rangle \quad (5.65)$$

Hence the CWT of  $f$  with respect to  $w$  can be written in the form,

$$W(u, \tau, f, w) = \sum_{j,k,j',k'} W(j', k', \psi, f) W(j, k, w) W(u + j - j', -2^{u+j-j'} k' + 2^j \tau + k, \psi, \psi) \quad (5.66)$$

This equation gives a method for computing the DWT efficiently. The actual computation of this equation involves six loops, two for running through the values of  $u$  and  $\tau$  and four for the four indices in the summation.

Notice that  $W(u, \tau, \psi, \psi)$  can be precomputed, since it depends only on the auxiliary DWT wavelet  $\psi$  used in the computation.

Now consider the case that the wavelet  $w$  is known. Then  $W(j, k, w, \psi)$  can also be precomputed. Besides the summation over  $j$  and  $k$  in Eqn. 5.66 can also be precomputed. Thus the computation of the CWT of  $f$  with respect to  $w$ , reduces to,

1. Compute the DWT of  $f(t)$
2. Do the summation over  $j'$  and  $k'$  in Eqn. 5.66 which is similar to doing one 2 dimensional convolution.

Now assume that one has to compute the CWT of  $f$  with respect to a number of wavelets, say  $w_l(t)$ . Such an instance occurs in the detection of multiple time-scale perturbed signals as discussed in the next chapter. Then, it is more efficient to compute the summation over  $j'$  and  $k'$ , initially than over  $j$  and  $k$ . Then the order of computation becomes,

1. Compute the DWT of  $f(t)$
2. Do the summation over the primed variables in Eqn. 5.66
3. Do the summation over  $j$  and  $k$  for all the wavelets  $w_l$ .

Assuming there are  $M - 1$  wavelets with respect to all of which the CWT is taken, the entire computation is approximately equivalent to  $M$  2 dimensional convolutions and a DWT computation. The alternative ordering is equivalent to approximately  $2(M - 1)$  2 dimensional convolutions. There is another way to interpret this sequence of operations,

1. Compute the CWT of  $f(t)$



2. Discretely convolve the CWT with the DWT of  $w_l$  for each  $l$  over the affine group

In all these computations, since the associated CWTs are available only as uniformly spaced samples, interpolation is necessary to get the exact value of  $W(u + j - j', -2^{u+j-j'}k' + 2^j\tau + k, \psi, \psi)$ , and this also introduces errors in the computation.

### 5.8 Computation of the DTWT

Again with scale sampled at  $L$  scales per octave we have to find for arbitrary  $f$ , for a given DTWT wavelet  $w$ , and for all integers  $j$  and  $k$ ,

$$W(u, k, f, w) = \sum_{j, l, j', l'} W(j', l', f) W(j, l, w) W(u + j - j', -2^{u+j-j'}l' + 2^j k + l, \psi, \psi) \quad (5.67)$$

This is similar to the equation for the CWT.

### 5.9 Computation of the DSWT

Once again we can show that the computation of the DSWT involves the computation of the following equation

$$W(j, \tau, f) = \sum_{l, k, l', k'} W(l', k', f) W(l, k, w) W(j + l - l', -2^{j+l-l'}k' + 2^l\tau + k, \psi, \psi) \quad (5.68)$$

This equation is similar to the implementation of the CWT and in fact the same program may be used.

## Chapter 6

### Detection of Time-Scale Perturbed Signals

#### 6.1 Introduction

This chapter addresses the problem of detection of time-scale perturbed signals. As an example consider the radar detection problem. In the detection of moving targets, the received radar signal is a scaled and translated version of the transmitted signal. Assuming that a moving target exists, the received signal results from a time-scale perturbation of the transmitted signal. The wavelet transforms give the necessary framework for the study of this and similar problems where scale is an important parameter. For narrowband signals like radar, scaling is approximated by shifts in the center (carrier) frequency of the transmitted signal. Hence it is possible to apply conventional time-frequency matched filtering to detect and estimate range and closings from radar echoes. However, for general wideband time-scale perturbed signals, scaling effects cannot be approximated by a frequency shift. The time-scale properties of the wavelet representation makes it a natural candidate for the analysis of such signals [22, 21]. This chapter uses the wavelet transforms to develop detectors for time-scale perturbed wideband signals. No performance measures have been obtained. Nor have any comparisons been made. The aim is to show that the wavelet transforms introduced earlier do indeed give a set of tools in order to do time-scale analysis. As an aside it must be mentioned that that the time variable could be replaced by space or any other variable.

## 6.2 The Problem

The goal is to detect multiple wideband signals that are time-scale perturbed and submerged in additive White Gaussian noise. The signals are assumed to reside in a finite dimensional subspace spanned by a finite set generated by a DWT wavelet,  $\psi$ , by certain translations and dilations.

Given the lack of translation invariance of the DWT, in order to say that a signal *resides* in a scale, a signal and its time translates ought to be distinguished. Consider a signal  $f(t) = \psi(t)$ . Then the signal is trivially one dimensional with respect to the DWT wavelet  $\psi$ . Now consider the signal  $f(t) = \psi(t - T)$ . It is quite possible that  $f(t)$  cannot be represented exactly (in the basis  $\psi_{j,k}$ ) unless all scales  $j$  are included, as can be expected from the lack of translation invariance of the DWT. Thus even though  $f(t)$  is one dimensional with respect to  $\psi$ ,  $f(t - T)$  is infinite dimensional with respect to the same wavelet  $\psi$ . This explains why general time-scale analysis cannot be done just with the DWT, even though the DWT is a complete and efficient representation and can be computed with  $O(N)$  complexity where  $N$  is the length of the input. Hence one is lead to consider finite dimensional approximations to the signal. Clearly, depending on the nature of the signals being studied, one of the many DWT wavelets may be used as the underlying DWT wavelet. The smoother the nature of the signals, one of the smooth DWT wavelets could be used.

