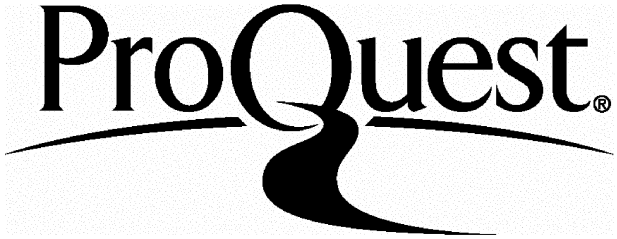
GENERIC DISCRETE WAVELETS

Benjamin La Borde

University College London
England

Submitted for
The Degree of Doctor of Philosophy
The University of London

September 1996



ABSTRACT

The work concentrates on orthogonal wavelets and documents a generic method for designing discrete wavelets which encompasses the well-documented Haar and Daubechies.

Only wavelets up to 6 taps are considered, these being derived from seed equations capable of explicit solution; higher degrees than this require iterative solution.

Much of the work describes the derivation of the generic method, leading to equations which although comprising simple algebra do become complicated, especially for the 6 tap case, which "reduces" to a single quartic equation of great complexity. The seed equations are the usual ones of orthonormality and moment conditions with the highest moment condition replaced by a generic parameterized "lock" condition which introduces a new degree of freedom not found in the Daubechies wavelets.

The 4 and 6 tap wavelets exist as two classes, each encompassing a continuous space of wavelets of their respective lengths, mutable by use of a single controlling parameter that can be exploited to change the wavelet's shape, regularity and support. Applications of compression are examined, using the control parameter as a tuning mechanism to best match given signals. Additionally, application to transient detection is presented, again using the new wavelet's tuning capability to optimize the detection process.

CONTENTS

ERRATA

page 14 1.6.1 2nd paragraph:	change *impossible* to *very difficult*
page 29:	change *the deltas equal 1* to *the integrals of the deltas equal 1*
page 31 3rd paragraph:	change *necessary condition* to *sufficient condition*
page 43 4.1:	change $f(x,c) = 0$ to $\sum_{k=0}^{N-1}(-1)^k f(x_k,c) = 0$
page 43 (4.1):	change *orthonormality* to *normalization*
page 51 first paragraph:	change *none zero* to *non-zero*
page 53 (5.1):	change *orthonormality* to *normalization*
page 54 5.3:	change *predominately* to *predominantly*
page 77 first paragraph:	change *attributable* to *due*

TABLES

GLOSSARY

ϕ, phi	scaling function
ψ, psi	wavelet
affine	property of translation and dilation
AKA	Also Known As
ANN	Artificial Neural Net, AKA Neural Net
APR	Automatic Pattern Recognition
ASSP	IEEE Transactions on Acoustics, Speech and Signal Processing
CFAR	Constant False Alarm Rate
CON	Complete Ortho-Normal
EMTP	ElectroMagnetic Transient Program (proprietary simulation software)
FT	Fourier Transform
IEEE	Institute of Electrical and Electronics Engineers Inc
JPEG	Joint Photographic Experts Group
ISAR	Inverse Synthetic Aperture Radar
MPEG	Moving Pictures Experts Group
NGC	The National Grid Company plc
NN	Neural Net
SAR	Synthetic Aperture Radar
SIAM	Society for Industrial and Applied Mathematics
SNR	Signal to Noise Ratio
SMPTE	Society of Motion Picture and Television Engineers
SP	Signal Processing
SPIE	Society of Photo-optical Instrumentation Engineers, AKA The International Society for Optical Engineering
TFTS	Time-Frequency/Time-Scale
UCL	University College London
WT	Wavelet Transform

ACKNOWLEDGEMENTS

This research was undertaken within the Postgraduate Training Partnership established between Sira Ltd and University College London. Postgraduate Training Partnerships are jointly sponsored by the Department of Trade and Industry and the Engineering and Physical Sciences Research Council. They are aimed at providing research training relevant to a career in industry and at fostering closer links between the science base, industrial research, and industry.

Consequently, supervisors from both UCL and Sira deserve recognition. They are listed here alphabetically by last name so as to offend none.

Dr Mike Cutter, Sira, Kent.

Professor Chris Pitt, University College London.

Dr Paul Radmore, University College London.

1 INTRODUCTION

1.1 Motivation

This thesis documents the PhD of Benjamin La Borde, started in October 1993 in the Department of Electronic and Electrical Engineering, University College London, England. At commencement the title was simply "Wavelets and the wavelet transform" without remit for specific research direction.

The achievement of this PhD is the development of a generic method for designing discrete wavelets which encompass the well-documented Haar and Daubechies wavelets, and brought them together in a relationship previously unexplored using a single parameter. They are termed the *abc* wavelets, the name deriving from the use of a single extra parameter, c, in their representation.

The focus on generic discrete wavelets defined with a single parameter was motivated by the fact that the problem was open and that the literature showed no method for closed-form discrete wavelet design modelled by a single parameter.

1.2 Thesis structure

The thesis comprises 9 chapters. This chapter, chapter 1, examines the literature of parameterized and general wavelets, and outlines historical development. The concept of parameterization is put into context with examples of extant systems.

Chapter 2 is an overview of the continuous wavelet transform. It shows how the inverse is calculated and how it depends on the admissibility condition. The importance of admissibility is explained as a necessary condition also in the derivation of discrete wavelets.

Chapter 3 examines the development of discrete wavelets, focusing on Daubechies' wavelets and the fast transform. Readers unfamiliar with the wavelet transform would benefit from an early reading of chapter 3.

Chapters 4 and 5 document the derivation of the generic 4 tap and 6 tap wavelets respectively.

Chapters 6, 7 and 8 concentrate on applications. Each chapter describes a separate aspect of the 6 tap abc wavelets:

Chapter 6 looks at the derivation and implementation of a specific 6 tap wavelet, useful in compression, being similar in quality to Daubechies 6, but offering a potential speed advantage in implementation.

Chapter 7 considers wavelet selection from the six tap class to optimize compression efficiency. Here the parameter is used to produce a set of wavelets from which the most efficient can be selected.

Chapter 8 illustrates application of the abc class to power system transient detection. Further literature background is presented, and a case study considers the selection of best wavelets for detecting different transient types occurring in the switching state of high voltage relays.

Chapter 9 considers future work and closes the discussion with an overview and conclusion.

1.3 Background

The use of wavelets has won recent favour in signal analysis due to the advent of digital computing. The father of the subject was Alfred Haar [27], a Hungarian mathematician who devised the first wavelet in 1910. Like the grandfather clock, who's owner died taking the key with him, wavelets fell into disuse with Alfred. The man died.

Nothing happened until the 1980's when the French resurrected the subject for use in seismic analysis. In 1984 Jean Morlet [44], a geophysicist, used complex continuous wavelets to provide variable resolution in both spatial and frequency domains. Compared with the conventional Fourier transform (FT), the wavelet provides location information as well as frequency content, and is more useful in analysing signals for which location information is important. The wavelet transform (WT) also has the advantage of automatically adjusting its window size to the resolution of the examining basis. Large scale (low frequency) uses a large window; small scale (high frequency) uses a small window. The concept is borrowed from quantum mechanics which prohibits simultaneous high precision in time and frequency (Heisenberg's principle). This is called a constant Q feature, $d\omega\, d\mu \geq c$, for some positive constant $c \sim 1$ (Kaiser [34]).

The symbol μ is used as the signal space parameter to represent time t or distance x, in a generic fashion without reference specifically to either dimension. Similarly, ω can be temporal or spatial frequency respectively.

The wavelet constant Q property is shown graphically in figure 1.1 compared with Fourier windowed sampling, figure 1.2. Mallat [40] shows that wavelets have good resolution in both time and frequency. It is the combined properties of constant Q and localization that often leads to the choice of the wavelet transform in preference to the Fourier transform.

Additionally, the wavelet representation is better at modelling abrupt changes and suffers less from overshoot and ringing effects, features of the Fourier representation due to sinusoidal bases (Szu [66] and Telfer [70]).

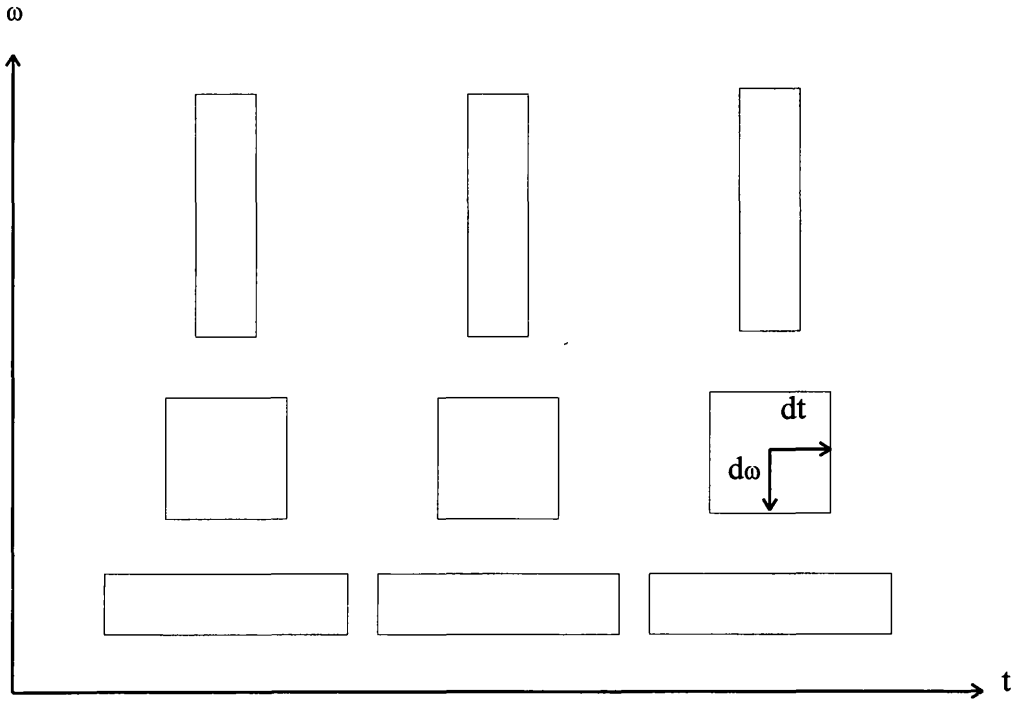


Figure 1.1: wavelet transform resolution cells

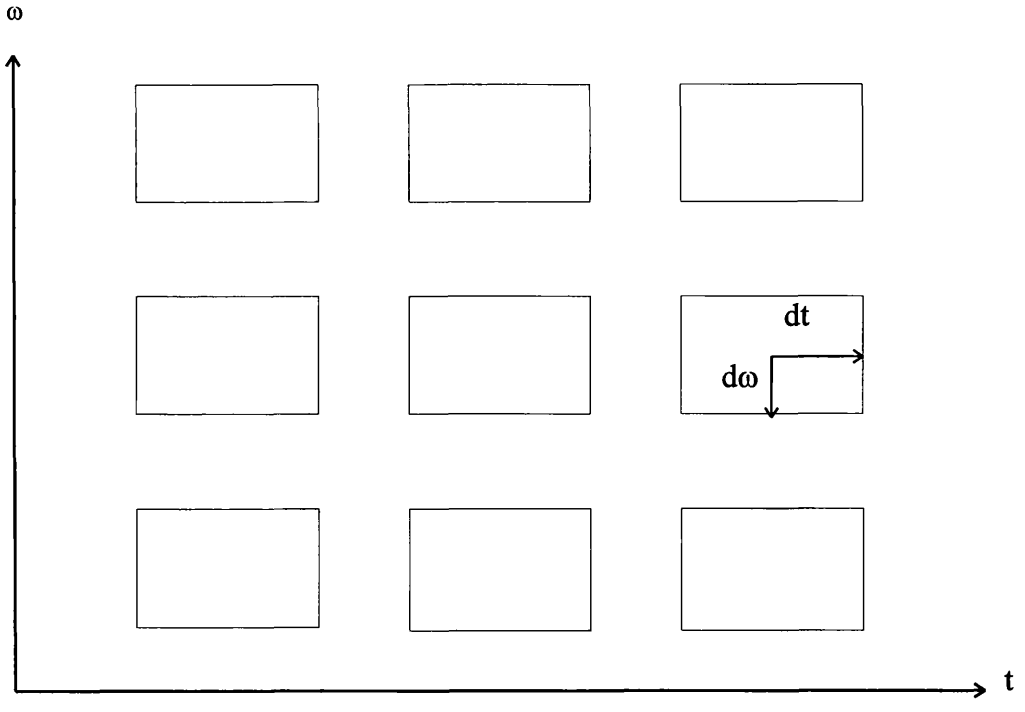


Figure 1.2: Fourier transform resolution cells

Finally, the discrete wavelet transform devised by Mallat [40, 51] offers order N complexity, thereby giving a speed advantage over fast Fourier which is itself order N log(N), (Strang [64]). This advantage rightly names Mallat's transform the *fast* transform. Details are given in chapter 3.

A recent JASON review [31] identified 3 main areas for wavelet application: data compression; de-noising; feature extraction. The growth of wavelet usage is manifested by the wealth of publications and conferences in the last 8 years. The number 8 is chosen, based on a quote of Stéphane Mallat from a 1989 paper [41] in which he thought it necessary to explain the term when introducing "a function $\Psi(x)$ *called* a wavelet."

Although much of the published material (about 90%) is from universities and government institutions, wavelets are slowly migrating out of academia and finding applications in commercial video and audio compression, as well as feature recognition in such areas as human signature validation (Tian [72]), machine health monitoring (Atlas [7]) and enhancement of Synthetic Aperture Radar (SAR) images (Tuthill [73]).

The field is still in a state of development with *wavelet* not yet a word in engineers' vocabularies, and certainly only those people actually engaged in their use know what wavelets are. This is contrary to the ubiquity of the Fourier transform, which even if a person doesn't use it, at least has heard of it. Bracewell [10] discusses the broader aspects of the transform method from the viewpoint of numerical computation.

Despite the advantages cited earlier, the slow uptake is due, in some part, to the simple reason that Fourier analysis does work and is a very useful tool in signal analysis.

1.4 History

Briefly, the significant waypoints in the journey to current theory are outlined by the names below:

• Fourier	1880	global	CON (Complete Ortho-Normal) periodic
• Haar	1910	local	CON non-periodic
• Gabor	1942	wavepacket	
• Morlet	1985	affine	CON
• Mallat	1989	fast transform	CON
• Daubechies	1989	discrete	CON beyond Haar

This is a concise historical review which represents the salient points in this field.

Going back to Jean Morlet in 1984/5; he invented the wavelet due to his dissatisfaction with the FT. It was his work which united the mathematicians (about 300 world-wide) with engineers and physicists, thus bringing together theory with application. Working in seismology, he first collaborated with Alexander Grossman. At first no one listened to his theories at geophysics conferences because they weren't in books. Grossman promoted the idea of scale analysis. He was interested in discrete bases but at this time no orthogonal ones existed. This inspired Ingrid Daubechies to pursue compactly supported discrete wavelets. Her major wavelet selling point was the compact energy representation of the WT. She sought to find discrete orthonormal wavelets which at the time were thought impossible, beyond the simple case of the two tap Haar. The property of orthonormality lends a degree of simplification to favour them over non-orthogonal wavelets, namely that orthonormality renders the analysis and synthesis matrices identical. The term *identical* here implicitly recognizes the transposition of the inverse matrix, there being a one-to-one correspondence of matrix elements.

It is discrete orthonormal wavelets to which this thesis dedicates further investigation.

1.5 Future

Wavelets are good at linear approximation, ie you just add together the contributions from each scale of the approximation. Fourier works the same way but has no sensible connection between scales (frequencies) because no location information exists. New directions for wavelets are *adaptive methods* and *irregular grids*, the major qualification being that direction will be application driven.

In considering image compression, JPEG decays suddenly whereas the WT decays gracefully (Huffman [30]). Sun [65] illustrates this succinctly in his scheme for image transmission. In adaptive pattern recognition, the challenge is to find the feature vector at an appropriate resolution, taking advantage of the WT's multiresolution analysis. Here also we want graceful degradation (Szu [68]).

Time-frequency joint representations traditionally do not use wavelets, and this will probably remain the case, their being better served by windowed Fourier and Wigner-Ville (Pan [49]). Rioul [54] compares the two methods, the wavelet and Wigner-Ville, and examines the role of scale in signal analysis. Automatic Pattern Recognition (APR), based on biorthogonal systems, are another area served well by multiresolution analysis: here the use of neural nets allows graceful degradation, and as Assef [6] states, "Artificial Neural Nets can construct relations between input and output data without any explicit analytical model". Whether this is seen as a merit or escape clause, the future *is* adaptive neural net transforms.

The discrete wavelet transform is chosen for computational efficiency. Wavelets allow you to work incrementally, by only computing those scales necessary for particular analyses. High frequency $\rightarrow$ small scale. Ie scale is opposite to frequency; "a" as in (a, b) is the inverse of ω. The properties of dilation (a) *and* translation (b) give a new two variable transform coined by the term *affine*, a concept fundamentally different from frequency. It is the wavelet's affine property which will give it a lasting place in the engineer's toolbox of tomorrow.

1.6 Wavelets

1.6.1 Form and function

The wavelet method is a branch of information theory concerned with forms of representation.

The form of representation used to describe a problem determines the degree of ease or difficulty with which that problem can be solved. As an example, consider dividing 100 by 10. In base 10 it is easy; a simple shift one place right. In base 2 the calculation is more lengthy, but it is still soluble. If, however, Roman numerals are used, the problem becomes impossible. $\frac{C}{X}$ is not defined. Roman numerals do not support division. To repeat: the system of representation used to describe a problem determines crucially whether the solution of that problem is easy, difficult or impossible.

All the words on this page, and all the pages in this thesis, and all the books ever written can be expressed in only 26 letters. This is an impressive feat for so small an alphabet. Wavelets are the mathematician's alphabet. Wavelets provide efficient descriptions of functions. The wavelet function and the wavelet transform depend for their success on the affine principle. Affine means the process of translation and dilation.

The symbol for the wavelet is psi, ψ, a function of x, $\psi(x)$. The wavelet is shifted by a parameter b, $\psi(x-b)$ so that as b increases ψ moves in the positive x direction. The wavelet is also dilated

by a parameter 'a', $\psi\left(\frac{x-b}{a}\right)$, such that as 'a' increases, $\psi\left(\frac{x-b}{a}\right)$ dilates, and as 'a' decreases, $\psi\left(\frac{x-b}{a}\right)$ contracts.

The wavelet is a bounded function, and is zero almost everywhere, except for some interval where it has an oscillatory form. The interval over which it is non-zero is called its region of support, or simply its support.

The wavelet transform is an inner product method such that the analysing wavelet is correlated with the underlying signal.

$$W(a,b) = \int f(x)\psi\left(\frac{x-b}{a}\right)dx$$

where W is the wavelet transform of f. (A more rigorous description is given in chapter 2).

W(a,b) is a function of 2 variables, a and b, and thus gives information about size and location. In wavelet terms, size in embodied in the concept of scale, which is the opposite of frequency in Fourier analysis. Large scale compares with low frequency; small scale compares with high frequency. The comparison is not exact since the wavelet is not sinusoidal, due to its finite support.

Thus W(a,b) is a description of f(x) in terms of its similarity with the analysing wavelet. Since W(a,b) is a two parameter transform, it describes f(x) in terms of the dilated and translated wavelet, and so identifies constituent features of the signal in terms of their size **and** position. This, in Fourier analysis is impossible, because it has no parameter for position.

Implicit in the wavelet transform is the concept of self similarity, such that any wavelet $\psi_{a1,b1}$ is a self-similar form of any shifted and dilated version of itself, $\psi_{a2,b2}$. Thus the wavelet is size-invariant; able to locate features within signals at any resolution.

Progressing to discrete wavelets, the wavelet transform is implemented as a matrix technique using an "analysis" matrix. Here the wavelet is represented by a small number of coefficients called taps. They take this name to avoid ambiguity with the wavelet transform coefficients, $W_{a,b}$, which are themselves called the wavelet coefficients. The number of taps within the analysis matrix determines the length of the wavelet, ie the support. Longer wavelets have more taps and can thus respond to more of the signal at once. The disadvantage is that the transform is slower.

The discrete wavelet transform is described in chapter 3, where the matrix method of shifting and dilating is explained.

However, one thing which must be brought into this overview is the inclusion of another function, the scaling function, which is a necessary component of the discrete wavelet transform. The scaling function has the symbol ϕ, phi, and like ψ is a shifted and dilated function of x: $\phi\left(\frac{x-b}{a}\right)$. The reason for having two functions is that ψ and ϕ extract different sorts of information from f. ϕ is an averaging function which gives a description of the smooth form of f at any resolution. ψ is a differencing operator and contains the fine detail which must be added to ϕ to recover the exact form of f at a particular resolution.

To summarize the form and function of the wavelet: the key properties of ψ and ϕ are translation and dilation, where ψ and ϕ are respectively the wavelet and scaling function. The key feature of the discrete wavelet transform is the analysis matrix composed of the wavelet taps. It is the values of these taps which determine the shape of the wavelet, and which determine the suitability of the wavelet for analysing a particular function.

1.6.2 abc wavelets

The new class of wavelets described in this thesis is called abc wavelets, first introduced in 1995 (La Borde [35]), and. further documented in 1996 (La Borde [36]). The name is drawn from an extra parameter, c, in addition to the conventional ones of "a" for dilation and "b" for translation: $\Psi(a,b,c)$. The abc class is discrete and orthogonal. Explicit solutions exist for 4 and 6 taps. The name *abc* is used as a simple identifier for these classes, although "a" and "b" are not strictly parameters in the discrete case, both being fixed to 2 for the fast Mallat transform. The name however, is indicative of parameterization and is a useful shorthand for "parameterized wavelets". Full derivation is given in chapters 4 and 5; their introduction here is to establish a name for these wavelets and put a handle on the thesis, as well as to familiarize the reader with the name "abc wavelets". The use of a parameter, c, in the wavelet derivation, leads to generic expressions for the taps, yielding a continuous set of discrete wavelets in a wavelet space defined in terms of this single parameter. To emphasize: the wavelets are discrete; the class is continuous. A particular value of c gives one particular discrete wavelet.

1.6.3 Wavelet hierarchy

Wavelets can be split into two categories: continuous and discrete. Of the discrete version, a further tripartite division exists of orthogonal, non-orthogonal and biorthogonal. Biorthogonal wavelets have advantages over orthogonal ones in that they are symmetric, and therefore better able to represent images because their compressed reconstructions yield less asymmetric distortion. Antonini [5] examines the psychovisual aspects of asymmetric distortion, as similarly does da Silva [17], who also emphasizes the impact of quantization and encoding on image efficiency and distortion.

In this thesis biorthogonal wavelets are not addressed and shall not be discussed beyond this overview. The category of discrete orthogonal wavelets can then be further subdivided into fixed wavelets and parameterized wavelets. The fixed wavelets are a subset of parameterized wavelets, the accepted standards being Daubechies wavelets, so chosen for their optimal smoothness. So dominant are the Daubechies wavelets that some authors describe them simply as *wavelets*, apparently assuming that there are no wavelets that are not Daubechies. Other times they are referred to as the *standard* wavelets (Scholl [58]). Following this practice lain out below, in figure 1.3, is a wavelet hierarchy in which the orthogonal wavelets are classed as Daubechies and not-Daubechies. This is a selective hierarchy pruned to show the location of the *abc* wavelets. Obviously, parameterization is not unique to orthogonal wavelets, and certainly the method exists on other leaves of the tree.

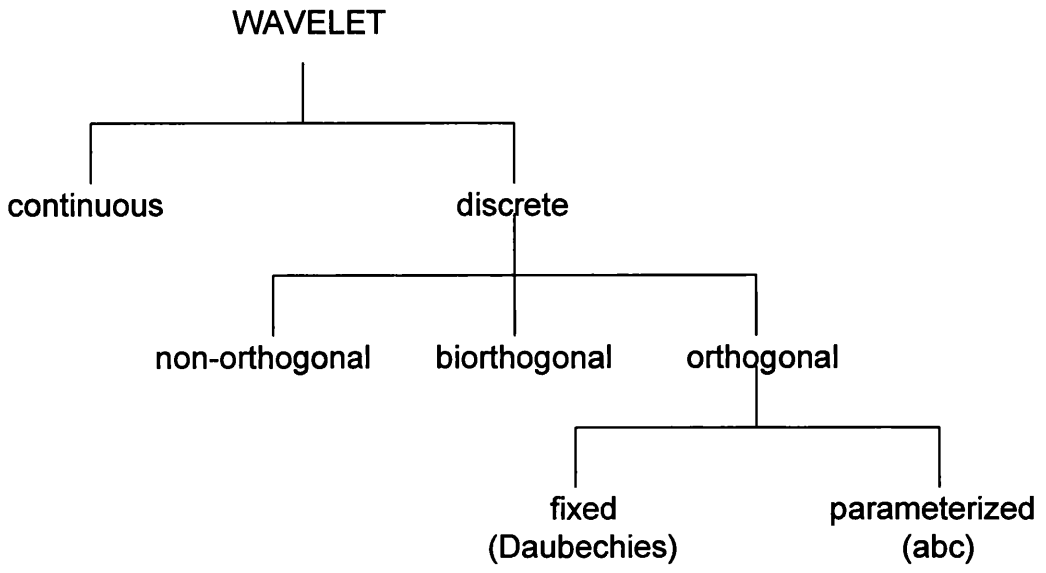


Figure 1.3: wavelet classification hierarchy

Of the parameterized types, there are 3 classes which are covered in this introduction, as a reference to which the abc class can be compared.

They are

- Vetterli
- Szu
- Pollen

These are 4 and 6 tap wavelets and are documented separately in the following sections.

1.6.4 Vetterli

Martin Vetterli [74, 75] has a four tap wavelet parameterized as (1, α, α, 1). It is not orthogonal and is stable only for $\alpha > 0$. It includes asymmetric and symmetric wavelets, including a quadratic spline for $\alpha = 3$. Figures 1.4 and 1.5 show a selection of Vetterli wavelets and corresponding scaling functions for $\alpha \in [-3, 3]$.

For negative α the wavelets are unstable, as shown by their high frequency oscillations. This means that they are unable to model any smooth features of signals, since the wavelets themselves are not smooth. In compression, this would manifest itself as alternate extreme values in the reconstructed signal. In images the effect would appear as neighbouring bright and dark spots throughout the image.

Although the Vetterli class does include symmetric wavelets, this property cannot be exploited due to their instability as illustrated in these figures. Another drawback is their inconsistent area due to unbounded sum of the taps. As explained in chapter 4, the area conservation requirement states

$$c_0 + c_1 + c_2 + c_3 = \sqrt{2}$$

which is clearly untrue for

$c_0 = 1$
$c_1 = \alpha$
$c_2 = \alpha$
$c_3 = 1$

Leaving aside the particular value of the sum, Σc, it remains that area is a function of α, so not even a normalizing factor could coerce area conservation. This wavelet is not only not orthogonal, it also cannot be normalized, in the sense of some sum being set to 1 or $\sqrt{2}$. Hence the wavelet is not orthonormal.

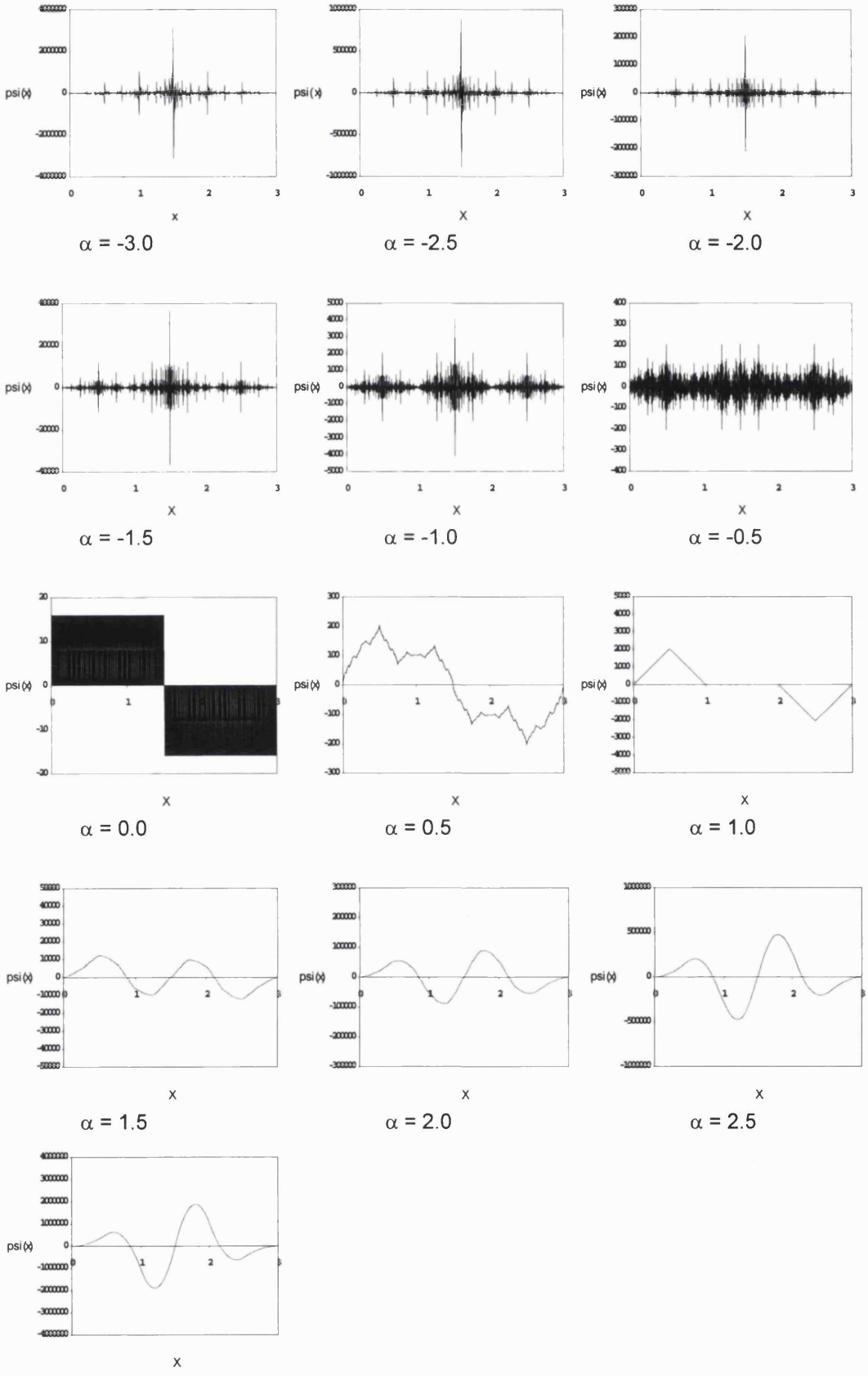


Figure 1.4: Vetterli wavelets (1, α, α, 1)

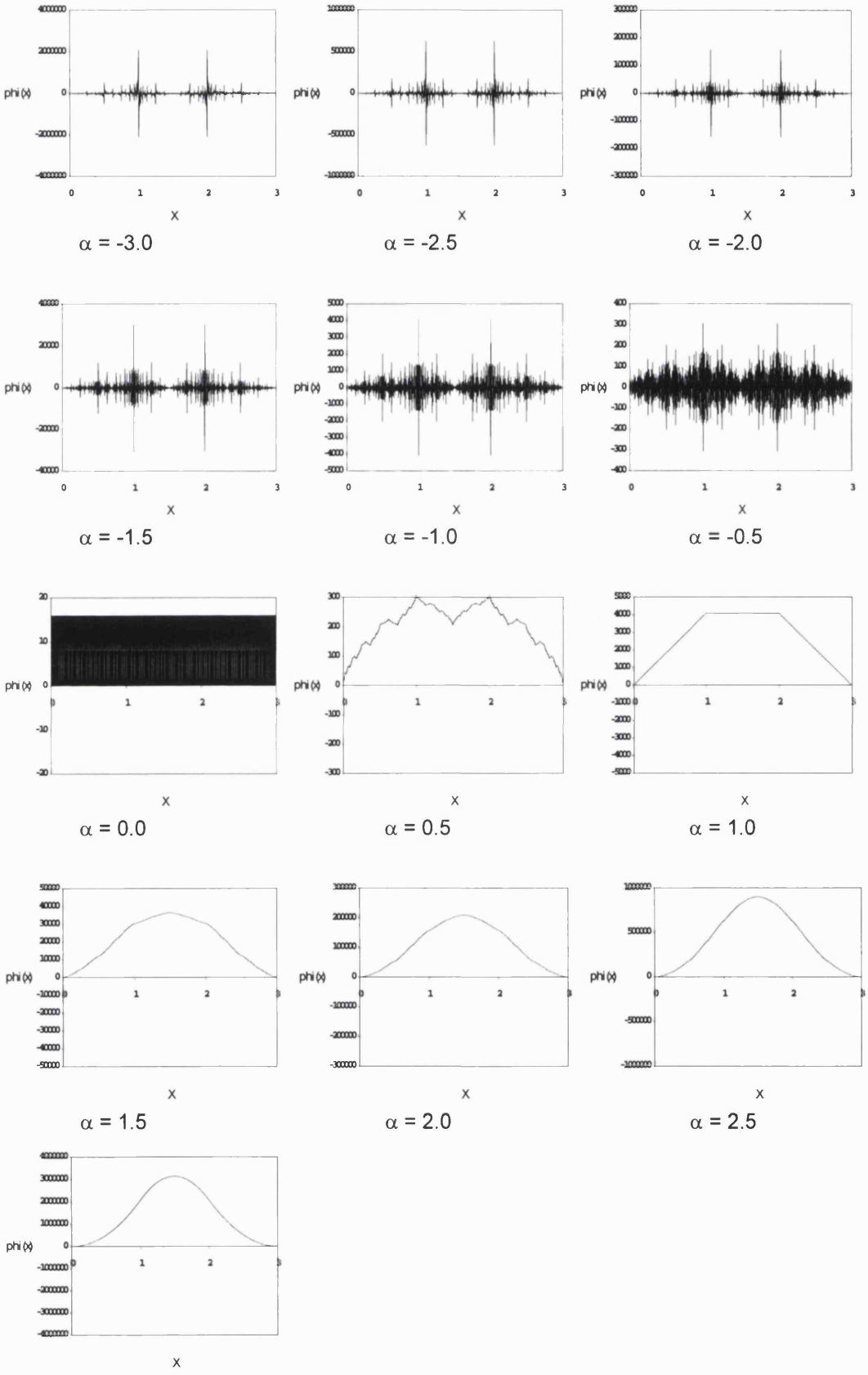


Figure 1.5: Vetterli scaling functions (1, α, α, 1)

1.6.5 Szu

Harold Szu [67] discusses a superset of the Vetterli wavelet parameterized as (1, α, β, 1). Again it is not orthogonal, but has the advantage of including both the Vetterli wavelet and wavelets which Szu claims includes the Daubechies 4. This is not true, since, although the above form allows arbitrary assignment of α and β to mimic Daubechies taps, the two unit taps prohibit an exact match. Figures 1.6 and 1.7 illustrate the Szu wavelets and scaling functions for a fixed value of β while varying α. This example is taken from his paper describing this parameterized wavelet and uses the fixed value $\beta = 3 - \sqrt{3}$. These wavelets are not orthogonal nor normalizable for the same reasons as for the Vetterli wavelets. Additionally, Szu's wavelets are everywhere unstable except for the Vetterli subset where $\alpha = \beta > 0$; and also the wavelets fail to integrate to zero, ie are not admissible.

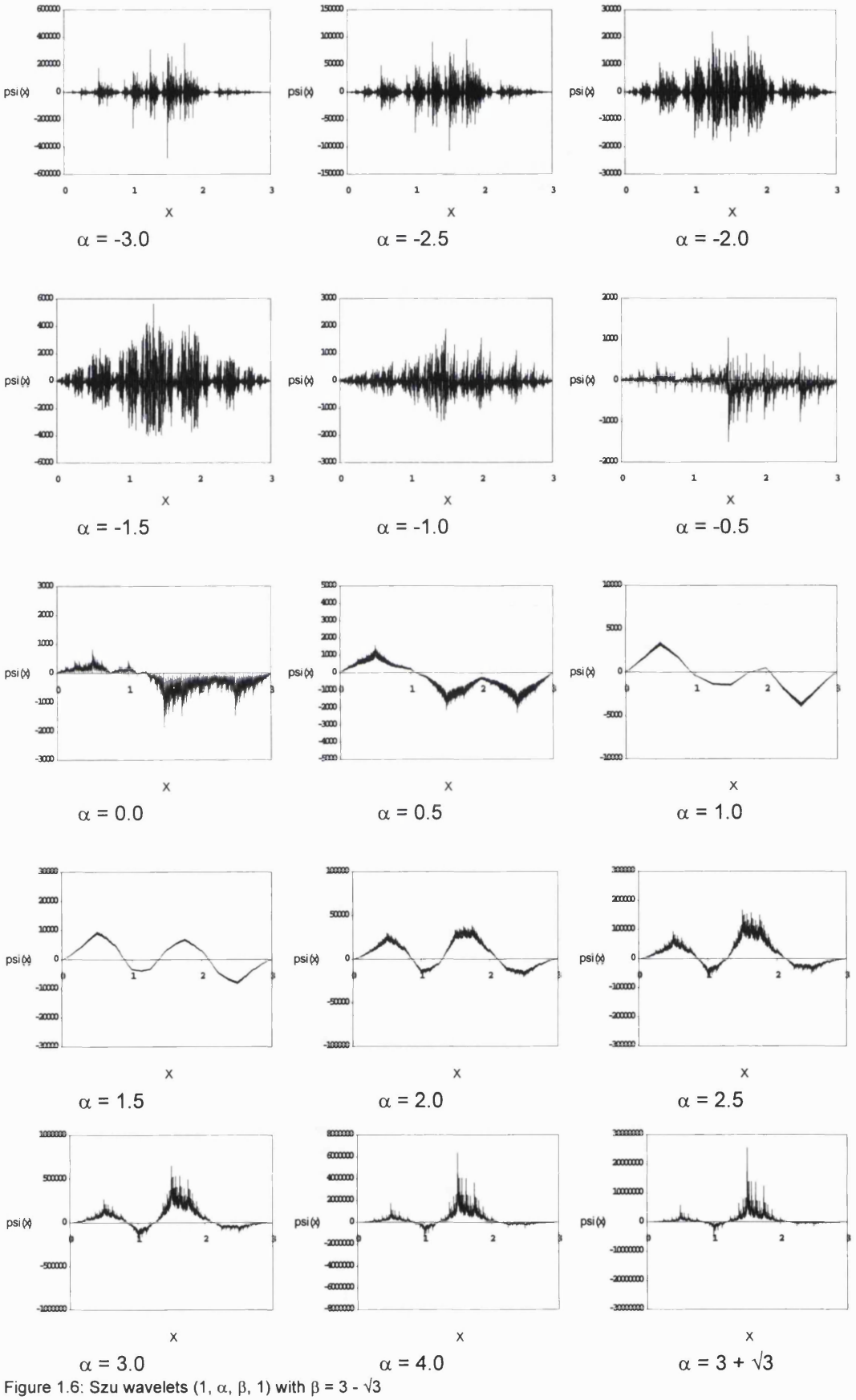


Figure 1.6: Szu wavelets (1, α, β, 1) with β = 3 - √3

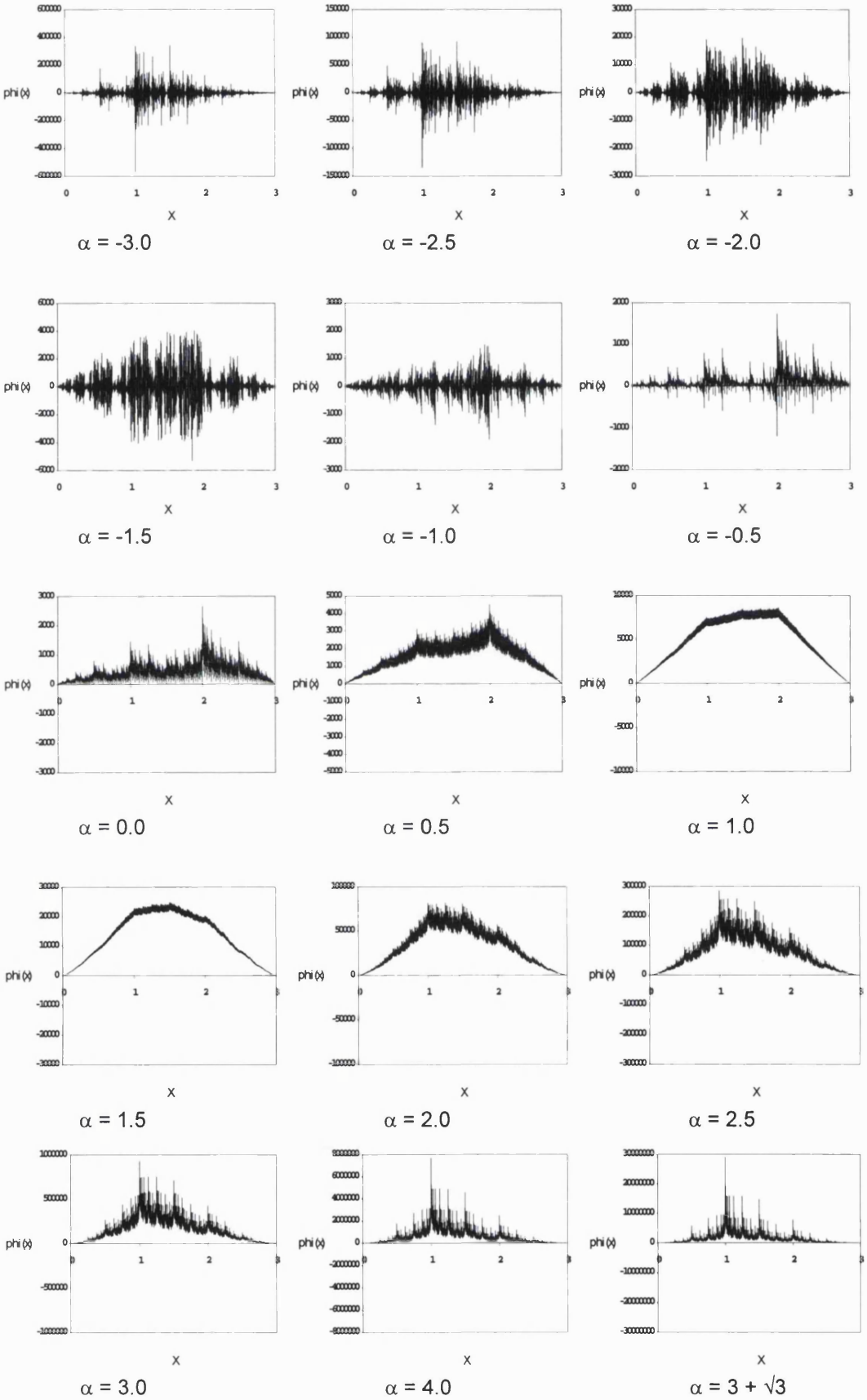


Figure 1.7: Szu scaling functions (1, α, β, 1) with β = 3 - √3

1.6.6 Pollen

David Pollen [14, 50] has a 6 tap wavelet, and like Szu uses two parameters, α and β to characterize his space. The 6 taps are expressed as:

$c_0 = [(1+\cos\alpha+\sin\alpha)(1-\cos\beta-\sin\beta)+2\sin\beta\cos\alpha]/4$
$c_1 = [(1-\cos\alpha+\sin\alpha)(1+\cos\beta-\sin\beta)-2\sin\beta\cos\alpha]/4$
$c_2 = [1+\cos(\alpha-\beta)+\sin(\alpha-\beta)]/2$
$c_3 = [1+\cos(\alpha-\beta)-\sin(\alpha-\beta)]/2$
$c_4 = 1-c_0-c_2$
$c_5 = 1-c_1-c_3$

with $\alpha, \beta \in [-\pi, \pi)$.

