Rafat Hussain

A Concise Introduction to Wavelets

July 17, 2011

Draft

Page Left Blank.

Contents

Acronyms

DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
IDFT	Inverse Discrete Fourier Transform
IDWT	Inverse Discrete Wavelet Transform
IFFT	Inverse Fast Fourier Transform

Chapter 1
Signal Processing

Abstract To be completed

1.1 Sampling,Interpolation and Aliasing

This chapter is meant to be a short introduction to Digital Signal processing. Most real life signals exist in analog (continuous time) domain but from a computational point of view it makes sense to discretize them first. These signals are first sampled and then digitized (sampling along the dependent axis) to convert them to digital signals. The conversion back to analog signal is called interpolation and utilizes a Digital to Analog converter. More on these conversions can be found in references.

1.1.1 Shannon's Sampling Theorem

In order to exactly recover a continuous time signal containing a maximum frequency of f_{max} , it should be sampled periodically at a rate of $f_s > 2f_{max}$. The lower bound on the sampling rate f_s is called Nyquist frequency $f_N = 2f_{max}$.

Let $x(t)$ be the continuous-time signal and $x(k)$ be the samples of this signal then according to Shannon Interpolation rule

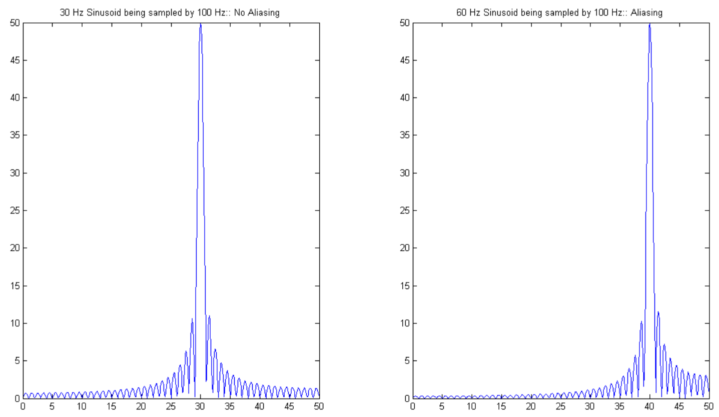
$$x(t) = \sum_{k=-\infty}^{k=\infty} x(k)h(t - kT_s)$$

where $T_s = \frac{1}{f_s}$ is the sampling rate and $h(t) = \frac{sin(\pi \frac{t}{T_s})}{\pi \frac{t}{T_s}}$ is a sinc-shaped envelope.

1.1.1.1 Aliasing

Aliasing occurs when a signal is sampled at a frequency less than the Nyquist frequency, $f_s < 2f_{max}$ which causes higher frequencies [$> f_s/2$] in the signal to appear as lower frequencies and distorts the reconstructed signal. An example of aliasing is shown in the figure. A sampling rate of $100Hz$ is set and two sinusoids with frequency components $30Hz$ and $60Hz$ respectively are sampled with this sampling frequency. Aliasing occurs in the second case and $60Hz$ sinusoid appears as $40Hz$. This phenomena is also known as folding as the signal appears to be folded back into the lower frequency.

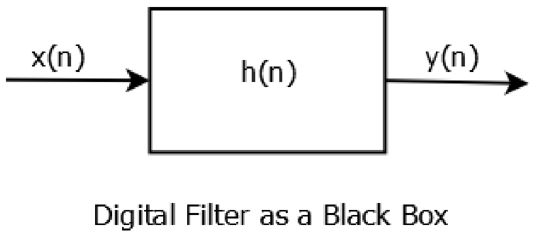
Fig. 1.1 Aliasing Demo: a) 30 Hz sinusoid sampled by 100 Hz results in no aliasing b) 60 Hz sinusoid sampled by 100 Hz is aliased and appears as 40 Hz



1.2 Digital FIR Filters

A digital filter is a system that processes a digital signal. In 1-D the filtering process usually consists of removing or enhancing certain bands of frequency.

Fig. 1.2 Digital Filters



In the figure above $h(n)$ is the filter transfer function while $x(n)$ and $y(n)$ are input and output signals, respectively.