Since there is no reason to expect the signal studied to be an exact linear combination of  $\psi_{j,k}$ , one has to find a finite dimensional *approximation* to it. The natural measure of approximation error is the  $L^2$  or energy norm. The approximation should not only try to minimize the squared error, but also try to make the dimension of the signal as small as possible. This would reduce the arithmetic complexity of the CWT which has to be computed to solve the detection problem. This makes the implicit assumption that the classes of signals considered (upto a translation) can be efficiently represented by DWT wavelets. A number of experiments with synthetic and natural (underwater transient) signals show that nearly 95% of all the energy

is concentrated in about three consecutive scales when the Daubechies wavelets for moderate  $N$  are used. That is, most of these signals are confined to a bandwidth spanning approximately three octaves.

In summary, given a signal two choices have to be made. Firstly, the DWT wavelet  $\psi$  to be used in the analysis. Secondly, the exact translation of the signal that minimizes the dimensionality of the signal for a given approximation error. The two are inter-related and we have no simple technique to make these choices. But once the DWT wavelet is fixed, the appropriate translation can be chosen. Let most of the signal energy be concentrated around some level  $J$ . Then if the signal is translated by  $2^{-J}$ , the DWT approximately shifts by 1 unit in the time index at that level. So by empirically trying out a number of translations by fractions of  $2^{-J}$ , a reasonable translation of the signal that simultaneously minimizes the energy and dimensionality can be found.

In summary, a signal being detected is of the form

$$w(t) = \sum_I a_I \psi_I(t) \quad (6.1)$$

for some DWT wavelet  $\psi$ , and where  $I$  is a finite set composed of elements of the form  $(j, k)$ ,  $j$  and  $k$  integral.

Let  $\{w_l(t) | l = 1, 2, \dots, M-1\}$ , be the set of all possible received signals. The received signal is then assumed to be of the form,

$$r(t) = \sum_{l=1}^{M-1} b_l A_l 2^{u_l/2} w_l(2^{u_l} t - \tau_l) + n(t) \quad (6.2)$$

where  $u_l, \tau_l$ , and  $A_l$  are defined to be the scale, time and amplitude perturbations respectively of the signal  $w_l$ , and  $b_l$  is a Boolean variable that is 0 or 1, depending on whether signal  $w_l$  is present or not. By assuming that at most one of the possible signals is present in the received signal, at most one of the  $b_l$ s is 1. This greatly simplifies the number of hypotheses being tested.

Given the received signal, the problem is to find the  $b_l$ s, and, given  $b_l$  is 1 for a particular  $l$ , to estimate the values of  $u_l$  and  $\tau_l$ , which are respectively, the scale and time perturbations.

### 6.3 Detection and Estimation

The detection problem can be classified as an  $M$ -ary hypothesis testing problem with unknown parameters. There is the null hypothesis and the  $M - 1$  hypotheses, one each corresponding to the presence of each of the signals. The unknown parameters are the scale, time and gain parameters. The unknown parameters are assumed to be non-random.

Let us recall a few classical results. For a known signal with an unknown translation signal  $w(t - t_0)$  that is submerged in additive white Gaussian noise, the optimum detector consists of a likelihood ratio test, that compares the output of a matched filter or correlation receiver to a pre-determined threshold. The threshold is set in order to fix a certain probability of error or detection etc. That is,

$$r(t) = Aw(t - t_0) + n(t) \quad (6.3)$$

$$\lambda(t) = \int r(\tau)w(t - \tau)d\tau \quad (6.4)$$

If  $\lambda(T)$  exceeds  $\alpha$ , the threshold, the signal is detected at  $T$ . Fig. 6.1 shows the detector. Moreover the output of the matched filter forms a sufficient statistic for the detection problem. In the  $M$ -ary hypothesis testing, the sufficient statistic is given by the vector of outputs from the matched filters corresponding to each of the signal  $w_l$ . measure required (using Bayes' criteria with suitable costs), decisions regions are formed. Depending on where the sufficient statistic vector falls, one or the other of the hypotheses is selected.

Notice that the gain parameter is not important in the above detection in the sense that it merely alters the threshold by a constant. Hence in the detection of a single signal subject to time-scale perturbations, once the scale and the time parameters

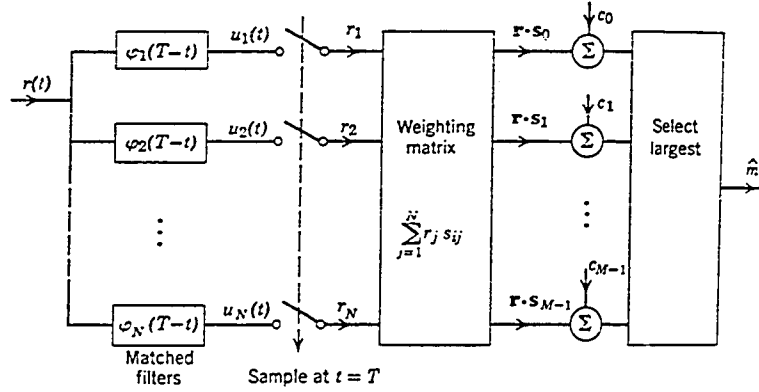


Figure 6.1: Matched filter Receiver

are known, the problem is trivial. Since the scale and time perturbation parameters are non-random, they cannot be estimated by a Bayesian technique. Therefore, a bank of matched filters for all possible values of scale perturbations must be used. This is exactly analogous to the use of the ambiguity function in the radar detection problem. For a single signal  $w(t)$ , and for a family of perturbations we must compute

$$\lambda(u, \tau) = \int r(t) 2^{u/2} s(2^u t - \tau) dt \quad (6.5)$$

Thus  $\lambda$  is a function on the affine group  $G_0$ . Moreover,  $\lambda$  is the CWT of the received signal with respect to the wavelet  $w(t)$ . The maximum likelihood estimate of  $u$  and  $\tau$  are given by the point in  $G_0$  where  $\lambda$  attains a maximum. Thus assuming a single signal  $s(t)$  is present, the maximum likelihood estimate of  $(u, \tau)$  is given by the maximum of the outputs of the matched filters. Furthermore, the maximum value gives the output of the matched filter assuming the signal transmitted had the estimated time-scale perturbations.