This is a very versatile system embodying 4 and 6 tap solutions for particular values of α and β. It includes stable and unstable solutions and is always orthogonal. The use of two parameters prohibits comprehensive illustration of Pollen's wavelets; to demonstrate their range, one parameter is set while the other is varied. Figure 1.8 shows the 4 tap case with $\beta = 0$ and a selection of values for α. Without deriving their forms, let it be quoted here that for this value of β, c_0 and c_5 are both zero, thus yielding a four tap solution. However there are also non-wavelet solutions, occurring for $\alpha = n\pi/2$. These are not shown since they are simply straight lines $\psi(x) \equiv 0$.

Figure 1.9 shows the full 6 tap case with α set to $\pi/3$ while β is varied. This is primarily a six tap class, although this range does also include some 4 tap solutions, and also the trivial non-solutions $\psi(x) \equiv 0$ already mentioned. The wavelet space covered by this two parameter set embodies the maxim that there are an infinite number of wavelets, most of which have little value. The use of two parameters leads to some difficulty in selecting appropriate values for useful stable wavelets.

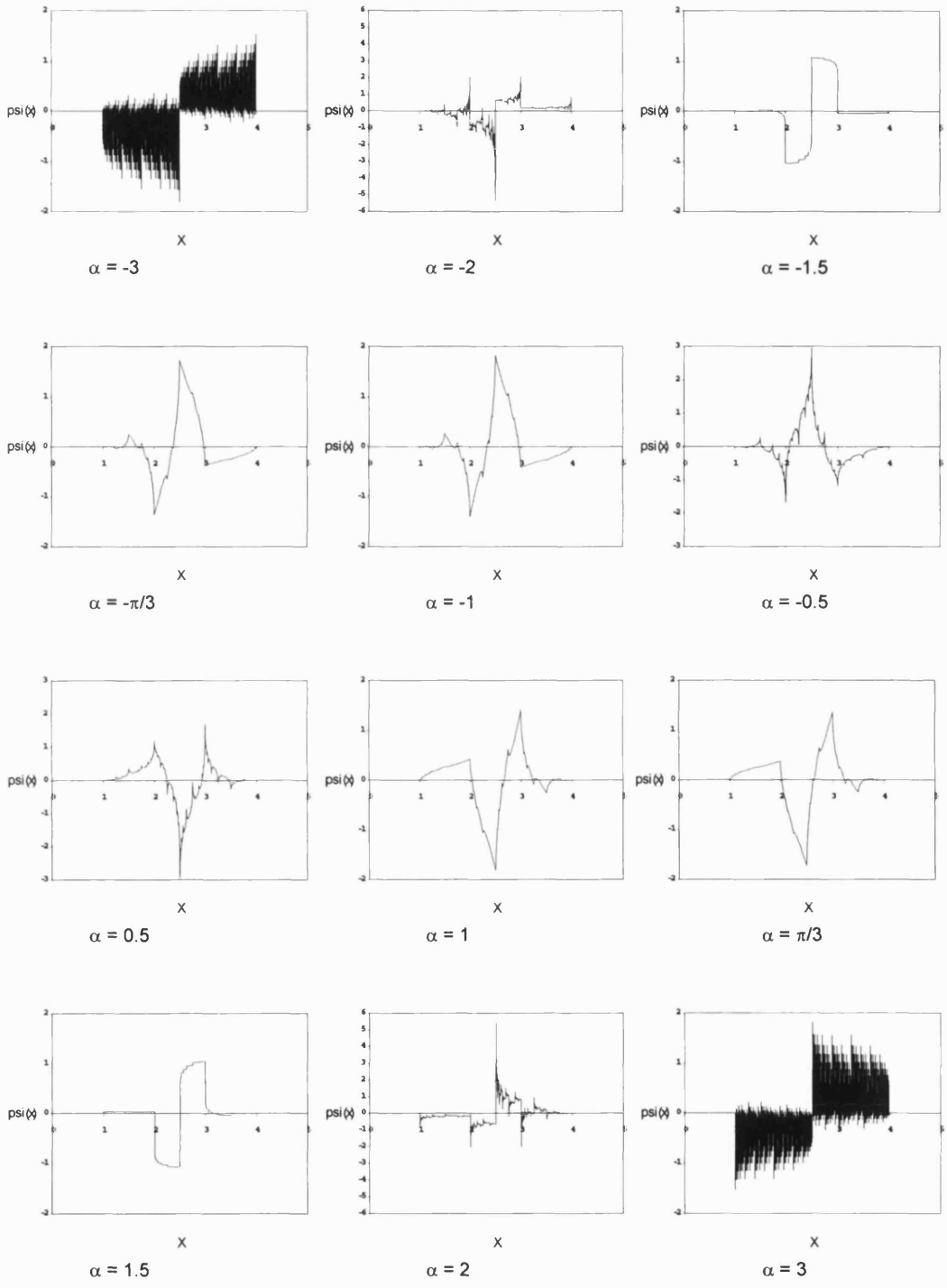


Figure 1.8: Pollen wavelets for $\beta = 0$

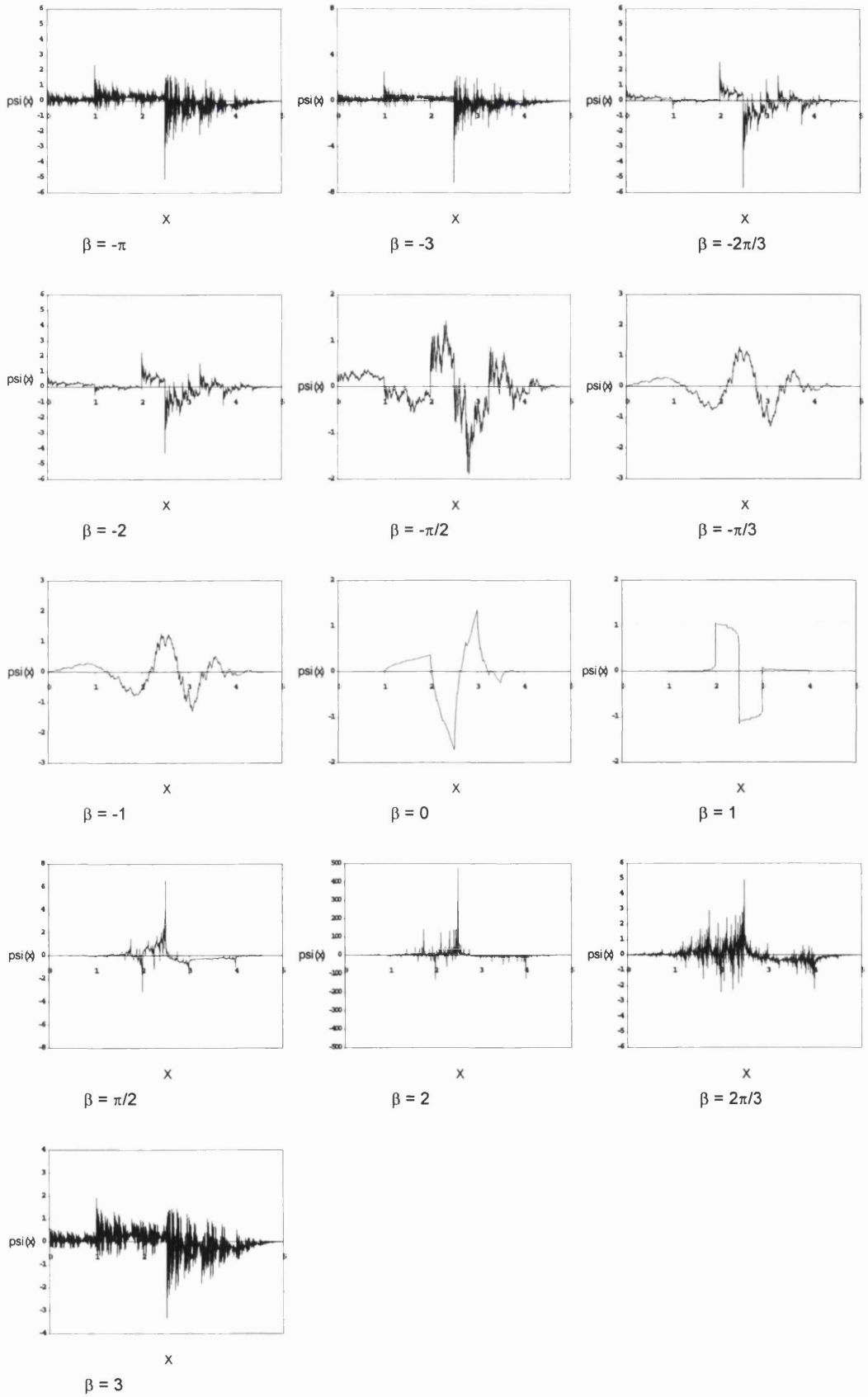
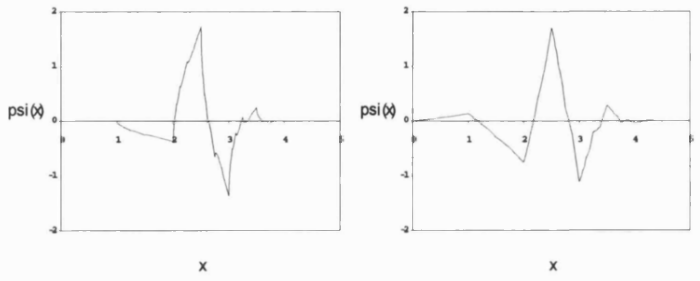


Figure 1.9: Pollen wavelets for $\alpha = \pi/3$

Gene Tagliarini [69] shows that the Pollen parameters can be searched in algorithmic fashion to find close approximations to desired wavelets. He finds solutions for close approximations to Daubechies wavelets. For the 4 tap Daubechies wavelet, the Tagliarini-Pollen parameters are $\alpha = 1.570887$, $\beta = -1.047168$ (ie the exact solution is $\alpha = \pi/2$, $\beta = -\pi/3$). In this case c_4 and c_5 are zero and are not used. His values for Daubechies 6 approximation are $\alpha = 1.360035$ and $\beta = -0.783036$ (these look seductively close to $\alpha = 3\pi/7$, $\beta = -\pi/4$, but this is misleading; these convenient pi multiples do not yield the desired Daubechies 6). Figure 1.10 shows the Tagliarini-Pollen-Daubechies 4 and 6 tap wavelets.



$\alpha = 1.570887$, $\beta = -1.047168$ $\alpha = 1.360035$, $\beta = -0.783036$

Figure 1.10: Tagliarini-Pollen approximations for Daubechies 4 and 6 tap wavelets

1.7 Summary

The background given here shows that wavelet parameterization is a plausible pursuit and that extant systems address the problem with various degrees of sophistication. The following chapters show that a system of wavelets can be derived from first principles to yield 4 and 6 tap analytic classes which always have all of the following properties:

- closed form solution for the taps
- orthonormality
- stability
- single parameter
- purpose

The use of a single parameter bypasses the instability problems of Pollen, while still allowing analytic forms for 6 taps. Orthonormality is achieved by making it a condition in the derivation. Purpose comes from application.

2 CONTINUOUS WAVELETS AND THEIR INFLUENCE ON DISCRETE WAVELETS

2.1 Introduction

Although this thesis concentrates on generic discrete wavelets, the continuous wavelet transform is documented here to illustrate an important relationship between continuous and discrete wavelets and to highlight a constraint imposed by the former on the latter. This constraint is called the *admissibility condition* and derives from a requirement of the inverse transform which states that continuous and discrete wavelets both have no dc component, ie that they integrate to zero. Without this condition there would be little justification in calling discrete filters 'wavelets' other than their having affine properties.

2.2 The continuous wavelet transform

The following description comes from Daubechies' "10 Lectures on Wavelets" [18], expanded to include intermediate steps in the derivation to give an easier progression to the conclusion, namely that $\hat{\psi}(0) = 0$, ie $\int_{-\infty}^{\infty} \psi(x)\,dx = 0$. This is another way of saying the wavelet has no area.

We start with the affine definition of a wavelet

$$\Psi^{a,b}(x) = |a|^{-\frac{1}{2}} \psi\left(\frac{x-b}{a}\right) \tag{2.1}$$

where $\psi\left(\frac{x-b}{a}\right)$ is the mother wavelet and $\Psi^{a,b}(x)$ is the (a,b) member of that wavelet family, also called the daughter wavelet. Parameters a and b are respectively the coefficients for dilation and translation. For the remainder of this analysis the (a,b) superscript will be dropped. In the following discussion, overbar ($\overline{\quad}$) denotes the complex conjugate. The wavelet transform is denoted tilde (~) and is defined as the inner product of the signal f(x) and the wavelet.

$$\tilde{f}(a,b) = |a|^{-\frac{1}{2}} \int_{-\infty}^{\infty} f(x)\overline{\psi\left(\frac{x-b}{a}\right)}\,dx \tag{2.2}$$

Given the wavelet definition and its transform, we must now engage in some foresight to derive the inverse transform, and in so doing also discover the zero area constraint on the wavelet.

The following equation comes from Daubechies book, page 24, and the reason for using it as a starting point is simply that it gives the required result. For algebraic simplicity the shorthand I is adopted for the integral which otherwise has no name.

$$I = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{a^2}\tilde{f}(a,b)\overline{\tilde{g}(a,b)}\,da\,db \tag{2.3}$$

Substitute in the wavelet definition for f and g to give

$$I = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{da\,db}{a^2}\left\{\int_{-\infty}^{\infty} dx\, f(x)|a|^{-\frac{1}{2}}\overline{\psi\left(\frac{x-b}{a}\right)}\right\}\left\{\int_{-\infty}^{\infty} dy\, \overline{g(y)}|a|^{-\frac{1}{2}}\psi\left(\frac{y-b}{a}\right)\right\}$$

The next step is to express f(x) and g(x) as the inverse Fourier transform of their Fourier transforms. The Fourier transform is denoted hat (^).

$$I = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{da\,db}{a^3}\int_{-\infty}^{\infty}dx\left\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{f}(\xi)e^{-ix\xi}d\xi\right\}\left\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\overline{\hat{\psi}(\theta)}e^{+i\theta\left(\frac{x-b}{a}\right)}d\theta\right\}$$
$$\int_{-\infty}^{\infty}dy\left\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\overline{\hat{g}(\xi')}e^{+iy\xi'}d\xi'\right\}\left\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{\psi}(\theta')e^{-i\theta'\left(\frac{y-b}{a}\right)}d\theta'\right\}$$

Exploiting the Dirac delta equivalence

$$\frac{1}{2\pi}\int_{-\infty}^{\infty}e^{\pm i\mu z}dz = \delta(\mu)$$

The I components above simplify as

$$\int_{-\infty}^{\infty}dx e^{-ix\xi}e^{+i\theta\left(\frac{x-b}{a}\right)} = \int_{-\infty}^{\infty}dx e^{-ix\left(\xi-\frac{\theta}{a}\right)}e^{-i\theta\frac{b}{a}} = 2\pi e^{-i\theta\frac{b}{a}}\,\delta\left(\xi-\frac{\theta}{a}\right) \text{ for } \xi,$$

and

$$\int_{-\infty}^{\infty}dy\, e^{+iy\xi'}e^{-i\theta'\left(\frac{y-b}{a}\right)} = \int_{-\infty}^{\infty}dy\, e^{+iy\left(\xi'-\frac{\theta'}{a}\right)}e^{+i\theta'\frac{b}{a}} = 2\pi e^{+i\theta'\frac{b}{a}}\,\delta\left(\xi'-\frac{\theta'}{a}\right) \text{ for } \xi'.$$

Then the deltas equal 1 when $\xi = \frac{\theta}{a}$ and $\xi' = \frac{\theta'}{a}$. With these simplifications the integral becomes

$$I = 2\pi\int_{-\infty}^{\infty}\frac{da}{a}\int_{-\infty}^{\infty}d\xi\hat{f}(\xi)\overline{\hat{g}(\xi)}\,\overline{\hat{\psi}(a\xi)}\hat{\psi}(a\xi)$$

$$= 2\pi\int_{-\infty}^{\infty}d\xi\hat{f}(\xi)\overline{\hat{g}(\xi)}\int_{-\infty}^{\infty}\frac{da}{a}\left|\hat{\psi}(a\xi)\right|^2$$

Now use the variable substitution $u = a\xi$.

$$I = \int_{-\infty}^{\infty}d\xi\hat{f}(\xi)\overline{\hat{g}(\xi)}.2\pi\int_{-\infty}^{\infty}\frac{du}{u}\left|\hat{\psi}(u)\right|^2$$

$= C_\psi \langle f,g \rangle$

where

$$C_\psi = 2\pi\int_{-\infty}^{\infty}\frac{du}{u}\left|\hat{\psi}(u)\right|^2$$

(2.3) now yields

$$I = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{a^2}\tilde{f}(a,b)\overline{\tilde{g}(a,b)}\,da\,db = C_\psi \langle f,g\rangle$$

$$\text{ie } \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{a^2}\tilde{f}(a,b)\left\{\int_{-\infty}^{\infty} dy\ \overline{g(y)}|a|^{-\frac{1}{2}}\psi\left(\frac{y-b}{a}\right)\right\} da\,db = C_\psi \langle f,g\rangle$$

Then

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{1}{a^2}\tilde{f}(a,b)\left\{\int_{-\infty}^{\infty} dy\ \overline{g(y)}\psi(y)\right\} da\,db = C_\psi \langle f,g\rangle$$

by virtue of (2.1).

Independence of the integrands allows rearrangement:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{da\,db}{a^2}\tilde{f}(a,b)\psi(y)\overline{g(y)}\,dy = C_\psi \langle f,g\rangle$$

Therefore, by the definition of the inner product, $\langle f,g\rangle = \int_{-\infty}^{\infty} f(x)\overline{g(x)}\,dx$, the identity becomes

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{da\,db}{a^2}\tilde{f}(a,b)\psi(x) = C_\psi f(x) \tag{2.4}$$

Where C_ψ^{-1} is the reconstruction constant and x has replaced y to adopt the more common notation.

Rearranging in terms of f(x) we get

$$f(x) = C_\psi^{-1}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{da\,db}{a^2}\tilde{f}(a,b)\psi(x) \tag{2.5}$$

This is the wavelet resolution of identity showing that the continuous wavelet transform is a reversible process. It shows how the original function f(x) can be reconstructed from its wavelet transform $\tilde{f}(a,b)$.

2.3 Admissibility condition

The start of this chapter justified the inclusion of continuous wavelets by necessity of the admissibility condition, which can now be defined. Validity of the resolution of identity, (2.5), requires that the reconstruction constant C_ψ^{-1} is finite.

Therefore it is required that

$$2\pi \int_{-\infty}^{\infty} \frac{du}{u} |\hat{\psi}(u)|^2 < \infty$$

and since u ranges over $[-\infty, \infty]$ then a necessary condition is that $\hat{\psi}(u) = 0$ when $u = 0$ to avoid a logarithmically divergent contribution to the integral in the neighbourhood of $u = 0$.

Ie $\hat{\psi}(0) = 0$. This is equivalent to stating $\int_{-\infty}^{\infty} \psi(x)\, dx = 0$.

Thus the derivation of the continuous wavelet inverse transform imposes an extra constraint on the wavelet, which although very simple is also very necessary. It is the invertability of the wavelet transform, (not the wavelet, nor the forward transform itself) which imposes this admissibility condition.

2.4 Conclusion

In this chapter the reconstruction constant has been derived for the continuous wavelet transform. Additionally, it has been proven that the admissibility condition must be satisfied in order that the transform have an inverse. Further, it is this same admissibility condition that exists in the discrete wavelet domain which imposes a zero area condition on the wavelet taps, where it manifests as the constant moment.

3 DISCRETE WAVELETS AND THE FAST WAVELET TRANSFORM

3.1 Introduction

This chapter documents 3 aspects of discrete wavelets: their derivation; implementation, and appearance. The wavelet transform used as the basis of this thesis is the Fast transform, or Mallat transform, known for its speed in transforming signals into their dyadic components in a logarithmically fast manner, such that for an N stage transform, the last N - 1 stages take the same computation time as the single first stage. Thus a doubling of the signal only leads to a doubling of the wavelet transform time, compared with the Fourier equivalent which would require a squaring of the time (Strang [64]), in this case four times the computational time. The fast wavelet transform is a matrix method which offers an additional advantage in that the matrix is inverse self-transpose, which means that the inverse transform is simply a reversal of the forward transform; thus the numerical implementation requires a single function for both the forward and inverse operations.

3.2 Derivation

The transform matrix contains the wavelet taps in line pairs in diagonal form, each line pair representing the scaling function above and wavelet below. This pairing places a size constraint on the matrix, the effect of which is to limit the number of taps, N, to be a multiple of 2, ie an even number. The term "tap" is used to avoid the ambiguous term "coefficient" which is generally reserved to describe the transform values. These taps are themselves not the wavelet, but instead form what is called the pre-wavelet (Stollnitz [61]), or simply the taps. The diagonal construction introduces the translation aspect. Dilation is implemented in the transform by vector decimation after successive matrix multiplications. The term "decimation" is peculiarly hijacked by the wavelet community to mean the removal of alternate components of the signal, ie reducing the signal resolution by a half.

For an N order matrix, N/2 orthonormality conditions guarantee perfect reconstruction when compression is not used, and N/2 equations remain for specifying the desired wavelet characteristics. Classically the moment conditions are used to provide polynomial accuracy of degree N/2 - 1 (Strang & Fix [62]).

Daubechies wavelets are the industry standard for signal compression. Their use in Mallat decomposition is based on these conditions of orthonormality, supplemented with N/2 moments, also known as Strang accuracy conditions [63]. The discrete wavelet technique depends on the success of the Daubechies wavelet matrix form which in addition to being inverse self-transpose, also amazingly offers degrees of freedom. The surprise comes from having N/2 degrees of freedom in a system of N equations in N variables.

It is this freedom that allows the derivation of wavelet families from what otherwise would simply be a system of N equations and N unknowns with a single solution for the matrix transpose. In video compression, the crucial aesthetic factor is smoothness of the synthesized image. The smoothness gained from Daubechies wavelets is clearly essential for perceptive quality, where regularity is more important than accuracy (Morris [45]).

In the following discussion, the value N = 4 is used for simplicity, with taps c_0, c_1, c_2, c_3. The method described extends for any even N, with taps c_0, c_1, c_2, c_3, c_4, ... c_{N-2}, c_{N-1}. The transformation is based on matrices derived from conditions of orthonormality and vanishing of the moments. These matrices are called the analysis and synthesis matrices, referred to in this discussion as A and A inverse, A^{-1}. Recall that by necessity $A^T = A^{-1}$.

Given a matrix A, $A^{-1} = A^T$ iff A is orthonormal. Thus the two matrices are

A

$$\begin{bmatrix} c0 & c1 & c2 & c3 & & & & \\ c3 & -c2 & c1 & -c0 & & & & \\ & & c0 & c1 & c2 & c3 & & \\ & & c3 & -c2 & c1 & -c0 & & \\ & & & & \ddots & & & \\ & & & & & \ddots & & \\ & & & & & c0 & c1 & c2 & c3 \\ & & & & & c3 & -c2 & c1 & -c0 \\ c2 & c3 & & & & & & c0 & c1 \\ c1 & -c0 & & & & & & c3 & -c2 \end{bmatrix}$$

$A^T = A^{-1}$

$$\begin{bmatrix} c0 & -c3 & & & & & & & c2 & -c1 \\ c1 & c2 & & & & & & & c3 & c0 \\ c2 & -c1 & c0 & -c3 & & & & & & \\ c3 & c0 & c1 & c2 & & & & & & \\ & & & \ddots & & & & & & \\ & & & & \ddots & & & & & \\ & & & & c2 & -c1 & c0 & -c3 & & \\ & & & & c3 & c0 & c1 & c2 & & \\ & & & & & & c2 & -c1 & c0 & -c3 \\ & & & & & & c3 & c0 & c1 & c2 \end{bmatrix}$$

Orthonormality conditions yield:

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$$

and

$$c_2c_0 + c_3c_1 = 0$$

Additionally, the first two moment conditions yield

$$c_3 - c_2 + c_1 - c_0 = 0$$

and

$$0c_3 - 1c_2 + 2c_1 - 3c_0 = 0$$

Solution gives

$$c_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}$$

$$c_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}$$

$$c_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}$$

$$c_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}$$

This is Daubechies 4; it is the most compact of a sequence of wavelet sets. For N coefficients there are N/2 orthogonality requirements and N/2 moment equations (Amaratunga [3]). This derivation is explained further by considering the scaling equation.

Given a scaling function

$$\phi(x) = c_0\phi(2x) + c_1\phi(2x-1) + c_2\phi(2x-2) + c_3\phi(2x-3)$$

then it is necessary to have a wavelet of the form

$$\psi(x) = c_3\phi(2x) - c_2\phi(2x-1) + c_1\phi(2x-2) - c_0\phi(2x-3)$$

simply because no other arrangement yields a flexible orthonormal system. There are other orthonormal systems, but none with any degrees of freedom, so they all reduce to the trivial case.

Examples follow. Consider a system of just $\phi(x)$ and no $\psi(x)$; then the system becomes

$$A \qquad\qquad A^T = A^{-1}$$

$$\begin{bmatrix} c0 & c1 & c2 & c3 & & & & & & \\ & c0 & c1 & c2 & c3 & & & & & \\ & & c0 & c1 & c2 & c3 & & & & \\ & & & c0 & c1 & c2 & c3 & & & \\ & & & & & \ddots & & & & \\ & & & & & & \ddots & & & \\ & & & & & & & \ddots & & \\ & & & & & & & & \ddots & \\ c3 & & & & & & & & & \\ c2 & c3 & & & & & & & c0 & c1 \\ c1 & c2 & c3 & & & & & & & c0 \end{bmatrix} \begin{bmatrix} c0 & & & & & & & & c3 & c2 & c1 \\ c1 & c0 & & & & & & & & c3 & c2 \\ c2 & c1 & c0 & & & & & & & & c3 \\ c3 & c2 & c1 & c0 & & & & & & & \\ & c3 & c2 & c1 & c0 & & & & & & \\ & & c3 & c2 & c1 & c0 & & & & & \\ & & & & & & \ddots & & & & \\ & & & & & & & \ddots & & & \\ & & & & & & & & & c0 & \\ & & & & & & & & & c1 & c0 \end{bmatrix}$$

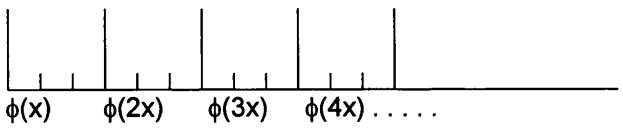
So for orthonormality the conditions are

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$$
$$c_0c_1 + c_1c_2 + c_2c_3 = 0$$
$$c_2c_0 + c_3c_1 = 0$$
$$c_0c_3 = 0$$

suppose $c_0 = 0$, $c_3 \neq 0$, then $c_1 = c_2 = 0 \Rightarrow c_3 = 1$ and the transformation matrix is

$$\begin{bmatrix} 0 & 0 & 0 & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & 0 \end{bmatrix}$$

So the solution is a set of Diracs, $\phi(x) = c_3\phi(2x-3)$



which can not even model intermediate function points!

Now suppose $c_3 = 0$, $c_0 \neq 0$,

therefore $c_1 = c_2 = 0$, $c_0 = 1$ and the transformation matrix is

$$\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}$$

$\phi(x) = c_0\phi(2x)$. Again there are no degrees of freedom, and also the Dirac approximations are collocated, so the system could only model a single point within a signal

$$\phi(x) = c_0\phi(2x) = c_0\phi(4x) = \ldots c_0\phi(2^n x).$$

Another example to consider does use a wavelet of the form

$$\psi(x) = c_3\phi(2x) + c_2\phi(2x-1) + c_1\phi(2x-2) + c_0\phi(2x-3)$$

and the transformation matrix is

$$A \qquad\qquad A^T = A^{-1}$$

$$\begin{bmatrix} c0 & c1 & c2 & c3 & & & & \\ c3 & c2 & c1 & c0 & & & & \\ & & c0 & c1 & c2 & c3 & & \\ & & c3 & c2 & c1 & c0 & & \\ & & & & \ddots & & & \\ & & & & & \ddots & & \\ & & & & c0 & c1 & c2 & c3 \\ & & & & c3 & c2 & c1 & c0 \\ c2 & c3 & & & & & c0 & c1 \\ c1 & c0 & & & & & c3 & c2 \end{bmatrix} \begin{bmatrix} c0 & c3 & & & & & c2 & c1 \\ c1 & c2 & & & & & c3 & c0 \\ c2 & c1 & c0 & c3 & & & & \\ c3 & c0 & c1 & c2 & & & & \\ & & & \ddots & & & & \\ & & \ddots & & & & & \\ & c2 & c1 & c0 & c3 & & & \\ & c3 & c0 & c1 & c2 & & & \\ & & & c2 & c1 & c0 & c3 & \\ & & & c3 & c0 & c1 & c2 & \end{bmatrix}$$

so for orthonormality the conditions are

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$$
$$c_3c_0 + c_2c_1 = 0$$
$$c_1c_0 = 0$$
$$c_3c_1 + c_2c_0 = 0$$

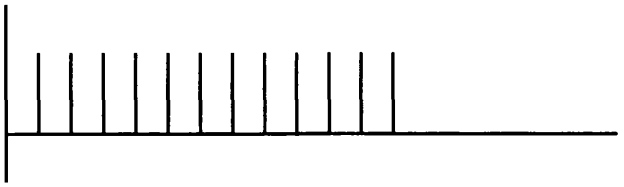
4 equations and 4 variables $\Rightarrow$ no degrees of freedom. Again there are 2 possible solutions:

[1] $c_0 = 1$, $c_1 = c_2 = c_3 = 0$ which yields the superimposed Diracs of the previous solution and is also unsatisfactory.

[2] $c_1 = 1, c_0 = c_2 = c_3 = 0$

$$\begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & & & & & 1 \\ 1 & & & & & & \end{bmatrix}$$

This yields Diracs displaced by unit interval



which is in fact usable, but yields a tautology: f(n) → f(n) which is not attractive for signal analysis nor compression because it can only model the signal as itself!

So $\phi(x)$ and $\psi(x)$ are necessary in their unique forms:

$$\phi(x) = c_0\phi(2x) + c_1\phi(2x-1) + c_2\phi(2x-2) + c_3\phi(2x-3) \quad \text{for the scaling function}$$

and

$$\psi(x) = c_3\phi(2x) - c_2\phi(2x-1) + c_1\phi(2x-2) - c_0\phi(2x-3) \quad \text{for the wavelet.}$$

This is the only form which yields orthonormality in only 2 equations; thus leaving 2 further equations free for conditions such as Strang's or others.

3.2.1 Daubechies' wavelets

The best known wavelets for compression are the Daubechies wavelets. 4 tap wavelets are commonest for fast applications, being simplest to implement; they also offer the advantage of sharp edge representation due to lower impact of local neighbourhood influence.

The Daubechies N wavelet uses the N/2 free conditions for accuracy. These accuracy conditions guarantee polynomial approximation to order N/2 - 1, ie 4 tap wavelets yield polynomial accuracy order 1; 8 tap wavelets yield polynomial accuracy of order 3, and so on. They are ascribed the notation D<N>, eg D4, D6, D8 and so on.

The complexity of the matrix solution means that wavelets of order only up to 6 can be solved analytically. For higher orders iteration is required.

The accuracy condition general form (Cohen, Vial & Daubechies [16]) is

$$\sum_{k=0}^{n-1} (-1)^k k^m c_{n-1-k} = 0, \quad \text{where n is the number of taps and m is the moment number.}$$

For 4 tap wavelets, the 4 conditions are:

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$$
$$c_0c_2 + c_3c_1 = 0$$
$$c_3 - c_2 + c_1 - c_0 = 0$$
$$-c_2 + 2c_1 - 3c_0 = 0$$

The first two equations ensure orthonormality. These conditions alone guarantee perfect reconstruction. The second two equations are the moment conditions for polynomial accuracy.

Complete solution is the tap set

$$c_0 = \frac{1 \pm \sqrt{3}}{4\sqrt{2}}$$
$$c_1 = \frac{3 \pm \sqrt{3}}{4\sqrt{2}}$$
$$c_2 = \frac{3 \mp \sqrt{3}}{4\sqrt{2}}$$
$$c_3 = \frac{1 \mp \sqrt{3}}{4\sqrt{2}}$$

with the particular solution

$$c_0 = \frac{1 + \sqrt{3}}{4\sqrt{2}}$$
$$c_1 = \frac{3 + \sqrt{3}}{4\sqrt{2}}$$
$$c_2 = \frac{3 - \sqrt{3}}{4\sqrt{2}}$$
$$c_3 = \frac{1 - \sqrt{3}}{4\sqrt{2}}$$

known as the D4. This is the minimum phase 4 tap wavelet, described by Daubechies as "the least asymmetric", so chosen from the two possible solutions for its property of introducing less asymmetric distortion in reconstruction (Walden [77]).

So ubiquitous are Daubechies' wavelets that they have come to be known as "The Daubechies wavelets"; the possessive apostrophe dropped to make *Daubechies* an adjective, thus: Daubechies wavelets.

3.3 Implementation

Discrete wavelets owe their wide acceptance to the Fast transform devised by Stéphane Mallat, which works on filters of even length like the Daubechies wavelet. The two are so commonly associated that they are often referred to as a single entity. Such is their interdependence that it is difficult to describe each separately without reference to the other. The previous section on Daubechies wavelets touched on the discrete wavelet transform, since it is the transform which determines the wavelet constraints. Now is the time to describe that transform itself.

3.3.1 Multiresolution decomposition

The technique of multiresolution decomposition comes from Mallat [40] and is based on the transpose self-inverse matrices for forward and backward transformations previously described.

Given a signal vector, y, and the tap matrix, A, a single step in the wavelet transform is represented as $y \rightarrow Ay$.

Repeated quadrature application of the wavelet transformation serves to shuffle and order the coarse and fine elements (the *smooth* and *detail*). At each step, the smooth elements are moved to the top of the vector and the subsequent transformation acts only on these elements; a successive halving of coefficients per step. The process continues until dim(A) = N. By necessity then, y must be of length 2^M, for some positive integer M such that $2^M > N$. These are details of implementation and do not affect the discrete wavelet method. A practical description of Mallat's implementation can be found in Numerical Recipes [51].

Let T_n represent a transformation of multiplication by the wavelet coefficient matrix, which itself intrinsically shuffles the elements. This vector is then ordered into its *smooth* and *detail* components. The terms *smooth* and *detail* refer to the signal scale, which in Fourier terms correspond to low and high frequency terms respectively (Jawerth [32]).

The dyadic transform is demonstrated by the following example in which the signal y has 16 elements.

Successive results in the filter sequence are denoted by **bold** and *italic* fonts.

y1		s1		s1		**s1**		**s1**		*s1*		*s1*
y2		d1		s2		**D1**		**s2**		*D1*		*s2*
y3		s2		s3		**s2**		**s3**		*s2*		*D1*
y4		d2		s4		**D2**		**s4**		*D2*		*D2*
y5		s3		s5		**s3**		**D1**		**D1**		**D1**
y6		d3		s6		**D3**		**D2**		**D2**		**D2**
y7	T_{16}	s4	order	s7	T_8	**s4**	order	**D3**	T_4	**D3**	order	**D3**
y8	→	d4	→	s8	→	**D4**	→	**D4**	→	**D4**	→	**D4**
y9		s5		d1		**d1**		d1		d1		d1
y10		d5		d2		**d2**		d2		d2		d2
y11		s6		d3		**d3**		d3		d3		d3
y12		d6		d4		**d4**		d4		d4		d4
y13		s7		d5		**d5**		d5		d5		d5
y14		d7		d6		**d6**		d6		d6		d6
y15		s8		d7		**d7**		d7		d7		d7
y16		d8		d8		**d8**		d8		d8		d8

Quadrature filter process

The final vector is the discrete wavelet transform of the signal y. It can be compressed by setting to zero so many of those coefficients with small moduli. Wavelet compression is an effective technique because most of the signal's energy is represented by a small number of the transform coefficients (Murenzi [46]).

The inverse is then a simple reversal of the forward transform.

3.4 Appearance

As mentioned in section 3.2, the wavelet taps are not the wavelet itself; instead they are the matrix representation of the filters used in the transform described above. To see the wavelets themselves it is necessary to iterate the dilation equation until the intermediate interval size gives sufficient resolution. By the nature of difference relations, the wavelet can never be drawn exactly since it has no closed form expression. Similarly for the scaling function.

An excellent example of this process comes from Newland [47]. The C code for viewing wavelets was developed from his book. Again, for simplicity, the 4 tap wavelet is illustrated. The method generalizes for higher orders; further examples are given in chapters 4 and 5.

The wavelet is derived by starting with the scaling function dilation equation. This is also known as the auxiliary equation (Waagen [76]) since it is used to generate the wavelet.

$$\phi(x) = c_0\phi(2x) + c_1\phi(2x-1) + c_2\phi(2x-2) + c_3\phi(2x-3)$$

Then each approximation ϕ_j is used to calculate the approximation ϕ_{j+1} of the next step.

$$\phi_{j+1}(x) = c_0\phi_j(2x) + c_1\phi_j(2x-1) + c_2\phi_j(2x-2) + c_3\phi_j(2x-3)$$

At each iteration the x interval decreases by a factor of two. As j increases, the scaling function converges asymptotically to its limiting shape as shown in figure 3.1.

The wavelet is calculated similarly using the same coefficients, reversed and with alternating signs as explained in section 3.2.

$$\psi(x) = c_3\phi(2x) - c_2\phi(2x-1) + c_1\phi(2x-2) - c_0\phi(2x-3)$$

The wavelet so developed is shown in figure 3.2. This recursion method applies for any wavelet; for this illustration the Daubechies 4 wavelet is shown.

An important condition to note is that the admissibility condition holds at each step, with the wavelet area always equalling zero.

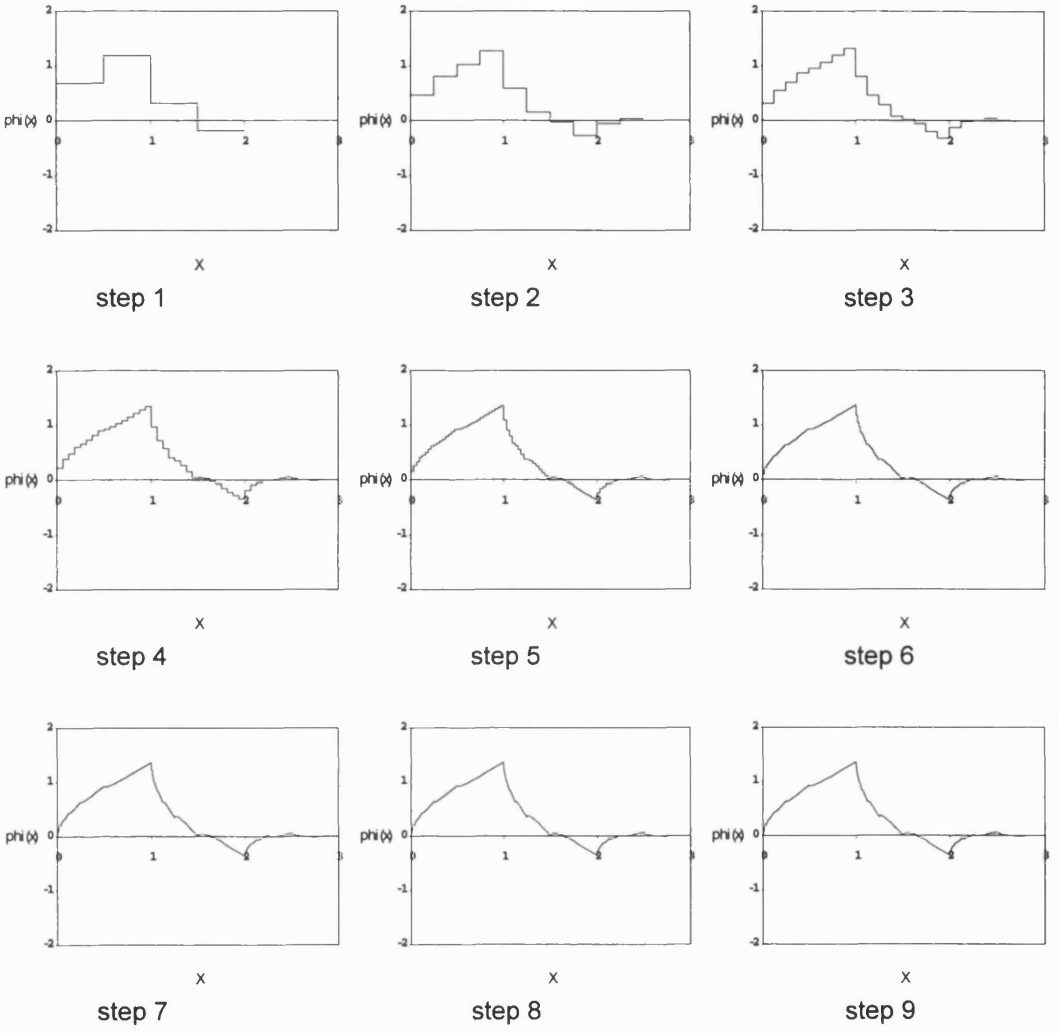


Figure 3.1: recursive development of the Daubechies 4 scaling function

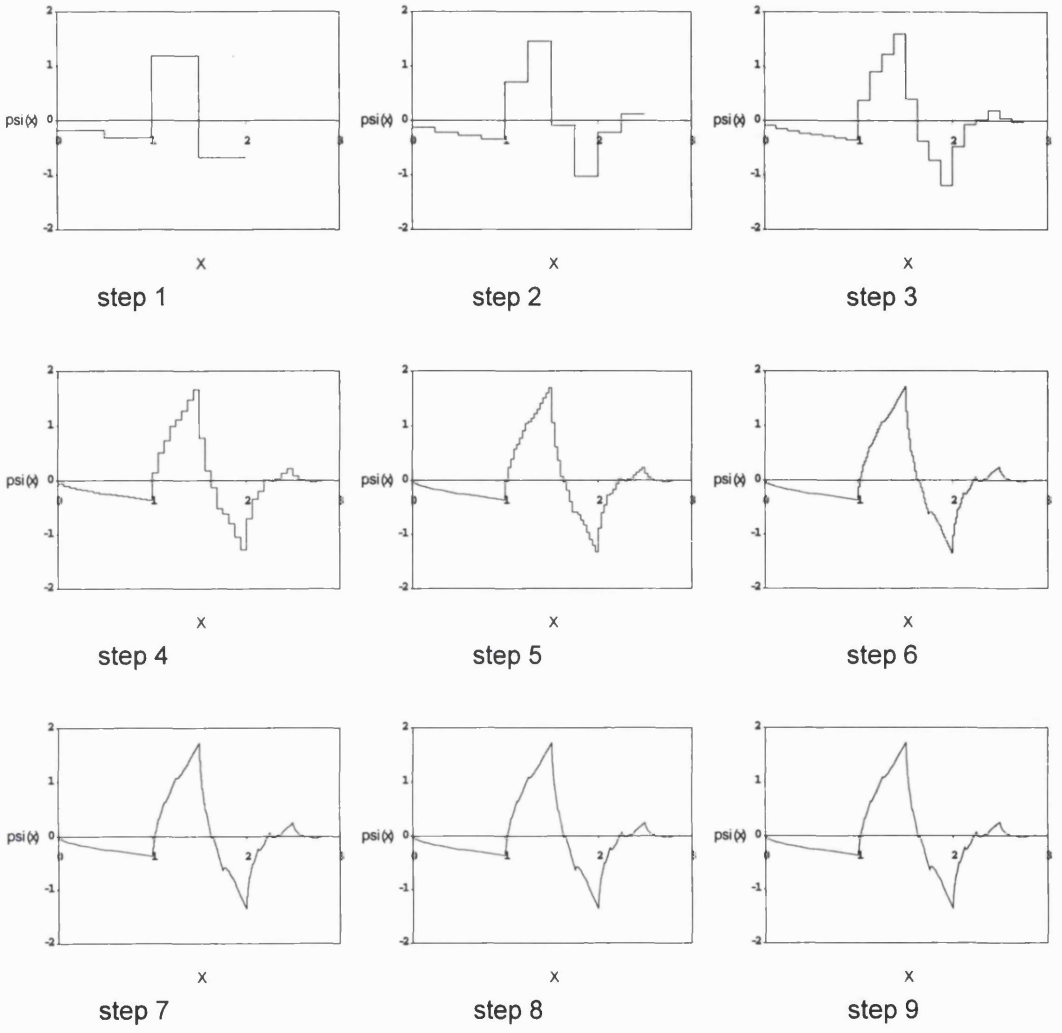


Figure 3.2: recursive development of the Daubechies 4 wavelet

3.5 CONCLUSION

This chapter has introduced the accepted method of implementation of the discrete wavelet transform for the well-known Daubechies wavelet, explaining the interdependence between the wavelet and the transform; and the necessary conditions on orthogonal wavelets in terms of their matrix representation.