1.2.0.2 Finite Impulse Response (FIR) Filters

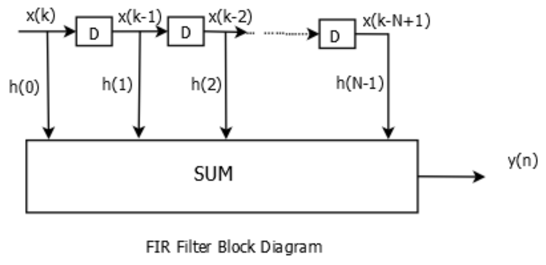
FIR filters have an impulse response that persists for only a finite number of samples and it is given by

$$h(n) = [h(0), h(1), ..., h(N-1)]$$

The output response of this filter is given by

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

Fig. 1.3 FIR Filters



1.2.0.3 Linear Phase FIR filters

Frequency response of any digital filter can be expressed in a magnitude-phase form.

$$H(\exp j\omega) = \|H(\omega)\| \angle \phi(\omega)$$

A Linear Phase filter is one whose phase ω is linear

$$\phi(\omega) = A\omega + B$$

A Linear Phase filter delays all frequency components equally (group delay) and it is highly desirable in phase-sensitive applications like image processing. Linear phase FIR filters can be easily designed by using symmetric impulse response. Following is a five tap FIR filter with coefficients

$$[-0.125, 0.250, 0.750, 0.250, -0.125]$$

and its magnitude-phase response.

Fig. 1.4 FIR Low Pass Filter Impulse Response

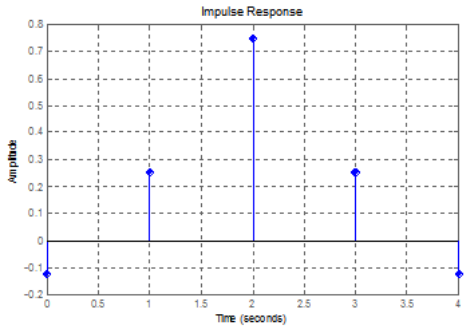
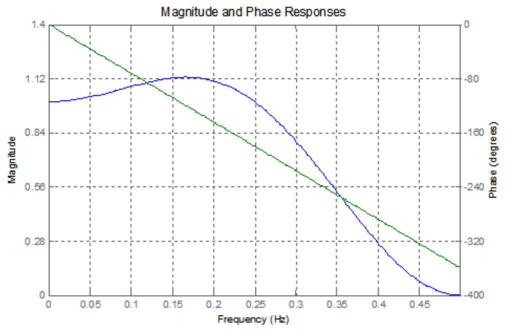


Fig. 1.5 FIR Low Pass Filter Magnitude-Phase Response

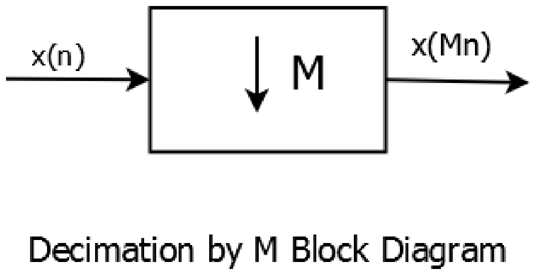


1.3 Multirate Systems and Filter Banks

Multirate systems are systems that contain multiple sample rates. There are a number of reasons why one may want such a system. Multirate may be needed to reduce computational needs of a system. Instead of ,say, N operations per cycle the system may need N/M operations if the sampling rate is reduced by M. It may also be needed to help reduce redundancy as in the case of multichannel communications and M-band filterbanks.