When multiple signals are present, the extension is obvious. Again a bank of matched filters is used, one for each signal. For each bank, the maximum of the output is obtained. The global maximum over these maxima gives the output of the matched filter corresponding to the maximum likelihood estimate of  $u_l$  and  $\tau_l$  of the signal.

### 6.3.1 $\tau, u$ known

For signal  $l$  both,  $u_l$  and  $\tau_l$  are known. The only unknowns are thus  $A_l$  and  $b_l$ . This is the M-ary hypothesis testing problem. In fact by redefining the signals, both  $u_l$  and  $\tau_l$  could be assumed to be zero. The optimum detector computes the matched filter output corresponding to each of the possible signals to obtain the vector of sufficient statistics  $\lambda_l$ . Let

$$\max_l \lambda_l = \Lambda \quad (6.6)$$

and the maximum be attained for  $l = L$ . Let the threshold be  $\alpha$ . Then,

$$\Lambda \begin{cases} > \alpha, \text{signal } L \text{ detected} \\ < \alpha, \text{NULL hypothesis} \end{cases} \quad (6.7)$$

Thus given the received signal  $r(t)$ , for signal  $w_l$ , one must either match it to  $L(u_l, \tau_l)w_l(t)$  or match  $L(-u, 2^{-u}\tau)r(t)$  to  $w_l(t)$ . It is more efficient to do the former, since the scaling of each signal becomes a precomputable operation. If the number of signals,  $M - 1$ , is large, then it may be more efficient to do the matched filtering in the DWT domain. This is because corresponding to every signal  $w_l(t)$ , there would be a window in the DWT domain,  $I_l$ , such that the DWT coefficients  $W(I, w_l)$  can be considered zero in  $I_l^c$ , the complement of  $I_l$ . Thus once the DWT of  $r(t)$  is obtained, each  $\lambda_l$  is obtained as,

$$\lambda_l = \sum_{I_l} W(I_l, w_l)W(I_l, r) \quad (6.8)$$

The cardinality of  $I_l$  is expected to be much less than the number of samples of the signal  $r(t)$ . Thus this method is superior to direct implementation of matched filtering in the time domain after scaling.

In summary, given the  $r(t)$ ,

1. Compute  $W(j, k, r)$
2. Compute  $\lambda_l$  as in Eqn. 6.8
3. Find maximum of  $\lambda_l$ , compare to threshold  $\alpha$  and make decision

For a single signal, it may be more efficient to directly implement the matching operation in the time domain.

### 6.3.2 $\tau$ unknown, $u$ known

For signal  $l$ ,  $\tau$  is unknown. From  $r(t)$ ,  $\lambda_l(\tau)$  is obtained by matched filtering,

$$\lambda_l(\tau) = \int r(t) 2^{u/2} w_l(2^u t - \tau) \quad (6.9)$$

Let,

$$\max_{l, \tau} \lambda_l(\tau) = \Lambda \quad (6.10)$$

and the maximum be attained for  $l = L$  and  $\tau = T$ . Let the threshold be  $\alpha$ . Then,

$$\Lambda \begin{cases} > \alpha, \text{signal } L \text{ detected} \\ < \alpha, \text{NULL hypothesis} \end{cases} \quad (6.11)$$

The maximum likelihood estimate for the time parameter is given by

$$\hat{\tau} = T$$

Moreover, the estimator for the physically relevant delay is

$$\hat{\alpha} = 2^{-uL} T \quad (6.12)$$



### 6.3.3 $\alpha$ known, $u$ unknown

The time parameter  $\tau$  is local and has no physical significance like the global time parameter  $\alpha$ . So it is more reasonable to assume that  $\alpha$  is known rather than that  $\tau$  is known. In other words we know exactly when the received signal might be the signal being detected. From  $r(t)$  we obtain,

$$\lambda_l(u) = \int r(t) 2^{u/2} w_l(2^u(t - \alpha)) \quad (6.13)$$

Let

$$\max_{l,u} \lambda_l(u) = \Lambda \quad (6.14)$$

and let the maximum be attained for  $l = L$  and  $u = U$ . Then,

$$\Lambda \begin{cases} > \alpha, \text{signal } L \text{ detected} \\ < \alpha, \text{NULL hypothesis} \end{cases} \quad (6.15)$$

The maximum likelihood estimate for the scale parameter  $u$  is given by

$$\hat{u} = U$$

### 6.3.4 $\tau, u$ unknown

This is the most general setting where none of the parameters are known. From the received signal  $r(t)$  we compute,

$$\lambda_l(u, \tau) = \int r(t) 2^{u/2} w_l(2^u t - \tau) dt \quad (6.16)$$

which is essentially the CWT of the received signal with respect to the wavelet  $w_l$  for each  $l$ . Let

$$\max_{l,u,\tau} \lambda_l(u, \tau) = \Lambda \quad (6.17)$$

and let the maximum be attained for  $l = L$  and  $u = U$  and  $\tau = T$ . Then

$$\Lambda \begin{cases} > \alpha, \text{signal } L \text{ detected} \\ < \alpha, \text{NULL hypothesis} \end{cases} \quad (6.18)$$

The maximum likelihood estimate for the scale parameter  $u$ , the time parameter  $\tau$  and the delay  $\alpha$  are given respectively by

$$\hat{u} = U$$

$$\hat{\tau} = T$$

$$\hat{\alpha} = 2^{-U}T$$

The computation of the CWT can be carried out efficiently using the algorithms developed in the previous chapter.