What follows in chapters 4 and 5 is the extension of this method to the derivation of generic 4 and 6 tap discrete wavelets maintaining all the properties so far described.

4 DERIVATION OF GENERIC 4 TAP WAVELETS

4.1 Introduction

This chapter looks at the novel technique for designing 4 tap generic wavelets. Based only on orthonormality and constant approximation, the technique exploits the degree of freedom in discrete wavelets when the first moment condition is abandoned. Classically for 4 tap wavelets, the moment conditions provide polynomial accuracy to degree 1, ie constant and linear approximation. For generic derivation the linear moment condition is replaced by a general lock condition f(x,c) = 0. This concept is explained more fully in chapter 5 when generic 6 tap wavelets are described, and a correspondence made with the Daubechies wavelets in terms of this lock condition. The purpose of this chapter is to introduce the concept of generic taps with the simplest example, ie that of 4 taps. Suffice to say here, the replacement of the linear moment by a parameterized function yields a variable solution for the taps. The term "lock condition" is used as being indicative of the fact that if a particular signal does indeed include this function as an explicit feature, then the wavelet transform will "lock" on to it by giving a zero response for the wavelet and a non-zero response for the scaling function.

4.2 Seed equations

The term "seed equation" is used to embody the sense of a starting point for the derivation. The constant moment is included implicitly in these seed equations as demonstrated below.

There are four design equations,

$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$ orthonormality (4.1)

$c_0 + c_1 + c_2 + c_3 = \sqrt{2}$ area conservation (4.2)

$c_3 - \alpha c_2 + \beta c_1 - \gamma c_0 = 0$ lock condition (4.3)

$c_0c_2 + c_1c_3 = 0$ orthogonality (4.4)

A simple rearrangement gives a form which is easier to solve.

Ignoring for the moment the lock condition,

$c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1$ (4.1)

$c_0 + c_1 + c_2 + c_3 = \sqrt{2}$ (4.2)

$c_0c_2 + c_1c_3 = 0$ (4.4)

$(4.1)^2 \Rightarrow$

$(c_0 + c_1 + c_2 + c_3)^2 = 2$

multiply out

$c_0^2 + c_1^2 + c_2^2 + c_3^2 + 2(c_0c_1 + c_0c_2 + c_0c_3 + c_1c_2 + c_1c_3 + c_2c_3) = 2$

substitute in (4.1) and (4.4)

$1 + 2(c_0c_1 + c_0c_3 + c_1c_2 + c_2c_3) = 2$

$\therefore 2(c_0c_1 + c_0c_3 + c_1c_2 + c_2c_3) = 1$

ie $(c_0 + c_2)(c_1 + c_3) = ½$

The only solution is $c_0 + c_2 = c_1 + c_3 = \frac{1}{\sqrt{2}}$

then

$c_3 - c_2 + c_1 - c_0 = 0$

which satisfies the constant moment condition as an additional benefit.

Then the system of design equations becomes:

$c_0 + c_1 + c_2 + c_3 = \sqrt{2}$ (4.2)

$c_3 - \alpha c_2 + \beta c_1 - \gamma c_0 = 0$ (4.3)

$c_0c_2 + c_1c_3 = 0$ (4.4)

$c_3 - c_2 + c_1 - c_0 = 0$ (4.5)

Thus the quadratic form (4.1) has been replaced by a linear equation, facilitating solution of (4.2) to (4.5) by standard row elimination.

4.3 Wavelet solution: a step by step derivation

The system of equations is now

$c_0 + c_1 + c_2 + c_3 = \sqrt{2}$	area conservation	(4.2)
$c_3 - \alpha c_2 + \beta c_1 - \gamma c_0 = 0$	lock condition	(4.3)
$c_0c_2 + c_1c_3 = 0$	orthogonality	(4.4)
$c_3 - c_2 + c_1 - c_0 = 0$	linear moment	(4.5)

Solution for c_0, c_1, c_2 and c_3 proceeds systematically, starting with the elimination of c_3:

(4.5) $\Rightarrow$

$c_3 = c_2 - c_1 + c_0$ (4.6)

Use (4.6) to eliminate c_3 from (4.2), (4.3) and (4.4).

Substitute (4.6) into (4.2)

$c_0 + c_1 + c_2 + (c_2 - c_1 + c_1) = \sqrt{2}$

$2c_0 + 2c_2 = \sqrt{2} \Rightarrow c_0 + c_2 = \frac{1}{\sqrt{2}}$

$$c_2 = \frac{1}{\sqrt{2}} - c_0 \tag{4.7}$$

Substitute (4.6) into (4.3)

$$c_2 - c_1 + c_0 - \alpha c_2 + \beta c_1 - \gamma c_0 = 0$$

$$c_0(1 - \gamma) + c_1(\beta - 1) + c_2(1 - \alpha) = 0 \tag{4.8}$$

substitute (4.6) into (4.4)

$$c_0 c_2 + c_1(c_2 - c_1 + c_0) = 0$$

$$c_0 c_2 + c_1 c_2 - c_1^2 + c_0 c_1 = 0$$

$$c_2(c_1 + c_0) = c_1(c_1 - c_0)$$

$$c_2 = \frac{c_1(c_1 - c_0)}{c_1 + c_0} \tag{4.9}$$

c_3 has been eliminated from (4.2), (4.3) and (4.4) to give (4.7), (4.8) and (4.9).

Now use (4.7) to eliminate c_2 from (4.8) and (4.9).

Substitute (4.7) into (4.8)

$$c_0(1 - \gamma) + c_1(\beta - 1) + (1 - \alpha)(\frac{1}{\sqrt{2}} - c_0) = 0$$

$$c_1(\beta - 1) = (1 - \alpha)(c_0 - \frac{1}{\sqrt{2}}) + c_0(\gamma - 1)$$

$$c_1 = \frac{c_0(1 - \alpha + \gamma - 1) - \frac{1}{\sqrt{2}} + \frac{\alpha}{\sqrt{2}}}{\beta - 1}$$

$$c_1 = \frac{c_0(\gamma - \alpha) + \frac{\alpha - 1}{\sqrt{2}}}{\beta - 1} \tag{4.10}$$

Substitute (4.7) into (4.9)

$$\frac{1}{\sqrt{2}} - c_0 = \frac{c_1(c_1 - c_0)}{c_1 + c_0}$$

$$(\frac{1}{\sqrt{2}} - c_0)(c_1 + c_0) = c_1(c_1 - c_0)$$

$$\frac{c_1}{\sqrt{2}} + \frac{c_0}{\sqrt{2}} - c_0 c_1 - c_0^2 + c_1 c_0 = 0$$

$$c_0^2 - \frac{c_0}{\sqrt{2}} + c_1\left(c_1 - \frac{1}{\sqrt{2}}\right) = 0 \tag{4.11}$$

c_2 has been eliminated from (4.8) and (4.9) to give (4.10) and (4.11).

Now eliminate c_1 by substituting (4.10) into (4.11)

$$c_0^2-\frac{c_0}{\sqrt{2}}+\frac{\left(c_0(\gamma-\alpha)+\frac{\alpha-1}{\sqrt{2}}\right)}{\beta-1}\left(\frac{c_0(\gamma-\alpha)+\frac{\alpha-1}{\sqrt{2}}}{\beta-1}-\frac{1}{\sqrt{2}}\right)=0$$

multiply out

$$c_0^2-\frac{c_0}{\sqrt{2}}+\frac{c_o^2(\gamma-\alpha)^2}{(\beta-1)^2}+\frac{2c_0(\gamma-\alpha)(\alpha-1)}{\sqrt{2}(\beta-1)^2}+\frac{\left(\frac{\alpha-1}{\sqrt{2}}\right)^2}{(\beta-1)^2}-\frac{c_0(\gamma-\alpha)}{\sqrt{2}(\beta-1)}-\frac{\alpha-1}{2(\beta-1)}=0$$

$$c_0^2\left(1+\frac{(\gamma-\alpha)^2}{(\beta-1)^2}\right)+c_0\left(\frac{\sqrt{2}(\gamma-\alpha)(\alpha-1)}{(\beta-1)^2}+\frac{\alpha-\gamma}{\sqrt{2}(\beta-1)}-\frac{1}{\sqrt{2}}\right)+\frac{(\alpha-1)^2}{2(\beta-1)^2}-\frac{\alpha-1}{2(\beta-1)}=0$$

tidy up this quadratic in c_0

$$c_0^2\left\{1+\left(\frac{\gamma-\alpha}{\beta-1}\right)^2\right\}+c_0\left\{\frac{\sqrt{2}(\gamma-\alpha)(\alpha-1)}{(\beta-1)^2}+\frac{1}{\sqrt{2}}\left(\frac{\alpha-\gamma}{\beta-1}-1\right)\right\}+\frac{\alpha-1}{2(\beta-1)}\left(\frac{\alpha-1}{\beta-1}-1\right)=0$$

note that the last term $\frac{\alpha-1}{\beta-1}-1\equiv\frac{\alpha-1-(\beta-1)}{\beta-1}\equiv\frac{\alpha-\beta}{\beta-1}$

then the quadratic becomes

$$c_0^2\left\{1+\left(\frac{\gamma-\alpha}{\beta-1}\right)^2\right\}+c_0\left\{\frac{\sqrt{2}(\gamma-\alpha)(\alpha-1)}{(\beta-1)^2}+\frac{1}{\sqrt{2}}\left(\frac{\alpha-\gamma}{\beta-1}-1\right)\right\}+\frac{\alpha-1}{2(\beta-1)}\left(\frac{\alpha-\beta}{\beta-1}\right)=0$$

solve by the quadratic formula:

$$c_0=\frac{\frac{\sqrt{2}(\gamma-\alpha)(1-\alpha)}{(\beta-1)^2}+\frac{1}{\sqrt{2}}\left(1-\frac{\alpha-\gamma}{\beta-1}\right)\pm\sqrt{\left\{\frac{\sqrt{2}(\gamma-\alpha)(\alpha-1)}{(\beta-1)^2}+\frac{1}{\sqrt{2}}\left(\frac{\alpha-\gamma}{\beta-1}-1\right)\right\}^2-4\left\{1+\left(\frac{\alpha-\gamma}{\beta-1}\right)^2\right\}\left\{\frac{\alpha-1}{2(\beta-1)}\left(\frac{\alpha-\beta}{\beta-1}\right)\right\}}}{2\left(1+\left(\frac{\gamma-\alpha}{\beta-1}\right)^2\right)}$$

use a common denominator $(\beta-1)^2$ throughout to aid cancellation:

$$c_0=\frac{\frac{2(\gamma-\alpha)(1-\alpha)+(\beta-1)^2-(\beta-1)(\alpha-\gamma)}{\sqrt{2}(\beta-1)^2}\pm\sqrt{\left\{\frac{2(\gamma-\alpha)(\alpha-1)+(\beta-1)(\alpha-\gamma)-(\beta-1)^2}{\sqrt{2}(\beta-1)^2}\right\}^2-4\left\{\frac{(\beta-1)^2+(\gamma-\alpha)^2}{(\beta-1)^2}\right\}\left\{\frac{\alpha-1}{2(\beta-1)}\left(\frac{\alpha-\beta}{\beta-1}\right)\right\}}}{2\left(\frac{(\beta-1)^2+(\gamma-\alpha)^2}{(\beta-1)^2}\right)}$$

take out $\frac{1}{(\beta-1)^4}$ from the square root term:

$$c_0 = \frac{\frac{2(\gamma-\alpha)(1-\alpha)+(\beta-1)^2-(\beta-1)(\alpha-\gamma)}{\sqrt{2}(\beta-1)^2} \pm \frac{1}{(\beta-1)^2}\sqrt{\left\{\sqrt{2}(\gamma-\alpha)(\alpha-1)+\frac{(\beta-1)(\alpha-\gamma)}{\sqrt{2}}-\frac{(\beta-1)^2}{\sqrt{2}}\right\}^2-\frac{4}{2}\left\{(\beta-1)^2+(\gamma-\alpha)^2\right\}(\alpha-1)(\alpha-\beta)}}{2\left(\frac{(\beta-1)^2+(\gamma-\alpha)^2}{(\beta-1)^2}\right)}$$

The $(\beta-1)^2$ term cancels from the denominator of each term in the numerator and denominator of the whole expression

$$c_0 = \frac{2(\gamma-\alpha)(1-\alpha)+(\beta-1)^2-(\beta-1)(\alpha-\gamma) \pm \sqrt{2}\sqrt{\left\{\sqrt{2}(\gamma-\alpha)(\alpha-1)+\frac{(\beta-1)(\alpha-\gamma)}{\sqrt{2}}-\frac{(\beta-1)^2}{\sqrt{2}}\right\}^2-2\left\{(\beta-1)^2+(\gamma-\alpha)^2\right\}(\alpha-1)(\alpha-\beta)}}{2\sqrt{2}\left((\beta-1)^2+(\gamma-\alpha)^2\right)}$$

Now, to simplify further progress use a shorthand notation and express c_0 as $\frac{L+S}{D}$ where L is the left hand term; S is the square root term; and D is the denominator, thus:

$$c_0 = \frac{L+S}{D}$$

where $D = 2\sqrt{2}\left((\beta-1)^2+(\gamma-\alpha)^2\right)$

and $L = 2(\gamma - \alpha)(1 - \alpha) + (\beta - 1)^2 - (\beta - 1)(\alpha - \gamma)$

$= 2\gamma - 2\alpha\gamma - 2\alpha + 2\alpha^2 + \beta^2 - 2\beta + 1 - \alpha\beta + \beta\gamma + \alpha - \gamma$

$L = 1 - \alpha + 2\alpha^2 - 2\beta - \alpha\beta + \beta^2 + \gamma - 2\alpha\gamma + \beta\gamma$

The square root term is $S = \sqrt{2}\sqrt{\left\{\sqrt{2}(\gamma-\alpha)(\alpha-1)+\frac{(\beta-1)(\alpha-\gamma)}{\sqrt{2}}-\frac{(\beta-1)^2}{\sqrt{2}}\right\}^2-2\left\{(\beta-1)^2+(\gamma-\alpha)^2\right\}(\alpha-1)(\alpha-\beta)}$

take the $\sqrt{2}$ inside the square root and multiply out.

$S = \{4(\gamma-\alpha)^2(\alpha-1)^2 + 4(\gamma-\alpha)(\alpha-1)(\beta-1)(\alpha-\gamma) - 4(\gamma-\alpha)(\alpha-1)(\beta-1)^2 + (\beta-1)^2(\alpha-\gamma)^2 - 2(\beta-1)^3(\alpha-\gamma)$

$+ (\beta-1)^4 - 4(\beta-1)^2(\alpha-1)(\alpha-\beta) - 4(\gamma-\alpha)^2(\alpha-1)(\alpha-\beta)\}^{1/2}$

collect the $(1-\beta)^2$ terms inside the square root

$S = \{(1-\beta)^2[-4(\gamma-\alpha)(\alpha-1) + (\gamma-\alpha)^2 + 2(1-\beta)(\alpha-\gamma) + (\beta-1)^2 - 4(\alpha-1)(\alpha-\beta)] + 4(\gamma-\alpha)^2(\alpha-1)^2$
$- 4(\gamma-\alpha)^2(\alpha-1)(\beta-1) - 4(\gamma-\alpha)^2(\alpha-1)(\alpha-\beta)\}^{1/2}$

$S = \{(1-\beta)^2[4(\alpha-1)(\alpha-\gamma+\beta-\alpha) + (\alpha-\gamma)(\alpha-\gamma+2-2\beta) + (\beta-1)^2] + 4(\gamma-\alpha)^2(\alpha-1)(\alpha-1-\beta+1+\beta-\alpha)\}^{1/2}$

Now take the $(1-\beta)^2$ outside the square root and multiply out:

$$S = (1-\beta)\sqrt{-4\alpha\gamma + 4\alpha\beta + 4\gamma - 4\beta + \alpha^2 - \alpha\gamma + 2\alpha - 2\alpha\beta - \alpha\gamma + \gamma^2 - 2\gamma + 2\beta\gamma + \beta^2 - 2\beta + 1}$$

$$S = (1-\beta)\sqrt{1 + 2\alpha + \alpha^2 - 6\beta + 2\alpha\beta + \beta^2 + 2\gamma - 6\alpha\gamma + 2\beta\gamma + \gamma^2}$$

Using $c_0 = \frac{L+S}{D}$, c_0 now becomes:

result:

$$c_0 = \frac{1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma \pm (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

Given c_0, c_2 is now derived using (vi), $c_2 = \frac{1}{\sqrt{2}} - c_0$

Then $$c_2 = \frac{1}{\sqrt{2}} - \left(\frac{1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma \pm (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)} \right)$$

use a common denominator

$$c_2 = \frac{2\left((1-\beta)^2+(\gamma-\alpha)^2\right) - 1+\alpha-2\alpha^2+2\beta+\alpha\beta-\beta^2-\gamma+2\alpha\gamma-\beta\gamma \mp (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_2 = \frac{2\left(1-2\beta+\beta^2\right)+2\left(\gamma^2-2\alpha\gamma+\alpha^2\right) - 1+\alpha-2\alpha^2+2\beta+\alpha\beta-\beta^2-\gamma+2\alpha\gamma-\beta\gamma \mp (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

result:

$$c_2 = \frac{1+\alpha-2\beta+\alpha\beta+\beta^2-\gamma-2\alpha\gamma-\beta\gamma+2\gamma^2 \mp (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

Also, given c_0, the expression for c_1 can be found using (4.10):

$$c_1 = \frac{c_0(\gamma-\alpha)+\frac{\alpha-1}{\sqrt{2}}}{\beta-1}$$

Again, to reduce complexity, adopt the shorthand $\sqrt{\cdots}$ for the square root term in c_0

then

$$c_1 = \frac{\left\{\frac{1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma \pm (1-\beta)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}\right\}(\gamma-\alpha)+\frac{\alpha-1}{\sqrt{2}}}{\beta-1}$$

$$c_1 = \frac{\left(\frac{\gamma-\alpha}{\beta-1}\right)\left(1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma\right) \pm (\alpha-\gamma)\sqrt{\cdots}+\frac{2(\alpha-1)}{\beta-1}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_1 = \frac{\frac{1}{\beta-1}\left\{\begin{matrix}\gamma-\alpha\gamma+2\alpha^2\gamma-2\beta\gamma-\alpha\beta\chi+\beta^2\gamma+\gamma^2-2\alpha\gamma^2+\beta\gamma^2-\alpha+\alpha^2-2\alpha^3+2\alpha\beta+\alpha^2\beta-\alpha\beta^2-\alpha\gamma+\\ 2\alpha^2\gamma-\alpha\beta\gamma+2(\alpha-1)\left(1-2\beta+\beta^2+\gamma^2-2\alpha\gamma+\alpha^2\right)\end{matrix}\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_1 = \frac{\frac{1}{\beta-1}\left\{\begin{matrix}\gamma-\alpha\gamma+2\alpha^2\gamma-2\beta\gamma-\alpha\beta\chi+\beta^2\gamma+\gamma^2-2\alpha\gamma^2+\beta\gamma^2-\alpha+\alpha^2-2\alpha^3+2\alpha\beta+\alpha^2\beta-\alpha\beta^2-\alpha\gamma+\\ 2\alpha^2\gamma-\alpha\beta\gamma+2\alpha-4\alpha\beta+2\alpha\beta^2+2\alpha\gamma^2-4\alpha^2\gamma+2\alpha^3-2+4\beta-2\beta^2-2\gamma^2+4\alpha\gamma-2\alpha^2\end{matrix}\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

collect terms together

$$c_1 = \frac{\frac{1}{\beta-1}\left\{-2+\alpha-\alpha^2+\overline{4\beta}-\overline{2\alpha\beta}+\gamma+2\alpha\gamma-\gamma^2-\overline{2\beta\gamma}+\alpha^2\beta-2\beta^2+\alpha\beta^2-2\alpha\beta\gamma+\beta^2\gamma+\beta\gamma^2\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

The overlined terms are now split up to facilitate a convenient factorization in the following step

$$c_1 = \frac{\frac{1}{\beta-1}\left\{-2+\alpha-\alpha^2+\overline{(2\beta+2\beta)}+\overline{(-\alpha\beta-\alpha\beta)}+\gamma+2\alpha\gamma-\gamma^2+\overline{(-\beta\gamma-\beta\gamma)}+\alpha^2\beta-2\beta^2+\alpha\beta^2-2\alpha\beta\gamma+\beta^2\gamma+\beta\gamma^2\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

Now rearrange the parenthesized terms:

$$c_1 = \frac{\frac{1}{\beta-1}\left\{\left(-2+\alpha-\alpha^2+2\beta-\alpha\beta+\gamma+2\alpha\gamma-\beta\gamma-\gamma^2\right)+\left(2\beta-\alpha\beta+\alpha^2\beta-2\beta^2-\beta\gamma+\alpha\beta^2-2\alpha\beta\gamma+\beta^2\gamma+\beta\gamma^2\right)\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

The parenthesized terms in braces can now be factorized with (β-1):

$$c_1 = \frac{\frac{1}{\beta-1}(\beta-1)\left\{2-\alpha+\alpha^2-2\beta+\alpha\beta-\gamma-2\alpha\gamma+\beta\gamma+\gamma^2\right\}\pm(\alpha-\gamma)\sqrt{\cdots}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

result:

$$c_1 = \frac{2-\alpha+\alpha^2-2\beta+\alpha\beta-\gamma-2\alpha\gamma+\beta\gamma+\gamma^2\pm(\alpha-\gamma)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

Having found c_0, c_1 and c_2, these solutions can now be substituted into (v) to give c_3:

$c_3 = c_2 - c_1 + c_0$

$$c_3 = \frac{\left\{\begin{matrix}\left(1+\alpha-2\beta+\alpha\beta+\beta^2-\gamma-2\alpha\gamma-\beta\gamma+2\gamma^2\right)-\left(2-\alpha+\alpha^2-2\beta+\alpha\beta-\gamma-2\alpha\gamma+\beta\gamma+\gamma^2\right)\\ +\left(1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma\right)+\left[\mp(1-\beta)\mp(\alpha-\gamma)\pm(1-\beta)\right]\sqrt{\cdots}\end{matrix}\right\}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

result:

$$c_3 = \frac{\alpha+\alpha^2-2\beta-\alpha\beta+2\beta^2+\gamma-2\alpha\gamma-\beta\gamma+\gamma^2 \mp (\alpha-\gamma)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

Collecting these terms together we have the final solution for the four taps:

$$c_0 = \frac{1-\alpha+2\alpha^2-2\beta-\alpha\beta+\beta^2+\gamma-2\alpha\gamma+\beta\gamma \pm (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_1 = \frac{2-\alpha+\alpha^2-2\beta+\alpha\beta-\gamma-2\alpha\gamma+\beta\gamma+\gamma^2 \pm (\alpha-\gamma)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_2 = \frac{1+\alpha-2\beta+\alpha\beta+\beta^2-\gamma-2\alpha\gamma-\beta\gamma+2\gamma^2 \mp (1-\beta)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

$$c_3 = \frac{\alpha+\alpha^2-2\beta-\alpha\beta+2\beta^2+\gamma-2\alpha\gamma-\beta\gamma+\gamma^2 \mp (\alpha-\gamma)\sqrt{1+2\alpha+\alpha^2-6\beta+2\alpha\beta+\beta^2+2\gamma-6\alpha\gamma+2\beta\gamma+\gamma^2}}{2\sqrt{2}\left((1-\beta)^2+(\gamma-\alpha)^2\right)}$$

4.4 Verification

These generic tap solutions can be validated by setting $\alpha = k$, $\beta = 2k$, $\gamma = 3k$ and letting $k \to \infty$ to yield the Daubechies taps.

For large k the second order terms dominate:

$$c_3 \sim \frac{k^2-2k^2+8k^2-6k^2-6k^2+9k^2 \mp (-2k)\sqrt{k^2+4k^2+4k^2-18k^2+12k^2+9k^2}}{2\sqrt{2}\left(4k^2+4k^2\right)} = \frac{4\pm 4\sqrt{3}}{16\sqrt{2}} = \frac{1\pm\sqrt{3}}{4\sqrt{2}}$$

Thus the Daubechies 4 c_3 tap is delivered. In similar fashion the other Daubechies taps are produced.

4.5 Appearance

Given the generic solution, the choice of f(x, c) can be made post derivation. As an example, figure 4.1 shows the 4 tap wavelet class using $f(x, c) = \exp(-x^2/c)$.

Thus

$\alpha = \exp(-1^2/c)$ ie $\alpha = e^{-1/c}$
$\beta = \exp(-2^2/c)$ ie $\beta = e^{-4/c}$
$\gamma = \exp(-3^2/c)$ ie $\gamma = e^{-9/c}$

Only a sample of wavelets is shown for an illustrative selection of discrete c values. The full set is a continuous class with solutions for all values of c. This sample, however, is sufficient to show how the wavelet varies with c; starting with a pseudo Haar function for small c and ranging through the Daubechies 4 and on to the Christmas tree wavelet at c = 100. Note the small c solution is not the actual Haar since it has 4 taps and support 3. Despite this, it does behave like

the Haar because its none zero interval has support 1 and as c becomes small, c_0 and $c_1 \rightarrow \frac{1}{\sqrt{2}}$; c_2 and $c_3 \rightarrow 0$.

The Daubechies 4 wavelet is included at c = 5.83967. There is no significance in this number; for a different lock function f(x, c), the Daubechies 4 wavelet would occur at a different c value.

4.6 Conclusion

This chapter has shown the derivation of 4 tap abc wavelets. It has demonstrated the general technique of generic wavelet derivation in the simplest case, ie 4 taps. This stands as a conceptual foundation extendible to cases of higher order. Chapter 5 goes on to develop this theory, deriving generic forms for the more complicated 6 tap case. The use of a parameterized lock condition produces a continuous range of wavelets as opposed to the Daubechies wavelet which is one specific solution of the seed equations. Thus the abc wavelets are a superset of Daubechies wavelets.

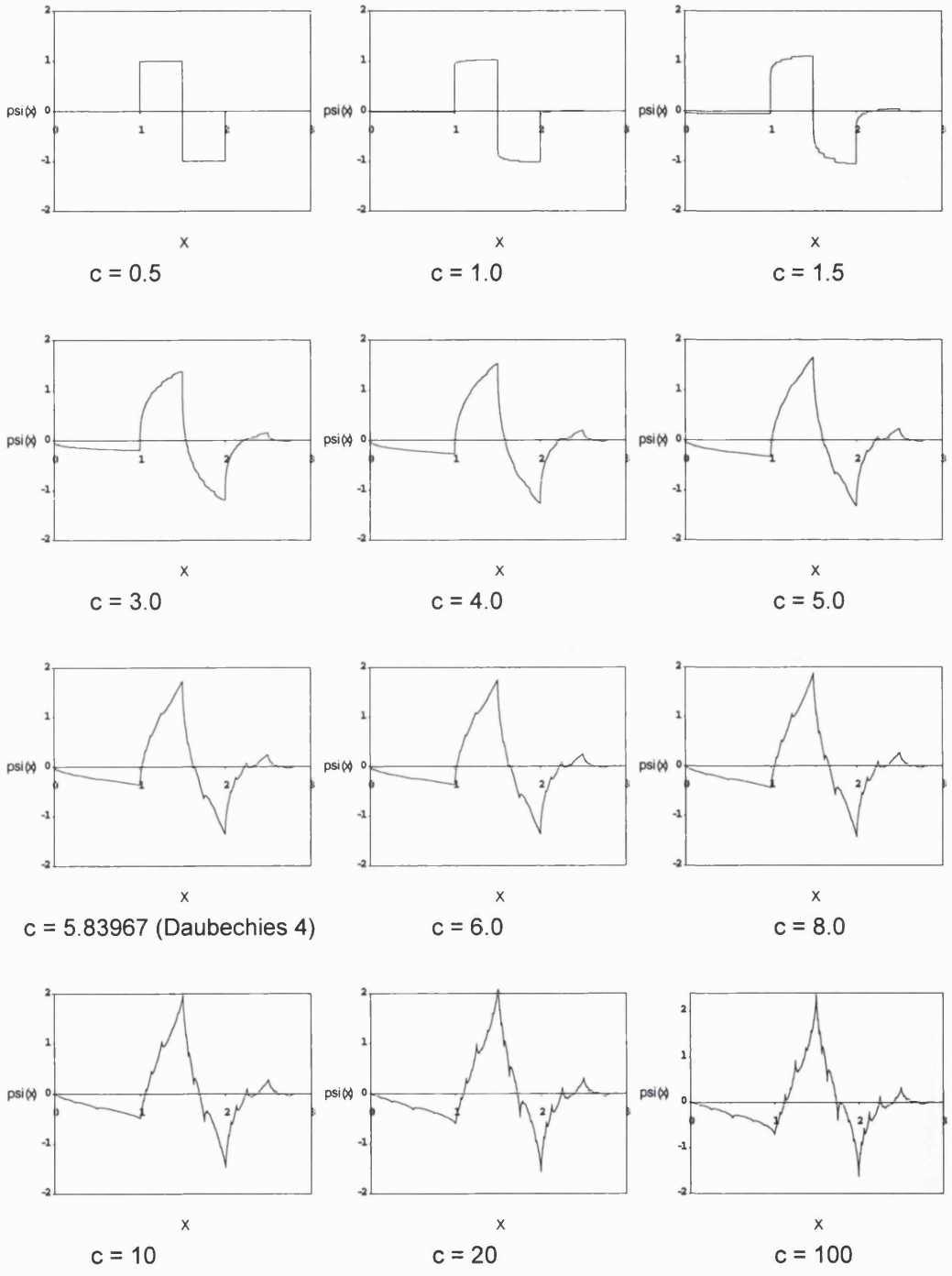


Figure 4.1: wavelets of the $\exp(-x^2/c)$ class

5 DERIVATION OF GENERIC 6 TAP WAVELETS

5.1 Introduction

In this chapter the concept of a dynamic wavelet class shall be further examined, taking the wavelet order N up to 6 taps, where the support is longer and the number of free conditions increases by 1. As in the case of the 4 tap class, a discrete system requires N/2 equations to satisfy conditions of orthonormality and orthogonality, leaving N/2 free equations to manipulate at choice. In this case, the first two moment conditions are used, leaving a third condition specified only in terms of the general lock function, f(x,c) = 0, as a surrogate for the second moment conventionally used in Daubechies 6.

5.2 Seed equations

The 6 tap derivation proceeds similarly to that for the 4 tap. Again, the wavelet described here deviates from Daubechies by introducing a generic form for the highest moment. The seed equations are

$$c_0^2 + c_1^2 + c_2^2 + c_3^2 + c_4^2 + c_5^2 = 1 \qquad \text{orthonormality} \qquad (5.1)$$

$$c_0c_2 + c_1c_3 + c_2c_4 + c_3c_5 = 0 \qquad \text{orthogonality} \qquad (5.2)$$

$$c_0c_4 + c_1c_5 = 0 \qquad \text{orthogonality} \qquad (5.3)$$

$$c_5 - c_4 + c_3 - c_2 + c_1 - c_0 = 0 \qquad \text{constant moment} \qquad (5.4)$$

$$-c_4 + 2c_3 - 3c_2 + 4c_1 - 5c_0 = 0 \qquad \text{linear moment} \qquad (5.5)$$

$$-\alpha c_4 + \beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0 = 0 \qquad \text{generic lock} \qquad (5.6)$$

The solution of these equations is greatly simplified by rearranging (5.1) into a linear form:

$$\begin{aligned}(5.1) \Rightarrow c_0^2 + c_1^2 + c_2^2 + c_3^2 + c_4^2 + c_5^2 = {} & (c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 \\ & - 2(c_0c_1 + c_0c_2 + c_0c_3 + c_0c_4 + c_0c_5 \\ & + c_1c_2 + c_1c_3 + c_1c_4 + c_1c_5 \\ & + c_2c_3 + c_2c_4 + c_2c_5 \\ & + c_3c_4 + c_3c_5 \\ & + c_4c_5)\end{aligned}$$

The left hand side = 1 by (5.1), so

$$\begin{aligned}(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + 2(& c_0c_1 + c_0c_2 + c_0c_3 + c_0c_4 + c_0c_5 \\ & + c_1c_2 + c_1c_3 + c_1c_4 + c_1c_5 \\ & + c_2c_3 + c_2c_4 + c_2c_5 \\ & + c_3c_4 + c_3c_5 \\ & + c_4c_5)\end{aligned}$$

Rearrange the right hand side,

$$\begin{aligned}(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 & + 2(c_0c_1 + c_0c_3 + c_0c_5 + c_1c_2 + c_1c_4 + c_2c_3 + c_2c_5 + c_3c_4 + c_4c_5) \\ & + 2(c_0c_2 + c_1c_3 + c_2c_4 + c_3c_5) \qquad \rightarrow 0 \text{ by (5.2)} \\ & + 2(c_0c_4 + c_1c_5) \qquad \rightarrow 0 \text{ by (5.3)}\end{aligned}$$

Factorize the right hand side

$$(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + 2(c_0 + c_2 + c_4)(c_1 + c_3 + c_5)$$

$$(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + (c_0 + c_2 + c_4)(c_1 + c_3 + c_5) + (c_0 + c_2 + c_4)(c_1 + c_3 + c_5)$$

Now, by (5.4),

$$c_0 + c_2 + c_4 = c_1 + c_3 + c_5$$

$$\therefore (c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + (c_0 + c_2 + c_4)(c_0 + c_2 + c_4) + (c_1 + c_3 + c_5)(c_1 + c_3 + c_5)$$

$$(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + (c_0 + c_2 + c_4)^2 + (c_1 + c_3 + c_5)^2$$

Multiply out

$$\begin{aligned}(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 &+ (c_0^2 + c_1^2 + c_2^2 + c_3^2 + c_4^2 + c_5^2) \\ &+ 2(c_0c_2 + c_1c_3 + c_2c_4 + c_3c_5) \quad \rightarrow 0 \text{ by (5.2)} \\ &+ 2(c_0c_4 + c_1c_5) \quad \rightarrow 0 \text{ by (5.3)}\end{aligned}$$

Then by (5.1)

$$(c_0 + c_1 + c_2 + c_3 + c_4 + c_5)^2 = 1 + 1$$

$$\Rightarrow c_0 + c_1 + c_2 + c_3 + c_4 + c_5 = \sqrt{2} \qquad (5.7)$$

Now (5.1) can be expressed in this linear form, thus simplifying the system (5.1) to (5.6). Area conservation (5.7) is the linear equivalent of $c_0^2 + c_1^2 + c_2^2 + c_3^2 + c_4^2 + c_5^2 = 1$. The generic lock (5.6) replaces the usual quadratic moment. The generic form allows the use of any function $f(x,c) = \alpha, \beta, \gamma, \delta, \varepsilon$ for $x = 1, 2, 3, 4, 5$ respectively. Daubechies 6 uses $f(x) = x^2$ (no c dependence).

$c_0c_2 + c_1c_3 + c_2c_4 + c_3c_5 = 0$	orthogonality	(5.2)
$c_0c_4 + c_1c_5 = 0$	orthogonality	(5.3)
$c_5 - c_4 + c_3 - c_2 + c_1 - c_0 = 0$	constant moment	(5.4)
$-c_4 + 2c_3 - 3c_2 + 4c_1 - 5c_0 = 0$	linear moment	(5.5)
$-\alpha c_4 + \beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0 = 0$	generic lock	(5.6)
$c_0 + c_1 + c_2 + c_3 + c_4 + c_5 = \sqrt{2}$	area conservation	(5.7)

5.3 Wavelet solution: a step by step derivation

Given this set of predominately linear seed equations, the generic taps can now be derived by standard row elimination. Solution for c_0, c_1, c_2 c_3, c_4 and c_5 proceeds systematically, starting with the elimination of c_5. The process is straightforward but lengthy. To help structure the process, successive elimination of each tap is shown underlined, as waypoints to guide the algebra. At the end of each step the set of equations is shown together, each a simplification of the preceding step, with one less tap.

$(5.4) \Rightarrow c_5 = c_4 - c_3 + c_2 - c_1 + c_0$ (5.8)

Use (5.8) to eliminate c_5 from (5.2), (5.3) and (5.7).

Substitute (5.8) into (5.2)

$c_0c_2 + c_1c_3 + c_2c_4 + c_3(c_4 - c_3 + c_2 - c_1 + c_0) = 0$

$c_0c_2 + c_2c_4 + c_3c_4 - c_3^2 + c_2c_3 - c_0c_3 = 0$

$c_4(c_2 + c_3) = c_3^2 - c_2c_3 - c_0c_3 - c_0c_2$

$$c_4 = \frac{c_3(c_3 - c_2 - c_0) - c_0c_2}{c_2 + c_3} \quad (5.9)$$

Substitute (5.8) into (5.3)

$c_0c_4 + c_1(c_4 - c_3 + c_2 - c_1 + c_0) = 0$

$c_4(c_0 + c_1) = c_1(c_3 - c_2 + c_1 - c_0)$

$$c_4 = \frac{c_1(c_3 - c_2 + c_1 - c_0)}{c_0 + c_1} \quad (5.10)$$

Substitute (5.8) into (5.7)

$c_0 + c_1 + c_2 + c_3 + c_4 + (c_4 - c_3 + c_2 - c_1 + c_0) = \sqrt{2}$

$2c_0 + 2c_2 + 2c_4 = \sqrt{2}$

$$c_0 + c_2 + c_4 = \frac{1}{\sqrt{2}} \quad (5.11)$$

There are now 5 equations in c_0, c_1, c_2, c_3, c_4.

$$c_4 = \frac{c_3(c_3 - c_2 - c_0) - c_0c_2}{c_2 + c_3} \quad (5.9)$$

$$c_4 = \frac{c_1(c_3 - c_2 + c_1 - c_0)}{c_0 + c_1} \quad (5.10)$$

$$c_4 = \frac{1}{\sqrt{2}} - c_0 - c_2 \quad (5.11)$$

$c_4 = 2c_3 - 3c_2 + 4c_1 - 5c_0$ (5.12) (from (5.5))

$$c_4 = \frac{\beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0}{\alpha} \quad (5.13) \text{ (from (5.6))}$$

Now use (5.12) to eliminate c_4 from (5.9), (5.10), (5.11) and (5.13).

Substitute (5.12) into (5.11)

$$2c_3 - 3c_2 + 4c_1 - 5c_0 = \frac{1}{\sqrt{2}} - c_0 - c_2$$

$$2c_3 = \frac{1}{\sqrt{2}} + 4c_0 - 4c_1 + 2c_2$$

$$c_3 = \frac{1}{2\sqrt{2}} + 2c_0 - 2c_1 + c_2 \qquad (5.14)$$

Substitute (5.12) into (5.9)

$$2c_3 - 3c_2 + 4c_1 - 5c_0 = \frac{c_3(c_3 - c_2 - c_0) - c_0c_2}{c_2 + c_3}$$

$$(2c_3 - 3c_2 + 4c_1 - 5c_0)(c_2 + c_3) = c_3(c_3 - c_2 - c_0) - c_0c_2$$

$$c_3(2c_3 - 3c_2 + 4c_1 - 5c_0 - c_3 + c_2 + c_0) = c_2(-c_0 - 2c_3 + 3c_2 - 4c_1 + 5c_0)$$

$$c_3(-4c_0 + 4c_1 + c_3) = c_2(-c_0 + 3c_2 - 4c_1 + 5c_0)$$

$$c_3^2 + c_3(-4c_0 + 4c_1) = c_2(4c_0 + 3c_2 - 4c_1) \qquad (5.15)$$

Substitute (5.12) into (5.10)

$$2c_3 - 3c_2 + 4c_1 - 5c_0 = \frac{c_1(c_3 - c_2 + c_1 - c_0)}{c_0 + c_1}$$

$$(2c_3 - 3c_2 + 4c_1 - 5c_0)(c_0 + c_1) = c_1(c_3 - c_2 + c_1 - c_0)$$

$$c_3(2c_0 + 2c_1 - c_1) = c_1(-c_2 + c_1 - c_0 + 3c_2 - 4c_1 + 5c_0) + c_0(3c_2 - 4c_1 + 5c_0)$$

$$c_3(2c_0 + c_1) = c_1(2c_2 - 3c_1 + 4c_0) + c_0(3c_2 - 4c_1 + 5c_0)$$

$$c_3(2c_0 + c_1) = c_1(2c_2 - 3c_1) + c_0(3c_2 + 5c_0)$$

$$c_3 = \frac{c_1(-3c_1 + 2c_2) + c_0(5c_0 + 3c_2)}{2c_0 + c_1} \qquad (5.16)$$

Substitute (5.12) into (5.13)

$$2c_3 - 3c_2 + 4c_1 - 5c_0 = \frac{\beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0}{\alpha}$$

$$2\alpha c_3 - 3\alpha c_2 + 4\alpha c_1 - 5\alpha c_0 = \beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0$$

$$(2\alpha - \beta)c_3 = (3\alpha - \gamma)c_2 + (\delta - 4\alpha)c_1 + (5\alpha - \varepsilon)c_0$$

$$c_3 = \frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta} \tag{5.17}$$

There are now 4 equations in c_0, c_1, c_2, c_3.

$$c_3 = \frac{1}{2\sqrt{2}} + 2c_0 - 2c_1 + c_2 \tag{5.14}$$

$$c_3^2 + c_3(-4c_0 + 4c_1) = c_2(4c_0 + 3c_2 - 4c_1) \tag{5.15}$$

$$c_3 = \frac{c_1(-3c_1+2c_2)+c_0(5c_0+3c_2)}{2c_0+c_1} \tag{5.16}$$

$$c_3 = \frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta} \tag{5.17}$$

Now use (5.17) to eliminate c_3 from (5.14), (5.15), (5.16).