1.3.0.4 Decimation

Fig. 1.6 Decimation



Decimation by an integer M corresponds to retaining every Mth sample and discarding $M-1$ samples. In frequency domain, the output $Y(e^{j\omega})$ is given by

$$Y(e^{j\omega}) = \frac{1}{M}X(e^{\frac{j\omega+2k\pi}{M}})$$

for $k = 0, 1, ..., M-1$ Aliasing is an issue as downsampling "stretches" the bandwidth by a factor M. Applying the Shannon sampling theorem, the sampling frequency must be $f_s > 2f_{max}M$ in order to prevent aliasing.An example of aliasing is shown below. A signal consisting of two sinusoids at 50Hz and 130Hz is originally sampled at 1000Hz. It is then downsampled by 2 and 5. In the first case $f_s = 1000Hz$ is greater than $2f_{max}M = 520Hz$ and, therefore, no aliasing occurs. In the second case, $f_s = 1000Hz$ is less than $2f_{max}M = 1300Hz$ and we see higher frequency appearing at 350 Hz.

Fig. 1.7 Decimation Example 1

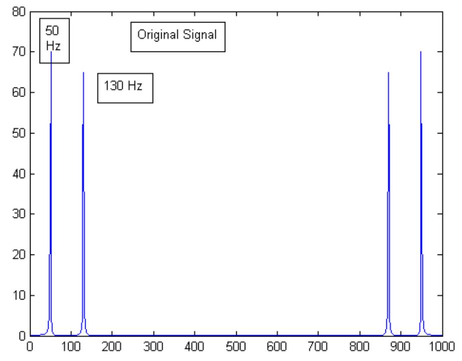


Fig. 1.8 Decimation Example 2

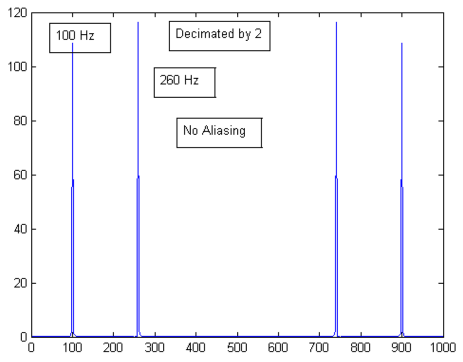
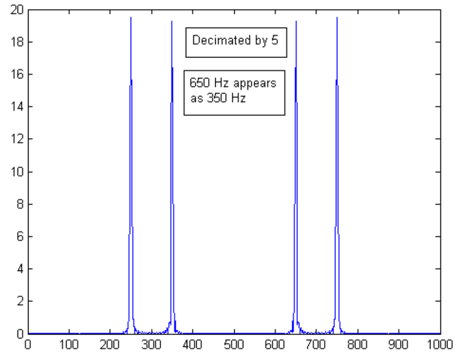


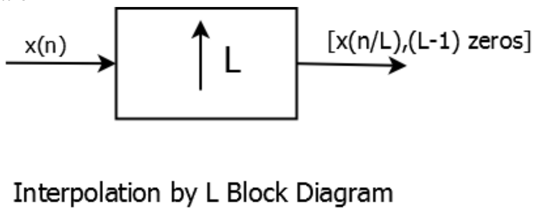
Fig. 1.9 Decimation Example 2



1.3.0.5 Interpolation

At its simplest, L-interpolation is the process of adding $L-1$ zeros between sample values of a signal x.

Fig. 1.10 Interpolation

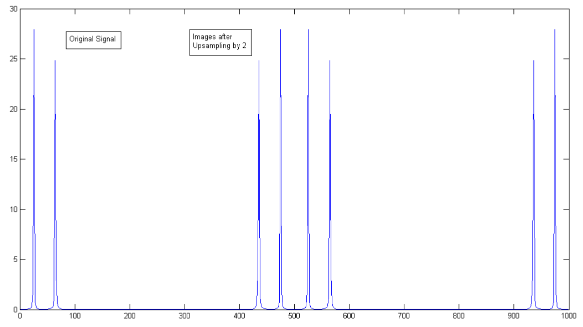


In frequency domain, the output $Y(e^{j\omega})$ is given by

$$Y(e^{j\omega}) = X(e^{jL\omega})$$

"Imaging" is a an upsampling phenomena that corresponds to aliasing in the downsampling case. Since an interpolator is compressing a signal, its images appear in the frequency band and depending on the interpolation factor we need a suitable low-pass filter (anti-imaging filter) to filter out these images. Shown below is an example of imaging after upsampling the signal from example above by a factor of 2.