The efficiency of the estimate is quantified by the covariance matrix of the random field  $(u, \tau)$ , whose asymptotic value is given by the Cramér-Rao bound. One can set up a minimization problem to minimize this covariance over all possible CWT wavelets. The functions that minimize this covariance are the Klauder wavelets [22]. That is, they give the maximum concentration in time-scale.

## 6.4 Conclusion

In summary, we have developed a framework for the scale-time analysis of signals, given efficient algorithms for the computation of the various wavelet transforms, and finally given an outline as to how the wavelet framework may be used in the detection of time-scale perturbed signals.

## Appendix A

### Program Listing

#### A.1 The Programs

##### A.1.1 $h_0$ the scaling filter - daubcmf.m

```
function h_0 = daubcmf(N2)
%Function to compute the Daubechies CMF. The program implements
%the CMF as described in the paper, "Orthonormal bases of compactly supported wavelets", CPAM, Oct.89. The program returns the
%minimum phase filter h_0 and may be modified to generate other
%filters. The polynomial R in the reference is set to 0.
%INPUT :
% N2 : N2 = N/2, where N is the length of the filter.
%OUTPUT :
% h_0 : Minimal phase Daubechies CMF of length 2*N2.
%by R.G. 10/10/88
p = 1;
q = 1;
h_0 = [1 1];
for j = 1:N2-1,
    a = -a*0.25*(j+N-1)/j;
    h_0 = [0 h_0] + [h_0 0];
    p = [0 -p] + [p 0];
    q = [0 -p] + [p 0];
    q = [0 q 0] + a*p;
end;
q = sort(roots(q));
h_0 = conv(h_0,real(poly(q(1:N-1))));
h_0 = h_0/sum(h_0);
```

##### A.1.2 $\phi$ - phi.m

```
function phi = phi(h_0,J);
%function phi = phi(h_0,J);
%Function to generate samples of phi, the scaling function
```

```

%at the dyadics. The program uses an efficient polyphase
%implementation of the infinite product method.
%INPUT :
% h_0 : The scaling function filter.
% J : an integer.
%OUTPUT :
% phi : Vector of samples of phi at  $2^{-J}k$ .
%by R.G. 10/11/88
if nargin < 2,
    J = 5;
end;
h_0_len = max(size(h_0));
h_0 = 2*h_0;
h0 = h_0(1:2:h_0_len);
h1 = h_0(2:2:h_0_len);
phi = h_0;
m = 3*h_0_len-2;
for i = 2:J,
    phi0 = conv(phi,h0);
    phi1 = conv(phi,h1);
    phi = zeros(1,m);
    phi(1:2:m) = phi0;
    phi(2:2:m) = phi1;
    m = m+m+h_0_len-2;
end;
%Adding zeros to make support [0,N);
phi = [phi zeros(1,h_0_len-1)];

```

### A.1.3 $\psi$ - psi.m

```

function psi = psi(h_0,J);
%function psi = psi(h_0,J);
%Function to compute the wavelet psi corresponding to the
%scaling filter h_0. The infinite product method is used
%with a polyphase implementation of the filters.
%INPUT :
% h_0 : The scaling filter.
% J : An integer.
%OUTPUT :
% psi : Vector of samples of psi at  $2^{-J}k$ 
%by R.G. 10/11/88
if nargin < 2,

```

```

J = 5;
end;
h_0_len = max(size(h_0));
h_0 = 2*h_0;
h0 = h_0(1:2:h_0_len);
h1 = h_0(2:2:h_0_len);
psi = h_0(h_0_len:-1:1);
psi(2:2:h_0_len) = -psi(2:2:h_0_len);
m = 3*h_0_len-2;
for i = 2:J,
psi0 = conv(psi,h0);
psi1 = conv(psi,h1);
psi = zeros(1,m);
psi(1:2:m) = psi0;
psi(2:2:m) = psi1;
m = m+m+h_0_len-2;
end;
%Adding zeros to make the support width N
psi = [psi zeros(1,h_0_len-1)];

```

#### A.1.4 The one level Analyzer - latdec.m

```

function [xl,xh] = latdec(x,k,beta);
%function [xl,xh] = latdec(x,k,beta);
%Function to implement the one level analyzer as a lattice.
%INPUT :
% x : The input scaling coeffs. (input)
% k : The vector of denormalized lattice coeffs.
% beta: The lattice normalization const.
%OUTPUT :
% xl: The scaling coeffs. at next coarser level. (low pass)
% xh: The wavelet coeffs. at next coarser level. (high pass)
%by R.G. 9/27/89. Last modified 11/6/89
k_len = max(size(k));
x_len = max(size(x));
flag = rem(x_len,2);
xl = x(1:2:x_len);
xh = [0 x(2:2:x_len)];
if flag == 0,
xl = [xl 0];
end;
t = xl-k(1)*xh;

```

```

xh = x1*k(1)+xh;
x1 = t;
for i=2:k_len,
t = [x1 0] - [0 k(i)* xh];
xh = [x1*k(i) 0] + [0 xh];
x1 = t;
end;
if nargin < 3,
beta = fbeta(k);
end;
x1 = x1*beta;
xh = xh*beta;
end;