Substitute (5.17) into (5.14)

$$\frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta} = \frac{1}{2\sqrt{2}} + 2c_0 - 2c_1 + c_2$$

$$(3\alpha - \gamma)c_2 + (\delta - 4\alpha)c_1 + (5\alpha - \varepsilon)c_0 = \frac{2\alpha-\beta}{2\sqrt{2}} + 2(2\alpha - \beta)c_0 - 2(2\alpha - \beta)c_1 + (2\alpha - \beta)c_2$$

$$c_2(3\alpha - \gamma - 2\alpha + \beta) = \frac{2\alpha-\beta}{2\sqrt{2}} + c_0(4\alpha - 2\beta - 5\alpha + \varepsilon) + c_1(2\beta - 4\alpha - \delta + 4\alpha)$$

$$c_2(\alpha + \beta - \gamma) = \frac{2\alpha-\beta}{2\sqrt{2}} + c_0(-\alpha - 2\beta + \varepsilon) + c_1(2\beta - \delta) \tag{5.18}$$

Substitute (5.18) into (5.15)

$$\left(\frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta}\right)^2 + \left(\frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta}\right)(-4c_0 + 4c_1) =$$

$$c_2(4c_0 + 3c_2 - 4c_1)$$

Multiply out

$$\frac{(3\alpha-\gamma)^2}{(2\alpha-\beta)^2}c_2^2+\frac{(\delta-4\alpha)^2}{(2\alpha-\beta)^2}c_1^2+\frac{(5\alpha-\varepsilon)^2}{(2\alpha-\beta)^2}c_0^2+2\frac{(3\alpha-\gamma)(\delta-4\alpha)}{(2\alpha-\beta)^2}c_1c_2+2\frac{(3\alpha-\gamma)(5\alpha-\varepsilon)}{(2\alpha-\beta)^2}c_0c_2$$

$$+2\frac{(\delta-4\alpha)(5\alpha-\varepsilon)}{(2\alpha-\beta)^2}c_0c_1$$

$$-4\frac{3\alpha-\gamma}{2\alpha-\beta}c_0c_2-4\frac{\delta-4\alpha}{2\alpha-\beta}c_0c_1-4\frac{5\alpha-\varepsilon}{2\alpha-\beta}c_0^2$$

$$+4\frac{3\alpha-\gamma}{2\alpha-\beta}c_1c_2+4\frac{\delta-4\alpha}{2\alpha-\beta}c_1^2+4\frac{5\alpha-\varepsilon}{2\alpha-\beta}c_0c_1=4c_0c_2+3c_2^2-4c_1c_2$$

Rearrange in terms of c_2

$$c_2^2\left\{\frac{(3\alpha-\gamma)^2}{(2\alpha-\beta)^2}-3\right\}$$

$$+c_2\left\{2\frac{(3\alpha-\gamma)(\delta-4\alpha)}{(2\alpha-\beta)^2}c_1+2\frac{(3\alpha-\gamma)(5\alpha-\varepsilon)}{(2\alpha-\beta)^2}c_0-4\frac{3\alpha-\gamma}{2\alpha-\beta}c_0+4\frac{3\alpha-\gamma}{2\alpha-\beta}c_1-4c_0-4c_1\right\}$$

$$+c_1^2\left\{\frac{(\delta-4\alpha)^2}{(2\alpha-\beta)^2}+4\frac{\delta-4\alpha}{2\alpha-\beta}\right\}+c_0^2\left\{\frac{(5\alpha-\varepsilon)^2}{(2\alpha-\beta)^2}-4\frac{5\alpha-\varepsilon}{2\alpha-\beta}\right\}$$

$$+c_0c_1\left\{2\frac{(\delta-4\alpha)(5\alpha-\varepsilon)}{(2\alpha-\beta)^2}-4\frac{\delta-4\alpha}{2\alpha-\beta}+4\frac{5\alpha-\varepsilon}{2\alpha-\beta}\right\}=0$$

Tidy up to produce a common denominator of $(2\alpha-\beta)^2$

$$\frac{c_2^2}{(2\alpha-\beta)^2}\{(3\alpha-\gamma)^2-3(2\alpha-\beta)^2\}$$

$$+\frac{c_2}{(2\alpha-\beta)^2}\{c_1[2(3\alpha-\gamma)(\delta-4\alpha)+4(3\alpha-\gamma)(2\alpha-\beta)+4(2\alpha-\beta)^2]+c_0[2(3\alpha-\gamma)(5\alpha-\varepsilon)-4(3\alpha-\gamma)(2\alpha-\beta)-4(2\alpha-\beta)^2]\}$$

$$+\frac{c_1^2}{(2\alpha-\beta)^2}\{(\delta-4\alpha)^2+4(\delta-4\alpha)(2\alpha-\beta)\}+\frac{c_0^2}{(2\alpha-\beta)^2}\{(5\alpha-\varepsilon)^2-4(5\alpha-\varepsilon)(2\alpha-\beta)\}$$

$$+\frac{c_0c_1}{(2\alpha-\beta)^2}\{2(\delta-4\alpha)(5\alpha-\varepsilon)-4(\delta-4\alpha)(2\alpha-\beta)+4(5\alpha-\varepsilon)(2\alpha-\beta)\}=0$$

Multiply throughout by $(2\alpha - \beta)^2$ to remove this denominator and multiply out all the α, β, γ, δ, ε terms.

$c_2^2(9\alpha^2 + \gamma^2 - 6\alpha\gamma - 12\alpha^2 - 3\beta^2 + 12\alpha\beta)$
$+c_2[c_1(6\alpha\delta - 24\alpha^2 - 2\gamma\delta + 8\alpha\gamma + 24\alpha^2 - 12\alpha\beta - 8\alpha\gamma + 4\beta\gamma + 16\alpha^2 + 4\beta^2 - 16\alpha\beta) +$
$c_0(30\alpha^2 - 6\alpha\varepsilon - 10\alpha\gamma + 2\gamma\varepsilon - 24\alpha^2 + 12\alpha\beta + 8\alpha\gamma - 4\beta\gamma - 16\alpha^2 - 4\beta^2 + 16\alpha\beta)]$
$+c_1^2(\delta^2 + 16\alpha^2 - 8\alpha\delta + 8\alpha\delta - 4\beta\delta - 32\alpha^2 + 16\alpha\beta)$
$+c_0^2(25\alpha^2 + \varepsilon^2 - 10\alpha\varepsilon - 40\alpha^2 + 20\alpha\beta + 8\alpha\varepsilon - 4\beta\varepsilon)$
$+c_0c_1(10\alpha\delta - 2\delta\varepsilon - 40\alpha^2 + 8\alpha\varepsilon - 8\alpha\delta + 4\beta\delta + 32\alpha^2 - 16\alpha\beta + 40\alpha^2 - 20\alpha\beta - 8\alpha\varepsilon + 4\beta\varepsilon) = 0$

Collect together the α, β, γ, δ, ε terms in each c_0, c_1, c_2 expansion

$c_2^2(-3\alpha^2 - 3\beta^2 + \gamma^2 + 12\alpha\beta - 6\alpha\gamma)$
$+c_2[c_1(16\alpha^2 + 4\beta^2 - 28\alpha\beta + 6\alpha\delta + 4\beta\gamma - 2\gamma\delta) + c_0(-10\alpha^2 - 4\beta^2 + 28\alpha\beta - 2\alpha\gamma - 6\alpha\varepsilon - 4\beta\gamma + 2\gamma\varepsilon)]$
$+c_1^2(-16\alpha^2 + \delta^2 + 16\alpha\beta - 4\beta\delta)$
$+c_0^2(-15\alpha^2 + \varepsilon^2 + 20\alpha\beta - 2\alpha\varepsilon - 4\beta\varepsilon)$
$+c_0c_1(32\alpha^2 - 36\alpha\beta + 2\alpha\delta + 4\beta\delta + 4\beta\varepsilon - 2\delta\varepsilon) = 0$ (5.19)

Now substitute (5.17) into (5.16)

$$\frac{(3\alpha-\gamma)c_2+(\delta-4\alpha)c_1+(5\alpha-\varepsilon)c_0}{2\alpha-\beta} = \frac{c_1(-3c_1+2c_2)+c_0(5c_0+3c_2)}{2c_0+c_1}$$

$[(3\alpha - \gamma)c_2 + (\delta - 4\alpha)c_1 + (5\alpha - \varepsilon)c_0](2c_0 + c_1) = [(-3c_1 + 2c_2)c_1 + (5c_0 + 3c_2)c_0](2\alpha - \beta)$

$(6\alpha - 2\gamma)c_0c_2 + (2\delta - 8\alpha)c_0c_1 + (10\alpha - 2\varepsilon)c_0^2 + (3\alpha - \gamma)c_1c_2 + (\delta - 4\alpha)c_1^2 + (5\alpha - \varepsilon)c_0c_1 =$
$(3\beta - 6\alpha)c_1^2 + (4\alpha - 2\beta)c_1c_2 + (10\alpha - 5\beta)c_0^2 + (6\alpha - 3\beta)c_0c_2$

$c_2[(6\alpha - 2\gamma)c_0 + (3\alpha - \gamma)c_1 + (2\beta - 4\alpha)c_1 + (3\beta - 6\alpha)c_0] =$
$(2\varepsilon - 10\alpha + 10\alpha - 5\beta)c_0^2 + (4\alpha - \delta + 3\beta - 6\alpha)c_1^2 + (8\alpha - 2\delta + \varepsilon - 5\alpha)c_0c_1$

$c_2[(6\alpha - 2\gamma - 6\alpha + 3\beta)c_0 + (3\alpha - \gamma + 2\beta - 4\alpha)c_1] = (2\varepsilon - 5\beta)c_0^2 + (-2\alpha + 3\beta - \delta)c_1^2 + (3\alpha - 2\delta + \varepsilon)c_0c_1$

$c_2[(3\beta - 2\gamma)c_0 + (-\alpha + 2\beta - \gamma)c_1] = (2\varepsilon - 5\beta)c_0^2 + (-2\alpha + 3\beta - \delta)c_1^2 + (3\alpha - 2\delta + \varepsilon)c_0c_1$ (5.20)

There are now 3 equations in c_0, c_1, c_2:

$$c_2 = \frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)} + \frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}c_0 + \frac{2\beta-\delta}{\alpha+\beta-\gamma}c_1 \quad (5.18)$$

$c_2^2(-3\alpha^2 - 3\beta^2 + \gamma^2 + 12\alpha\beta - 6\alpha\gamma)$
$+ c_2[c_1(16\alpha^2 + 4\beta^2 - 28\alpha\beta + 6\alpha\delta + 4\beta\gamma - 2\gamma\delta) + c_0(-10\alpha^2 - 4\beta^2 + 28\alpha\beta - 2\alpha\gamma - 6\alpha\varepsilon - 4\beta\gamma + 2\gamma\varepsilon)]$
$+ c_1^2(-16\alpha^2 + \delta^2 + 16\alpha\beta - 4\beta\delta)$
$+ c_0^2(-15\alpha^2 + \varepsilon^2 + 20\alpha\beta - 2\alpha\varepsilon - 4\beta\varepsilon)$
$+ c_0c_1(32\alpha^2 - 36\alpha\beta + 2\alpha\delta + 4\beta\delta + 4\beta\varepsilon - 2\delta\varepsilon) = 0$ (5.19)

$c_2[(3\beta - 2\gamma)c_0 + (-\alpha + 2\beta - \gamma)c_1] = (2\varepsilon - 5\beta)c_0^2 + (-2\alpha + 3\beta - \delta)c_1^2 + (3\alpha - 2\delta + \varepsilon)c_0c_1$ (5.20)

Now use (5.18) to eliminate c_2 from (5.19) and (5.20). Substitute (5.18) into (5.19)

$$\left[\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}c_0+\frac{2\beta-\delta}{\alpha+\beta-\gamma}c_1\right]^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)$$

$$+\left[\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}c_0+\frac{2\beta-\delta}{\alpha+\beta-\gamma}c_1\right]\left[\begin{array}{l}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)c_1+\\(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)c_0\end{array}\right]$$

$$+c_1^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)$$
$$+c_0^2(-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)$$
$$+c_0c_1(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)=0$$

Multiply out and rearrange in terms of c_0^2, c_1^2, c_0, c_1 and c_0c_1.

$$c_1^2\left[\frac{(2\beta-\delta)^2}{(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\beta-\delta}{\alpha+\beta-\gamma}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\right]$$

$$+c_1\left[\frac{(2\alpha-\beta)}{\sqrt{2}(\alpha+\beta-\gamma)}\frac{(2\beta-\delta)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\right]$$

$$+c_0^2\left[\frac{(-\alpha-2\beta+\varepsilon)^2}{(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma-2\gamma\varepsilon)+(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\right]$$

$$+c_0\left[\frac{(2\alpha-\beta)}{\sqrt{2}(\alpha+\beta-\gamma)}\frac{(-\alpha-2\beta+\varepsilon)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma-2\gamma\varepsilon)\right]$$

$$+c_0c_1\left[\begin{array}{l}\frac{2(-\alpha-2\beta+\varepsilon)}{(\alpha+\beta-\gamma)}\frac{(2\beta-\delta)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\+\frac{2\beta-\delta}{\alpha+\beta-\gamma}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon\end{array}\right]$$

$$+\frac{(2\alpha-\beta)^2}{8(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)=0$$

Rearrange as a quadratic in c_1:

$$c_1^2\left[\frac{(2\beta-\delta)^2}{(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\beta-\delta}{\alpha+\beta-\gamma}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\right]$$

$$+\ c_1\left[\begin{array}{l}\frac{(2\alpha-\beta)}{\sqrt{2}(\alpha+\beta-\gamma)}\frac{(2\beta-\delta)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\ \left[\begin{array}{l}\frac{2(-\alpha-2\beta+\varepsilon)}{(\alpha+\beta-\gamma)}\frac{(2\beta-\delta)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{(-\alpha-2\beta+\varepsilon)}{(\alpha+\beta-\gamma)}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\ \frac{(2\beta-\delta)}{(\alpha+\beta-\gamma)}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)\end{array}\right]c_0\end{array}\right]$$

$$+\ c_0^2\left[\begin{array}{l}\frac{(-\alpha-2\beta+\varepsilon)^2}{(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+\\ (-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)\end{array}\right]$$

$$+\ c_0\left[\frac{(2\alpha-\beta)}{\sqrt{2}(\alpha+\beta-\gamma)}\frac{(-\alpha-2\beta+\varepsilon)}{(\alpha+\beta-\gamma)}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)\right]$$

$$+\frac{(2\alpha-\beta)^2}{8(\alpha+\beta-\gamma)^2}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)=0$$

Rearrange to give a common $(\alpha+\beta-\gamma)^2$ denominator

$$\frac{c_1^2}{(\alpha+\beta-\gamma)^2}\left[\begin{array}{l}(2\beta-\delta)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(2\beta-\delta)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\ (\alpha+\beta-\gamma)^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\end{array}\right]$$

$$+\frac{c_1^2}{(\alpha+\beta-\gamma)^2}\left[\begin{array}{l}\frac{(2\alpha-\beta)(2\beta-\delta)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{(2\alpha-\beta)(\alpha+\beta-\gamma)}{2\sqrt{2}}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\ \left[\begin{array}{l}2(-\alpha-2\beta+\varepsilon)(2\beta-\delta)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(-\alpha-2\beta+\varepsilon)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\ (\alpha+\beta-\gamma)(2\beta-\delta)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(\alpha+\beta-\gamma)^2(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)\end{array}\right]c_0\end{array}\right]$$

$$+\frac{c_0^2}{(\alpha+\beta-\gamma)^2}\left[\begin{array}{l}(-\alpha-2\beta+\varepsilon)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(-\alpha-2\beta+\varepsilon)(\alpha+\beta-\gamma)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+\\ (\alpha+\beta-\gamma)^2(-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)\end{array}\right]$$

$$+\frac{c_0}{(\alpha+\beta-\gamma)^2}\left[\frac{(2\alpha-\beta)(-\alpha-2\beta+\varepsilon)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{(2\alpha-\beta)(\alpha+\beta-\gamma)}{2\sqrt{2}}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)\right]$$

$$+\frac{1}{(\alpha+\beta-\gamma)^2}\frac{(2\alpha-\beta)^2}{8}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)=0 \qquad (5.21)$$

Then the leading $(\alpha+\beta-\gamma)^2$ denominator can be removed from all terms. This leaves a quadratic in c_1 in terms of α, β, γ, δ, ε and c_0 of the form

$$ac_1^2 + bc_1 + c = 0$$

The terms a, b and c are

$$a = (2\beta-\delta)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma) + (\alpha+\beta-\gamma)(2\beta-\delta)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta) + (\alpha+\beta+\gamma)^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)$$

$$b = \frac{(2\alpha-\beta)(2\beta-\delta)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma) + \frac{(2\alpha-\beta)(2\beta-\delta)}{2\sqrt{2}}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)$$
$$+ c_0\begin{bmatrix}2(-\alpha-2\beta+\varepsilon)(2\beta-\delta)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(-\alpha-2\beta+\varepsilon)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)+\\(\alpha+\beta-\gamma)(2\beta-\delta)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(\alpha+\beta-\gamma)^2(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)\end{bmatrix}$$

$$c = c_0^2\begin{bmatrix}(-\alpha-2\beta+\varepsilon)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(-\alpha-2\beta+\varepsilon)(\alpha+\beta-\gamma)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+\\(\alpha+\beta-\gamma)^2(-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)\end{bmatrix}$$

$$+ c_0\left[\frac{(2\alpha-\beta)(-\alpha-2\beta+\varepsilon)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\frac{(2\alpha-\beta)(\alpha+\beta-\gamma)}{2\sqrt{2}}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)\right]$$

$$+ \frac{(2\alpha-\beta)^2}{8}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)$$

Then by the quadratic formula

$$c_1 = \frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

This is not the final solution because b and c are functions of c_0.

Going back to the elimination of c_2, substitute (5.18) into (5.20):

$$\left[\frac{2\alpha-\beta}{2\sqrt{2}(\alpha+\beta-\gamma)}+\frac{-\alpha-2\beta+\varepsilon}{\alpha+\beta-\gamma}c_0+\frac{2\beta-\delta}{\alpha+\beta-\gamma}c_1\right][(3\beta-2\gamma)c_0+(-\alpha+2\beta-\gamma)c_1] =$$

$$(2\varepsilon-5\beta)c_0^2+(-2\alpha+3\beta-\delta)c_1^2+(3\alpha-2\delta+\varepsilon)c_0c_1$$

Multiply throughout by $(\alpha+\beta-\gamma)$ to eliminate this denominator. Also rearrange as a quadratic in c_1.

$$c_1^2\{(2\beta-\delta)(-\alpha+2\beta-\gamma)+(2\alpha-3\beta+\delta)(\alpha+\beta-\gamma)\}$$

$$+ c_1\left\{\frac{(2\alpha-\beta)}{2\sqrt{2}}(-\alpha+2\beta-\gamma)+c_0[(-\alpha-2\beta+\varepsilon)(-\alpha+2\beta-\gamma)+(2\beta-\delta)(3\beta-2\gamma)-(3\alpha-2\delta+\varepsilon)(\alpha+\beta-\gamma)]\right\}$$

$$+ c_0^2\{(-\alpha-2\beta+\varepsilon)(3\beta-2\gamma)+(5\beta-2\varepsilon)(\alpha+\beta-\gamma)\}$$

$$+ c_0\frac{(2\alpha-\beta)}{2\sqrt{2}}(3\beta-2\alpha) = 0 \qquad (5.22)$$

There is now a further quadratic in c_1 in terms of α, β, γ, δ, ε and c_0:

$$Ac_1^2 + Bc_1 + C = 0$$

The terms A, B, C are

$$A = (2\beta - \delta)(-\alpha + 2\beta - \gamma) + (2\alpha - 3\beta + \delta)(\alpha + \beta - \gamma)$$

$$B = \frac{(2\alpha - \beta)}{2\sqrt{2}}(-\alpha + 2\beta - \gamma) + c_0[(-\alpha - 2\beta + \varepsilon)(-\alpha + 2\beta - \gamma) + (2\beta - \delta)(3\beta - 2\gamma) - (3\alpha - 2\delta + \varepsilon)(\alpha + \beta - \gamma)$$

$$C = [(-\alpha - 2\beta + \varepsilon)(3\beta - 2\gamma) + (5\beta - 2\varepsilon)(\alpha + \beta - \gamma)]c_0^2 + \frac{(2\alpha - \beta)}{2\sqrt{2}}(3\beta - 2\alpha)c_0$$

and c_1 can be found by the quadratic formula as

$$c_1 = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

Again, this is not the final solution because B and C contain c_0.

There are now two equations for c_1 given by the quadratic formulas (5.21) and (5.22) as follows:

(5.21) gives $ac_1^2 + bc_1 + c = 0 \Rightarrow$ $\quad c_1 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ (5.23)

(5.22) gives $Ac_1^2 + Bc_1 + C = 0 \Rightarrow$ $\quad c_1 = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$ (5.24)

Equate c_1 in these two equations to get

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

Substituting in the terms for a, b, c and A, B, C yields a single equation in α, β, γ, δ, ε and c_0 which shall be called *equation abc*. It is rather unmanageable. This is the final stage in the elimination that started from the seed equations (5.1) to (5.6). To remind the reader, the purpose of this process was to find expressions for the taps in terms of the lock condition so that a class of wavelets could be generated. The process of elimination, having reduced the seed equations to this single expression for c_0, can facilitate calculation of the other taps, once c_0 itself has been determined.

$$-\left\{\begin{array}{l}\dfrac{(2\alpha-\beta)(2\beta-\delta)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\dfrac{(2\alpha-\beta)(2\beta-\delta)}{2\sqrt{2}}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +c_0\left[\begin{array}{l}2(-\alpha-2\beta+\varepsilon)(2\beta-\delta)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(-\alpha-2\beta+\varepsilon)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +(\alpha+\beta-\gamma)(2\beta-\delta)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(\alpha+\beta-\gamma)^2(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)\end{array}\right]\end{array}\right\}$$

$$\pm\sqrt{\left\{\begin{array}{l}\dfrac{(2\alpha-\beta)(2\beta-\delta)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\dfrac{(2\alpha-\beta)(2\beta-\delta)}{2\sqrt{2}}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +c_0\left[\begin{array}{l}2(-\alpha-2\beta+\varepsilon)(2\beta-\delta)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(-\alpha-2\beta+\varepsilon)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +(\alpha+\beta-\gamma)(2\beta-\delta)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(\alpha+\beta-\gamma)^2(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)\end{array}\right]\end{array}\right\}^2+}$$

$$-4\left\{\begin{array}{l}(2\beta-\delta)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)\\ +(\alpha+\beta-\gamma)(2\beta-\delta)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +(\alpha+\beta-\gamma)^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\end{array}\right\}\left\{\begin{array}{l}c_0^2\left[\begin{array}{l}(-\alpha-2\beta+\varepsilon)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(-\alpha-2\beta+\varepsilon)(\alpha+\beta-\gamma)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+\\ (\alpha+\beta-\gamma)^2(-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)\end{array}\right]\\ +c_0\left[\dfrac{(2\alpha-\beta)(-\alpha-2\beta+\varepsilon)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+\dfrac{(2\alpha-\beta)(\alpha+\beta-\gamma)}{2\sqrt{2}}(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)\right]\\ +\dfrac{(2\alpha-\beta)^2}{8}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)\end{array}\right\}}$$

$$2\left\{\begin{array}{l}(2\beta-\delta)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)\\ +(\alpha+\beta-\gamma)(2\beta-\delta)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)\\ +(\alpha+\beta-\gamma)^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)\end{array}\right\}$$

=

$$-\left\{\frac{(2\alpha-\beta)}{2\sqrt{2}}(-\alpha+2\beta-\gamma)+c_0[(-\alpha-2\beta+\varepsilon)(-\alpha+2\beta-\gamma)+(2\beta-\delta)(3\beta-2\gamma)-(3\alpha-2\delta+\varepsilon)(\alpha+\beta-\gamma)]\right\}\pm$$

$$\sqrt{\left\{\frac{(2\alpha-\beta)}{2\sqrt{2}}(-\alpha+2\beta-\gamma)+c_0[(-\alpha-2\beta+\varepsilon)(-\alpha+2\beta-\gamma)+(2\beta-\delta)(3\beta-2\gamma)-(3\alpha-2\delta+\varepsilon)(\alpha+\beta-\gamma)]\right\}^2}$$

$$-4\{(2\beta-\delta)(-\alpha+2\beta-\gamma)+(2\alpha-3\beta+\delta)(\alpha+\beta-\gamma)\}\left\{c_0^2[(-\alpha-2\beta+\varepsilon)(3\beta-2\gamma)+(5\beta-2\varepsilon)(\alpha+\beta-\gamma)]+\frac{(2\alpha-\beta)}{2\sqrt{2}}(3\beta-2\gamma)c_0\right\}$$

$$2\{(2\beta-\delta)(-\alpha+2\beta-\gamma)+(2\alpha-3\beta+\delta)(\alpha+\beta-\gamma)\}$$

Equation abc can be made more manageable by grouping together terms into larger components with the introduction of some new identifiers. The a and A of (5.21) and (5.22) remain, and the new set becomes:

$A = A$
$B = D + Ec_0$
$C = Gc_0 + Hc_0^2$

where

$$A = (2\beta - \delta)(-\alpha + 2\beta - \gamma) + (2\alpha - 3\beta + \delta)(\alpha + \beta - \gamma)$$

$$D = \frac{(2\alpha - \beta)}{2\sqrt{2}}(-\alpha + 2\beta - \gamma)$$

$$E = (-\alpha - 2\beta + \varepsilon)(-\alpha + 2\beta - \gamma) + (2\beta - \delta)(3\beta - 2\gamma) - (3\alpha - 2\delta + \varepsilon)(\alpha + \beta - \gamma)$$

$$G = \frac{(2\alpha - \beta)}{2\sqrt{2}}(3\beta - 2\alpha)$$
$$H = (-\alpha - 2\beta + \varepsilon)(3\beta - 2\gamma) + (5\beta - 2\varepsilon)(\alpha + \beta - \gamma)$$

and

$a = a$
$b = d + ec_0$
$c = f + gc_0 + hc_0^2$

where

$$a = (2\beta-\delta)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma) + (\alpha+\beta-\gamma)(2\beta-\delta)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta) + (\alpha+\beta+\gamma)^2(-16\alpha^2+\delta^2+16\alpha\beta-4\beta\delta)$$

$$d = \frac{(2\alpha - \beta)(2\beta - \delta)}{\sqrt{2}}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma) + \frac{(2\alpha - \beta)(\alpha + \beta - \gamma)}{2\sqrt{2}}(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta)$$

$$e = 2(-\alpha-2\beta+\varepsilon)(2\beta-\delta)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(\alpha+\beta-\gamma)(-\alpha-2\beta+\varepsilon)(16\alpha^2+4\beta^2-28\alpha\beta+6\alpha\delta+4\beta\gamma-2\gamma\delta) + (\alpha+\beta-\gamma)(2\beta-\delta)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon)+(\alpha+\beta-\gamma)^2(32\alpha^2-36\alpha\beta+2\alpha\delta+4\beta\delta+4\beta\varepsilon-2\delta\varepsilon)$$

$$f = \frac{(2\alpha-\beta)^2}{8}(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)$$

$$g = \frac{(2\alpha - \beta)}{\sqrt{2}}(-\alpha-2\beta+\varepsilon)(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma) + \frac{(2\alpha - \beta)}{2\sqrt{2}}(\alpha+\beta-\gamma)(-10\alpha^2-4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon+4\beta\gamma+2\gamma\varepsilon)$$

$$h = (-\alpha-2\beta+\varepsilon)^2(-3\alpha^2-3\beta^2+\gamma^2+12\alpha\beta-6\alpha\gamma)+(-\alpha-2\beta+\varepsilon)(\alpha+\beta-\gamma)(-10\alpha^2+4\beta^2+28\alpha\beta-2\alpha\gamma-6\alpha\varepsilon-4\beta\gamma+2\gamma\varepsilon) + (\alpha+\beta-\gamma)^2(-15\alpha^2+\varepsilon^2+20\alpha\beta-2\alpha\varepsilon-4\beta\varepsilon)$$

Then the equation

$$\frac{-b \pm \sqrt{b^2-4ac}}{2a} = \frac{-B \pm \sqrt{B^2-4AC}}{2A}$$

becomes

$$\frac{-d-ec_0 \pm \sqrt{(d+ec_0)^2-4a(f+gc_0+hc_0^2)}}{2a} = \frac{-D-Ec_0 \pm \sqrt{(D+Ec_0)^2-4A(Gc_0+Hc_0^2)}}{2A}$$

$$-2A(d+ec_0) \pm 2A\sqrt{(d+ec_0)^2-4a(f+gc_0+hc_0^2)} = -2a(D+Ec_0) \pm 2a\sqrt{(D+Ec_0)^2-4A(Gc_0+Hc_0^2)}$$

Divide throughout by 2 and multiply out the terms within the square roots:

$$-A(d+ec_0) \pm A\sqrt{d^2+e^2c_0^2+2dec_0-4af-4agc_0-4ahc_0^2} = -a(D+Ec_0) \pm a\sqrt{D^2+E^2c_0^2+2DEc_0-4AGc_0-4AHc_0^2}$$

Collect together the c_0 terms in each square root term:

$$-A(d+ec_0) \pm A\sqrt{(d^2-4af)+(2de-4ag)c_0+(e-4ah)c_0^2} = -a(D+Ec_0) \pm a\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}$$

Consider the positive roots and collect them:

$$a(D+Ec_0)-A(d+ec_0) = a\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2} - A\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Square both sides:

$$(aD - Ad)^2 + 2(aD - Ad)(aE - Ae)c_0 + (aE - Ae)^2c_0^2 =$$

$$a^2[D^2 + (2DE - 4AG)c_0 + (E^2 - 4AH)c_0^2]$$

$$+ A^2[(d^2 - 4af) + (2de - 4ag)c_0 + (e^2 - 4ah)c_0^2]$$

$$- 2aA\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Rearrange the square roots alone on one side

$$(aD - Ad)^2 - a^2D^2 - A^2(d^2 - 4af) + [2(aD - Ad)(aE - Ae) - a^2(2DE - 4AG) - A^2(2de - 4ag)]c_0$$

$$+ [(aE - Ae)^2 - a^2(E^2 - 4AH) - A^2(e^2 - 4ah)]c_0^2$$

$$= -2aA\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Expand the left hand side

$$a^2D^2 + A^2d^2 - 2aDAd - a^2D^2 - A^2d^2 + 4A^2af$$

$$+ [2aDaE - 2aDAe - 2AdaE + 2AdAe - 2a^2DE + 4a^2AG - 2A^2de + 4A^2ag]c_0$$

$$+ [a^2E^2 + A^2e^2 - 2aEAe - a^2E^2 + 4a^2AH - A^2e^2 + 4A^2ah]c_0^2$$

$$= -2aA\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Rewrite, taking advantage of left hand side cancellation

$$-2aDAd + 4A^2af + [-2aDAe - 2AdaE + 4a^2AG + 4A^2ag]c_0 + [-2aEAe + 4a^2AH + 4A^2ah]c_0^2$$

$$= -2aA\sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Divide throughout by -2aA

$$Dd - 2Af + [De + dE - 2aG - 2Ag]c_0 + [Ee - 2aH - 2Ah]c_0^2$$

$$= \sqrt{D^2+(2DE-4AG)c_0+(E^2-4AH)c_0^2}\sqrt{(d^2-4af)+(2de-4ag)c_0+(e^2-4ah)c_0^2}$$

Square both sides

$$\begin{aligned}&(Dd - 2Af)^2 + (De + dE - 2aG - 2Ag)^2c_0^2 + (Ee - 2aH - 2Ah)^2c_0^4\\ &+ 2(Dd - 2Af)(De + dE - 2aG - 2Ag)c_0 + 2(Dd - 2Af)(Ee - 2aH - 2Ah)c_0^2\\ &+ 2(De + dE - 2aG - 2Ag)(Ee - 2aH - 2Ah)c_0^3\end{aligned}$$

$$= [D^2 + (2DE - 4AG)c_0 + (E^2 - 4AH)c_0^2][(d^2 - 4af) + (2de - 4ag)c_0 + (e^2 - 4ah)c_0^2]$$

Multiply out the right hand side

$$\begin{aligned}&(Dd - 2Af)^2 + (De + dE - 2aG - 2Ag)^2c_0^2 + (Ee - 2aH - 2Ah)^2c_0^4\\ &+ 2(Dd - 2Af)(De + dE - 2aG - 2Ag)c_0 + 2(Dd - 2Af)(Ee - 2aH - 2Ah)c_0^2\\ &+ 2(De + dE - 2aG - 2Ag)(Ee - 2aH - 2Ah)c_0^3\end{aligned}$$

$$\begin{aligned}&= D^2(d^2 - 4af) + D^2(2de - 4ag)c_0 + D^2(e^2 - 4ah)c_0^2\\ &+ (2DE - 4AG)(d^2 - 4af)c_0 + (2DE - 4AG)(2de - 4ag)c_0^2 + (2DE - 4AG)(e^2 - 4ah)c_0^3\\ &+ (E^2 - 4AH)(d^2 - 4af)c_0^2 + (E^2 - 4AH)(2de - 4ag)c_0^3 + (E^2 - 4AH)(e^2 - 4ah)c_0^4\end{aligned} \quad (5.25)$$

Collect the powers of c_0

$$\begin{aligned}&(Dd - 2Af)^2 - D^2(d^2 - 4af)\\ &+ c_0[2(Dd - 2Af)(De + dE - 2aG - 2Ag) - D^2(2de - 4ag) - (2DE - 4AG)(d^2 - 4af)]\\ &+ c_0^2[(De + dE - 2aG - 2Ag)^2 + 2(Dd - 2Af)(Ee - 2aH - 2Ah) - D^2(e^2 - 4ah) -\\ &\quad (2DE - 4AG)(2de - 4ag) - (E^2 - 4AH)(d^2 - 4af)]\\ &+ c_0^3[2(De + dE - 2aG - 2Ag)(Ee - 2aH - 2Ah) - (2DE - 4AG)(e^2 - 4ah) - (E^2 - 4AH)(2de - 4ag)]\\ &+ c_0^4[(Ee - 2aH - 2Ah)^2 - (E^2 - 4AH)(e^2 - 4ah)]\\ &= 0\end{aligned} \quad (5.26)$$

This can be solved by the standard quartic formula for c_0 (Abramowitz & Stegun [2]). The other taps c_1, c_2, c_3, c_4, and c_5 can then be calculated by back substitution as follows:

Given c_0 as a solution of (5.26) for a given lock condition f(x,c), c_1 is given by (5.24):

$$c_1 = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \quad (5.24)$$

where

$$B = D + Ec_0$$
$$C = Gc_0 + Hc_0^2$$

Given c_0 and c_1, c_2 can then be calculated from any one of (5.18), (5.19), (5.20). Of these, (5.18) is the simplest; it is the only expression in explicit form:

$$c_2 = \frac{2\alpha - \beta}{2\sqrt{2}(\alpha + \beta - \gamma)} + \frac{-\alpha - 2\beta + \varepsilon}{\alpha + \beta - \gamma} c_0 + \frac{2\beta - \delta}{\alpha + \beta - \gamma} c_1 \qquad (5.18)$$

Given c_0, c_1 and c_2, c_3 can be calculated from any one of (5.14), (5.15), (5.16), (5.17). Of these, (5.14) is the simplest:

$$c_3 = \frac{1}{2\sqrt{2}} + 2c_0 - 2c_1 + c_2 \qquad (5.14)$$

Given c_0, c_1, c_2 and c_3, c_4 can be calculated from any one of (5.9), (5.10), (5.11), (5.12), (5.13). Of these, (5.11) is the simplest:

$$c_4 = \frac{1}{\sqrt{2}} - c_0 - c_2 \qquad (5.11)$$

Finally, any one of (5.2) to (5.7) gives c_5. (5.3) is the simplest algebraic form: $c_5 = -\frac{c_0 c_4}{c_1}$, and (5.4) is the simplest computational form, having no division: $c_5 - c_4 + c_3 - c_2 + c_1 - c_0 = 0$. Rearrange for c_5:

$$c_5 = c_4 - c_3 + c_2 - c_1 + c_0 \qquad (5.8)$$

This system of equations, (5.8), (5.11), (5.14), (5.18), (5.24) and (5.26) gives all the six taps for the generic form (5.1) to (5.6). Given these expressions, individual wavelets can be generated for different values of c in the lock condition.

5.4 Verification

Equation (5.26) can be verified by substituting in the Daubechies lock condition, the second moment given by $f(x,c) = x^2$, ie no c dependency.

Then

$\alpha = 1^2 = 1$
$\beta = 2^2 = 4$
$\gamma = 3^2 = 9$
$\delta = 4^2 = 16$
$\varepsilon = 5^2 = 25$

and

$a = -256$
$d = 32\sqrt{2}$
$e = 1024$
$f = 12$
$g = -32\sqrt{2}$
$h = -1280$
$A = -8$
$D = \sqrt{2}$
$E = 0$
$G = 3\sqrt{2}$
$H = 24$

and the quartic (5.26) becomes

$$36864 - 393216\sqrt{2}\,c_0 - 12582912c_0^2 - 33554432\sqrt{2}\,c_0^3 + 268435456c_0^4 = 0$$

Divide throughout by 1048576

$$\frac{9}{256} - \frac{3}{8}\sqrt{2}\,c_0 - 12c_0^2 - 32\sqrt{2}\,c_0^3 + 256c_0^4 = 0$$

This happily factorizes; evidence in itself of being a Daubechies wavelet:

$$(16c_0^2 - (1+\sqrt{10})\sqrt{2}\,c_0 + \frac{3}{16})(16c_0^2 - (1-\sqrt{10})\sqrt{2}\,c_0 + \frac{3}{16}) = 0$$

The right hand quadratic has no real solution.

The left hand quadratic solved by the quadratic formula gives

$$c_0 = \frac{(1+\sqrt{10})\sqrt{2} \pm \sqrt{2(1+\sqrt{10})^2 - 4\cdot 16\cdot \frac{3}{16}}}{2\cdot 16}$$

$$c_0 = \frac{(1+\sqrt{10})\sqrt{2} \pm \sqrt{2}\sqrt{(1+\sqrt{10})^2 - 6}}{2\cdot 16}$$

$$c_0 = \frac{1+\sqrt{10} \pm \sqrt{1+10+2\sqrt{10}-6}}{16\sqrt{2}}$$

$$c_0 = \frac{1+\sqrt{10} \pm \sqrt{5+2\sqrt{10}}}{16\sqrt{2}}$$

This is the exact formula for Daubechies c_0. The positive root gives the Daubechies minimum phase solution:

$$c_0 = \frac{1+\sqrt{10}+\sqrt{5+2\sqrt{10}}}{16\sqrt{2}} = 0.3326705$$

This can be used for calculating the other taps by back substitution.

$$B = \sqrt{2} + 0$$

$$C = 3\sqrt{2}\,(0.3326705) + 24(0.3326705)^2 = 4.06747$$

and c_1 by (5.24) has two solutions: -0.6301148 and 0.8068915. Choose $c_1 = 0.8068915$ because this is the Daubechies 6 minimum phase solution.

Then

c_2 from (5.18) is $c_2 = 0.4598775$
c_3 from (5.14) is $c_3 = -0.135011$
c_4 from (5.11) is $c_4 = -0.854412$
c_5 from (5.8) is $c_5 = 0.0352262$

The production of these Daubechies values serves to verify the generic derivation.

Given this corroboration it is now safe to proceed with other lock conditions having c dependency. In this fashion, an entire wavelet class can be generated from the generic method.

In deciding the characteristics of a useful class the only starting point is the Daubechies wavelet itself, a known industry standard. It would be useful to constrain the new class to "seed" from this wavelet. Consequently, a useful constraint on the lock condition would be a requirement to look at least something like the second moment, x^2. For approximation to Daubechies 6, f(x,c) is a function that's close to points (0, 0) and (5, 25), the D6 second moment end points.

The function $f(x,c) = \exp(x^n/c) - 1$ is examined for various n values. This is a monotonic increasing function that passes through the origin.

Case n = ½: $f(x,c) = e^{\sqrt{x}/c} - 1$

This satisfies f(0,c) = 0, and f(x,c) must also pass through (5,25):

f(5,c) = 25

ie $e^{\sqrt{25}/c} - 1 = 25$

$$\frac{\sqrt{5}}{c} = \ln 26$$

$c = \sqrt{5}/\ln 26 = 0.6863111$

Values for f(x,c) over x = 0, 1, 2, 3, 4, 5 are:

x	$f(x,c) = e^{\sqrt{x}/0.686} - 1$
0	0
1	3.29
2	6.85
3	11.48
4	17.45
5	25.03

Case n = 1: $f(x,c) = e^{x/c} - 1$

f(0,c) = 0, and f(x,c) must also pass through (5,25):

f(5,c) = 25

ie $e^{5/c} - 1 = 25$

$e^{5/c} = 26$

$5/c = \ln 26$

then $c = 5/\ln 26 = 1.5346384$

Values for f(x,c) over x = 0, 1, 2, 3, 4, 5 are:

x	$e^{x/1.534}$ -1
0	0
1	0.919
2	2.683
3	6.068
4	12.56
5	25.03

Case n = 2: $f(x,c) = \exp(x^2/c) - 1$

This satisfies f(0,c) = 0, and f(x,c) must also pass through (5,25):

f(5,c) = 25

ie $e^{25/c} - 1 = 25$

$e^{25/c} = 26$

25/c = ln26

c = 25/ln26 = 7.6731919

Values for f(x,c) over x = 0, 1, 2, 3, 4, 5 are:

x	$f(x,c) = \exp(x^2/7.67) - 1$
0	0
1	0.14
2	0.68
3	2.23
4	7.05
5	25.03

Case n = 3: $f(x,c) = \exp(x^3/c) - 1$

f(0,c) = 0 and $e^{125/c} - 1 = 25$

$e^{125/c} = 26$

125/c = ln26

c = 125/ln26 = 38.36596

Values for f(x,c) over x = 0, 1, 2, 3, 4, 5 are:

x	$f(x,c) = \exp(x^3/38.36) - 1$
0	0
1	0.03
2	0.23
3	1.02
4	4.30
5	25.01

These curves are shown in figure 5.1

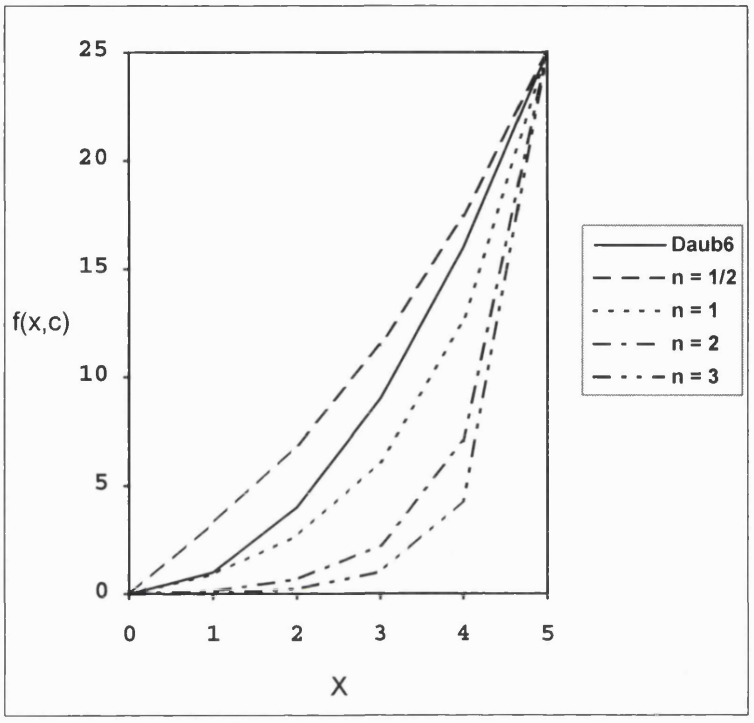


Figure 5.1: $f(x,c) = \exp(x^n/c) - 1$ for n = ½, 1, 2, 3

The two functions closest to the Daubechies second moment are $f(x,c) = \exp(x/c) - 1$ and $f(x,c) = \exp(\sqrt{x}/c) - 1$. The function $f(x,c) = \exp(x/c) - 1$ is examined further, and is used subsequently to generate a full class of generic wavelets.

In the following example for $f(x,c) = \exp(x/c) - 1$, c ranges over [0.2, 10000]. The solutions are shown in table 5.1. They were found using Mathematica [43]. Recall (5.26) is a quartic in c_0, so for each discrete c value there are four solutions for c_0. Only the real solutions are tabulated. Furthermore, for the other taps, c_1, c_2, c_3, c_4, c_5, only the minimum phase solution is shown.

c	c_0	c_1	c_2	c_3	c_4	c_5
0.2	-0.0008717 0.0000219 $5.873x10^{-6}$ 0.00325407	0.48856225	0.83473573	0.21767278	-0.1308803	0.00087174
0.3	-0.00458506 -0.00061455 0.000163688 0.017214144	0.51176911	0.82630575	0.19074919	-0.1641311	0.00458846
0.4	-0.01021879 -0.00327202 0.000847268 0.039463464	0.54619953	0.81050415	0.15058541	-0.1428608	0.01032183
0.5	-0.01532194 -0.00932071 0.002211535 0.064575722	0.58166241	0.78972320	0.10910320	-0.1471921	0.01634116
0.6	0.004089041 0.089157052	0.61327785	0.76686774	0.07217953	-0.1489180	0.02164939
0.7	0.006218378 0.111683432	0.63983972	0.74407861	0.04131941	-0.1486552	0.02594763
0.8	0.008390033 0.131686264	0.66166284	0.72255961	0.01615983	-0.1471390	0.02928409
0.9	0.010475503 0.149196112	0.67950075	0.70284015	-0.0042157	-0.1449294	0.03182176
1.0	0.012410181 0.164444586	0.69412628	0.68505613	-0.0207538	-0.1423939	0.03373436
1.1	0.014169969 0.177721008	0.70620169	0.66914228	-0.0342657	-0.1397565	0.03517078
1.2	0.015753482 0.189309424	0.71625808	0.65494423	-0.0453996	-0.1371468	0.03624838
1.3	0.017170764 0.199464462	0.72471015	0.64422783	-0.0546596	-0.1346359	0.03705632
1.4	0.018436758 0.208404958	0.73187887	0.63096107	-0.0624333	-0.1322592	0.03766126
1.5	0.019567760 0.216315164	0.73801254	0.62082281	-0.0690185	-0.1300311	0.03811279
1.6	0.020579625 0.223348729	0.74330413	0.61171226	-0.0746451	-0.1279542	0.03844780
2.0	0.023703927 0.244969290	0.75863222	0.58316586	-0.0906066	-0.1210283	0.03908117
3.0	0.027915106 0.274542394	0.77736574	0.54287571	-0.1092176	-0.1103113	0.03895866
4.0	0.029948710 0.283904918	0.78582051	0.52213801	-0.1171686	-0.1044217	0.03845490
5.0	0.031120131 0.298247087	0.79056448	0.50961348	-0.1214679	-0.1007537	0.03801021
10	0.033307662 0.315713442	0.79926831	0.48458415	-0.1289721	-0.0931908	0.03681066
20	0.034304965 0.324268737	0.08032163	0.47216841	-0.1321733	-0.0893303	0.03606381
50	0.034867398 0.329329849	0.80545419	0.46477611	-0.1339191	-0.0869991	0.03557176
100	0.035048492 0.310036687	0.80617830	0.46232362	-0.1344722	-0.0862205	0.03540073
1000	0.035208661 0.332504182	0.80682067	0.46012181	-0.1349577	-0.8551921	0.35243888
10000	0.035224531 0.322653817	0.80688425	0.45990198	-0.1350055	-0.0854490	0.03522820

Table 5.1: tap values for f(x,c) = exp(x/c) - 1

As c decreases c_0 and c_5 approach zero and taps c_1, c_2, c_3, c_4 approach the Daubechies 4 taps. As c increases, the taps approach the Daubechies 6 taps.