Fig. 1.11 Interpolation Example



1.3.0.6 Polyphase Representation

Consider a time series $X(z) = \sum_{n=-\infty}^{\infty} x(n)z^{-n}$. We can use the decimation by 2 idea and separate it into its even and odd coefficients.

$$X(z) = \sum_{n=-\infty}^{\infty} x(2n)z^{-2n} + z^{-1} \sum_{n=-\infty}^{\infty} x(2n+1)z^{-(2n)}$$

Let $P_0(z) = \sum_{n=-\infty}^{\infty} x(2n)z^{-n}$ and $P_1(z) = \sum_{n=-\infty}^{\infty} x(2n+1)z^{-n}$ then $X(z)$ can be written as

$$X(z) = P_0(z^2) + z^{-1}P_1(z^2)$$

This idea can be extended to any value M instead of just 2.

$$X(z) = \sum_{n=-\infty}^{\infty} x(Mn)z^{-Mn} + z^{-1} \sum_{n=-\infty}^{\infty} x(Mn+1)z^{-(Mn)} + ... + z^{-(M-1)} \sum_{n=-\infty}^{\infty} x(Mn+M-1)z^{-(Mn)}$$

This representation is known as polyphase representation and is used to implement efficient filter banks :

$$X(z) = \sum_{i=0}^{M-1} P_i(z^M)z^{-i}$$

1.3.1 Filter Banks

Low pass and High pass filters process the signal by allowing only specific frequencies to go through- low pass and high pass frequencies, respectively. This obviously serves a very useful purpose but if these filters are used together then with proper design they can also reconstruct the original signal. A filter bank is a system in which two or more filters are used together to analyze and process a given signal.

Fig. 1.12 Two Channel Filter Bank

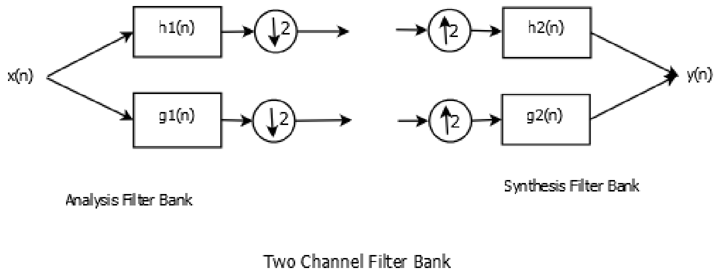


Figure above shows an example of a two channel filter bank. Analysis part of the filter bank decomposes a signal while the synthesis part reconstructs it. $h_1(n)$ and $h_2(n)$ are the low pass filters while $g_1(n)$ and $g_2(n)$ are complementary high pass filters.

1.3.1.1 Perfect Reconstruction property

Using interpolation and decimation frequency equations, the outputs at the two output nodes of analysis filters are $\frac{1}{2}(X(\frac{\omega}{2})H_1(\frac{\omega}{2})+X(\frac{\omega}{2}+\pi)H_1(\frac{\omega}{2}+\pi))$ and $\frac{1}{2}(X(\frac{\omega}{2})G_1(\frac{\omega}{2})+X(\frac{\omega}{2}+\pi)G_1(\frac{\omega}{2}+\pi))$

After feeding these two outputs to the two input nodes of synthesis filter, we get

$$Y(\omega)=\frac{1}{2}(H_1(z)H_2(z)+G_1(z)G_2(z))X(z)+\frac{1}{2}(H_1(-z)H_2(z)+G_1(-z)G_2(z))X(-z)$$

For perfect reconstruction, we need $Y(z)=X(z)$, this gives us two equations

$$\frac{1}{2}(H_1(z)H_2(z)+G_1(z)G_2(z))=z^{-L}$$

and

$$\frac{1}{2}(H_1(-z)H_2(z)+G_1(-z)G_2(z))=0$$

First equation is called the no distortion equation and the right hand term is a delay element. Second equation is the anti-aliasing equation and needs to be zero for Perfect Reconstruction Filter Bank to serve its purpose.