```

#### A.1.5 The one level Synthesizer - latrec.m

```

function x = latrec(xl,xh,k,beta);
%function x = latrec(xl,xh,k,beta);
%Function to implement the one level synthesizer as a
%denormalized lattice.
%INPUT :
% xl : The scaling coeffs (low pass)
% xh : The wavelet coeffs (high pass)
% k: The lattice coeffs. vector
% beta: The lattice normalization const.
%OUTPUT :
% x : The output vector.
% xh : The high freq. output.
%by R.G. 9/27/89. Last modified 11/6/89
k_len = max(size(k));
if max(size(xl)) ~= max(size(xh))
disp('error:both channels must have same length');
end;
t = xl;
x1 = xh;
xh = t;
k = k(k_len:-1:1);
t = x1-k(1)*xh;
xh = x1*k(1)+xh;
x1 = t;
for i=2:k_len,
t = [x1 0] - [0 k(i)* xh];

```

```

    xh = [x1*k(i) 0] + [0 xh];
    x1 = t;
end;
if nargin < 4,
    beta = fbeta(k);
end;
x1 = x1*beta;
xh = xh*beta;
x_len = max(size(x1));
x_len = x_len + x_len;
x = zeros(1,x_len);
x(1:2:x_len)=x1;
x(2:2:x_len)=xh;
end;

```

#### A.1.6 The modified Analyzer - mlatdec.m

```

function [x1,xh] = modlatdec(x,k,beta,M);
%function [x1,xh] = modlatdec(x,k,beta,M);
%Function to implement the modified one level analyzer as
%a lattice.
%INPUT :
% x: The input scaling coeffs. at spacing of 1/M.
% k: The denormalized lattice coeffs.
% beta: The normalization constant.
% M : No. of samples per unit time parameter
%OUTPUT :
% x1: the low freq. output.
% xh: the high freq. output.
%by R.G 1/7/89
k_len = max(size(k));
x_len = max(size(x));
tmp = zeros(1,M);
tmp1 = zeros(1,floor(M/2));
Mflag = rem(M,2);
if (Mflag == 0),
    x = x(1:2:x_len);
    x1 = [x tmp1] - [tmp1 k(1)*x];
    xh = [x*k(1) tmp1] + [tmp1 x];
else
    xflag = rem(x_len,2);
    x1 = x(1:2:x_len);

```

```

xh = [0 x(2:2:x_len)];
if xflag == 0,
xl = [xl 0];
end;
t = [xl tmp1] - [tmp1 k(i)*xh];
xh = [xl*k(1) tmp1] + [tmp1 xh];
xl = t;
end;
for i=2:k_len,
t = [xl tmp] - [tmp k(i)*xh];
xh = [xl*k(i) tmp] + [tmp xh];
xl = t;
end;
if nargin < 3,
beta = fbeta(k);
end;
xl = xl*beta;
xh = xh*beta;
end;

```

#### A.1.7 The modified Synthesizer - mlatrec.m

```

function x = mlatrec(xl,xh,k,beta,M);
%function [xl,xh] = mlatrec(xl,xh,k,beta,M);
%Function to implement the modified one level analyzer as
%a lattice.
%INPUT :
% xl : The scaling coeffs (low pass)
% xh : The wavelet coeffs (high pass)
% k: The denormalized lattice coeffs.
% beta: The normalization constant.
% M : No. of samples per unit time parameter; M is odd;
%OUTPUT :
% x : The output vector.
%by R.G 1/7/89
Mflag = rem(M,2);
if Mflag == 0,
'M must be odd for modified synthesizer';
end;
k_len = max(size(k));
xl_len = max(size(xl));
xh_len = max(size(xh));

```



```

if (xl_len ~= xh_len),
    'both xl and xh must be of equal length';
end;
tmp = zeros(1,M);
tmp0 = floor(M/2);
tmp1 = zeros(1,tmp0);
t = xl;
xl = xh;
xh = t;
k = k(k_len:-1:1);
t = xl-k(1)*xh;
xh = xl*k(1)+xh;
xl = t;
for i=2:k_len,
    t = [xl tmp] - [tmp k(i)* xh];
    xh = [xl*k(i) tmp] + [tmp xh];
    xl = t;
end;
if nargin < 4,
    beta = fbeta(k);
end;
xl = xl*beta;
xh = xh*beta;
keyboard;
tmp2 = max(size(xh));
xh = xh(1:tmp2-tmp0-1);
xl = xl(2+tmp0:tmp2);
x_len = tmp2-tmp0+tmp2-tmp0 -2;
x(1:2:x_len) = xh;
x(2:2:x_len) = xl;
end;

```

#### A.1.8 The DWT - dwt

```

function [d,idx] = dwt(f,n_level,k,beta);
%function [d,idx] = dwt(f,n_level,k,beta);
%Function to compute the DWT of $f$ upto n_level
%levels.
%INPUT :
% f : The samples of the input signal.
% n_level : The number of levels.
% k : The denormalized lattice parameters.

```

```

% beta : The normalization constant.
%OUTPUT :
% d : Vector of DWT of f arranged back to back from
%      the finest level to the coarsest level. The
%      DWT starts at f_time_start - (N-1), where N
%      is the length of the scaling filter.
% idx : is a vector of pointers that point to where the
%      nth level starts in d.
%by R.G. 9/28/89
f_len = max(size(f));
%Projection onto finest scale of interest.
s = [zeros(1,6) f];
if nargin < 4,
    beta = fbeta(k);
end;
%DWT analyzer.
d = [];
index = 1;
for i=1:n_level,
    [s,w] = latdec(s,k,beta);
    idx(i) = itmp;
    index = index + max(size(w));
d = [d w];
end;
end;

```

#### A.1.9 The DSWT with respect to $\psi$ - dswtpsi

```

function [dswt,idxdswt]= dswtpsi(f,k,beta,n_level,M);
%function [dswt,idxdswt]= dswtpsi(f,k,beta,n_level,M);
%Function to compute the DSWT with respect to DWT wavelet psi.
%INPUT :
% f : Samples of the function at M times the rate
%      corresponding to the finest scale i.e  $f(2^{\{-n\_level-1\}}k/M$ 
% k : The denormalized lattice parameters.
% beta : The normalization constant.
% n_level : The index of the finest scale.
% M : The sampling rate of time in Hz.
%OUTPUT :
% dswt : The DSWT with scales arranged back to back starting
%      with the finest scale.
% idxdswt : Index for scale into the DSWT vector.