These taps are shown plotted in figure 5.2.

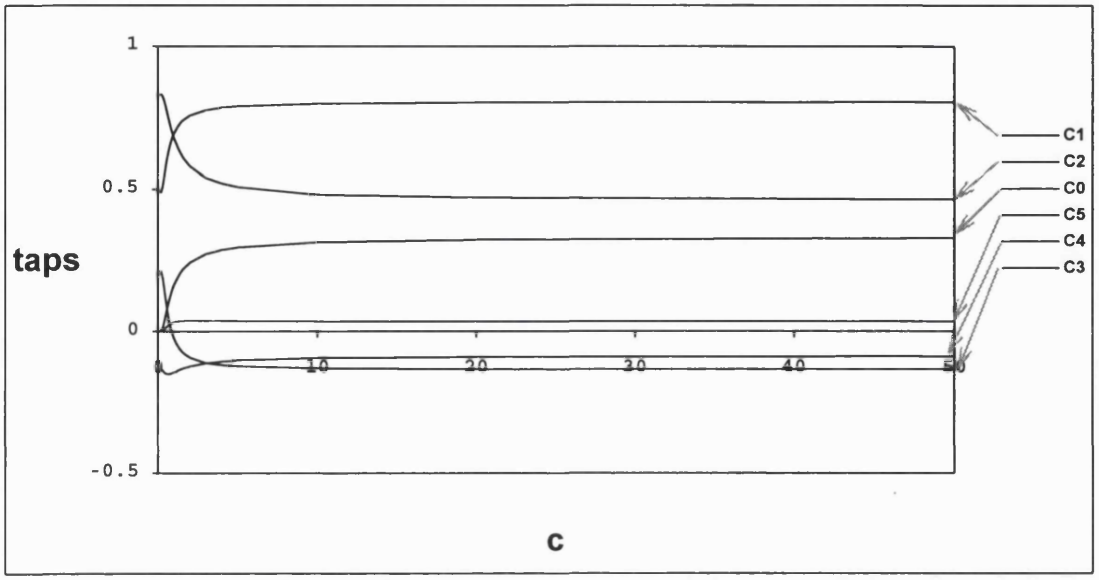


Figure 5.2: plot of the taps as functions of c

5.5 Appearance

The wavelet class derived above is shown in figures 5.3 and 5.4 for an illustrative selection of c values. Figure 5.3 shows the scaling functions and figure 5.4 shows the corresponding wavelets.

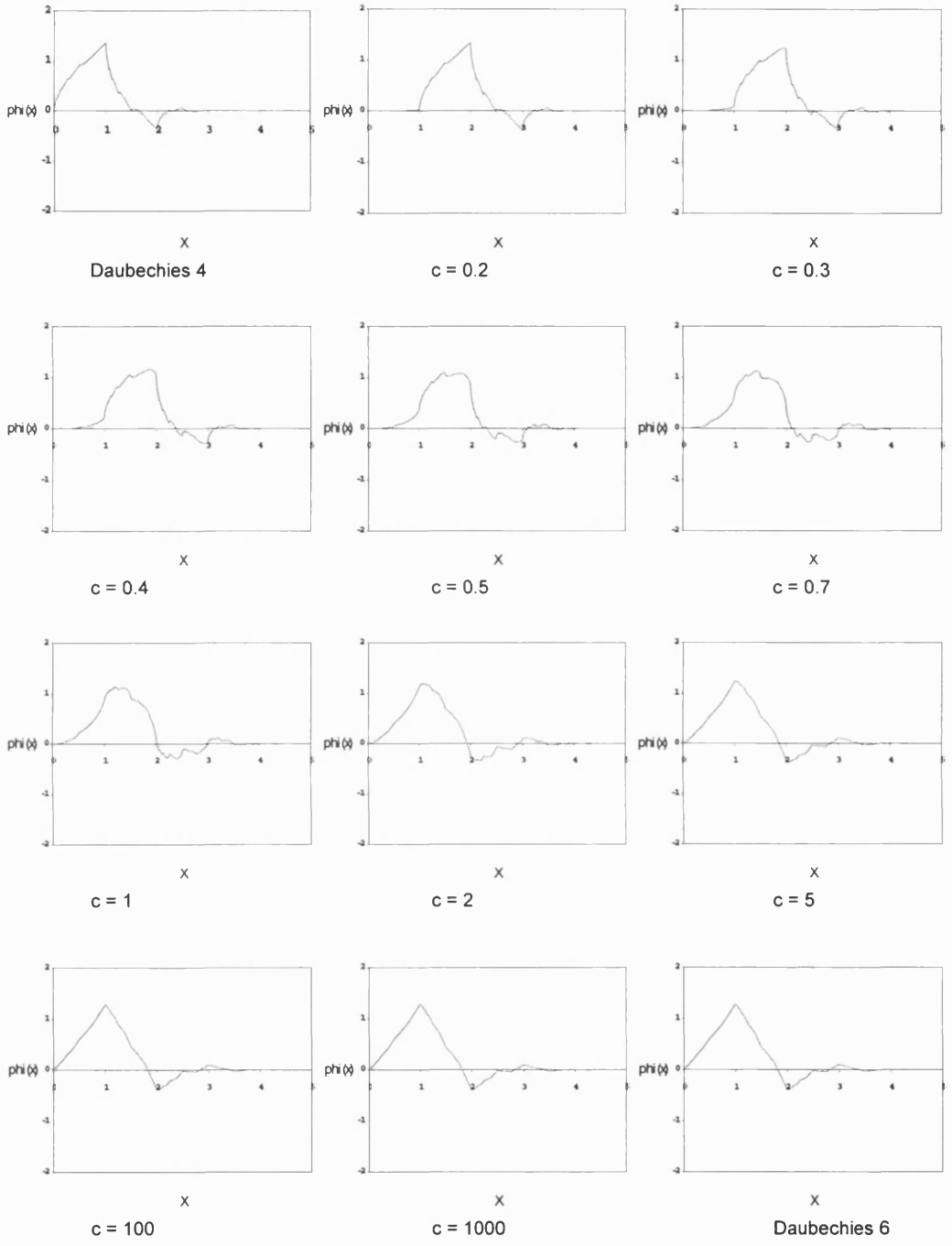


Figure 5.3: scaling functions of the exp(x/c) - 1 class

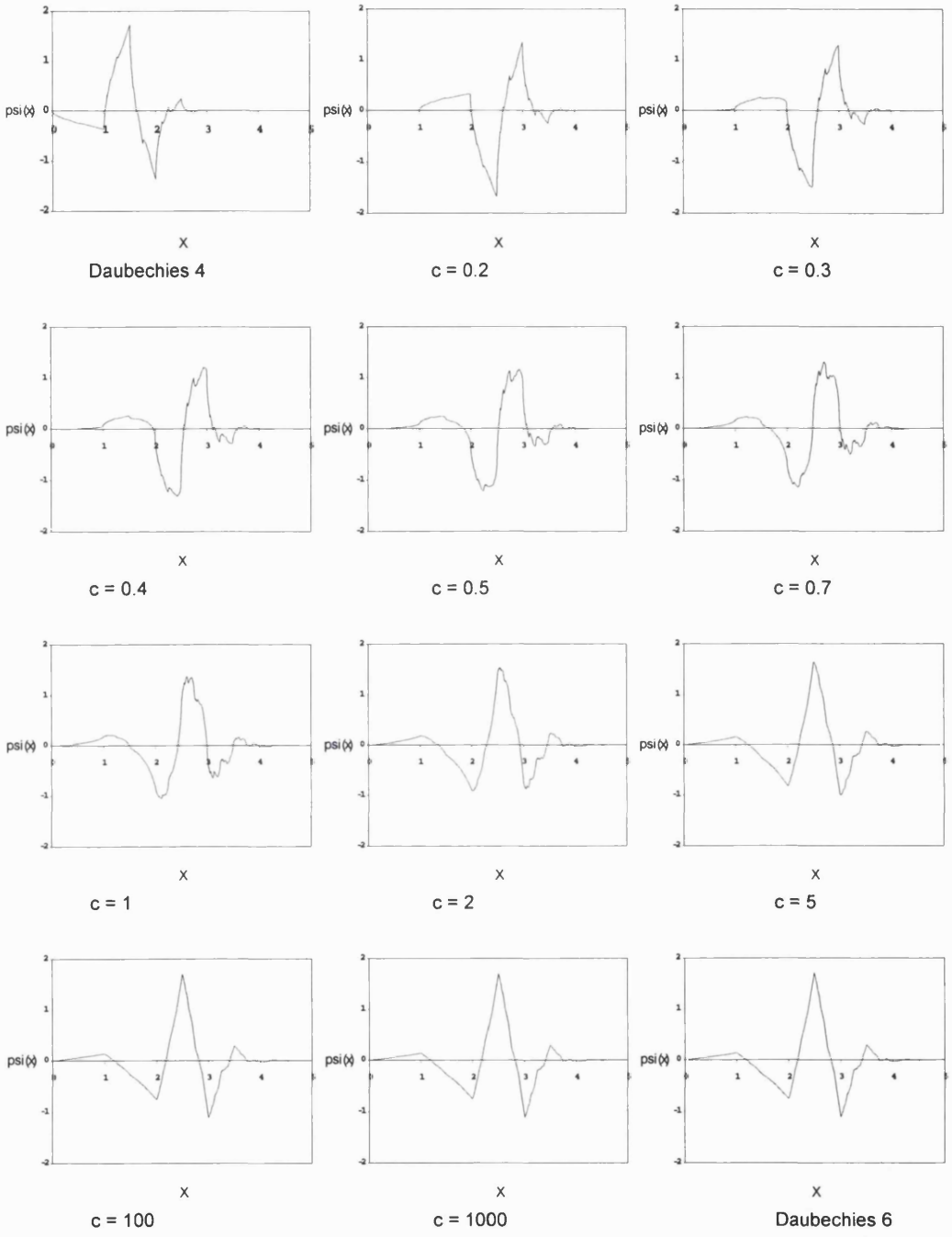


Figure 5.4: wavelets of the exp(x/c) - 1 class

5.6 Frequency response

The frequency response of the generic wavelet for a specific set of coefficients c_n is calculated as

$$m(\omega) = \sum_{n=0}^{5} c_n e^{-i\pi n\omega}$$

This expression is attributable to Benedetto [9].

Then the absolute frequency response is

$$|m(\omega)| = \sqrt{(R_e m(\omega))^2 + (I_m m(\omega))^2}$$

and the phase response is

$$\arg[m(\omega)] = \arctan\left(\frac{I_m m(\omega)}{R_e m(\omega)}\right)$$

These two components of the frequency response are shown respectively in figures 5.5 and 5.6.

For each plot in figure 5.5, the absolute frequency response of both scaling function (solid) and wavelet (dash) are shown. In comparison with figures 5.3 and 5.4, the changes between plots is slight and only noticeable when shown in comparison with the Daubechies 6 absolute frequency response (dot). A significant feature of the dynamic wavelet class is its fidelity to the Daubechies 6 in the frequency domain. This means that each wavelet covers the entire frequency domain.

Figure 5.6 tells a similar story, although the phase response is more varied with respect to that of Daubechies 6 (dot). The main difference is that the dynamic wavelet class lacks the π discontinuity at $\omega = ½$ except for high values of c as the wavelet approaches Daubechies 6.

Ignoring all those jumps in the phase plots, all that's left is something that gets gradually less linear as it changes from D4 to D6, and so will distort the reconstructed signal more with increasing c. However, these filters are so short that this effect is minimal so the consequences are still workable (Smith & Barnwell [60]).

The very similarity of the Daubechies 6 wavelet and the new wavelet class at $c \sim 5$ shall be further discussed in the next chapter, where the variability of the new class can be exploited by tuning the wavelet to satisfy specific conditions.

5.7 Conclusion

The derivation of 6 tap abc wavelets shows that the use of a parameterized lock condition produces a continuous class of wavelets as a superset of the Daubechies 6. The 6 tap class includes the D4 and D6 with a continuous set in-between. Similarly, as was shown in chapter 4, the 4 tap abc wavelets yield a continuous range from Haar (D2) to D4. Thus a degree of flexibility is offered, giving continuous classes of wavelets complementing the Daubechies integer solutions, D2, D4, D6.

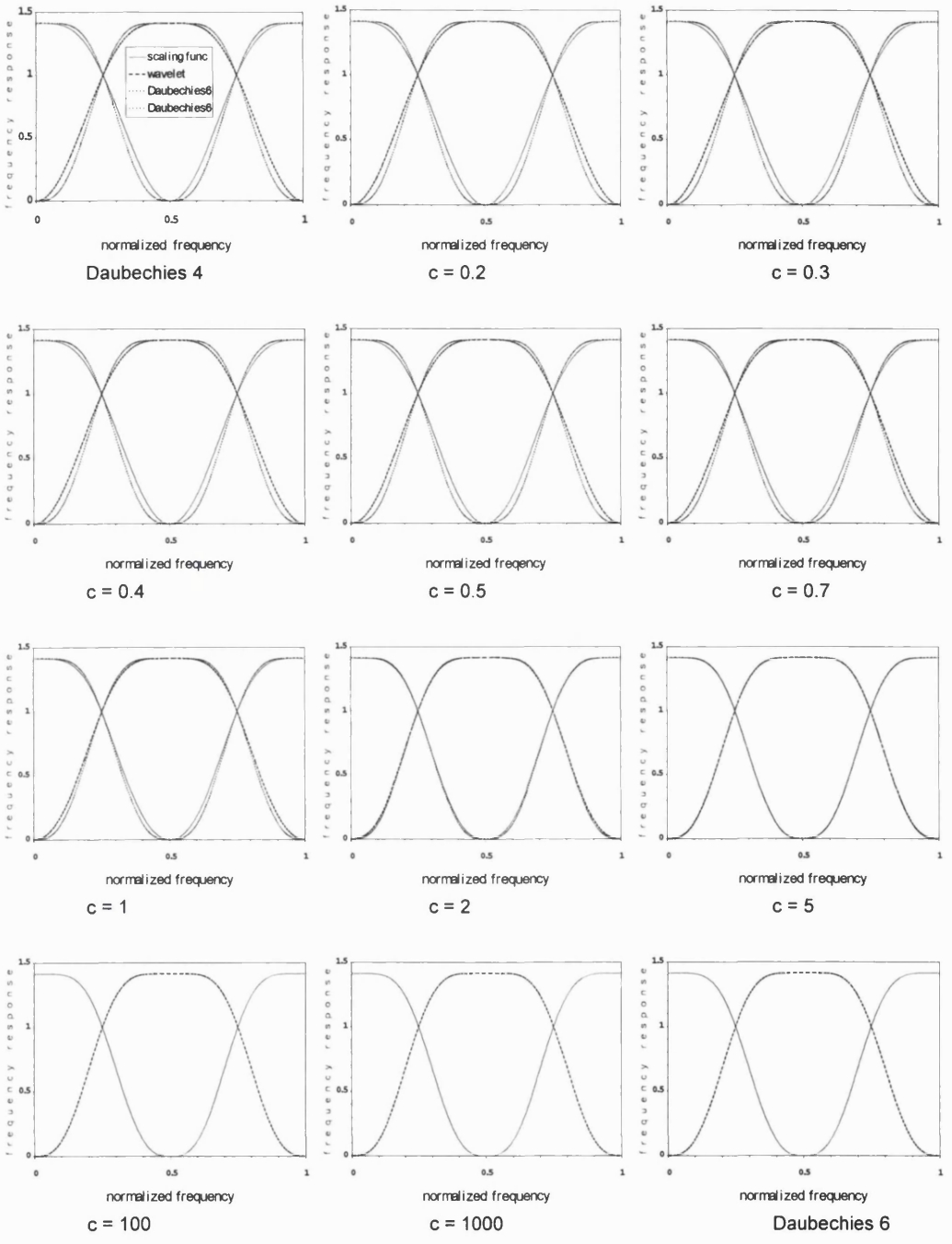


Figure 5.5: absolute frequency response plotted against normalized frequency ω for the exp(x/c) - 1 wavelet class compared with Daubechies 6 (dot)

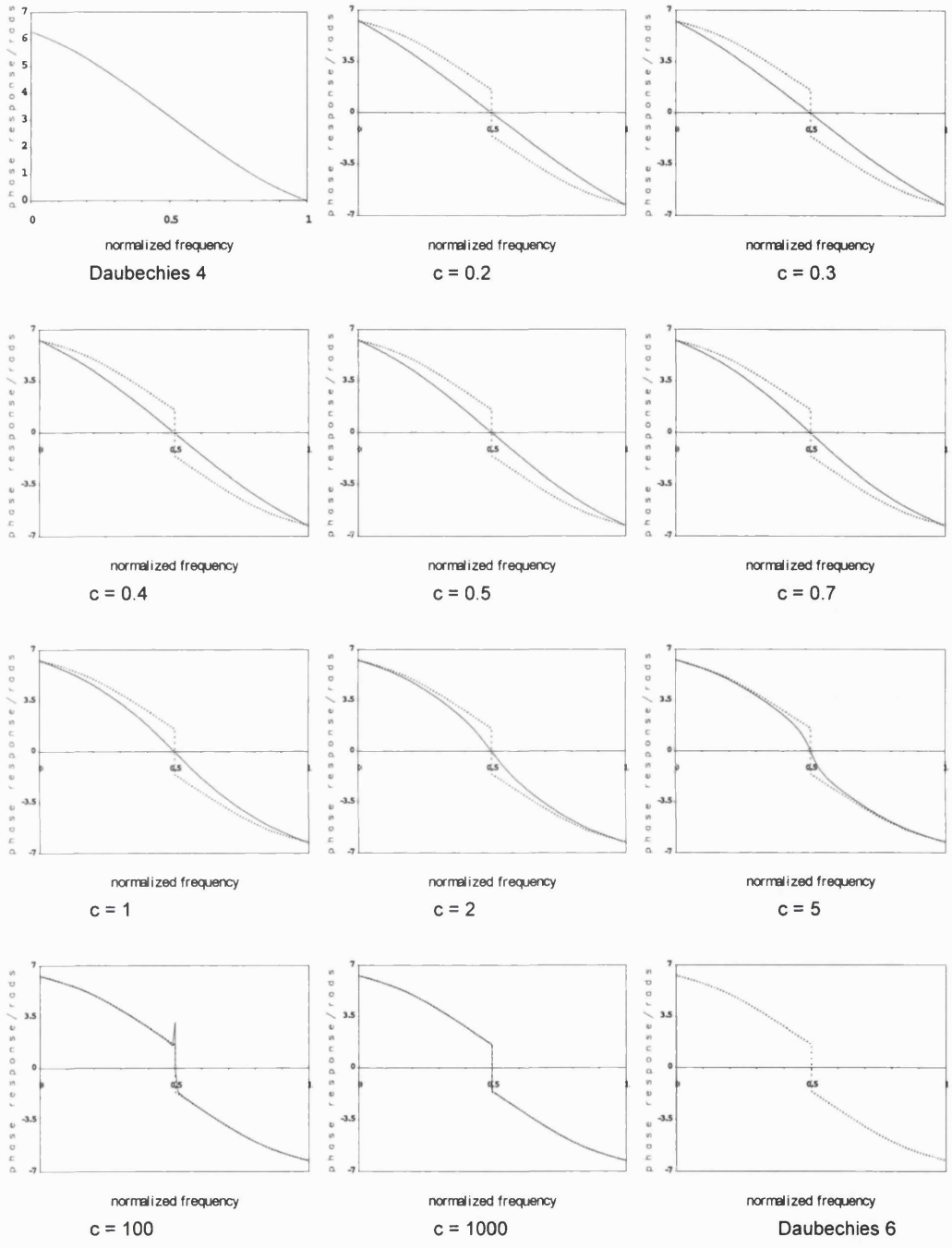


Figure 5.6: phase response of the exp(x/c) - 1 wavelet class (solid) and Daubechies 6 phase response (dot)

6 APPLICATION OF A SPECIFIC GENERIC 6 TAP WAVELET, THE B6

6.1 Introduction

This chapter describes one specific wavelet from the generic 6 tap class, first introduced in 1996 (La Borde [37]). It is designated "B6", meaning **B**inary **6** tap. It has a high c value of about 6 which gives a good emulation of Daubechies 6, and it offers potential to speed the wavelet transform and its inverse by setting one tap exactly equal to a half and adjusting the other taps to maintain perfect reconstruction and smoothness. The use of a binary power tap facilitates the use of arithmetic shift in place of floating point multiplication for one of the correlation components at each scale. It is this property which offers the speed potential. The binary tap wavelet is presented as an interesting "free extra" from the generic class, which although does have this potential to speed up the transform, must be recognized as only marginally beneficial.

6.2 The fast wavelet, B6

Recall the seed equations from chapter 5

$c_0c_2 + c_1c_3 + c_2c_4 + c_3c_5 = 0$	orthogonality	(6.1) (from (5.2))
$c_0c_4 + c_1c_5 = 0$	orthogonality	(6.2) (from (5.3))
$c_5 - c_4 + c_3 - c_2 + c_1 - c_0 = 0$	constant moment	(6.3) (from (5.4))
$-c_4 + 2c_3 - 3c_2 + 4c_1 - 5c_0 = 0$	linear moment	(6.4) (from (5.5))
$-\alpha c_4 + \beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0 = 0$	generic lock	(6.5) (from (5.6))
$c_0 + c_1 + c_2 + c_3 + c_4 + c_5 = \sqrt{2}$	area conservation	(6.6) (from (5.7))

The generic solution can be rearranged as a quadratic in c_3 simultaneously with c_2 chosen as a free parameter for reasons of speed explained below.

$$
\begin{aligned}
& c_3^2\,[-4\alpha^2 + 32\beta^2 + 4\delta^2 - 4\varepsilon^2 - 8\alpha\delta + 8\alpha\varepsilon - 32\beta\delta + 8\delta\varepsilon] \\
+\ & c_3\,[\sqrt{2}(4\alpha^2 - 6\delta^2 + 4\varepsilon^2 - 20\alpha\beta + 18\alpha\delta - 8\alpha\varepsilon + 20\beta\delta - 12\beta\varepsilon - 2\delta\varepsilon) + (32\alpha\beta - 16\alpha\delta - 64\beta\gamma + \\
& \quad 32\beta\varepsilon + 32\gamma\delta - 16\delta\varepsilon)c_2] \\
+\ & c_2^2\,[4\alpha^2 + 32\gamma^2 - 4\delta^2 + 4\varepsilon^2 - 32\alpha\gamma - 8\alpha\delta + 24\alpha\varepsilon - 32\gamma\varepsilon + 8\delta\varepsilon] \\
+\ & c_2\,\sqrt{2}[-6\alpha^2 + 4\delta^2 - 2\varepsilon^2 + 20\alpha\gamma + 18\alpha\delta - 24\alpha\varepsilon - 20\gamma\delta + 12\gamma\varepsilon + 2\delta\varepsilon] \\
+\ & 5\alpha^2/2 + 5\delta^2/2 - 3\varepsilon^2/2 - 20\alpha\delta + 15\alpha\varepsilon = 0
\end{aligned}
$$

The use of the lock condition (6.5) facilitates tuning of the wavelet around the Daubechies 6 while still retaining perfect reconstruction.

The Daubechies 6 minimum phase wavelet taps are:

$c_0 = 0.3326705530$
$c_1 = 0.8068915093$
$c_2 = 0.4598775021$
$c_3 = -0.1350110200$
$c_4 = -0.08544127388$
$c_5 = 0.03522629189$

Coercion of $c_2 = ½$ then yields solution of (6.1) to (6.6) as

$c_0 = 0.3049914936$
$c_1 = 0.7940277829$
$c_2 = 0.5$
$c_3 = -0.1245191882$
$c_4 = -0.09788471237$
$c_5 = 0.03759818644$

with properties very similar to Daubechies 6. This wavelet is denoted B6, meaning **B**inary **6** tap.

The assignment of $c_2 = ½$ then allows the wavelet transform to use arithmetic shift for this tap, in place of floating point multiplication, with a potential speed gain of about 14% for the forward and inverse transforms, based on the removal of one in six multiplications from each correlation of each step of the wavelet transform. These are essentially benefits gratis because very little is lost in terms of smoothness and accuracy. Figures 6.1 to 6.5 illustrate.

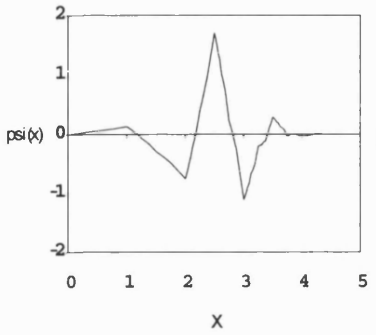


Fig 6.1: Daubechies 6

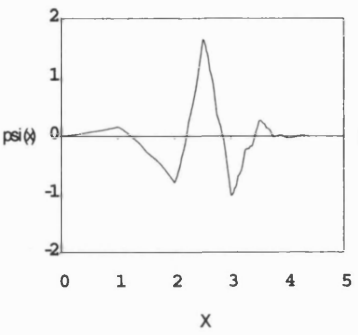


Fig 6.2: Binary tap wavelet B6

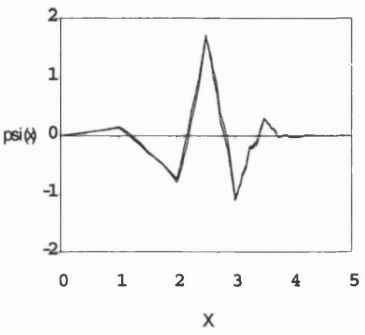


Fig 6.3: D6 dot; B6 solid

Figures 6.1 to 6.3: comparison of Daubechies 6 and binary tap wavelet B6

Figures 6.1 to 6.3 show the Daubechies 6 and the binary tap wavelet B6 in comparison. Figures 6.1 and 6.2 show the D6 and B6 respectively. Figure 6.3 shows these two wavelets superimposed to illustrate their closeness. Naturally such similar wavelets have similar properties. In applications, the B6 wavelet offers performance similar to that of Daubechies 6.

Figure 6.4 shows an electrocardiograph signal, recorded at 1000 samples per second. Figure 6.5 compares reconstruction errors using 3 different wavelets: Daubechies 4, Daubechies 6 and the new binary tap B6 wavelet. The wavelets are distinguished as: Daubechies 4 dash; Daubechies 6 dot; binary tap B6 solid. The graph shows compression error (vertical axis) against compression percentage (horizontal axis). The compression was done by simple thresholding of the wavelet transform, setting to zero so many percent of the coefficients as shown. Notice that up to 94% compression the 3 wavelets offer comparable fidelity. Between 94 and 99 percent compression, the Daubechies 6 is noticeably better than Daubechies 4. However, in this range, the B6 performs much like the Daubechies 6, thus enabling it to be used in place of the Daubechies 6 without serious loss of accuracy, and at the same time offering the additional speed already documented.

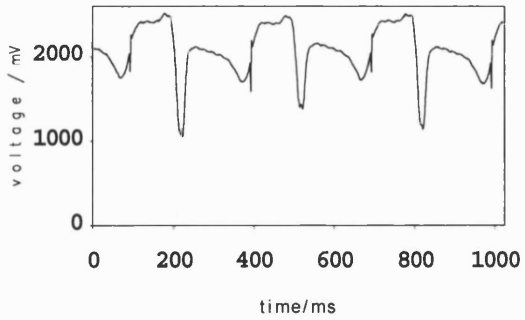


Fig 6.4: ECG signal

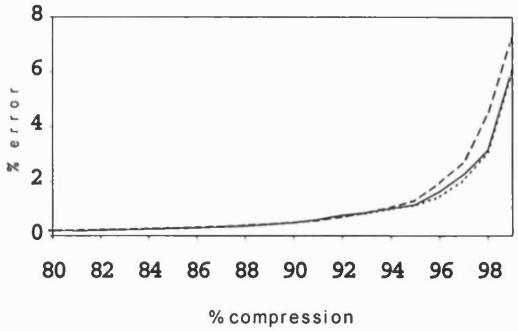
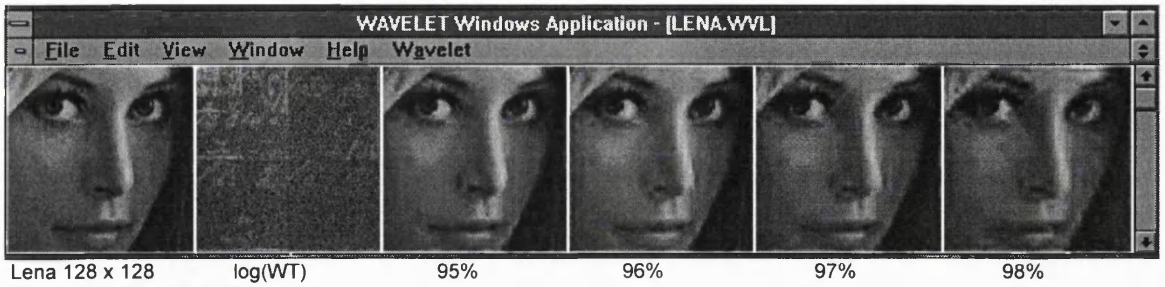


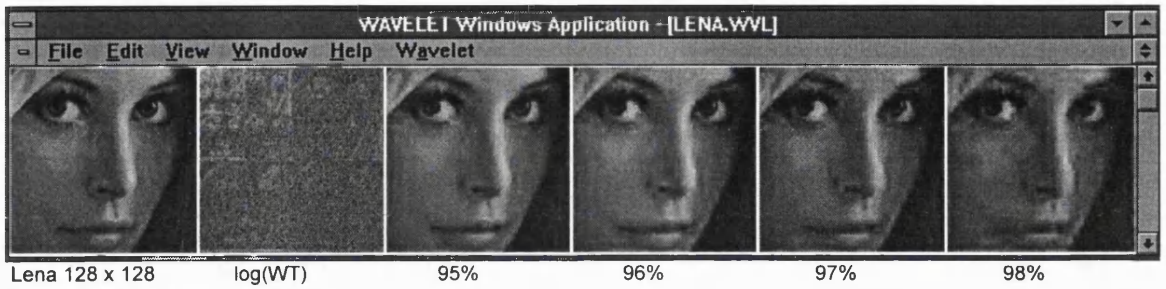
Fig 6.5: % error against % compression.
D4 dash, D6 dot, B6 solid

Figures 6.4 and 6.5: electrocardiograph compression; error comparison for Daubechies 4 and 6 and the B6 wavelet

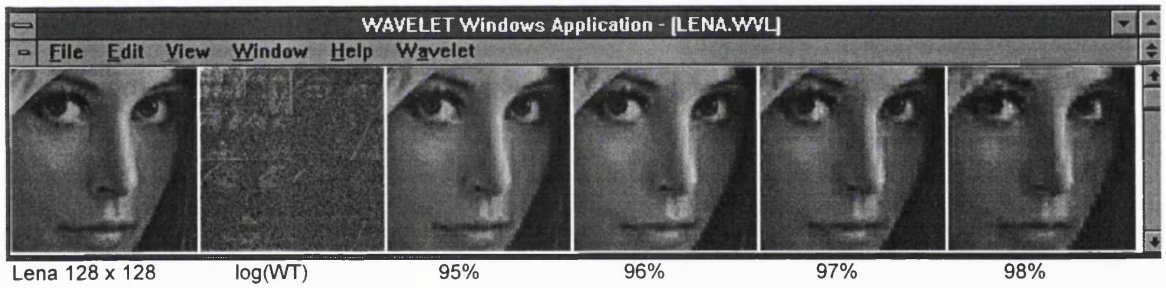
Figures 6.6 to 6.8 show how the B6 wavelet compares with the Daubechies 4 and 6 tap wavelets. Figure 6.9 shows L^1 reconstruction errors (vertical axis) plotted against compression percentage (horizontal axis) for a 128 x 128 pixel greyscale Lena image. This is a comparative graph to show the similarity of Daubechies 4, Daubechies 6 and the binary tap B6 wavelet. As in figure 6.5, the lines are keyed as: Daubechies 4 dash; Daubechies 6 dot; binary tap B6 solid. Clearly there is little to choose on grounds of this measure and it is well recognized that images rely more on aesthetic properties dependent on the smoothness of the underlying wavelet, rather than the quantitative simplicity of absolute error (de Queiroz [20], Li [38]). Figures 6.6 to 6.8 illustrate these effects. Figures 6.6 and 6.7 show the Lena image with reconstructions under 95% to 98% compression. The two sequences show that smoothness is maintained equally well with the B6 as compared with the D6, whereas figure 6.8 shows the smoothness degrading with higher compression for the D4 wavelet.



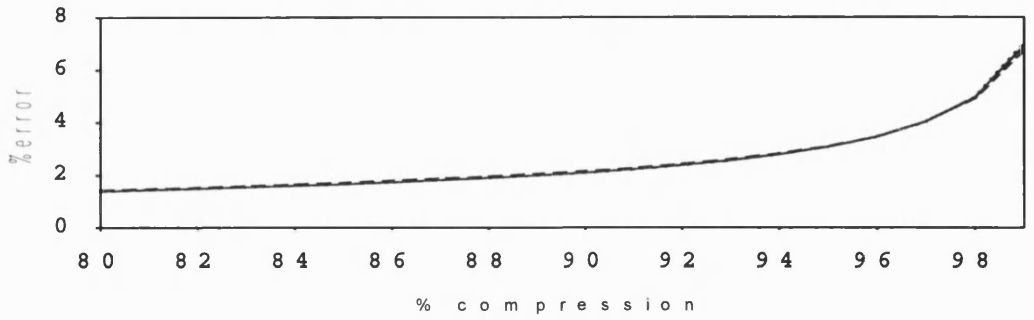
Lena 128 x 128 log(WT) 95% 96% 97% 98%
Figure 6.6: D6 Lena compression



Lena 128 x 128 log(WT) 95% 96% 97% 98%
Figure 6.7: B6 Lena compression



Lena 128 x 128 log(WT) 95% 96% 97% 98%
Figure 6.8: D4 Lena compression



% error against % compression: D4 dash; D6 dot; B6 solid

Figure 6.9: compression error comparison for Daubechies 4 and 6 and B6 on 2D Lena

To summarize this section, the graphic illustration highlights the similarity of properties between the D6 and B6 wavelets thus enabling the B6 to be used wherever the D6 is used; the advantage being that B6 offers a speed improvement from its specially derived taps, thus enabling a general acceleration of the wavelet transform with minimal loss of quality.

6.3 Software implementation

Throughout this discussion the virtue of the B6 is based on c_2's exact equality to ½, thus enabling an arithmetic shift to replace a floating point multiplication with consequent saving of execution time. Standard software languages and off-the-shelf computers do not support float shift operation so the shift must be performed at byte level. In C and C++ with four byte floats the operation to shift right one place for x is

```
*((int*)&x+1)-=128;
```

In tests on a 65MHz dan 486 PC the speed ratio of shift to multiply is 13.69% ie 0.1369.

This ratio was calculated by running the respective code segments in *for loops*. The variable name *halfa2* is descriptive for this exercise only and is not used beyond this example.

500,000,000 loops
assignment and shift

time: 657 seconds

```
{
   halfa2 = a[2];
   *((int*)&halfa2+1)-=128;
}
```

500,000,000 loops
assignment and float multiply

time: 1073 seconds

```
{
   halfa2 = a[2];
   y = C2*halfa2;
}
```

500,000,000 loops
assignment only

time: 591 seconds

```
{
   halfa2 = a[2];
}
```

Subtracting the time for assignment only yields:

float shift = 657 - 591 = 66 seconds

float multiply = 1073 - 591 = 482 seconds

Hence the speed ratio given above is explained: $\frac{66}{482}$ = 0.1369 ie 13.69%.

Given that one 6 tap correlation needs six multiplications the relative D6 cost is $\frac{6}{6}$ and the relative B6 cost is $\frac{5.1369}{6}$ = 85.615% ie the B6 gives ~14% improvement.

The C code above for the float shift is explained by considering the register representation of a floating point number, x. It has four bytes, The binary exponent is held in the second pair of bytes in the shaded bits. The float shift is achieved by decrementing this exponent by 1.

The float can be thought of as two pairs of two bytes, each pair having an integer pointer address.

The first operation

(int*)&x

coerces the address of x into an integer pointer.

((int)&x+1)

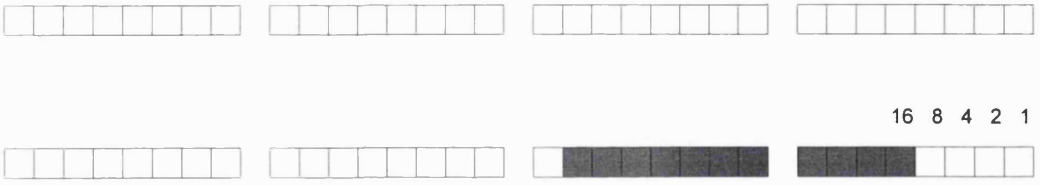
then increments this address by 1 to access the second pair of bytes and return the integer value.

The next operation,

((int)&x+1)-=128;

subtracts 128 from this integer, ie the binary exponent is decremented by 1. The value 128 in the integer is equivalent to 1 in the shaded bits because the first shaded bit is the highest bit of the last byte, as shown above; its value is 128.

The arrangement for a double is similar:



In this case there are 8 bytes and the exponent is held in 11 bits of the last two bytes. The corresponding C code statement to shift a double, x, is

((int)&x+3)-=16;

The result of these float and double shift operations is to halve the argument, x, in each case. This has no practical use in the standard wavelet transform in the correlation process, $\mathbf{y} = A\mathbf{x}$ (as explained in chapter 3) because the values of x (vector) must be preserved for subsequent rows; it is the analysis matrix A which contains the taps, and so the halving occurs as a result of x_k (say) being multiplied by $c_2 \equiv ½$ in A. As a reminder, the wavelet taps are held in A in the form:

$$A = \begin{bmatrix} c0 & c1 & c2 & c3 & c4 & c5 & & & & \\ c5 & -c4 & c3 & -c2 & c1 & -c0 & & & & \\ & & c0 & c1 & c2 & c3 & c4 & c5 & & \\ & & c5 & -c4 & c3 & -c2 & c1 & -c0 & & \\ & & & & & \ddots & & & & \\ & & & & & & \ddots & & & \\ & & & & & & & \ddots & & \\ & & & & & & & & \ddots & \\ c2 & c3 & c4 & c5 & & & & & c0 & c1 \\ c3 & -c2 & c1 & -c0 & & & & & c5 & -c4 \end{bmatrix}$$

Thus a particular y_i becomes

$$y_i = c_0x_k + c_1x_{k+1} + c_2x_{k+2} + c_3x_{k+3} + c_4x_{k+4} + c_5x_{k+5} \qquad (6.7)$$

This assignment uses 6 multiplications, but by virtue of $c_2 \equiv ½$ simplifies to

$$y_i = c_0x_k + c_1x_{k+1} + ½x_{k+2} + c_3x_{k+3} + c_4x_{k+4} + c_5x_{k+5}$$

In this case x_{k+2} avails itself to the shift halving in the correlation, but must not itself be changed. This is overcome by using modified taps T_n equal to twice the actual c_n values, and exploiting the shift half on y_i which does itself then need to be halved. The notation T_n is chosen to mean Twice the c_n value. Thus the above operation is replaced by

$$\left.\begin{array}{l} y_i = T_0x_k + T_1x_{k+1} + x_{k+2} + T_3x_{k+3} + T_4x_{k+4} + T_5x_{k+5} \\ y_i \mathrel{/}= 2 \end{array}\right\} \qquad (6.8)$$

which uses only 5 multiplications and one shift.

(The notation /= is the divide and assign operator meaning divide the left operand by the right operand. Thus y /= 2 is equivalent to y becomes y/2. The actual division is done by the bit shift operation outlined above).

This saving occurs for every vector multiplication at every scale in the wavelet transform. Thus any saving in this single operation should pro rata be represented throughout the computation. The transforms using operations (6.7) (conventional Fast Mallat transform) and (6.8) shall be identified by the names "D6 WT" and "B6 WT" respectively.

6.4 Computational results

The C code segments representative of the above two assignments are:

```
wksp[i]=C0*a[ii]+C1*a[ii+1]+C2*a[ii+2]+C3*a[ii+3]+C4*a[ii+4]+C5*a[ii+5];     }
wksp[nh+i]=C5*a[ii]-C4*a[ii+1]+C3*a[ii+2]-C2*a[ii+3]+C1*a[ii+4]-C0*a[ii+5];  }  D6 WT
```

for (6.7) and

```
wksp[i]=T0*a[ii]+T1*a[ii+1]+a[ii+2]+T3*a[ii+3]+T4*a[ii+4]+T5*a[ii+5];        }
*((int*)&wksp[i]+3)-=16;                                                     }
                                                                             }  B6 WT
wksp[nh+i]=T5*a[ii]-T4*a[ii+1]+T3*a[ii+2]-a[ii+3]+T1*a[ii+4]-T0*a[ii+5];     }
*((int*)&wksp[nh+i]+3)-=16;                                                  }
```

for (6.8).

The duplication of lines in the workspace (wksp) assignment attributes to the two separate correlations for the scaling function and wavelet. These two code segments were timed using the C *time* function, which has 1 second precision. As such, large loops were used for the timings. The results, in seconds, are shown in table 6.1.

	D6 WT	B6 WT	loop only
500,000 loops	4	4	0
5,000,000 loops	38	35	3
50,000,000 loops	382	350	32

Table 6.1: comparative time in seconds of the D6 and B6 transform segments

The times adjusted for the loop overhead are calculated by simply subtracting the relevant loop cost: table 6.2.

	D6 WT	B6 WT
500,000 loops	4	4
5,000,000 loops	35	32
50,000,000 loops	350	318

Table 6.2: comparative time in seconds of the D6 and B6 transform segments adjusted for the loop overhead

This indicates that the B6 WT is faster than the D6 WT by some 8.6% in 5 million loops.

The cause of this rather modest reduction is to be found in the respective times of multiplication and addition: table 6.3.

	double addition	double multiplication		
	a[0]+a[1]	C0*a[1]	a[0]*a[1]	loop only
5,000,000 loops	7	8	7	3
50,000,000 loops	66	78	75	32
500,000,000 loops	657	787	746	320

Table 6.3: comparative time in seconds of double addition and multiplication

Table 6.4 shows these times adjusted for the loop overhead.

	double addition	double multiplication	
	a[0]+a[1]	C0*a[1]	a[0]*a[1]
5,000,000 loops	4	5	4
50,000,000 loops	34	46	42
500,000,000 loops	327	467	426

Table 6.4: comparative time in seconds of double addition and multiplication adjusted for the loop overhead

Thus the ratio of multiplication to addition is 467/327 = 1.428. Hence the addition cost is not insignificant compared to the multiplication cost. This is disappointing because it means the B6 wavelet cannot gain significant advantage over the D6 by the removal of a single multiplication. The situation deteriorates further when considering the whole wavelet transform. Table 6.5 shows the times for the two wavelets using their respective wavelet transforms in full on a signal of length 1024.

	full D6 WT	full B6 WT
500 loops	12	12
5,000 loops	118	117

Table 6.5: comparative time in seconds of the full D6 and B6 wavelet transforms

The new B6 WT is only marginally faster than the standard Mallat/D6 transform. It's not appropriate here to enter a detailed analysis of computer hardware and cycle costs (such issue are addressed by Cody [15], Dress [22] and Resnikoff [53]), but it must be noted that the computation time is dominated by loops and register access, thus reducing the shift benefit.