```

```

%by R.G.
s = [zeros(1,6*M) 1];
M2 = M/2;
index = 1;
dswt = [];
for j = 1:n_level,
[s,w] = modlatdec(s,k,beta,M);
ixdswt(j) = index;
index = index + max((size(w)));
dswt = [dswt w];
end;
end;

```

#### A.1.10 The DTWT with respect to $\psi$ - dtwtpsi

```

function [dtwt,ixdtwt] = dtwtpsi(f,k,beta,n_level,L);
%function [dtwt,ixdtwt] = dtwtpsi(f,k,beta,n_level,L);
%Function to compute the DTWT with respect to DWT wavelet psi.
%INPUT :
% f : Samples of f-Projection onto finest scale.
% k : The denormalized lattice parameters.
% beta : The normalization constant.
% n_level : The number of levels.
% L : The number of subscales per octave.
%OUTPUT :
% dtwt : The vector of DTWT coeffs.
% ixdtwt : The index into dtwt.
%by R.G.
if nargin < 5,
L = 8;
end;
for i = 1:L,
ni = num2str(i-1);
di = ['d',ni];
ixci = ['ixc',ni];
fi = scale(f,(i-1)/L);
eval(['[',di,',',ixci,'] = dwtpsi(fi,k,beta,n_level);']);
end;
%Creation of a single vector c
c = [];
j = 1;
index = 1;

```

```

for l = 1:n_level,
for i = 1:L,
ixc(j) = index;
ni = num2str(i-1);
di = eval(['d',ni]);
ixci = eval(['ixc',ni]);
i1 = ixci(i);
if l < n_level,
i2 = ixci(l+1) - 1;
else
i2 = max(size(di));
end;
c = [c di(i1:i2)];
index = index + i2 - i1 + 1;
j = j+1;
end;
end;

```

#### A.1.11 The CWT with respect to $\psi$ - cwtpsi

```

function [c,ixc] = cwtw(f,k,beta,n_level,L,M);
%function [c,ixc] = cwtw(f,h_0,n_level,L,M);
%Function to compute the CWT with respect to DWT wavelet psi.
%INPUT :
% f : Samples of f at  $2^{\{-n\_level-1\}k}$ 
% k : The denormalized lattice parameters.
% beta : The normalization constant.
% L : No. of samples of scale per octave.
% M : No. of samples of time per second.
%OUTPUT :
% c : Vector of CWT outputs arranged back to back starting
%      with the finest scale.
% ixc : Index into c where the jth scale begins.
%by R.G.
if nargin < 4,
L = 8;
M = 8;
end;
for i = 1:L,
ni = num2str(i-1);
di = ['d',ni];
ixci = ['ixc',ni];

```

```

fi = scale(f,(i-1)/L);
eval(['[',di,',',ixci,'] = dswtpsi(fi,k,beta,n_level,M);']);
end;
%Creation of a single vector c
c = [];
j = 1;
index = 1;
for l = 1:n_level,
for i = 1:L,
ixc(j) = index;
ni = num2str(i-1);
di = eval(['d',ni]);
ixci = eval(['ixc',ni]);
i1 = ixci(1);
if l < n_level,
i2 = ixci(l+1) - 1;
else
i2 = max(size(di));
end;
c = [c di(i1:i2)];
index = index + i2 - i1 + 1;
j = j+1;
end;
end;

```

#### A.1.12 The CWT - cwt.c

```

#define MAX(a,b) ((a > b) ? a : b)
#define MIN(a,b) ((a > b) ? b : a)
#define LOG2 6.93147180559945e-01
#define SUBSCALES 8
#define RATE 8
#define F_ORIGIN -57
#define W_ORIGIN -7
#define PSI_LENGTH 8
/* maximum length of the input arrays */
#define MAXINDEX 100
#include <stdio.h>
#include <math.h>
#include <errno.h>

struct vec {

```

```

double      *data[MAXINDEX];
int          index[MAXINDEX];

};

int
main(argc, argv)
int          argc;
char         *argv[];

{
/* temp declarations */
register     i, j, k;
int          itmp;
double       tmp, tmp1;

char         *filename;
struct vec   f, w, c;
FILE         *fopen(), *fp, *fp1;

/* declarations */
double       f_finest_scale, w_finest_scale, c_finest_scale;
double       f_coarsest_scale, w_coarsest_scale, c_coarsest_scale;
int          f_no_ofscales, w_no_ofscales, c_no_ofscales, nw, nf, nc;
int          c_no_oftimes;
int          w_scale_index, f_scale_index, c_scale_index;
int          w_time_index, f_time_index, c_time_index;
int          c_index;
double       f_time_indexxx;
double       w_time, f_time, c_time;
double       w_scale, f_scale, c_scale;
double       w_scale2;
double       c_time_min, c_time_max;
int          scale_count, time_count;
double       scale_spacing, time_spacing;
double       w_support_width;
double       value, sum;

scale_spacing = 1.0 / SUBSCALES;
time_spacing = 1.0 / RATE;

/* read index for f */