Still, the B6 does hold promise for a custom chip which could be better adapted to exploit this. At this time though, the discussion is terminated with the observation that while the B6 documented here yields no immediate step forward, it does have the *potential* to accelerate the wavelet transform, as illustrated by the above figures.

6.5 B6's place among generic 6 taps

Recall that the generic 6 tap wavelets of chapter 5 were derived from 6 seed equations containing the lock condition

$$-\alpha c_4 + \beta c_3 - \gamma c_2 + \delta c_1 - \varepsilon c_0 = 0 \qquad \text{generic lock} \qquad (6.5)$$

The specific 6 tap wavelet, B6, derived in this chapter came from a slightly different route in that no lock condition was specified; rather a requirement that $c_2 = ½$ consumed the degree of freedom usually found in the lock condition. It is possible, however, to go back and investigate what particular value of c in the lock condition would yield the B6. This then accommodates the specific B6 in the generic class.

There is no explicit method for such a calculation, instead Mathematica was used to search iteratively using the 5 conventional equations of orthonormality and moment conditions with the additional lock condition f(x,c) = exp(x/c) - 1. A good starting point in the search is c = 5 since it is known that the B6 looks like the D6 and that the generic wavelets start to look like Daubechies 6 at about c = 5 (figure 5.4 in chapter 5). Figure 6.10 shows a close-up of the search region for taps varying with c, (compare figure 5.2 of chapter 5) and the value ½ is crossed on the c_2 curve. For clarity only the c_0, c_1 and c_2 curves are shown. This shows why the value of ½ was drawn from c_2 in preference to c_1 which also intersects ½; namely that such a solution would yield a wavelet very close to Daubechies 4, and clearly offer no advantage since 6 taps would be needed to approximate a 4 tap wavelet.

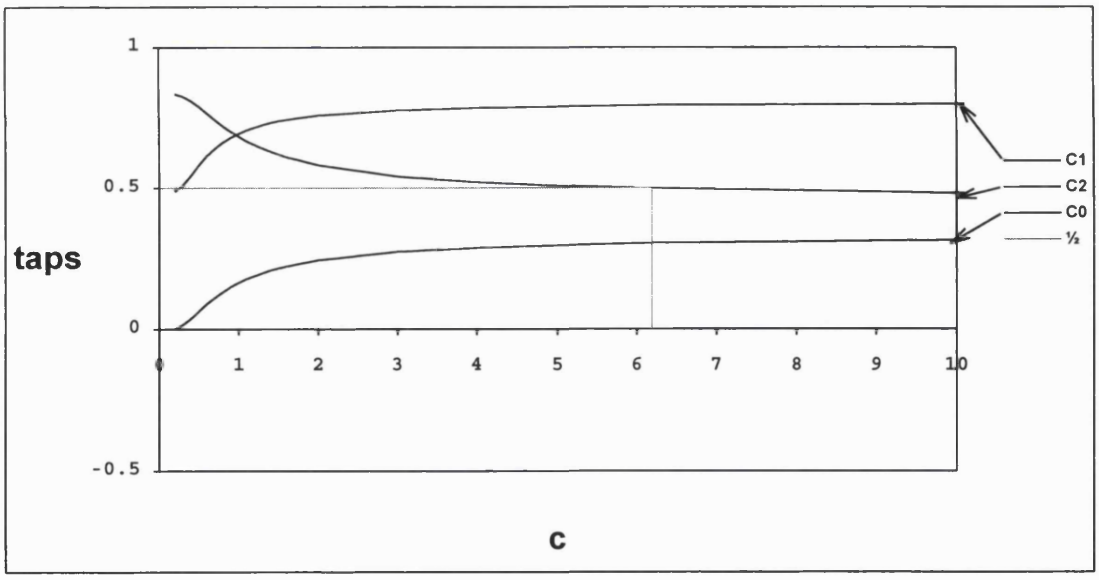


Figure 6.10: close-up of the taps in the $c_2 = ½$ search region

Table 6.6 shows the results of the iterative search, not exactly binary since intuitive input lead directly to the outcome. The reader is spared intermediate dead ends. Iteration stops at 7 decimals where results are sufficient to demonstrate B6 correspondence.

c	c_0	c_1	c_2	c_3	c_4	c_5
5	0.2982470	0.7905644	0.5096134	-0.1214679	-0.1007537	0.0380102
6	0.3041125	0.7935837	0.5012564	-0.1241324	-0.0982622	0.0376555
6.1	0.3045916	0.7938260	0.5005717	-0.1243436	-0.0980565	0.0376243
6.2	0.3050550	0.7940597	0.4999091	-0.1245470	-0.0978573	0.0375940
6.15	0.3048252	0.7939439	0.5002377	-0.1244463	-0.0979562	0.0376091
6.16	0.3048715	0.7939673	0.5001715	-0.1244665	-0.0979363	0.0376060
6.17	0.3049176	0.7939905	0.5001056	-0.1244868	-0.0979164	0.0376030
6.18	0.3049635	0.7940137	0.5000399	-0.1245069	-0.0978967	0.0376000
6.19	0.3050093	0.7940369	0.4999744	-0.1245270	-0.0978770	0.0375970
6.185	0.3049865	0.7940252	0.5000071	-0.1245170	-0.0978868	0.0375985
6.186	0.3049910	0.7940275	0.5000005	-0.1245190	-0.0978848	0.0375982
6.18608	0.3049914	0.7940276	0.5000000	-0.1245190	-0.0978847	0.0375981

Table 6.6: iterative search for $c_2 = ½$

It must be noted that this exercise is presented only to illustrate that a particular c value does exist corresponding to the B6. For the actual derivation, as explained in section 6.2, c_2 is *set* to ½ in the seed equations and the other taps solved explicitly by back substitution as documented in chapter 5. Thus c_2 does not approximate ½ (as table 6.6 might suggest); it is exactly equal to ½.

6.6 Conclusion

The B6 has interesting mathematical properties, namely that it is a very close substitute for the Daubechies 6. Its advantage is seen here to be theoretical, and while this is satisfactory from a mathematical viewpoint, it leads to no application for off-the-shelf computation. Its saving grace is that a usable speed gain may in future be won by specific hardware customization.

As shown in section 6.4 the B6 WT is faster than the D6 WT by some 8.6% in 5 million loops, however, this could be improved with dedicated hardware architecture.

The following chapter pursues generic 6 tap wavelets further, concentrating on the entire class for use in adaptive selection for 1 dimensional signal compression.

7 APPLICATION OF THE GENERIC 6 TAP WAVELET CLASS

7.1 Introduction

This chapter looks at chirps in relation to compression using generic 6 tap wavelets compared with Daubechies. For this analysis it must be kept in mind that linear phase filters offer advantages on step type signals, ie ones displaying sharp edges (Abousleman [1], Heller [28]). Thus the Daubechies 4 is better than Daubechies 6 for step-like functions. Another way of looking at this is to consider the filter length with respect to the local signal feature. The Daubechies 4 wavelet will give smaller errors on sharp edges due to the smaller local neighbourhood contribution. Conversely, the longer Daubechies 6 will give smaller errors on smoother longer curves precisely because of their greater local neighbourhood contributions, ie averaging over longer regions.

Consider the simple generalized chirp, $\sin(e^x)$, modified in some way to add sharp edges. The Daubechies 4 will work well on these step-like features, and the Daubechies 6 will do better on the smoother components. A wavelet somewhere in-between D4 and D6 would intuitively be the best single wavelet for such a compression. The c class generic wavelets offer such a middle option, and it is possible, for a given desired error, to find that one specific c value which gives the optimum wavelet for this task.

7.2 Top and bottom chirps

The generalized chirp $\sin(e^x)$, is augmented with step-like features by saturating the amplitude for all positive values such that

$$y = \begin{cases} 1, & \sin(e^x) > 0 \\ \sin(e^x), & \sin(e^x) \le 0 \end{cases} \qquad \text{denote as } y = top[\sin(e^x)]$$

This is denoted the *top* function since the top of the chirp is saturated.

Similarly, negative values of the chirp can be bound to -1 to give negative step-like features.

$$y = \begin{cases} \sin(e^x), & \sin(e^x) \ge 0 \\ -1, & \sin(e^x) < 0 \end{cases} \qquad \text{denote as } y = bottom[\sin(e^x)]$$

This is denoted the *bottom* function since the bottom of the chirp is saturated.

The compression discussed here works on a sample of length 1024, and so to avoid an astronomical frequency range on $x \in [0, 1023]$, the variable x is reduced by a factor 200 to give a more comfortable x range of zero to about 5. Also, chirps most commonly found in nature are down chirps, indicating a loss of energy with time (Freeman [23]), so the x value is subtracted from 6 to give lowering of frequency with x.

The basic down chirp is shown in figure 7.1.

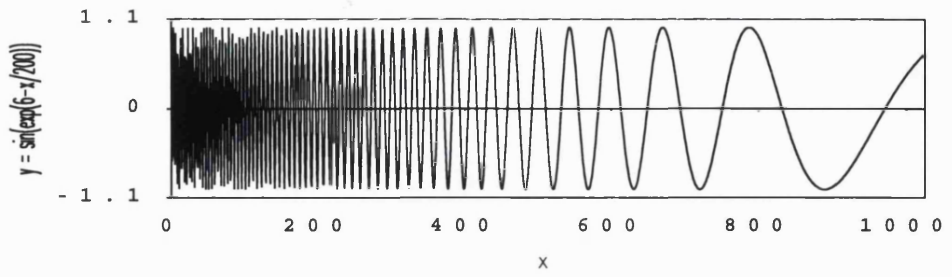


Figure 7.1: down chirp, y = sin(exp(6-x/200))

This basic chirp is modified as described above to produce a top chirp, figure 7.2, and a bottom chirp, figure 7.3, both of which have the step edges suitable to Daubechies 4, and the smooth curves suitable to Daubechies 6.

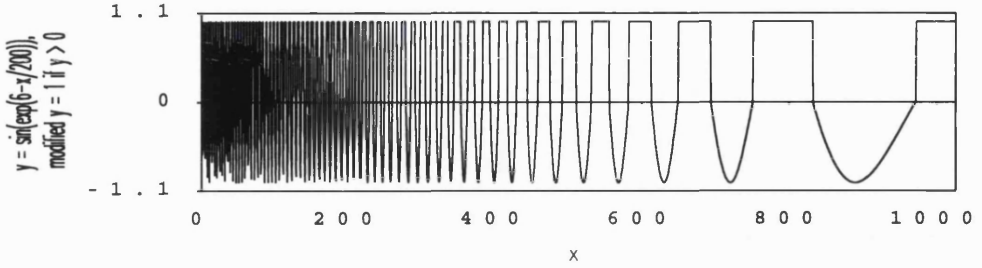


Figure 7.2: top chirp, y = sin(exp(6-x/200)), modified y → 1 if y > 0

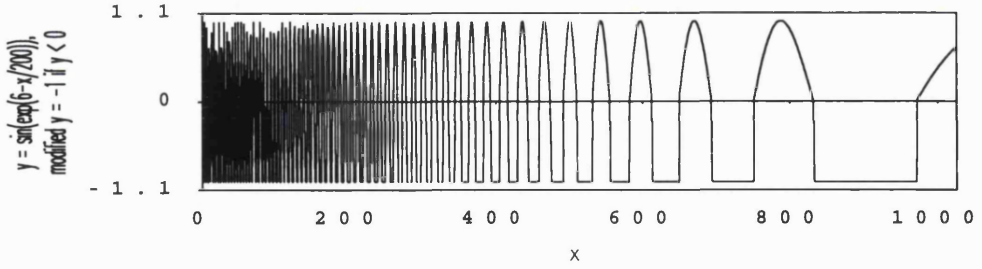


Figure 7.3: bottom chirp, y = sin(exp(6-x/200)), modified y → -1 if y < 0

The objective now is to find the best c class wavelet to compress the signal. In this trial, 20 different chirps are considered, formed by dilating the basic top and bottom chirps by a factor n:

$$y = top[\sin(\exp(6-x/200)/n)]$$

and

$$y = bottom[\sin(\exp(6-x/200)/n)]$$

n ranging over [1, 10]. The two classes of top and bottom chirps are shown respectively in figures 7.4 and 7.5.

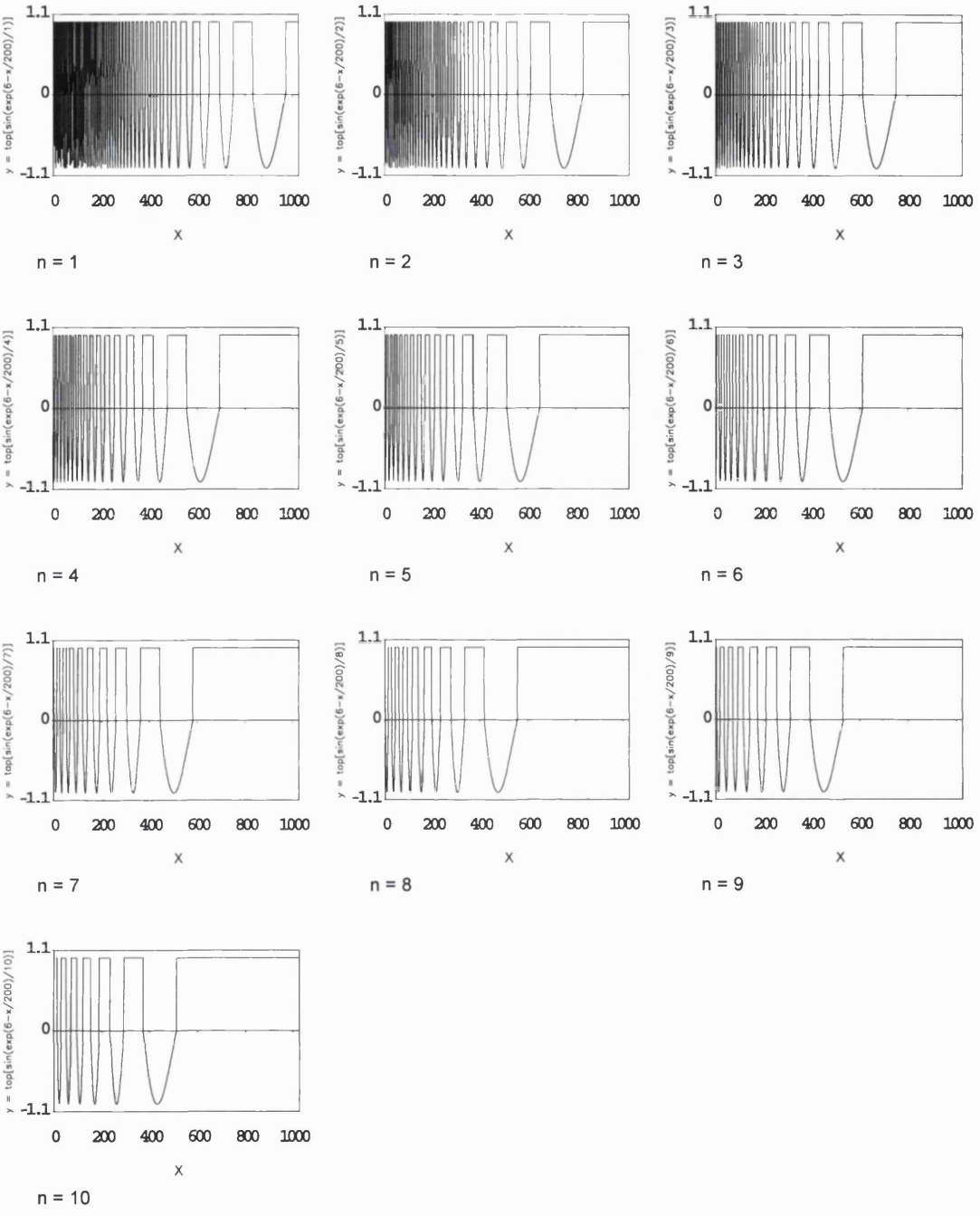


Figure 7.4: class of top chirps, y = top[sin(exp(6-x/200)/n)] for n = 1 to 10

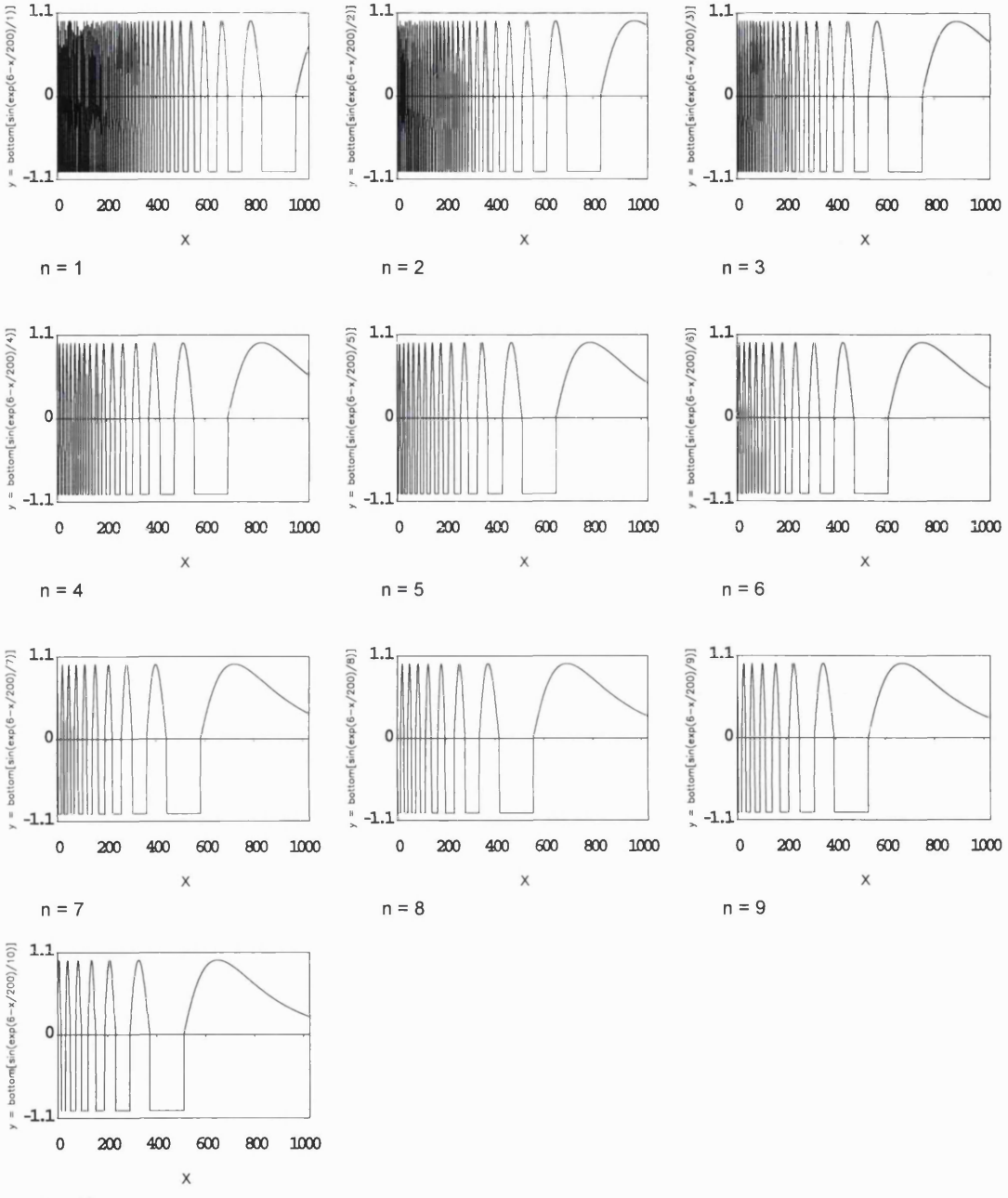
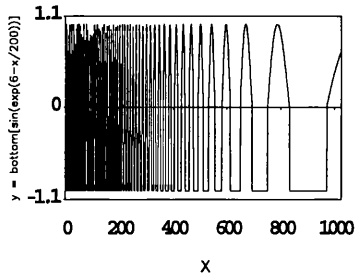


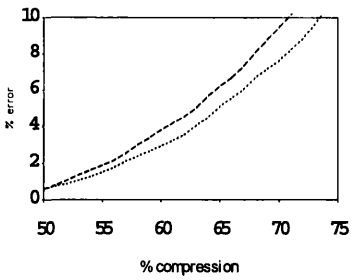
Figure 7.5: class of bottom chirps, y = bottom[sin(exp(6-x/200)/n)] for n = 1 to 10

For these ranges of chirps, the c class reconstruction errors are compared with those of the Daubechies 4 and 6, and the "best" wavelet is deemed to be the one which offers the highest compression rate for a given reconstruction error of 5%; this number being chosen as a reasonable benchmark for this test. While other acceptable error levels would produce similar selection of a best wavelet, the exact c value would not always be the same due to the varying place of intersection of the c class and Daubechies error curves. The range is illustrated in figure 7.6 for bottom chirp n = 1. In this case the best wavelet occurs at c = 2.0, where for high compression rates, this particular wavelet gives smaller reconstruction errors than Daubechies 4 and 6.

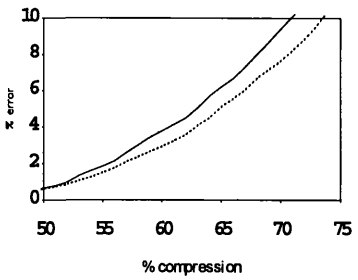
Figure 7.7 shows the reconstructions of the n = 1 bottom chirp.



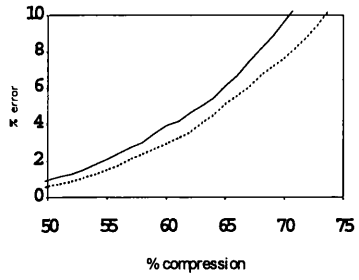
down chirp y = bottom[sin(exp(6-x/200))]



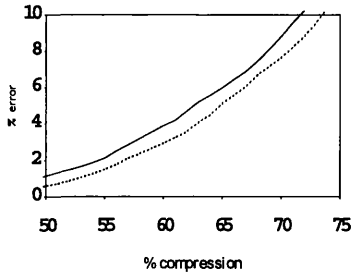
Daubechies 6 error (dot)
Daubechies 4 error (dash)



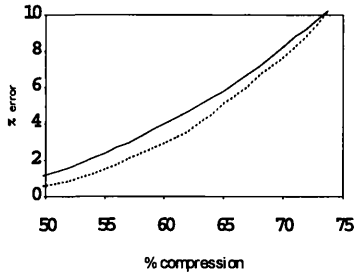
c = 0.2



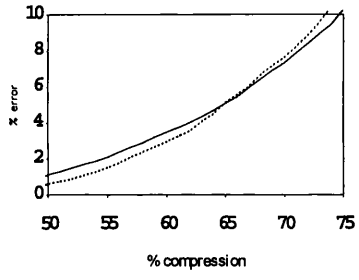
c = 0.4



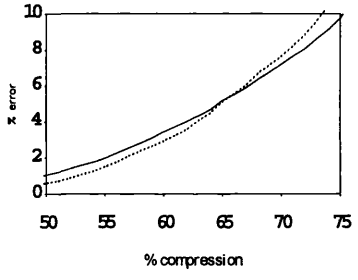
c = 0.6



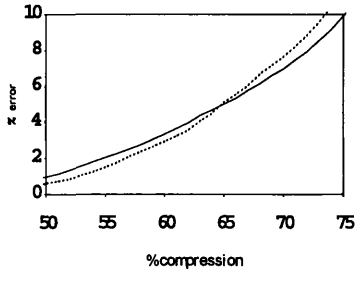
c = 0.8



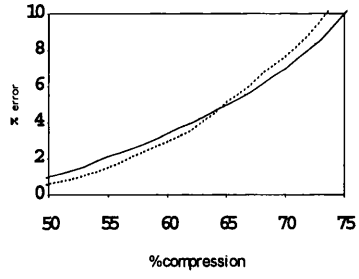
c = 1.0



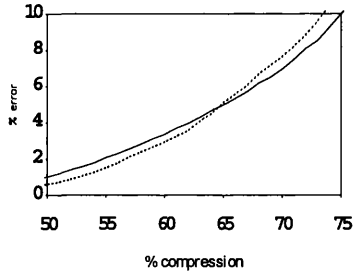
c = 1.2



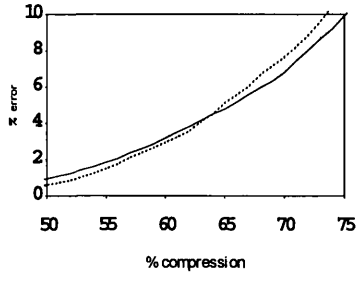
c = 1.4



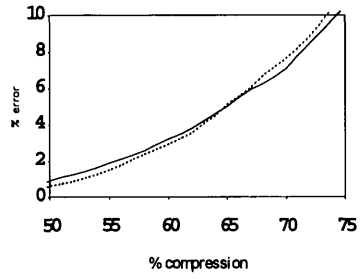
c = 1.5



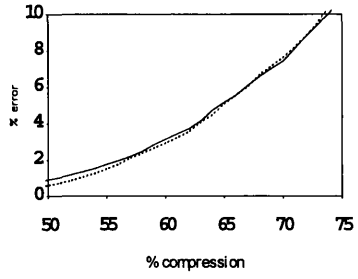
c = 1.6



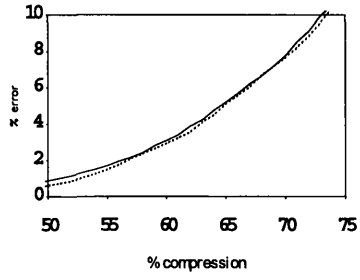
c = 2.0 (best)



c = 3

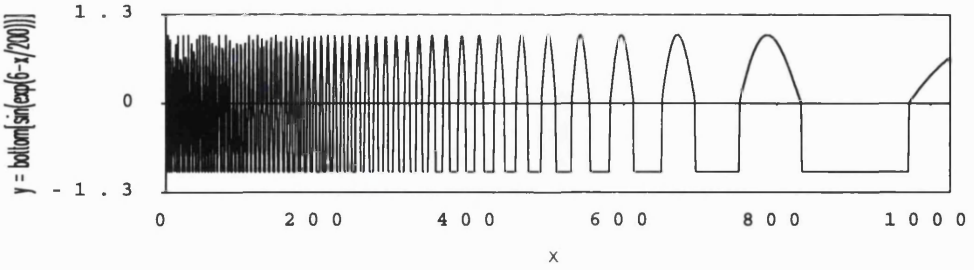


c = 4

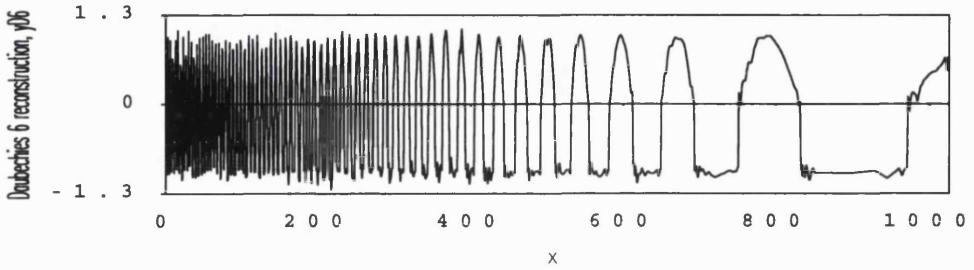


c = 5

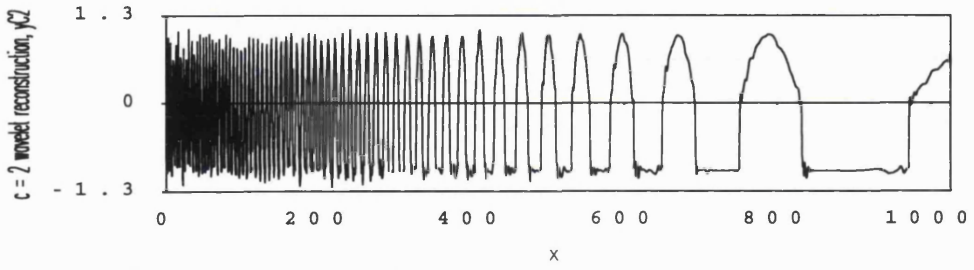
Figure 7.6: % error against % compression for generic wavelets compared with Daubechies 6 (dot)



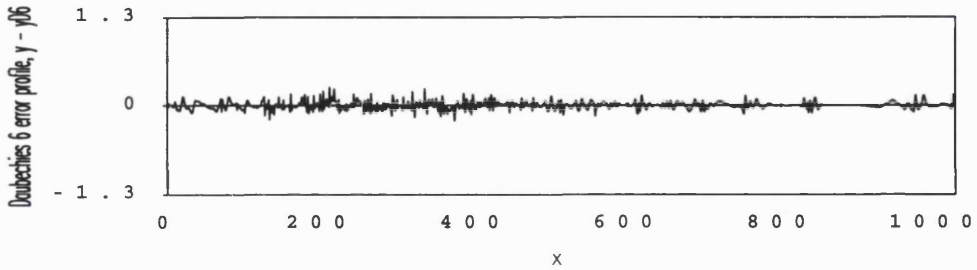
Original signal: down chirp y = bottom[sin(exp(6-x/200))]



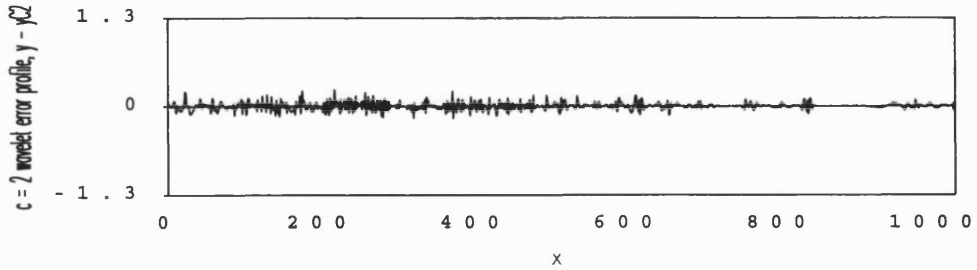
Daubechies 6 reconstruction, yD6



c = 2 wavelet reconstruction, yC2



Daubechies 6 error profile, y - yD6



c = 2 wavelet error profile, y - yC2

Figure 7.7: comparison of reconstructed signal and errors for 65% compression using Daubechies 6 and c = 2 wavelet

Figure 7.7 is shown for completeness only and is not meant to be illustrative of any wavelet merits because it shows the reconstructions at that compression rate where the two wavelet errors are most similar, ie just past the intersection of error curves in figure 7.6 for c = 2.

Considering again figure 7.6, the significant point is that the c = 2 error curve *does* intersect the Daubechies 6 error curve in a useful place, ie where the compression error ≈ 5%. The same advantage occurs for more than half of the 20 chirps considered in this exercise. The results are shown in table 7.1.

n	"top" chirp best c	"bottom" chirp best c
1	2.0	2.0
2	Daubechies 6	Daubechies 6
3	4.0	Daubechies 6
4	Daubechies 6	2.0
5	1.4	4.0
6	Daubechies 6	0.5
7	1.6	4.0
8	Daubechies 4	1.3
9	Daubechies 6	2.0
10	1.4	Daubechies 6

Table 7.1: best wavelet choice for compression of the chirp class y = top/bottom[sin(exp(6-x/200)/n)], x ∈ [0,1023]

7.3 Conclusion

This chapter has examined the selection of particular c class wavelets for compression of chirps. In this case selection was by linear search, a continual sweep of c values being examined for a particular chirp in order to determine the best; best being defined as the one which error curve most quickly intersects the Daubechies 6 with respect to increasing compression.

8 TRANSIENT DETECTION

8.1 Introduction

As already documented, one of the principal advantages of the wavelet transform is its ability to locate features within signals. The application discussed here is the detection and location of transients within high voltage power transmission systems. The specific area of examination is the transients that occur when switching off (open circuit) high voltage relays. Data were provided by Dr Kwong Tong of The National Grid Company (NGC), Leatherhead office. His interests lie in condition monitoring, machine health and incident investigation. The National Grid Company is concerned with two types of transient events: planned and unplanned. Various unplanned transients occur across the country on transmission lines, caused by equipment failure and lightning strikes. A group of researchers at the University of Bath (Johns, Aggarwal and Bo [33]) has developed an analogue transient detector specifically for NGC. It uses analogue inductor/capacitor filtering to identify high frequency events, and is dedicated to detecting fault conditions. The other aspect of electrical transients is that of planned events. The case discussed in this chapter is concerned with health monitoring of high voltage relays in their switching state.

The relays operate in the range of 100,000 volts, 2,000 amps. Signals are recorded in 3-phase voltage and current. The work here looks only at one phase, and only voltage. The signal is recorded in an interval around contact separation.

The switching process displays two or three transient characteristics, the location of which give information about the health of the relay. Ideally, contact separation should occur at maximum voltage. If the relay first fails to open, this is called "reignition". The relay can make a second attempt at the next voltage extremum, ie half a cycle later. The time interval of these events indicates whether the relay is operating optimally or not, and can aid preventive maintenance based on this interval's drift from π. Hence the wavelet's capacity to locate separate transient features can immediately quantify relay health.

8.2 Background

In addition to the Bath group already mentioned, many other people are using wavelets in transient analysis. The most practical of these is Robertson [55] who examines 3-phase capacitor switching on 60Hz transmission lines in the USA. He is concerned more with fault classification than exact pinpointing in time. In Robertson's case discrete quadratic splines are used to give 7 levels of decomposition, the local extrema of which are used as input for a Bayesian classifier. In this study the ElectroMagnetic Transient Program (EMTP) was used to provide simulated transients.

Similarly, Assef [6] has designed a method for classifying fault currents on power distribution systems. She uses a modified continuous Morlet wavelet, multiplied by a quadratic with a parameter constraint to ensure admissibility. Again, this is concerned with fault classification, rather than temporal location, with the objective of damage limitation. The events under study are intermittent self-extinguishing arc faults where transient components dominate steady state components. The wavelet transform is used as a pre-processor for an artificial neural net (ANN) to quickly identify faults in relays to limit further damage. The use of neural nets is justified on the dual grounds of inadequate mathematical model and system complexity with its inherent combinatorial possibilities. Classification is binary: faulty signals / sound signals, without further classification of fault type. A 20kV, 50Hz power transmission is simulated with high speed power system transients, again using EMTP, as in Robertson's case. The actual wavelet used is $\Psi(t) = (1+\sigma|t|+\frac{1}{2}\sigma^2t^2)e^{-\sigma|t|}e^{2i\pi t}$, with σ chosen to satisfy admissibility. The examining region around the transient samples extend for ~2 cycles from the instant of relay contact closure and

are themselves decaying sinusoids of frequency ~800Hz. A sliding window gives data for 31 sample intervals. The final decision, fault / no fault, is based on a voting decision from the 31 ANN outputs, one for each of the 31 windows. Performance degrades with shortening transients due to their decreased presence from latter windows.

Another use of the continuous Morlet wavelet is demonstrated by Pan [49]; here this wavelet is compared with the discrete Daubechies, the windowed Fourier transform and the Wigner-Ville transform. In this comparison paper, one highlight is the inability of Fourier analysis to identify the location of features within signals. Another problem is the Wigner-Ville's cross terms which mask interesting time-frequency information and also introduce false information. However, the signals examined are specifically frequency dominant in nature; in this situation, time-frequency analysis has clear advantages over time waveforms, and his simulations show better performance with spectrograms, smoothed Wigner-Ville and the Morlet transform. The discrete Daubechies lacks fidelity due to its dyadic limitations, allowing a scale display of only 8 levels. Clearly, where the signal comprises sets of basic frequencies, either stationary or shifting, frequency methods such as the Fourier transform and spectrograms are more appropriate than wavelet analysis and scalograms.

Another author using scalograms is Guillemain [26]. He uses a continuous Gabor transform of transients to classify monocomponent acoustic signals such as chirps. A Gabor "wavelet" is not rigorously a wavelet at all, because it is not admissible; actually it is a real Morlet, and in this context is used to classify smooth transients bounded in time and frequency and fading asymptotically, so there is no idea of suddenness. Also, the transient is the only thing being examined, rather than it being an unexpected event on some underlying carrier. The classifier distinguishes the two cases of linear and hyperbolic chirps for which the scalogram is a single line either linear or hyperbolic. He calls these curves "ridges", and it is the shape of these ridges which facilitate signal classification.

The Gabor transform is also used by De Lassus [19], chosen in preference to the Fourier transform for reasons of speed. It is used as input for a combination of neural nets and look up tables of rules for signal classification. Training gives estimates of likelihood distributions which are then used in Bayesian fashion to identify four classes. Human classification rates ~60% success versus 78% for the NN.

One use of wavelets for localization of short-time events comes from Ravier [52]. Continuous Malvar wavelets [44] are used in a simulated environment for detecting quiet submarines with low SNR signatures. In this context, the signature is acoustic, coming from propeller sounds. Malvar wavelets are continuous oscillatory functions in a bathtub-like envelope, and can be used for detecting pure tones. Despite their name, they are related more to windowed Fourier analysis than wavelets. The classification problem reduces to a two class system: noise / transient, with consequent localization of short-time events. The method seeks to identify any ordered feature in a background of white noise by examining probability density functions. However, it does not work well in real noise which is coloured rather than white. Coloured noise has a non-uniform frequency distribution; some frequencies being more dominant than others. For example, red noise has a greater low frequency component; blue noise has a greater high frequency component.

In a genuine classification problem, Barrows [8] classifies pulses using specific continuous Morlet "atoms". No location is attempted, and the method assumes that the transient is already located and isolated awaiting classification. It finds a best match wavelet set from a redundant library. A probability method is used to estimate the likelihood of a pulse belonging to a particular class based on the collection of atoms used in its best basis representation. This classifier is trained on a large set of sample pulses, and is very successful with about 97% classification accuracy. However, it is very slow even on a parallel computer, due to its algorithmic nature.

Carmona [11] addresses the transient classification problem using Mallat's gradient operator wavelet [42]. This is not orthogonal, and is best known for its use as an edge detector in images.

The wavelet is chosen because sharp time localization of the transients make Fourier inadequate. Carmona's version uses local extrema information, similar in fashion to the method of Robertson. WT extrema profiles work as signature identifiers for 4 different types of transients. The method can distinguish between cyclic skews, exponentially damped sines, chirps and exponentially skewed non-linear sine waves (these are his terms), and is specially designed to work with low SNRs. Here, as with CFAR (Constant False Alarm Rate) applications, noise statistics are used to estimate a threshold parameter, α, which Carmona calls "the level of significance".

A common element of all these techniques is the absence of an analytic discrete method. They all depend on either continuous wavelets, or neural net post-processing of the wavelet transform. To conclude this review, a broad description of the continuous wavelet method for transient analysis is supplied by Anant [4], Friedlander [24], Loe [39], Önsay [48], Rosiene [56] and Saracco [57].

One straightforward discrete orthogonal approach is outlined by Carter [12] in a blue sky study based on adaptive thresholds for classifying simulated sonar signals. She compares Daubechies 8 and the Haar wavelet, each across a range of thresholds. The main purpose of this investigation is to determine a good thresholding regime. The simulated signals comprise occasional high frequency transients in a background of high frequency noise. In this study, soft thresholding works better. Thresholding is an important issue, being one of the few quantitative tools useful in the transform domain. The use of threshold adaptivity can greatly improve classification success; it is also a tuneable control capable of limiting false alarm rates. Chen [13] uses adaptive thresholding to control false classification in a CFAR technique by increasing threshold sensitivity as noise increases in time frequency space. Noise is uniformly distributed but the signal is local, so target frequency events can be identified by focusing on a local time frequency window.

Other authors discussing discrete wavelets applied to transient detection and location are Del Marco, Frisch and Hewer.

Del Marco [21] uses discrete wavelet packets to obviate the WT's translation dependence, somewhat undermining one of its major advantages. This is done to enable template matching in the wavelet domain. This translation invariant method eliminates the wavelet's locality property and is not appropriate for transient location. Its application lies in transient detection.

Frisch [25] uses an unspecified discrete wavelet as a matched filter to detect a known waveshape with an unknown arrival time and unknown amplitude in white Gaussian noise. Here again, the method is not analytic, and uses a maximum likelihood test. This technique suffers from the major drawback that it can only detect wavelets.

Hewer [29] uses several ingenious variations to aid detection of radar signals of broadly known forms. His method is based on Carmona's, and uses a non-decimated decomposition, ie a redundant transform. As in Chen's case, a CFAR solution is important. The D4 is chosen for its shortness since it is good at detecting short transients, and he observes generally that the WT is better able to locate signal discontinuities than the FT. Only the first five levels of the wavelet transform are used. Also, a repeated WT is used to reduce noise, thus reducing the need for a precise noise model.

The nature of electrical transients in their own right, ignoring methods of analysis, are documented by Shi and Thomas.

Shi [59] is concerned with the computational complexity of numerical methods. To overcome this, current signals are resolved into a series of known ramp basis functions, whereas most studies in the literature try to approximate currents with exponential functions. Here the objective is to model rather than measure the line switching operations and to calculate field transients

derived from electromagnetic field theory of a power transmission line. This is more to do with control than instrumentation, and illustrates the alternative approach of understanding through knowledge before the event.

Likewise, Thomas [71] predicts electromagnetic field and current transients in power transmission and distribution systems with a time domain model. The environment is "switching actions in high voltage substations" and protection standards for grounding faults. The transients are of 5 to 20 μs duration, with ω in the 100MHz range. Most calculations of radiated fields are traditionally performed in the frequency domain. However, Thomas cites the Fourier costs as reasons to work in the time domain. Here again, ramp-shaped basis functions are used to describe transients.

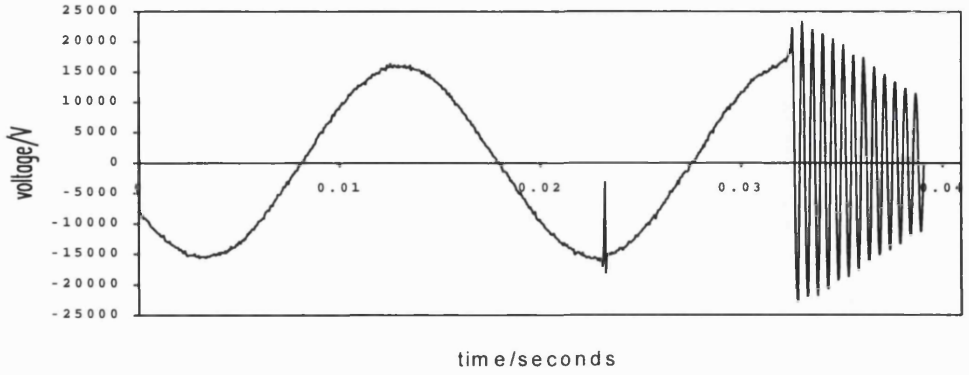
One thing to note in these papers is the urge for simplification, or cost saving. This advantage is offered by the discrete wavelet transform; not only is it faster than Fourier, it is also faster than the continuous wavelet transform.

Recalling the parameterized wavelets of the abc class, this chapter examines the use of wavelet selection combined with threshold adjustment to identify the best discrete orthogonal wavelet for finding two different types of transient.