```

```

filename = (char *) "ixf";
if ((fp = fopen(filename, "r")) == (FILE *) NULL) {
    fprintf(stderr, "%s:cannot open file %s\n", argv[0], filename);
    exit(1);
}
for (j = 0; j < MAXINDEX; j++) {
    if (fscanf(fp, "%le", &tmp) == EOF) {
        break;
    }
    f.index[j] = (int) tmp;
    /* printf("%ld\n", f.index[j]); */
}
f_finest_scale = (double) f.index[--j];
f_no_ofscales = --j;
printf("f_no_ofscales = %d\n", f_no_ofscales);
f_coarsest_scale = f_finest_scale - scale_spacing * (f_no_ofscales - 1);
printf("f_coarsest_scale = %le\n", f_coarsest_scale);
fclose(fp);

/* read f */
filename = (char *) "f";
if ((fp = fopen(filename, "r")) == (FILE *) NULL) {
    fprintf(stderr, "%s:cannot open file %s\n", argv[0], filename);
    exit(1);
}
for (j = 0; j < f_no_ofscales; j++) {
    itmp = f.index[j + 1] - f.index[j];
    if ((f.data[j] = (double *) malloc(sizeof(double) * itmp)) == NULL) {
        perror("malloc failed for array f");
        exit(1);
    }
    for (k = 0; k < itmp; k++) {
        if (fscanf(fp, "%le", &tmp) == EOF) {
            break;
        }
        f.data[j][k] = tmp;
        /* printf("%22.16e\n", tmp); */
    }
}
fclose(fp);

/* read w.index */

```

```

filename = (char *) "ixw";
if ((fp = fopen(filename, "r")) == (FILE *) NULL) {
    fprintf(stderr, "%s:cannot open file %s\n", argv[0], filename);
    exit(1);
}
for (j = 0; j < MAXINDEX; j++) {
    if (fscanf(fp, "%le", &tmp) == EOF) {
        break;
    }
    w.index[j] = (int) tmp;
    /* printf("%ld\n", w.index[j]); */
}
w_finetest_scale = (double) w.index[--j];
w_no_ofscales = --j;
w_coarsest_scale = w_finetest_scale - w_no_ofscales + 1;
fclose(fp);

/* read w */
filename = (char *) "w";
if ((fp = fopen(filename, "r")) == (FILE *) NULL) {
    fprintf(stderr, "%s:cannot open file %s\n", argv[0], filename);
    exit(1);
}
for (j = 0; j < w_no_ofscales; j++) {
    itmp = w.index[j + 1] - w.index[j];
    if ((w.data[j] = (double *) malloc(sizeof(double) * itmp)) == NULL) {
        perror("cannot malloc for w");
        exit(1);
    }
    for (k = 0; k < itmp; k++) {
        if (fscanf(fp, "%le", &tmp) == EOF) {
            break;
        }
    }
    w.data[j][k] = tmp;
    /* printf("%22.16e\n", w.data[j][k]); */
}
}
fclose(fp);

/* ranges for c_scale */
c_finetest_scale = f_finetest_scale - w_coarsest_scale;
c_coarsest_scale = f_coarsest_scale - w_finetest_scale;

```



```

c_no_ofscales = SUBSCALES * (c_fineest_scale - c_coarsest_scale) + 1;

fp = fopen("c.mat", "w");
fp1 = fopen("ixc.mat", "w");

c_index = 1;
c_scale_index = -1;
for (c_scale = c_fineest_scale; c_scale >= c_coarsest_scale; c_scale -= scale_spa
c_scale_index++;
nw = w_no_ofscales;
modf((double)(f_no_ofscales - 1) * scale_spacing, &tmp);
nf = (int) tmp;
nf++;
modf((c_scale_index * scale_spacing), &tmp);
nc = (int) tmp;
nc++;
if (nc < nw) {
scale_count = nc;
} else if (nc <= nf) {
scale_count = nw;
} else {
scale_count = nf + nw - nc;
}
w_scale_index = w_no_ofscales;
f_scale_index = c_scale_index + SUBSCALES;
w_scale2 = exp((w_coarsest_scale - 1) * LOG2);

for (i = 0; i < scale_count; i++) {
w_scale_index--;
f_scale_index -= SUBSCALES;
w_scale2 += w_scale2;
/* estimate w_support_width */
itmp = w.index[w_no_ofscales] - w.index[w_no_ofscales - 1] + PSI_LENGTH;
w_support_width = exp(-LOG2 * w_coarsest_scale) * itmp;
c_time_min = -floor(w_support_width);
tmp = f.index[f_scale_index + 1] - f.index[f_scale_index] + F_ORIGIN;
c_time_max = time_spacing * tmp;
c_no_oftimes = (int) (floor((c_time_max - c_time_min) * SUBSCALES));
c.index[c_scale_index] = c_index;
c_index += c_no_oftimes;
if (fprintf(fp1, "%le ", (double) c.index[c_scale_index]) == EOF) {
perror("cannot write ixc");
}
}

```

```

    exit(1);
}
if ((c.data[c_scale_index] = (double *) malloc(sizeof(double) * (c_no_oftimes)))
perror("cannot malloc for c.data");
exit(1);
}
c_time_index = -1;
for (c_time = c_time_min; c_time < c_time_max; c_time += time_spacing) {
    c_time_index++;
    f_time = w_scale2 * c_time + W_ORIGIN;
    time_count = w.index[w_scale_index + 1] - w.index[w_scale_index];
    sum = 0;
    for (w_time_index = 0; w_time_index < time_count; w_time_index++) {
        f_time++;
        f_time_indexx = (f_time) * RATE - F_ORIGIN;
        f_time_index = (int) floor(f_time_indexx);
        if (f_time_index < 0) {
            continue;
        }
        if (f_time_index >= (f.index[f_scale_index + 1] - f.index[f_scale_index] - 1)) {
            break;
        }
        tmp = f.data[f_scale_index][f_time_index];
        tmp1 = f.data[f_scale_index][f_time_index + 1];
        value = tmp + (f_time_indexx - f_time_index) * (tmp1 - tmp) * RATE;
        value = w.data[w_scale_index][w_time_index] * value;
        sum += value;
    }
    c.data[c_scale_index][c_time_index] = sum;
    if (fprintf(fp, "%22.16le ", sum) == EOF) {
        perror("cannot write c");
    }
}
}
}
}
}
}
}

```

## Bibliography

- [1] V. Belevitch. *Classical Network Synthesis*. Holden-Day, San Francisco, CA, 1968.
- [2] G. F. Bordereaux-Bartels. *Time-Frequency Signal Processing Algorithms : Analysis and Synthesis using Wigner Distribution*. PhD thesis, Rice University, Houston, TX, 1983.
- [3] T. A. C. M. Classen and W. F. G. Mecklenbrauker. The Wigner Distribution - A Tool for Time-Frequency Signal Analysis : Part i - Continuous-time Signals. *Philips Journal of Research*, 35(3):217–250, 1980.
- [4] T. A. C. M. Classen and W. F. G. Mecklenbrauker. The Wigner Distribution - A Tool for Time-Frequency Signal Analysis : Part ii - Relationship with other Time-Frequency Signal Transformations. *Philips Journal of Research*, 35(6):372–389, 1980.
- [5] T.A. C. M. Classen and W. F. G. Mecklenbrauker. The Wigner Distribution - A Tool for Time-Frequency Signal Analysis : Part ii - Discrete-time Signals. *Philips Journal of Research*, 35(4/5):276–300, 1980.
- [6] L. Cohen. Time-frequency distributions - a review. *Proceedings of the IEEE*, 77(7):941–981, July 1989.
- [7] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications in Pure and Applied Mathematics*, (41):909–996, October 1988.
- [8] I. Daubechies. The wavelet transform,time-frequency localization and signal analysis, September 1988.

- [9] I. Daubechies and J. C. Lagarias. Two-scale difference equations : Global regularity of solutions.
- [10] D.Gabor. Theory of communication. *Journal of the Institute for Electrical Engineers*, 93:429–439, 1946.
- [11] Z. Doganata, P. P. Vaidyanathan, and T. Q. Nguyen. General Synthesis Procedures for FIR Lossless Transfer Matrices, for Perfect-Reconstruction Multirate Filter Bank Applications. *IEEE Trans. in ASSP*, 36(10):1561–1574, October 1988.
- [12] D. Esteban and C. Galand. Applications of quadrature mirror filters to split band voice coding schemes. In *Proceedings of ICASSP*, 1977.
- [13] B. Friedlander and B. Porat. Detection of transient signals by the Gabor representation. *IEEE Transactions on ASSP*, 37(2), February 1989.
- [14] R. A. Gopinath. Lattice structure for vlsi fixed point implementation cmfs. Technical report, Aware Inc., Cambridge, MA-02138, October 1989.
- [15] A. Grossman and J. Morlet. Decomposition of Hardy functions into square integrable wavelets of constant shape. *SIAM Journal of Mathematical Analysis*, (15):723–736, 1984.
- [16] P. Groupillaud, A. Grossman, and J. Morlet. Cyclo-octave and related transforms in seismic signal analysis. *Geoeexploration*, (23), 1984.
- [17] H.I.Choi and W.J.Williams. Improved time-frequency representation of multi-component signals using exponential distributions. *IEEE Trans. in ASSP*, 37, 1989.
- [18] I.Daubechies and J. C. Lagarias. Two-scale difference equations : Local regularity, infinite products of matrices and fractals.

- [19] D. L. Jones. Study of windows for the Short Time Fourier Transform. Master's thesis, Rice University, Houston, TX, 1986.
- [20] D. L. Jones. *A high resolution data-adaptive time-frequency representation*. PhD thesis, Rice University, Houston, TX, 1987.
- [21] W. M. Lawton. Private communication.
- [22] W. M. Lawton. Applications and implementations of the cwt. Technical report, Aware Inc., Cambridge, Massachussets.
- [23] W. M. Lawton. General theory of wavelets and qmfs. Technical report, Aware Inc., Cambridge, Massachussets, August 1988.
- [24] W. M. Lawton. Tight frames of compactly supported wavelets. *Journal of Mathematical Physics*, 1990. to be published.
- [25] S. Mallat. Multiresolution approximations and wavelets. Technical report, GRASP Lab, Dept. of Computer and Information Science, University of Pennsylvania, September 1987.
- [26] Y. Meyer. Ondelettes, fonctions splines et analyses graduées.
- [27] T. Q. Nguyen and P. P. Vaidhyathan. Maximally Decimated Perfect-Reconstruction FIR Filter Banks with Pairwise Mirror-Image Analysis and Synthesis Frequency Responses. *IEEE Trans. in ASSP*, 36(5):693-706, May 1988.
- [28] P. Auscher. *Thèse de doctorat*. PhD thesis, 1988.
- [29] L. Rabiner and D. Crochiere. *Multirate Digital Signal Processing*. Prentice-Hall, 1983.
- [30] S. Mallat. Review of Multifrequency Channel Decompositions of Images and Wavelet Models. Invited paper to IEEE Trans. on ASSP.

- [31] M.J. Smith and T. P. Barnwell. Exact Reconstruction techniques for tree-structured subband coders. *IEEE Trans. in ASSP*, 34:434–441, 1986.
- [32] P. P. Vaidyanathan. Theory and design of M-channel maximally decimated QMF's having the perfect-reconstruction property. *IEEE Trans. in ASSP*, 35(4):476–492, April 1987.
- [33] M. Vetterli and C. Herley. Wavelets and filter banks: Relationships and new results. April 1990.
- [34] M. J. Vetterli. Multirate Filter Banks. *IEEE Trans. in ASSP*, 35:356–372, March 1987.
- [35] G. Wornell. Karunhen-loève like expansion for  $1/f$  noise processes. *IEEE Trans. in ASSP*.