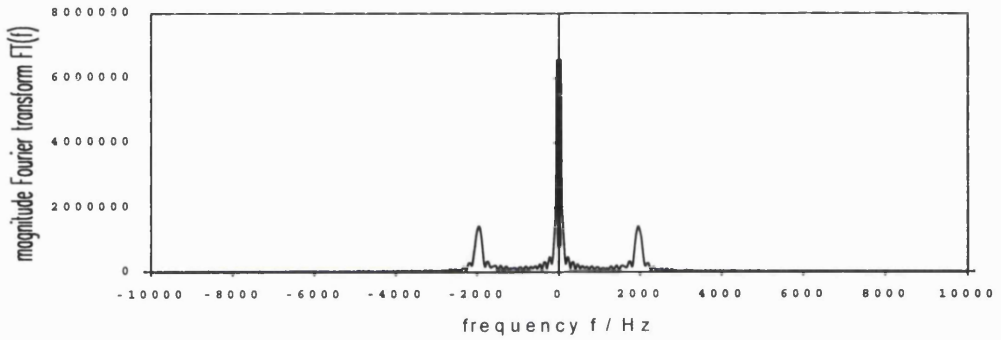
The method described here has various advantages in that it is discrete, fast and analytic. Also, while the solution developed here is presented as a demonstrator application, it does offer flexible extension to the neural net method for larger sets of signals.

To summarize and justify the use of wavelets in preference to the Fourier transform for transient detection, the following illustration shows a sample signal comprising low frequency carrier with two transients; one sharp and the other extended. Figure 8.1 shows the Fourier transform and figure 8.2 shows the wavelet transform of the same signal.

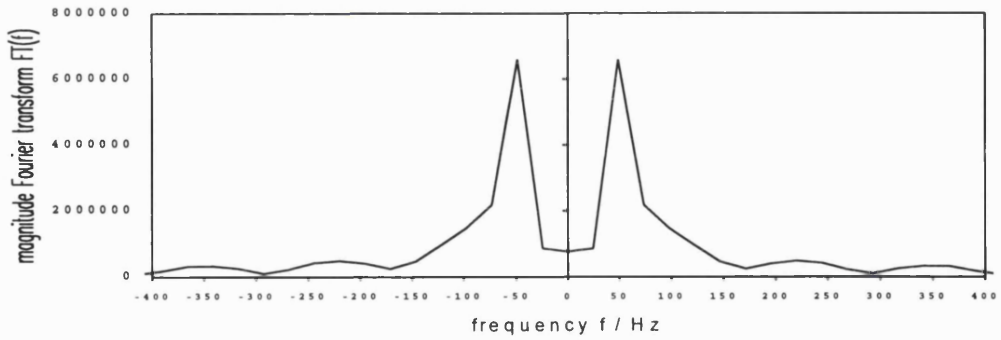
The Fourier transform is appropriate for identifying fundamental frequencies; here 50Hz and 1950Hz, but it contains no insight about locations of features within the signal. The wavelet transform, on the other hand, gives an immediate focus on both the sharp and extended transient. Lower scales (levels 0 to 5) identify the sharp transient, whereas higher scales (levels 4 to 7) identify the extended transient. Not only this, but the respective scale responses match the transient durations in the signal. Hence the WT can be used to identify the onset of events, and also their duration. In the application described later, this is exactly what is required. The use of the abc wavelets gives an extra degree of flexibility in tuning the wavelet response to more accurately align with the signal. In the following discussion the term "level" is used instead of the recognized wavelet term "scale" since it gives a stronger meaning of presence within the wavelet transform. For purposes of display, the individual scales of the wavelet transform are split into their respective levels and plotted separately as shown. Also, this form of representation corresponds to the wavelet map format, or scalogram, the greyscale equivalent of separate level plots for the various scale responses.



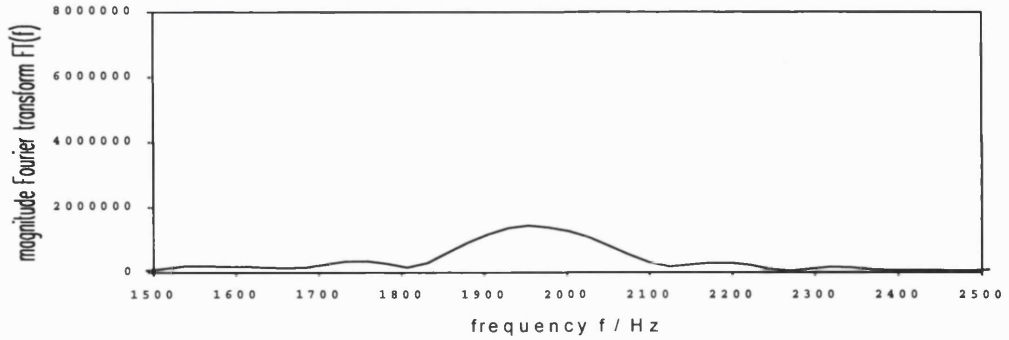
signal; relay switching transients



Fourier transform; absolute values



close-up on 50 Hz



close-up on 1950 Hz

Figure 8.1: Fourier transform of high voltage transients

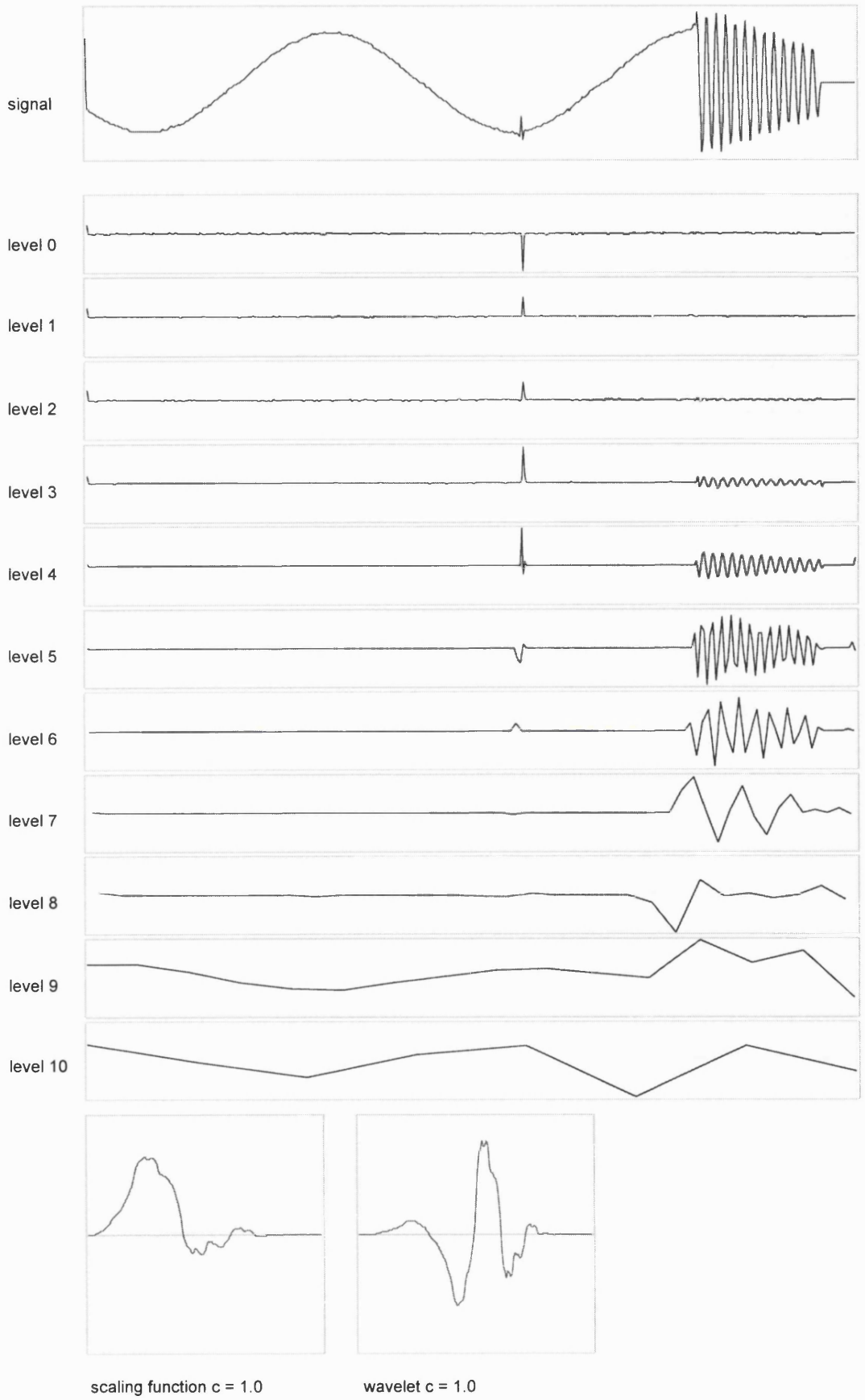


Figure 8.2: wavelet transform of high voltage transients

8.3 Data description

The National Grid Company is currently investigating relay transients with an experimental recorder operating at 40MHz. (Compare this with the standard sampling rate for field recorders of 2KHz).

There are 3 such signals used in this wavelet analysis, supplied by NGC in ASCII and binary files having the file extension .dat. Each is about 0.04 seconds long, and for identification in this study they are called wave1.dat, wave2.dat and wave3.dat. Later these extensions will be dropped to give the names wave1, wave2, and wave3.

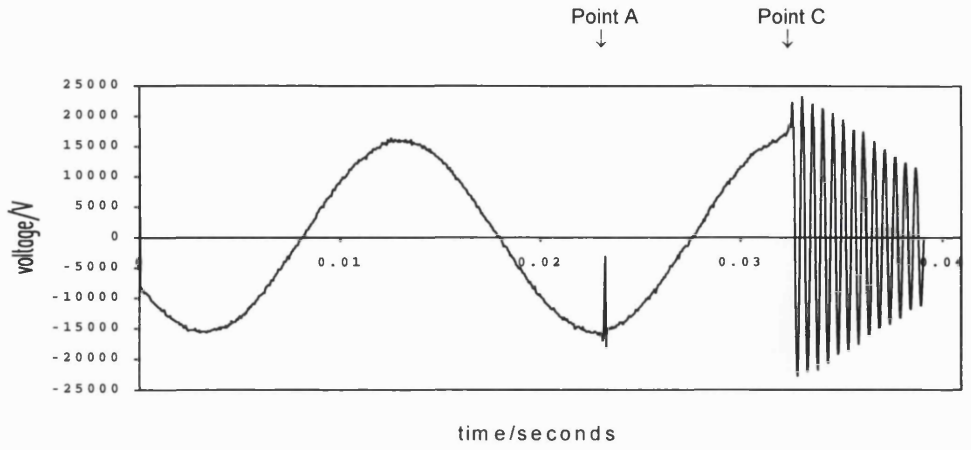
Each file is 4MB and contains voltage values in binary integer format.

Wave1.dat is an unprocessed file containing example transients with typical noise levels found in normal operation.

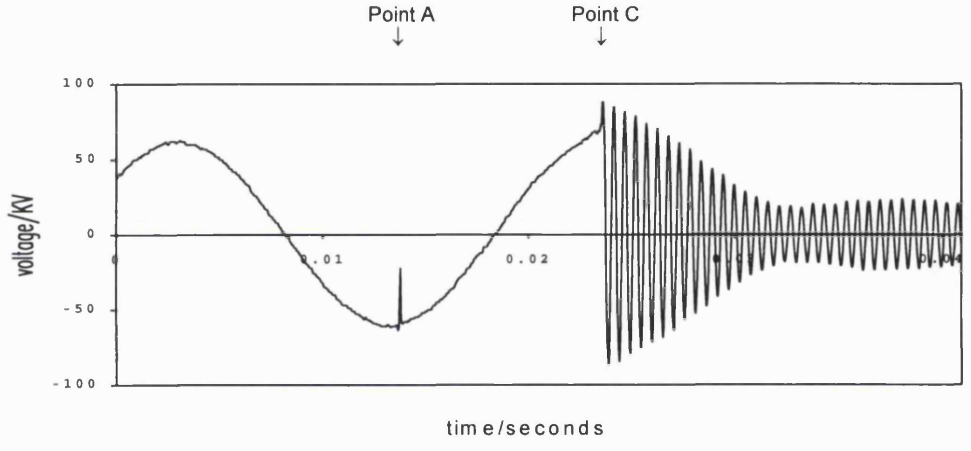
Wave2.dat and wave3.dat differ from wave1.dat in that they have been downsampled and upsampled with linear interpolation. The downsampling was performed by NGC to facilitate transmission by email in ASCII. Upsampling and ASCII-to-binary conversion was then performed to provide file formats equivalent to those actually used by NGC.

Additionally, wave3.dat derives from the downsampled wave2 ASCII file, shifted temporally to test the wavelet's ability to correctly identify shifted transients.

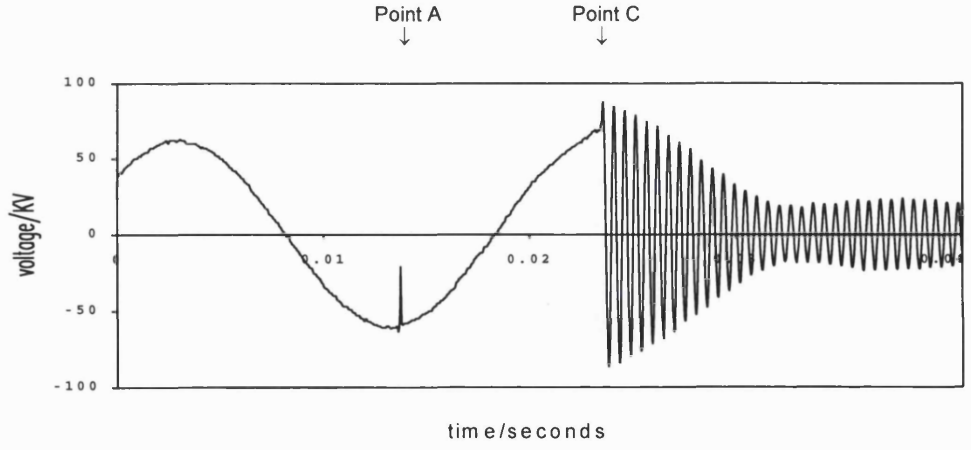
These three signals are illustrated in figure 8.3.



wave1



wave2



wave3

Figure 8.3: signals used in the transient analysis

In each case, reignition occurs before contact separation succeeds. The momentary sharp transient is called "point A". In practice, reignition can recur; this would be called "point B". Final clearance is denoted "point C", the start of the high frequency damping before final contact separation.

8.4 Practical considerations

Computing restraints prevented the full 4MB being used, due to segment limitation in C++ memory model. A segment is 64K (65536 bytes), equivalent to 16384 floating point numbers. For this reason, the .dat binary files were downsampled by the wavelet application at a rate of 1 in 100. Although clearly scaled down in quantity, the use of 16K points is still sufficient to demonstrate the suitability of wavelet analysis on these transients.

This size limitation means specifically that a 16K float array can be accessed as elements 0 to 16383. (The range 1 to 16384 would push the array size to 64K + 1, breaching the segment limit) This differs from the Numerical Recipes [51] code in which the signal and transform arrays always start at 1 (not zero). The significance of this becomes more apparent when aspects of level combinations are considered later. The dilated and shifted wavelet transform vector, as described in chapter 3, yields a single WT array equal in size to the signal, and having all the levels ordered with the lowest scales at the bottom of the vector and the highest scales at the top of the vector. To clarify the nature of the wavelet transform in computer memory, the signal and WT vector representations are shown in figure 8.4.

signal

0	16383

wavelet transform

0	level 10	15	16	level 9	31	32	level 8	63	64	level 7	127

128	level 6	255	256	level 5	511	512	level 4	1023	1024	level 3	2047

2048	level 2	4095	4096	level 1	8191	8192	level 0	16383

Figure 8.4: signal and wavelet transform memory representation

The numbers represent the array indices for the first and last points of the respective level. The size of any level n is twice that of level n + 1, since by the nature of wavelet decimation, each scale is a subsampled and averaged version of the scale below.

8.5 Transient location

8.5.1 Role of the abc wavelets

In this discussion the purpose of testing the abc class is to determine which wavelet is best for locating transients of the type described and illustrated above. Clearly wavelets are effective at locating features within signals due to their affine nature. The task now is to optimize the wavelet technique specifically through use of parametric adjustment. It remains now to determine which wavelet is the best. The performance metric in this context is the accuracy of transient location. In the analysis that follows, the 6 tap wavelet class is tested over the range c = 0.2 to 1000, as tabulated and illustrated in chapter 5. They are represented by the wavelet identifiers 0 to 23 as follows:

c	identifier	c	identifier	c	identifier	c	identifier
0.2	0	0.8	6	1.4	12	5	18
0.3	1	0.9	7	1.5	13	10	19
0.4	2	1.0	8	1.6	14	20	20
0.5	3	1.1	9	2.0	15	50	21
0.6	4	1.2	10	3.0	16	100	22
0.7	5	1.3	11	4.0	17	1000	23

8.5.2 Test framework

Attention now turns to the business of locating points A and C in the relay contact separation. For purposes of assessing the proposed method, the actual points A and C in the test signals need to be established. Define an interval as the time between two successive points in the signal. Then a sampling rate of 400KHz gives one interval equal to $\frac{1}{400,000}$ seconds, ie 2.5×10^{-6} seconds; 2.5 μs. Again, due to wavelet decimation of scales, the time interval for each scale is twice that of the preceding scale. The intervals for point separation are given below, together with the number of points for each scale, and the total length represented:

	interval	# points		time length
signal:	2.5μs	x 16384	=	0.04096 s
level 0:	5μs	x 8192	=	0.04096 s
level 1:	10μs	x 4096	=	0.04096 s
level 2:	20μs	x 2048	=	0.04096 s
level 3:	40μs	x 1024	=	0.04096 s
level 4:	80μs	x 512	=	0.04096 s
level 5:	160μs	x 256	=	0.04096 s

and so on.

The start of the signal is taken as the time origin, $t_0 = 0$, and t_A and t_C are the times of point A and point C with respect to t_0, ie just t_A and t_C themselves, since t_0 is zero. Actual times t_A and t_C are first measured by visual examination of the signal. Figure 8.5 shows successive close-ups of point A for wave1. Here point A is measured as 9310, ie 9310 in the sample range [0, 16383] as depicted in the signal representation in figure 8.4. The time of this event is therefore given by

$t_A = 9310 \times 2.5 \times 10^{-6} = 0.023275$ s

Similarly, figure 8.6 shows close-ups of point C. The onset of separation occurs at point 13005, and t_C is given by

$t_C = 13005 \times 2.5 \times 10^{-6} = 0.0325125$ s

In the same fashion, t_A and t_C for wave2 and wave3 are measured from close-ups of their respective signals as shown in figures 8.7 to 8.10.

For wave2

$t_A = 5509 \times 2.5 \times 10^{-6} = 0.0137725$ s
$t_C = 9424 \times 2.5 \times 10^{-6} = 0.0235600$ s

For wave3

$t_A = 5492 \times 2.5 \times 10^{-6} = 0.0137300$ s
$t_C = 9407 \times 2.5 \times 10^{-6} = 0.0235175$ s

Summary:

signal	t_A	t_C
wave1	0.0232750	0.0325125
wave2	0.0137725	0.0235600
wave3	0.0137300	0.0235175

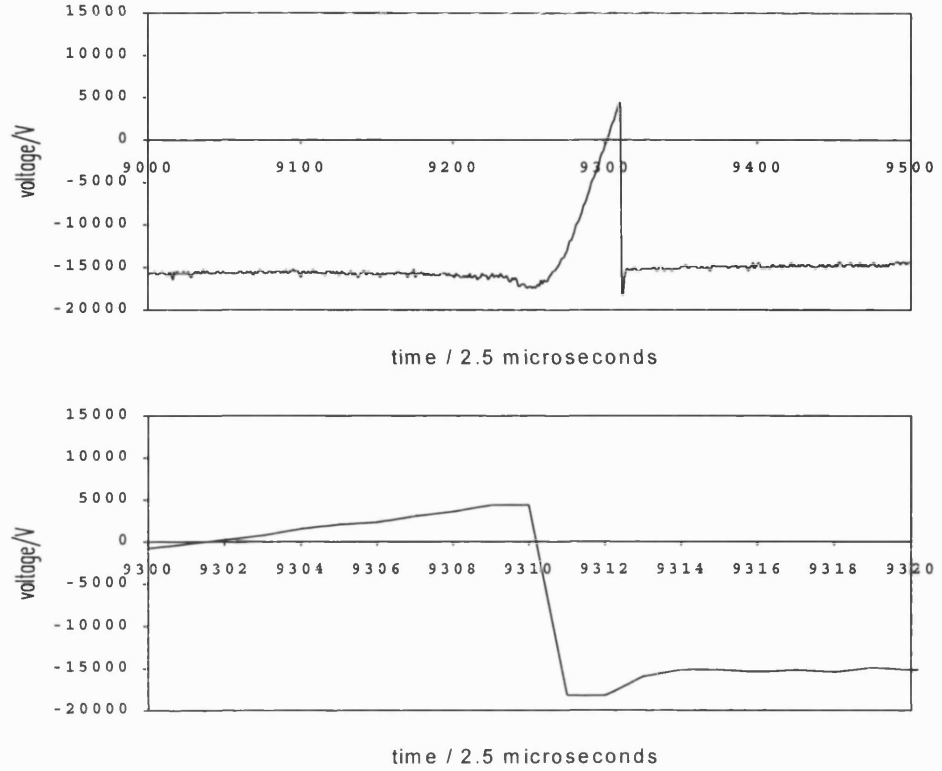


Figure 8.5: wave1 contact separation at point A. Successive close-ups

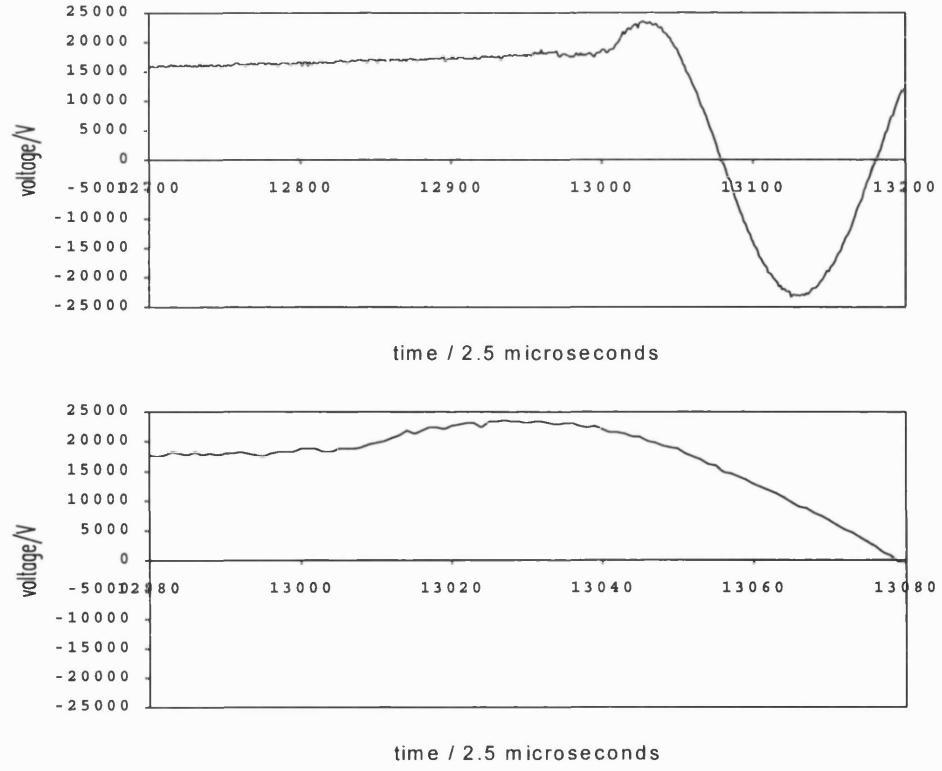


Figure 8.6: wave1 contact separation at point C. Successive close-ups

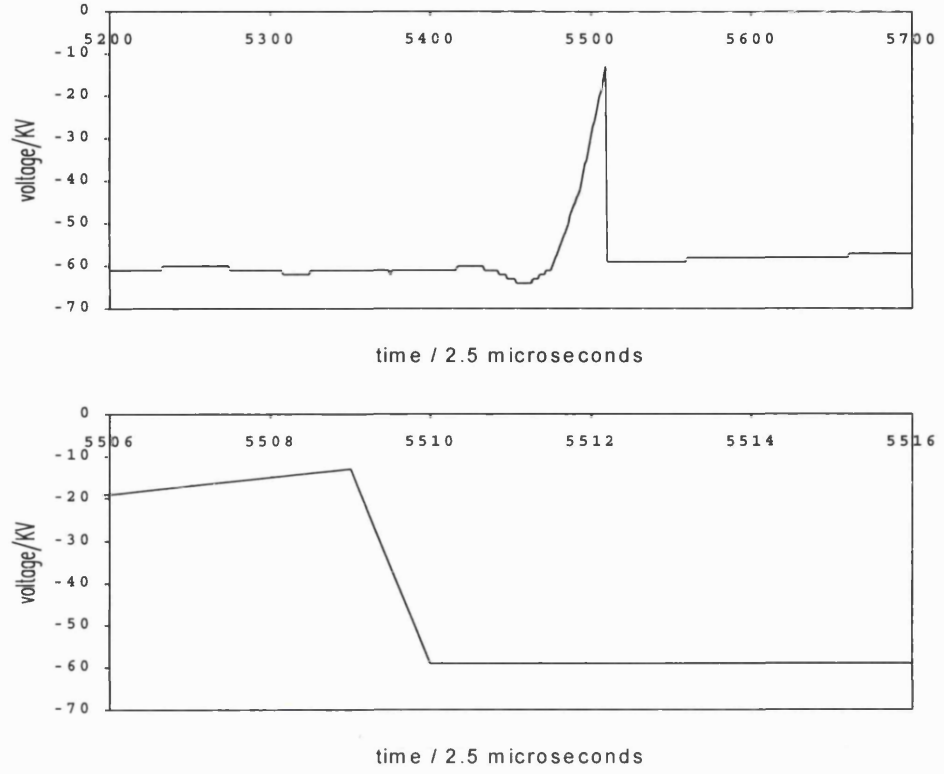


Figure 8.7: wave2 contact separation at point A. Successive close-ups

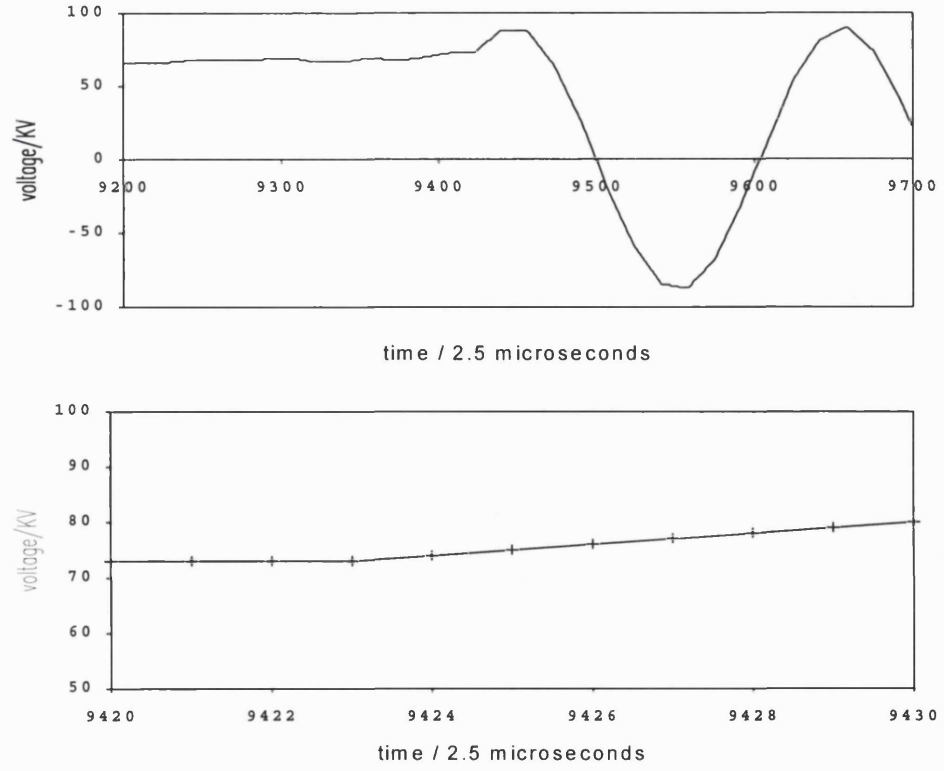


Figure 8.8: wave2 contact separation at point C. Successive close-ups

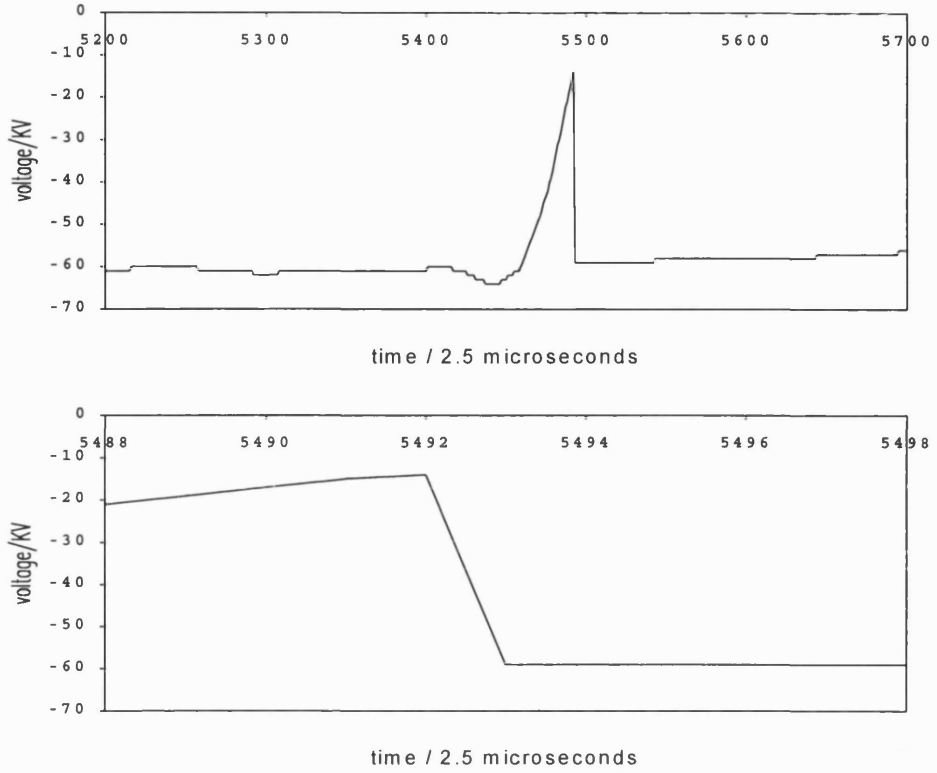


Figure 8.9: wave3 contact separation at point A. Successive close-ups

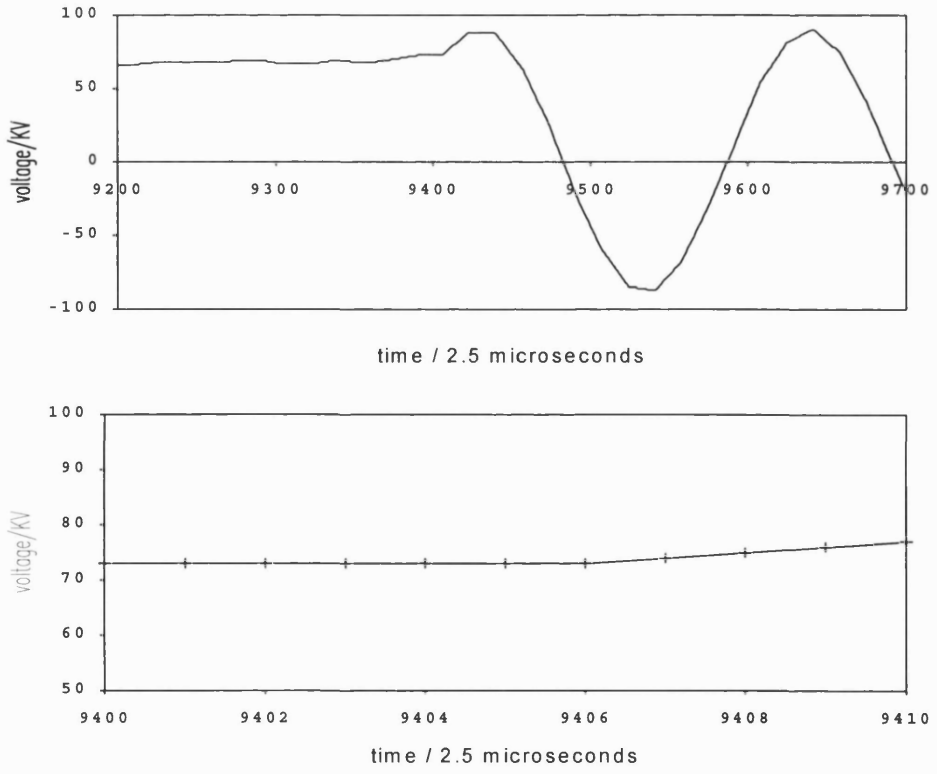


Figure 8.10: wave3 contact separation at point C. Successive close-ups

8.5.3 Expected results

Having measured the actual transient locations, t_A and t_C, they can now be used to predict how the wavelet transform should behave. The purpose of this exercise is to find a relationship between the signal transients and the corresponding local extrema of different levels of the wavelet transform. Having established this relationship, an inverse rule can then be determined to calculate signal transient locations from the responses of the various WT levels.

Consider first of all t_A in wave1 shown in figure 8.2. For this transient it is the low WT levels that locate t_A. The first 3 levels are sufficiently sharp to locate t_A; higher scales are not necessary. Also, much higher levels, 3 and beyond, are specifically excluded since at this scale the WT starts to show a response to point C.

Based on figure 8.4 the ideal wavelet response would yield extrema at the WT indices as follows:

	actual	rounded
level 0: $\frac{0.023275}{5 \times 10^{-6}} + 8192 = 12847$		12847
level 1: $\frac{0.023275}{10 \times 10^{-6}} + 4096 = 6423.5$		6423
level 2: $\frac{0.023275}{20 \times 10^{-6}} + 2048 = 3211.75$		3211

The actual wavelet extrema should occur almost at these rounded points. The respective times, based on the formulas above would give

$$\left.\begin{array}{l} \text{for level 0: } t_A = 5 \times 10^{-6} (j - 8192) \\ \text{for level 1: } t_A = 10 \times 10^{-6} (j - 4096) \\ \text{for level 2: } t_A = 20 \times 10^{-6} (j - 2048) \end{array}\right\} \quad (8.1)$$

where j is the index of the extremum in the respective WT level.

Table 8.1 shows the actual wavelet extrema locations, j, for levels 0, 1 and 2, together with the calculated times based on equations (8.1).

index	c	level 0		level 1		level 2	
		j	t_A	j	t_A	j	t_A
0	0.2	12846	0.02327	6422	0.02326	3211	0.02326
1	0.3	12846	0.02327	6422	0.02326	3210	0.02324
2	0.4	12846	0.02327	6422	0.02326	3211	0.02326
3	0.5	12846	0.02327	6422	0.02326	3211	0.02326
4	0.6	12846	0.02327	6422	0.02326	3211	0.02326
5	0.7	12846	0.02327	6422	0.02326	3211	0.02326
6	0.8	12846	0.02327	6422	0.02326	3211	0.02326
7	0.9	12846	0.02327	6423	0.02327	3211	0.02326
8	1.0	12846	0.02327	6423	0.02327	3211	0.02326
9	1.1	12846	0.02327	6423	0.02327	3210	0.02324
10	1.2	12846	0.02327	6423	0.02327	3210	0.02324
11	1.3	12846	0.02327	6423	0.02327	3210	0.02324
12	1.4	12846	0.02327	6423	0.02327	3210	0.02324
13	1.5	12846	0.02327	6423	0.02327	3210	0.02324
14	1.6	12846	0.02327	6423	0.02327	3210	0.02324
15	2.0	12846	0.02327	6423	0.02327	3210	0.02324
16	3.0	12846	0.02327	6423	0.02327	3210	0.02324
17	4.0	12846	0.02327	6423	0.02327	3210	0.02324
18	5.0	12846	0.02327	6423	0.02327	3210	0.02324
19	10	12846	0.02327	6423	0.02327	3210	0.02324
20	20	12846	0.02327	6422	0.02326	3210	0.02324
21	50	12846	0.02327	6422	0.02326	3210	0.02324
22	100	12846	0.02327	6422	0.02326	3210	0.02324
23	1000	12846	0.02327	6422	0.02326	3210	0.02324

Table 8.1: wavelet extrema locations for the range of wavelets 0 to 23

These results are broadly consistent across the wavelet class, with actual extrema indices 1 less than predicted. Hence the offset 8191 instead of 8192 in (8.1) should be used to calculate t_A. Where local extrema collocate in levels 0, 1 and 2, the time value t_A can be taken simply as

$$t_A = 5 \times 10^{-6} (j - 8191) \qquad (8.2)$$

where j is the level 0 index of the local extrema for any wavelet.

This result, together with similar consistent responses for wave2 and wave3, justify the use of (8.2) as an accurate transient locator for t_A.

8.6 Locating short transient, t_A

8.6.1 Voting system

Given the localization of high frequency events in the lower levels of the wavelet transform, a good detector of short transients could be based on the lowest three levels, 0, 1, 2 in a voting fashion, each one contributing to the decision. 3 is a good number because it's quite small and thus simple to compute; also the higher scales become more influenced by lower frequency events. The voting system used here works on the product of levels 0, 1, 2; each time point in the transform having a decision value. If all three scales have a large response at a particular time, as in figure 8.2 then the product vote for that time will be an extremum. If this extremum is globally large with respect to some threshold, then the particular time value of that vote is taken to be t_A. The decision as to the existence of a contact separation event, t_A, is determined by this

voting mechanism. The actual time of t_A is given by the location of the response in level 0 because it has the highest resolution and hence greatest precision (and hence potential accuracy) of all the WT levels. The voting system is chosen in preference to level 0 extrema alone to avoid false decisions caused by noise occurring in just one or two levels. Also note that a threshold must be used rather than just seeking the global extremum because multiple reignitions may occur as discussed in section 8.3 giving the so-called point B events.

Referring to figure 8.4, the WT representation has level 0 ranging over [8192, 16383]. The corresponding levels 1 and 2 range over [4096, 8191] and [2048, 4095] respectively. Figure 8.11 shows this arrangement aligned dyadically. The numbers in cells represent the array indices for the wavelet transform starting at t_0 on the left for each scale.

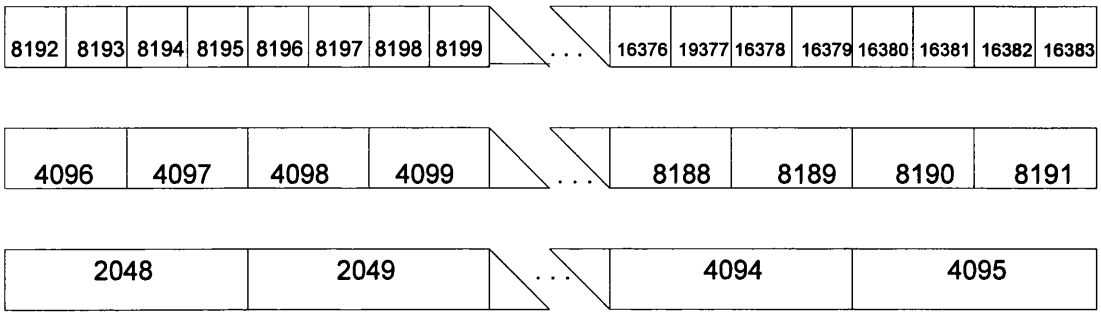


Figure 8.11: wavelet transform dyadic memory representation

Then for any element j of level 0, the corresponding WT coefficients in levels 1 and 2 are j/2 and j/4 with any remainder ignored. In C and C++ integer division by 2 is very efficiently implemented with the shift operator >>, meaning shift the left operand right by a number of places indicated by the right operand.

Thus for any index j in level 0, the three level voting product is simply

VOTE[j] = | WT[j] * WT[j>>1] * WT[j>>2] |

where j ranges over [8192, 16383] and WT is the name of the array holding the wavelet transform. The absolute value is used since the sign of the extremum holds no useful information. (|•| indicates modulus).

Note that this shifting technique would not work with the Numerical Recipes version of the wavelet transform, because of its use of arrays starting with element 1; thus the dyadic alignment of figure 8.11 would be invalid.

8.6.2 Decision criterion

The criterion for deciding whether or not a transient exists at a particular time is based on the vote for that particular time. Using the level 0 index j, VOTE[j] must exceed X times the mean vote where X is determined experimentally.

8.6.3 Results

Figure 8.12 shows the votes around point A for a sample of wavelets, c = 0.2, c = 1 and c = 1000. The index j refers to the level 0 position.

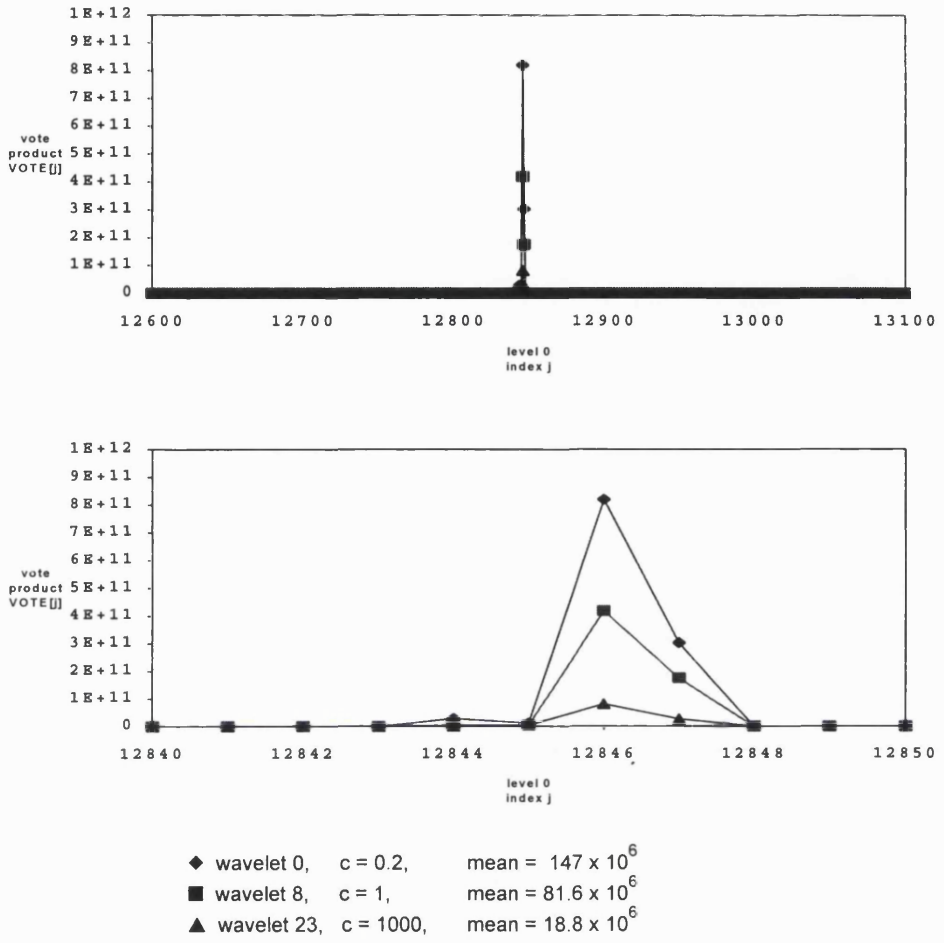


Figure 8.12: transient votes for wavelets c = 0.2, 1.0, 1000. Successive close-ups for wave1 on point A

Figure 8.12 shows two j values with votes significantly above the average. Call them point 1 for the first and point 2 for the second. Rough ratios of these points over the mean for their respective levels are:

	point 1	point 2
c = 0.2, wavelet 0:	5000	2000
c = 1.0, wavelet 8:	4900	1400
c = 1000, wavelet 23:	5300	2000

Using these ratios as estimates for a useful threshold, the accuracy of decision making can be gauged by varying the threshold around these values. If VOTE[j] exceeds this threshold, X times the mean, then t_A is taken from equation (8.2) as

$$t_A = 5 \times 10^{-6} (j - 8191)$$

Table 8.2 shows the t_A results for threshold = 2000 mean vote. **Bold** text indicates correct t_A values. Plain text indicates an incorrect hit. It must be noted that the actual value of t_A for wave2 is beyond the resolution of level 0, which can only measure to the nearest 5μs. In this case 0.013770 and 0.013775 are equally valid, both being as close to the actual t_A as can be determined by the wavelet transform. In this case take t_A to be 0.013770, the dominant result in the wave2 column.

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	**0.023275**	0.013775	**0.013730**
		0.023280		
1	0.3	**0.023275**	0.013775	**0.013730**
		0.023280		0.013735
2	0.4	**0.023275**	**0.013770**	**0.013730**
		0.023280	0.013775	0.013735
3	0.5	**0.023275**	**0.013770**	**0.013730**
		0.023280	0.013775	
4	0.6	**0.023275**	**0.013770**	**0.013730**
		0.023280	0.013775	
5	0.7	**0.023275**	**0.013770**	**0.013730**
		0.023280	0.013775	
6	0.8	**0.023275**	**0.013770**	**0.013730**
		0.023280	0.013775	
7	0.9	**0.023275**	**0.013770**	**0.013730**
		0.023280		
8	1.0	**0.023275**	**0.013770**	**0.013730**
		0.023280		
9	1.1	**0.023275**	**0.013770**	**0.013730**
		0.023280		
10	1.2	**0.023275**	**0.013770**	**0.013730**
		0.023280		
11	1.3	**0.023275**	**0.013770**	**0.013730**
12	1.4	**0.023275**	**0.013770**	**0.013730**
13	1.5	**0.023275**	**0.013770**	**0.013730**
14	1.6	**0.023275**	**0.013770**	**0.013730**
15	2.0	**0.023275**	**0.013770**	**0.013730**
16	3.0	**0.023275**	none	**0.013730**
17	4.0	**0.023275**	**0.013770**	**0.013730**
18	5.0	**0.023275**	**0.013770**	**0.013730**
19	10	**0.023275**	**0.013770**	**0.013730**
20	20	**0.023275**	**0.013770**	**0.013730**
21	50	**0.023275**	**0.013770**	**0.013730**
22	100	**0.023275**	**0.013770**	**0.013730**
23	1000	**0.023275**	**0.013770**	**0.013730**

Table 8.2: t_A results for threshold = 2000 mean vote for the range of wavelets 0 to 23

Significantly, the actual t_A is located for every wavelet for all three signals (except wavelet 16 on wave2). However, for low c values there are also false alarms; the double entries attributable to the pair of responses in the vote plot, figure 8.12. Increasing the threshold eliminates this problem. For X equal to 3000 most false alarms are eliminated. The cost of increasing the threshold, however, is the increasing failure of the method to find any t_A value at all. In no case is t_A ever incorrect; if it is found, then it has the correct numerical value. The problem reduces to one of succeeding or failing to find point A.

The adjustment of the threshold can then be used to reduce false alarms. Full tabular results over a range of thresholds are given in appendix A. This process shows that as the threshold increases, more and more wavelets of high c values fail to find any t_A values. In the limiting case, it is the shortest wavelet, c = 0.2 which alone can find t_A for all three signals under the highest threshold. This occurs for X = 5550, ie threshold = 5550 times the mean vote.

8.6.4 Conclusion

For the very short transient, point A, the shortest wavelet c = 0.2 is the most robust wavelet for always correctly finding the transient location, t_A, for increasing threshold values and for all three signals.

For this type of transient, in the limiting threshold case, there is no advantage tuning the wavelet in search of optimality.

8.7 Locating extended transient, t_C

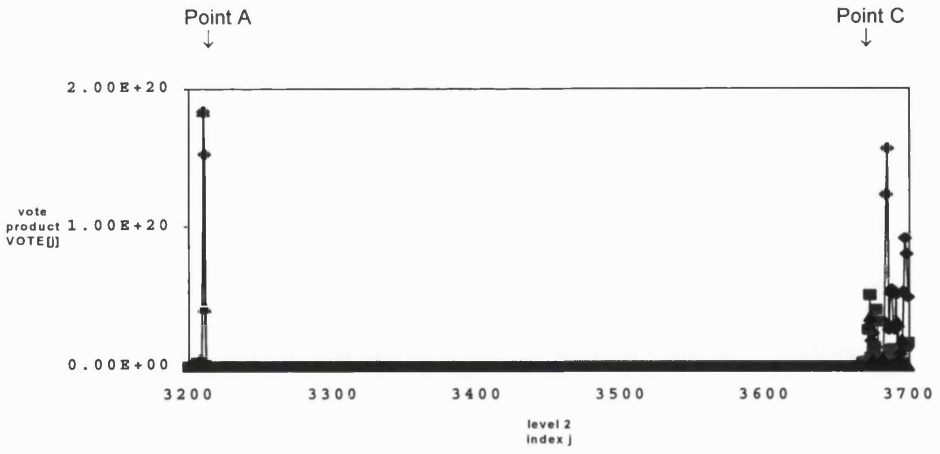
8.7.1 Voting system

The voting system for point C derives in similar fashion to that for point A, the difference being that the identification of point C depends on the response of the higher levels of the wavelet transform. Taking guidance from figure 8.2, levels 2, 3, 4, 5 and 6 are used to give the voting product. Level 2, while not giving a large response, is included for its ability to give better resolution than the higher levels. Using level 2 as the index range for voting, and referring to figure 8.4, gives j in the range [2048, 4095]. Then

VOTE[j] = | WT[j] * WT[j>>1] * WT[j>>2] * WT[j>>3] * WT[j>>4] |

As with the voting system for t_A, the absolute value is used since the sign of the extremum holds no useful information.

Figure 8.13 shows the vote product over the interval for point A and point C. Both points are detected.



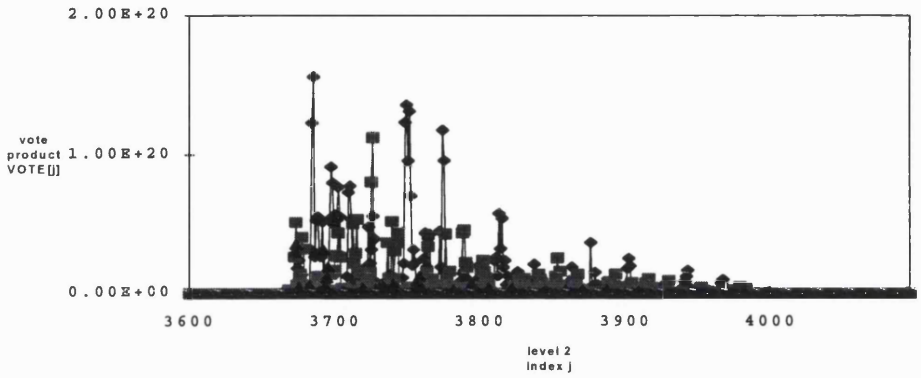
◆ wavelet 0, c = 0.2, mean = 2.05×10^{18}
■ wavelet 8, c = 1, mean = 9.36×10^{17}
▲ wavelet 23, c = 1000, mean = 5.08×10^{17}

Figure 8.13: transient votes for wavelets c = 0.2, 1.0, 1000, for wave1 on points A and C, using levels 2,3,4,5,6

This is not the desired result: only point C should be detected. This problem is solved by reducing the search space to the right hand side of (and not including) point A, which has already been found accurately using the first voting method based on levels 0, 1, 2. (It would not

be sensible to abandon the first method for t_A location in favour of the t_C method, which finds both t_A and t_C, because this would inevitably reduce resolution accuracy, by eliminating level 0 from the t_A calculation).

In this new search interval the range of j is reduced and the large values at t_A excluded, so the mean vote values are slightly different. The vote product in the reduced search interval is shown in figure 8.14.



◆ wavelet 0, c = 0.2, mean = 4.32×10^{18}
■ wavelet 8, c = 1, mean = 2.16×10^{18}
▲ wavelet 23, c = 1000, mean = 3.84×10^{17}

Figure 8.14: transient votes for wavelets c = 0.2, 1.0, 1000, for wave1 on point C, using levels 2,3,4,5,6

8.7.2 Decision criterion

There are many local maxima in figure 8.14, and clearly the decision criterion cannot be based simply on levels above a given threshold because there is only one point C event, (final clearance), so the voting method must be used in combination with another rule to locate t_C correctly. The extra condition necessary to find a single point C specifies that the local maximum must be the *first* local maximum above a given threshold. The combination of accepting only a j index for the *first* local maximum *and* which is also above a given threshold ensures that not the *very* first local maximum is used. This dual condition precludes ubiquitous small local extrema attributable to noise.

Having found the relevant j index in level 2, t_C is calculated as was t_A in (8.1),

$$t_C = 20 \times 10^{-6} (j - 2048)$$

8.7.3 Results

Referring to figure 8.6 for wave1, the actual point C occurs at point 13005 in the signal. Transforming this signal index into a WT index using figure 8.4 gives a level 0 index of

$$\frac{13005}{2} + 8192 = 14694 \text{ (rounded)}$$

Recalling the dyadic nature of the wavelet transform, the j index in level 2 is therefore

14694 >> 2 = 3673

A close-up of the votes around this point is shown in figure 8.15.

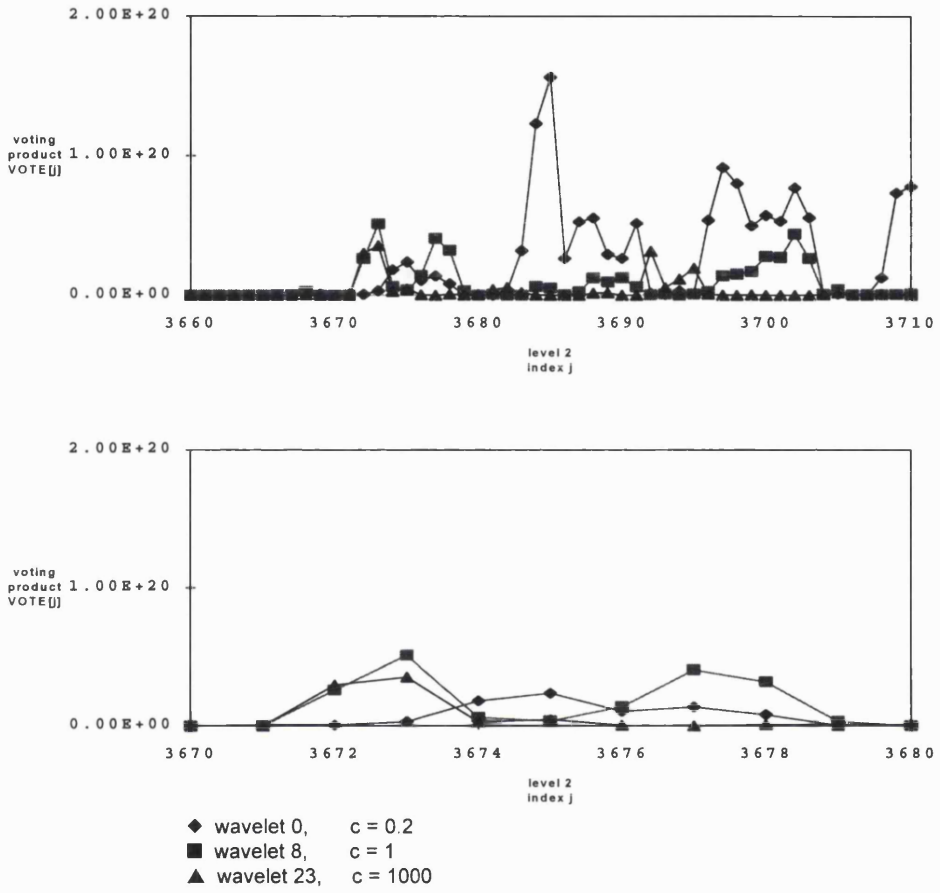


Figure 8.15: transient votes for wavelets c = 0.2, 1.0, 1000, for wave1. Successive close-ups on point C, using levels 2,3,4,5,6

From this sample of three wavelets it can be seen that there is a range of success for the accuracy of t_C. Some wavelets work better than others. Here (as opposed to the case for t_A), there is potential to tune the wavelet optimally to find t_C.

First, however, the issue of resolution must be considered. Recall from section 8.5.2 that level 2 has an interval size of 20 μs, ie a resolution of 2×10^{-5}. Therefore the actual t_C values need to be modified by rounding to the nearest 0.00002.

signal	t_C	20 μs rounding
wave1	0.0325125	0.03252
wave2	0.0235600	0.02356
wave3	0.0235175	0.02352

Table 8.3 shows the t_C results for threshold = 20 mean vote. **Bold** text indicates correct t_C values. Plain text indicates an incorrect hit.

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	0.03276	0.02380	0.02376
1	0.3	0.03276	0.02380	0.02376
2	0.4	0.03276	0.02384	0.02376
3	0.5	0.03300	0.02364	0.02380
4	0.6	0.03306	0.02364	0.02382
5	0.7	0.03306	0.02364	0.02382
6	0.8	**0.03252**	0.02364	0.02360
7	0.9	**0.03252**	0.02364	0.02360
8	1.0	**0.03252**	0.02364	0.02360
9	1.1	0.03260	0.02364	0.02360
10	1.2	0.03260	0.02364	0.02360
11	1.3	0.03260	0.02364	0.02360
12	1.4	0.03260	0.02364	0.02360
13	1.5	0.03282	0.02364	**0.02352**
14	1.6	**0.03252**	**0.02356**	**0.02352**
15	2.0	**0.03252**	**0.02356**	**0.02352**
16	3.0	**0.03252**	**0.02356**	**0.02352**
17	4.0	**0.03252**	**0.02356**	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	**0.02356**	**0.02352**
20	20	**0.03252**	**0.02356**	**0.02352**
21	50	**0.03252**	**0.02356**	**0.02352**
22	100	**0.03252**	**0.02356**	**0.02352**
23	1000	**0.03252**	**0.02356**	**0.02352**

Table 8.3: t_C results for threshold = 20 mean vote for the range of wavelets 0 to 23

In this case, only wavelets 14 to 23 (ie c = 1.6 to 1000) give correct answers for all three signals.

The adjustment of the threshold can then be used to reduce false alarms. Full tabular results over a range of thresholds are given in appendix B. This process shows that as the threshold increases, more and more wavelets fail to correctly find t_C for all three signals. In the limiting case only wavelet 18 (c = 5) succeeds. This occurs for X = 63, ie threshold = 63 times the mean vote.

8.7.4 Conclusion

For the extended transient starting at point C, an optimally robust wavelet exists and can be found experimentally by ramping a threshold to reduce the solution space of successful wavelets until a single one is revealed as the best.

The type of extended oscillatory transient studied, showed that wavelet 18 (c = 5) is the most robust for always correctly finding the transient location, t_C, for increasing threshold values and for all three signals.

8.8 Summary

This investigation has shown that the two types of transients occurring when high voltage, high current relay contacts open can be located using two different strategies and two different wavelets appropriate to each type of transient. In the case of very short transients, the shortest wavelet performs best. In the case of the extended transient, a longer wavelet performs better. For a given set of test signals, the optimal wavelet can be found for those particular signals.

This has been a demonstrator study, and while in this limited scenario only three signals were considered, the method serves to illustrate how suitable wavelets can be selected for different types of transients. With a larger set of test signals, an extension of the method could be derived to provide a neural net training system, to find optimal wavelets and thresholds automatically.

9 CONCLUSION

This chapter closes the presentation of generic discrete wavelets, the abc class. In addition to the final conclusion, there is concise discussion on future work. The merits and limitations of the work are assessed. The section on future work identifies potential areas of research that exist as natural extensions to this work, but which due to constraints of time were not pursued in the current investigation.

Additionally, as mentioned in the acknowledgements, this research was undertaken within the Postgraduate Training Partnership, a scheme established to increase industrial awareness and responsibility, and to produce a professional of greater commercial ability. In recognition of this, it should be noted that this project was completed on time and on budget.

9.1 Overview of the programme of work

The principal result of this work is the production of a new class of discrete wavelets called *abc wavelets*. The name comes from the addition of a single parameter c in the wavelet derivation which allows a degree of flexibility in designing 4 and 6 tap wavelets. These wavelets differ from the extant systems of parameterized wavelets in that they are all orthonormal and stable. Two separate derivations have been produced for the 4 and 6 tap classes. The solutions are analytic and thus wavelet taps can be found without iteration. The algebraic complexity of wavelet derivation prohibits analytic expressions for the taps of higher order wavelets, 8, 10 and so on. In this respect the work goes as far as it can in this direction. Each abc class includes the respective Daubechies wavelet of the same order, as a particular solution, and also the Daubechies wavelet of the immediately lower order. For example, the 6 tap wavelets include D6 and D4. The use of the parameter, c, enables the production of wavelets anywhere in-between these two Daubechies wavelets, thus giving a continuous set encompassing the two Daubechies.

The abc wavelets were tested in 3 areas of application to determine the merits of using the parameter c.

The applications addressed were:

- compression
- speed of transform
- transient detection and location

For compression, a selection of 1 dimensional test signals were produced, having both smooth and sharp features, which are characteristically best modelled by Daubechies 6 and Daubechies 4 respectively. The hypothesis was that such signals would best be compressed by a wavelet somewhere in-between D4 and D6. This in fact proved to be true in the majority of cases. Thus for the class of signals considered, the abc wavelet offers a middle ground between two extremes and the c parameter can be used to find the best wavelet for a particular signal. In this context *best* is a measure of efficiency, and it refers to the one particular wavelet which most accurately represents the signal at a particular rate of compression.

The speed of the wavelet transform was addressed with one specific wavelet from the generic 6 tap class. In this case one particular tap was set exactly equal to a half, thus giving an advantage in speed by shifting in place of floating point multiplication. The specific wavelet used is almost identical to Daubechies 6, thus giving comparable qualitative performance with a speed advantage. The fast wavelet is called B6, meaning **B**inary **6** tap.

The problems of transient detection and location were addressed with the objective of finding a best wavelet for detecting and locating different types of transients in high voltage relays. In this application real data were used. The purpose was to accurately locate in time the two different sorts of transients typically occurring at contact separation. Here again, the abc class proved useful in producing a range of wavelets from which the best one could be chosen. In this case the performance metric was the accuracy with which the different transient types were detected. The study showed that for each type of transient, a different wavelet from the abc class can be found which always accurately finds the transient. Thus in this case the wavelet can effectively be tuned with the c parameter, appropriately for different transient types.

9.2 Significant findings

The generic class was derived as a mathematical exercise and stands as a contribution to wavelet theory. The three examples cited above have been demonstrated to offer advantages in their respective areas of application. In each of these three cases, the benefits were incremental rather than revolutionary, offering solutions marginally better than are currently available. No order of magnitude improvements were achieved; but all great journeys comprise small steps, and this new wavelet class can at least answer yes to the question, "Does it work?" A further acid test question is whether the benefit of using these wavelets exceeds the cost of so doing. In the case of compression the answer is probably no. In the case of wavelet transform speed, the B6 offers a definite advantage, and as was commented in chapter 6, is a free benefit, ie the cost is zero. In the case of transient detection the answer is also affirmative; different types of transient *are* better detected with different types of wavelet.

So in assessing the worth of parameterized wavelets, in the cases studied here at least, the score has to be two out of three; and two out of three is good.

9.3 Future work

Two aspects of the abc class stand out as areas of potential benefit worthy of further research.

The first aspect concerns a search mechanism for optimal wavelets for compression. Given that orthogonality exists in the generic class without actually setting the taps, an interesting investigation would be to perform the wavelet transform analytically in terms of generic taps without setting specific numerical values. This would enable the wavelet transform itself to be optimized according to some performance metric (probably energy compaction). Tap solution in the wavelet domain would guarantee optimal compression of the given signal.

The second aspect worthy of further investigation is the exploitation of different lock functions. It may be possible to use the lock function shape to identify target features in signals. Originally this was the stance on which contact with NGC was made. Unfortunately in this case, no interest was raised in this possibility, and the matter was not pursued.

Of the two aspects mentioned here, the latter offers greater excitement, being more advantageous, novel and challenging than compression, which is "merely" a matter of calculus.

APPENDIX A WAVELET SELECTION FOR t_A

This appendix supplements chapter 8. Through a series of tabular results of increasing threshold, it is shown how the best wavelet is selected for locating the transient inception time t_A.

As the threshold increases, a single wavelet (c = 0.2) remains as the one able to locate t_A for all three signals, wave1, wave2 and wave3.

As previously noted in chapter 8, the issue relates more to whether or not t_A is found, rather than the accuracy with which it is found. Hence with increasing thresholds, the entries are either correct, or none found.

Also as was noted in chapter 8, the actual value of t_A for wave2 is beyond the resolution of level 0, which can only measure to the nearest 5μs. In this case 0.013770 and 0.013775 are equally valid, both being as close to the actual t_A as can be determined by the wavelet transform.

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	0.023275	0.013775	0.013730
1	0.3	0.023275	0.013775	0.013735
2	0.4	0.023275	0.013775	0.013730
3	0.5	0.023275	0.013770 0.013775	0.013730
4	0.6	0.023275	0.013770 0.013775	0.013730
5	0.7	0.023275	0.013770	0.013730
6	0.8	0.023275	0.013770	0.013730
7	0.9	0.023275	0.013770	0.013730
8	1.0	0.023275	0.013770	0.013730
9	1.1	0.023275	0.013770	0.013730
10	1.2	0.023275	0.013770	0.013730
11	1.3	0.023275	0.013770	0.013730
12	1.4	0.023275	0.013770	0.013730
13	1.5	0.023275	0.013770	0.013730
14	1.6	0.023275	0.013770	0.013730
15	2.0	none	0.013770	0.013730
16	3.0	0.023275	none	0.013730
17	4.0	0.023275	0.013770	0.013730
18	5.0	0.023275	0.013770	0.013730
19	10	0.023275	0.013770	0.013730
20	20	0.023275	0.013770	0.013730
21	50	0.023275	0.013770	0.013730
22	100	0.023275	0.013770	0.013730
23	1000	0.023275	0.013770	0.013730

Table A.1: t_A results for threshold = 3000 mean vote

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	0.023275	0.013775	0.013730
1	0.3	0.023275	0.013775	none
2	0.4	0.023275	0.013775	0.013730
3	0.5	0.023275	0.013775	0.013730
4	0.6	0.023275	0.013770	0.013730
5	0.7	0.023275	0.013770	0.013730
6	0.8	0.023275	0.013770	0.013730
7	0.9	0.023275	0.013770	0.013730
8	1.0	0.023275	0.013770	0.013730
9	1.1	0.023275	0.013770	0.013730
10	1.2	0.023275	0.013770	0.013730
11	1.3	0.023275	0.013770	0.013730
12	1.4	0.023275	0.013770	0.013730
13	1.5	0.023275	0.013770	0.013730
14	1.6	0.023275	0.013770	0.013730
15	2.0	none	0.013770	0.013730
16	3.0	none	none	0.013730
17	4.0	0.023275	0.013770	0.013730
18	5.0	0.023275	0.013770	0.013730
19	10	0.023275	0.013770	0.013730
20	20	0.023275	0.013770	0.013730
21	50	0.023275	0.013770	0.013730
22	100	0.023275	0.013770	0.013730
23	1000	0.023275	0.013770	0.013730

Table A.2: t_A results for threshold = 4000 mean vote

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	0.023275	0.013775	0.013730
1	0.3	0.023275	0.013775	none
2	0.4	0.023275	none	none
3	0.5	0.023275	none	0.013730
4	0.6	0.023275	none	0.013730
5	0.7	0.023275	none	0.013730
6	0.8	0.023275	0.013770	0.013730
7	0.9	0.023275	0.013770	0.013730
8	1.0	0.023275	0.013770	0.013730
9	1.1	0.023275	0.013770	0.013730
10	1.2	none	0.013770	0.013730
11	1.3	none	0.013770	0.013730
12	1.4	none	0.013770	0.013730
13	1.5	none	0.013770	0.013730
14	1.6	none	0.013770	0.013730
15	2.0	none	0.013770	0.013730
16	3.0	none	none	0.013730
17	4.0	none	none	0.013730
18	5.0	none	0.013770	0.013730
19	10	none	0.013770	0.013730
20	20	none	0.013770	0.013730
21	50	none	0.013770	0.013730
22	100	none	0.013770	0.013730
23	1000	none	0.013770	0.013730

Table A.3: t_A results for threshold = 5000 mean vote

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	0.023275	0.013775	0.013730
1	0.3	0.023275	0.013775	none
2	0.4	none	none	none
3	0.5	none	none	0.013730
4	0.6	none	none	0.013730
5	0.7	none	none	0.013730
6	0.8	none	none	0.013730
7	0.9	none	none	0.013730
8	1.0	none	0.013770	0.013730
9	1.1	none	0.013770	0.013730
10	1.2	none	0.013770	0.013730
11	1.3	none	0.013770	0.013730
12	1.4	none	0.013770	0.013730
13	1.5	none	0.013770	0.013730
14	1.6	none	0.013770	0.013730
15	2.0	none	0.013770	0.013730
16	3.0	none	none	0.013730
17	4.0	none	none	0.013730
18	5.0	none	none	0.013730
19	10	none	0.013770	0.013730
20	20	none	0.013770	0.013730
21	50	none	0.013770	0.013730
22	100	none	0.013770	none
23	1000	none	0.013770	none

Table A.4: t_A results for threshold = 5500 mean vote

wavelet	c	wave1 t_A (actual 0.023275)	wave2 t_A (actual 0.0137725)	wave3 t_A (actual 0.013730)
0	0.2	0.023275	0.013775	0.013730
1	0.3	none	0.013775	none
2	0.4	none	none	none
3	0.5	none	none	0.013730
4	0.6	none	none	0.013730
5	0.7	none	none	0.013730
6	0.8	none	none	0.013730
7	0.9	none	none	0.013730
8	1.0	none	0.013770	0.013730
9	1.1	none	0.013770	0.013730
10	1.2	none	0.013770	0.013730
11	1.3	none	0.013770	0.013730
12	1.4	none	0.013770	0.013730
13	1.5	none	0.013770	0.013730
14	1.6	none	0.013770	0.013730
15	2.0	none	0.013770	0.013730
16	3.0	none	none	0.013730
17	4.0	none	none	0.013730
18	5.0	none	none	0.013730
19	10	none	0.013770	0.013730
20	20	none	0.013770	0.013730
21	50	none	0.013770	0.013730
22	100	none	0.013770	none
23	1000	none	0.013770	none

Table A.5: t_A results for threshold = 5550 mean vote

Higher thresholds prevent any wavelet finding t_A for all three signals. Thus c = 0.2 is the limiting wavelet for detection of point A events.

APPENDIX B WAVELET SELECTION FOR t_C

This appendix supplements chapter 8. Through a series of tabular results of increasing threshold, it is shown how the best wavelet is selected for locating the transient inception time t_C.

As the threshold increases, a single wavelet (c = 5) remains as the one able to locate t_C for all three signals, wave1, wave2 and wave3.

In the results that follow, in keeping with the format introduced in chapter 8, **bold** text indicates correct t_C values and plain text indicates an incorrect hit.

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	0.03276	0.02380	0.02376
1	0.3	0.03276	0.02380	0.02376
2	0.4	0.03358	0.02406	0.02380
3	0.5	0.03310	0.02406	0.02380
4	0.6	0.03310	0.02364	0.02382
5	0.7	0.03358	0.02364	0.02382
6	0.8	0.03358	0.02364	0.02402
7	0.9	0.03358	0.02364	0.02360
8	1.0	0.03358	0.02364	0.02360
9	1.1	0.03358	0.02364	0.02360
10	1.2	0.03358	0.02364	0.02360
11	1.3	0.03358	0.02364	0.02360
12	1.4	0.03358	0.02364	0.02360
13	1.5	0.03358	0.02364	0.02360
14	1.6	0.03358	0.02364	0.02360
15	2.0	**0.03252**	**0.02356**	**0.02352**
16	3.0	**0.03252**	**0.02356**	**0.02352**
17	4.0	**0.03252**	**0.02356**	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	**0.02356**	**0.02352**
20	20	**0.03252**	**0.02356**	**0.02352**
21	50	**0.03252**	**0.02356**	**0.02352**
22	100	**0.03252**	**0.02356**	**0.02352**
23	1000	**0.03252**	**0.02356**	**0.02352**

Table B.1: t_C results for threshold = 30 mean vote

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	none	0.02380	0.02376
1	0.3	none	0.02384	0.02380
2	0.4	none	0.02406	0.02380
3	0.5	0.03358	0.02410	0.02380
4	0.6	0.03358	0.02364	0.02382
5	0.7	0.03358	0.02364	0.02402
6	0.8	0.03358	0.02364	0.02410
7	0.9	0.03358	0.02364	0.02410
8	1.0	0.03358	0.02364	0.02360
9	1.1	0.03358	0.02364	0.02360
10	1.2	0.03358	0.02364	0.02360
11	1.3	0.03358	0.02364	0.02360
12	1.4	none	0.02364	0.02360
13	1.5	none	0.02364	0.02360
14	1.6	none	0.02364	0.02360
15	2.0	**0.03252**	0.02364	0.02360
16	3.0	**0.03252**	**0.02356**	**0.02352**
17	4.0	**0.03252**	**0.02356**	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	**0.02356**	**0.02352**
20	20	**0.03252**	**0.02356**	**0.02352**
21	50	**0.03252**	**0.02356**	**0.02352**
22	100	**0.03252**	**0.02356**	**0.02352**
23	1000	**0.03252**	**0.02356**	**0.02352**

Table B.2: t_C results for threshold = 40 mean vote

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	none	0.02380	0.02380
1	0.3	none	0.02384	0.02380
2	0.4	none	0.02410	0.02380
3	0.5	none	0.02410	0.02380
4	0.6	0.03358	0.02410	0.02406
5	0.7	0.03358	0.02364	0.02410
6	0.8	0.03358	0.02364	0.02440
7	0.9	0.03358	0.02364	0.02440
8	1.0	0.03358	0.02364	0.02436
9	1.1	none	0.02364	0.02436
10	1.2	none	0.02364	0.02436
11	1.3	none	0.02364	0.02436
12	1.4	none	0.02364	0.02436
13	1.5	none	0.02364	0.02436
14	1.6	none	0.02364	0.02436
15	2.0	**0.03252**	0.02364	0.02436
16	3.0	**0.03252**	**0.02356**	**0.02352**
17	4.0	**0.03252**	**0.02356**	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	**0.02356**	**0.02352**
20	20	**0.03252**	none	**0.02352**
21	50	**0.03252**	none	**0.02352**
22	100	**0.03252**	none	**0.02352**
23	1000	**0.03252**	none	**0.02352**

Table B.3: t_C results for threshold = 50 mean vote

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	none	0.02380	0.02380
1	0.3	none	0.02380	0.02380
2	0.4	none	0.02410	0.02380
3	0.5	none	0.02410	0.02406
4	0.6	none	0.02460	0.02406
5	0.7	none	0.02364	none
6	0.8	none	0.02364	none
7	0.9	none	0.02364	0.02440
8	1.0	none	0.02364	0.02440
9	1.1	none	0.02364	0.02440
10	1.2	none	0.02364	0.02360
11	1.3	none	0.02364	0.02360
12	1.4	none	0.02364	0.02360
13	1.5	none	0.02364	0.02360
14	1.6	none	0.02364	0.02360
15	2.0	none	0.02364	0.02440
16	3.0	**0.03252**	0.02524	0.02520
17	4.0	**0.03252**	**0.02356**	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	none	**0.02352**
20	20	**0.03252**	none	**0.02352**
21	50	**0.03252**	none	**0.02352**
22	100	**0.03252**	none	**0.02352**
23	1000	**0.03252**	none	**0.02352**

Table B.4: t_C results for threshold = 60 mean vote

wavelet	c	wave1 t_C (actual 0.03252)	wave2 t_C (actual 0.02356)	wave3 t_C (actual 0.02352)
0	0.2	none	0.02384	0.02380
1	0.3	none	0.02410	0.02380
2	0.4	none	0.02410	0.02380
3	0.5	none	0.02414	0.02406
4	0.6	none	0.02460	none
5	0.7	none	0.02364	none
6	0.8	none	0.02364	none
7	0.9	none	0.02364	0.02440
8	1.0	none	0.02364	0.02440
9	1.1	none	0.02364	0.02440
10	1.2	none	0.02364	0.02440
11	1.3	none	0.02364	0.02360
12	1.4	none	0.02364	0.02360
13	1.5	none	0.02364	0.02360
14	1.6	none	0.02364	0.02360
15	2.0	none	0.02364	0.02440
16	3.0	**0.03252**	0.02524	0.02520
17	4.0	**0.03252**	0.02524	**0.02352**
18	5.0	**0.03252**	**0.02356**	**0.02352**
19	10	**0.03252**	none	**0.02352**
20	20	**0.03252**	none	**0.02352**
21	50	**0.03252**	none	**0.02352**
22	100	**0.03252**	none	**0.02352**
23	1000	**0.03252**	none	**0.02352**

Table B.5: t_C results for threshold = 63 mean vote

Higher thresholds prevent any wavelet finding t_C for all three signals. Thus c = 5 is the limiting wavelet for detection of point C events.

REFERENCES

[1] Abousleman G, 1995, Hyperspectral image coding using wavelet transforms and trellis coded quantization, *SPIE Proc Wavelet applications II,* **2491**, 1096-1106

[2] Abramowitz M & Stegun I, 1970, *Handbook of mathematical functions*, (Dover: New York)

[3] Amaratunga K, 1995, Time integration using wavelets, *SPIE Proc Wavelet applications II,* **2491**, 894-902

[4] Anant K, 1994, Detection of the electro-cardiogram P-wave using wavelet analysis, *SPIE Proc Wavelet applications*, **2242**, 744-749

[5] Antonini M, Barlaud M & Daubechies I, 1992, Image coding using wavelet transform, *IEEE Trans Image processing*, **1**, 205-220

[6] Assef Y, 1996, Classification of power distribution system fault currents using wavelets associated to artificial neural networks, *IEEE-SP Proc TFTS*, 421-424

[7] Atlas L, 1996, Automatic feature finding for time frequency distribution, *IEEE-SP Proc TFTS*, 333-336

[8] Barrows G, 1996, A mutual information measure for feature selection with application to pulse classification, *IEEE-SP Proc TFTS*, 249-252

[9] Benedetto J, 1995, Pyramidal Riesz products associated with subband coding and self-similarity, *SPIE Proc Wavelet applications II,* **2491**, 212-221

[10] Bracewell R, 1990, Numerical transforms, *Science*, **248**, 697-704

[11] Carmona R, 1993, Wavelet identification of transients in noisy time series, *SPIE Proc Mathematical imaging*, **2034**, 392-400

[12] Carter P, 1994, Unknown transient detection using wavelets, *SPIE Proc Wavelet applications*, **2242**, 803-814

[13] Chen V, 1996, CFAR detection and extraction of unknown signal in noise with time frequency Gabor transform, *SPIE Proc Wavelet applications III*, **2762**, 285-294

[14] Cody M, 1992, The fast wavelet transform, *Dr Dobb's Journal*, **April**, 16-28

[15] Cody M, 1993, A wavelet analyzer, *Dr Dobb's Journal*, **April**, 44-54

[16] Cohen A, Vial P & Daubechies I, 1993, Wavelets on the interval and fast wavelet transforms, *Applied and computational harmonic analysis*, **1**, 54-81

[17] da Silva E, 1995, Compression of super high definition images using successive approximation wavelet lattice quantisation, *IEE Proc Image processing and its applications*, **410**, 21-25

[18] Daubechies I, 1992, *Ten lectures on wavelets*, (SIAM: Philadelphia)

[19] De Lassus H, 1996, Neural network clusters and cellular automata for the detection and classification of overlapping transient signals on radio astronomy spectrograms from spacecraft, *IEEE-SP Proc TFTS*, 253-256

[20] de Queiroz R, 1994, Wavelet transforms in JPEG-like image coder, *SPIE Proc Visual communications and image processing*, **2308**, 161

[21] Del Marco S, 1995, Use of wavelet transform for improved CFAR detection in CW radar seekers, *SPIE Proc Wavelet applications II,* **2491**, 351-360

[22] Dress W & Kercel S, 1995, A hardware implementation of multiresolution filtering for broadband instrumentation, *SPIE Proc Wavelet applications II,* **2491**, 1014-1028

[23] Freeman M, 1995, Transionospheric chirp event classifier, *IEEE-SP Proc TFTS*, 144-150

[24] Friedlander B, 1992, Performance analysis of transient detectors based on a class of linear data transforms, *IEEE Trans Information theory*, **38**, 665-673

[25] Frisch M, 1991, Detection of a transient signal of unknown scaling and arrival time using the discrete wavelet transform, *IEEE Trans Information theory*, 1313-1316

[26] Guillemain P, 1996, Ridges associated to continuous linear time-frequency representations of asymptotic and transient signals, *IEEE-SP Proc TFTS*, 177-180

[27] Haar A, 1910, Zur Theorie der Orthogonalen Funktionensysteme, *Math Ann*, **69**, 331-371

[28] Heller P & Shapiro J, 1995, Optimally smooth symmetric quadrature mirror filters for image coding, *SPIE Proc Wavelet applications II,* **2491**, 119-130

[29] Hewer G, 1996, Adaptive wavelet detection of transients using the bootstrap, *SPIE Proc Wavelet applications III*, **2762**, 105-114

[30] Huffman J, 1994, Wavelets and image compression, *SMPTE Journal*, **November**, 723-727

[31] JASON committee, 1996, *Wavelet applications review report*, (JASON: La Jola)

[32] Jawerth B & Sweldons W, 1994, An overview of wavelet based multiresolution analyses, *SIAM Review*, **36**, 377-412

[33] Johns A, Aggarwal R & Bo Z, 1994, Non-unit protection technique for EHV transmission systems based on fault-generated noise, *IEE Proc Generation transmission and distribution*, **141**, 133-147

[34] Kaiser G, 1994, *A friendly guide to wavelets*, (Birkhäuser: Boston)

[35] La Borde B, 1995, New wavelet class for fine structure identification, *SPIE Proc Wavelet applications II,* **2491**, 1073-1085

[36] La Borde B, 1996, Generic explicit wavelet tap derivation, *SPIE Proc Wavelet applications III*, **2762**, 94-104

[37] La Borde B, 1996, New fast discrete wavelet, *IEEE-SP Proc TFTS*, 41-44

[38] Li J, 1995, A wavelet transform approach to video compression, *SPIE Proc Wavelet applications II,* **2491**, 1107-1118

[39] Loe R, 1994, Comparative analysis for underwater transient classification, *SPIE Proc Wavelet applications*, **2242**, 815-823

[40] Mallat S, 1989, A theory of multiresolution signal decomposition: the wavelet representation, *IEEE Trans Pattern recognition and machine intelligence*, **11**, 674-693

[41] Mallat S, 1989, Multiresolution approximations and wavelet orthonormal bases of $L^2(R)$, *Trans American mathematical society*, **315**, 69-87

[42] Mallat S & Zhong S, 1992, Characterization of signals from multiscale edges, *IEEE Trans Pattern recognition and machine intelligence*, **14**, 710-732

[43] Mathematica 2.2, 1993, Wolfram Research Inc

[44] Meyer Y, 1993, *Wavelets: algorithms and applications*, (SIAM: Philadelphia)

[45] Morris J, 1995, More results on orthogonal wavelets with optimum time-frequency resolution, *SPIE Proc Wavelet applications II,* **2491**, 52-62

[46] Murenzi, R & Antoine J-P, 1995, Target detection and recognition using two-dimensional isotropic and anisotropic wavelets, *SPIE Proc Wavelet applications in signal and image processing III*, **2569**

[47] Newland D, 1993, *Random vibrations, spectral & wavelet analysis*, (Longman Scientific & Technical: New York)

[48] Önsay T & Haddow A, 1994, Wavelet transform analysis of transient wave propagation in a dispersive medium, *Journal Acoustical society of America*, **95**, 1441-1449

[49] Pan M, 1996, Nonstationary time-frequency analysis for machine condition monitoring, *IEEE-SP Proc TFTS*, 477-480

[50] Pollen D, 1990, $SU_I(2, F[z, 1/z])$ for *F* A subfield of C, *Journal American mathematical society*, **3**, 611-624

[51] Press W, Teukolsky S, Vetterling W & Flannery B, 1992, *Numerical recipes in C*, (Cambridge University Press: Cambridge)

[52] Ravier P, 1996, Using Malvar wavelets for transient detection, *IEEE-SP Proc TFTS*, 229-232

[53] Resnikoff H, 1992, Wavelets and adaptive signal processing, *Optical Engineering*, **31**, 1229-1234

[54] Rioul O & Vetterli M, 1991, Wavelets and signal processing, *IEEE SP magazine*, **October**, 14-38

[55] Robertson D, 1994, Wavelets and power system transients, feature detection and classification, *SPIE Proc Wavelet applications*, **2242**, 474-488

[56] Rosiene J, 1996, Wavelet spatial filtering and transient signals, *SPIE Proc Wavelet applications III*, **2762**, 267-269

[57] Saracco G, 1994, Propagation of transient waves through a stratified fluid medium: Wavelet analysis of a nonasymptotic decomposition of the propagator, Part I. Spherical waves through a two-layered system, *Journal Acoustical society of America*, **95**, 1191-1205

[58] Scholl J & Rogovin D, 1994, Audio signal compression with circular wavelet packets, *SPIE Proc Wavelet applications in signal and image processing II*, **2303**, 518-529

[59] Shi R, 1995, Computation of transient electromagnetic fields radiated by a transmission line, *IEEE Trans Magnetics*, **31**, 2423-2431

[60] Smith M & Barnwell T,1986, Exact reconstruction techniques for tree structured subband coders, *IEEE Trans ASSP*, **34**, 434-441

[61] Stollnitz E, DeRose A & Salesin D, 1995, Wavelets for computer graphics: A Primer, Part 2, *IEEE Computer graphics & applications*, **July**, 75-85

[62] Strang G & Fix G, 1972, *Constructive aspects of functional analysis*, (Prentice-Hall: New Jersey)

[63] Strang G, 1989, Wavelets and dilation equations: a brief introduction, *SIAM Review*, **31**, 614-627

[64] Strang G, 1993, Wavelet transforms versus Fourier transforms, *Bulletin American mathematical society*, **28**, 288-305

[65] Sun Z, 1996, Analysis of a wavelet based compression scheme for wireless image communication, *SPIE Proc Wavelet applications III*, **2762**, 454-465

[66] Szu H, 1993, Why adaptive wavelet transform?, *SPIE Proc Information processing II*, ***1961***, 280-292

[67] Szu H & Telfer B, 1994, Mathematical theorems of adaptive wavelet transforms, *SPIE Proc Wavelet applications*, **2242**, 606-636

[68] Szu H, 1996, Review of wavelet transforms for pattern recognition, *SPIE Proc Wavelet applications III*, **2762**, 2-22

[69] Tagliarini G, 1996, Genetic algorithms for adaptive wavelet design, *SPIE Proc Wavelet applications III*, **2762**, 82-93

[70] Telfer B & Szu H, 1992, New wavelet transform normalization to remove frequency bias, *Optical Engineering*, **31**, 1830-1834

[71] Thomas D, 1989, Prediction of electromagnetic field and current transients in power transmission and distribution systems, *IEEE Trans Power delivery*, **4**, 744-755

[72] Tian J, 1996, Image data compression methodologies using discrete wavelets, *SPIE Proc Wavelet applications III*, **2762**,188-199

[73] Tuthill T, 1993, Wavelet packet-based nonlinear speckle reduction in optical synthetic aperture radar image analysis, *SPIE Proc Wavelet applications III*, **2762**, 295-300

[74] Vetterli M & Herley C, 1992, Wavelets and filter banks: Theory and design, *IEEE Trans SP*, **40**, 2207-2232

[75] Vetterli M & Kovacevic J, 1995, *Wavelets and subband coding*, (Prentice Hall: New Jersey)

[76] Waagen D, Argast J & M^{c}Donnell J, 1994, Stochastic determination of optimal wavelet compression strategies, *SPIE Proc Visual communications and image processing*, **2308**, 1711-1722

[77] Walden A, 1994, Multiresolution analysis and wavelets I: Deterministic results for continuous functions and discrete sequences, *Imperial College technical report*, **TR-94-08**, 1-51