1.3.1.2 Orthogonal Filter Banks

Before discussing Orthogonal Filter Banks it is important to discuss orthogonality with respect to a filter $h(n)$. In wavelet theory, double shift orthogonal filters play a very important role. A filter $h(n)$ is double shift orthogonal if

$$< h(n), h(n-2k) >= \delta(k)$$

In the case of two channel filter banks, we can obtain the other three filters just by designing one filter and then using the orthogonality and Perfect Reconstruction properties. Because of double shift orthogonality, we can only design even orthogonal filters. It has been shown that for a given low pass filter $h_1(n)$, the high pass filter can be obtained by alternating flip method, ie., coefficients of low pass filters are flipped and then the sign of every alternate coefficient is flipped. The filters of synthesis filter banks can be obtained by using the no-distortion and no-aliasing equations of the Perfect Reconstruction property. It turns out that for a given $N+1$-tap low pass analysis filter $H_1(z)$, the other three filters are
High Pass Analysis Filter $H_2(z) = -z^{-N}H_1(z)$
Low Pass Synthesis Filter $G_1(z) = H_2(-z)$
High pass Synthesis Filter $G_2(z) = -H_1(z)$

1.3.1.3 Biorthogonal Filter Banks

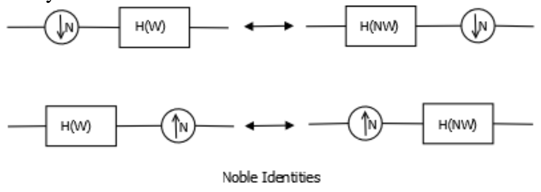
As has been mentioned previously, linear phase filter response is a desirable property in a number of applications and even with all their positives, Orthogonal filter banks are not linear phase filters. Only Haar filter bank is orthogonal and linear phase. Biorthogonal filter banks consists of filters where the filters are not orthogonal to themselves. They are constructed such that the analysis filters are orthogonal to the synthesis filters in addition to being linear phase. Using orthogonality and perfect reconstruction properties of biorthogonal filters, it can be shown that if given two FIR filters $H_1(z)$ and $G_1(z)$, the other two filters in the biorthogonal filter bank can be constructed as
$H_2(z) = G_1(-z)$ and $G_2(z) = -H_1(-z)$

1.4 Iterated Filter Banks

1.4.1 Noble Identities

Noble Identities are utilized in multirate systems to reverse the order of resampling and filtering.

Fig. 1.13 Noble Identity



The first row shows the noble identity of downsampling while the second one shows upsampling. Downsampling by N followed by a filtering operation with $H(omega$ is equivalent to filteing with an N-interpolated version $H(N\omega$ and then downsampling by N. Noble identities can be utilized to make computation more efficient or to simplify a multi stage multirate system.

1.4.2 Iterated Filter Banks

In the classical Discrete Wavelet transform algorithm, Iterated filter banks are constructed by iterating over low pass channels of a two-channel filter bank.Other constructions are also possible depending on specific needs but I will concentrate on Low Pass iterations for obvious reasons.

Fig. 1.14 Analysis Bank

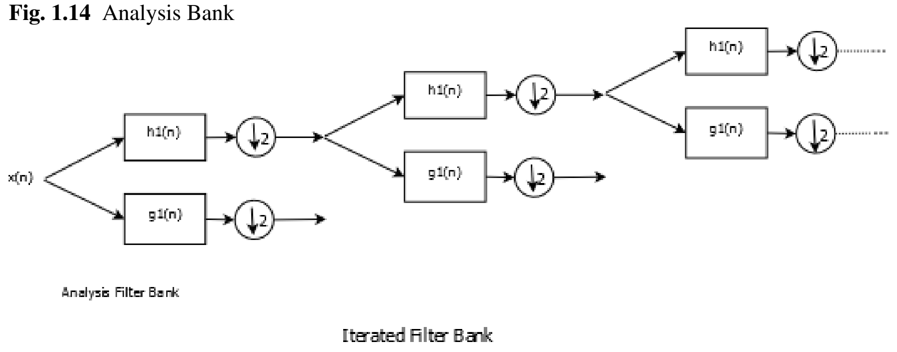
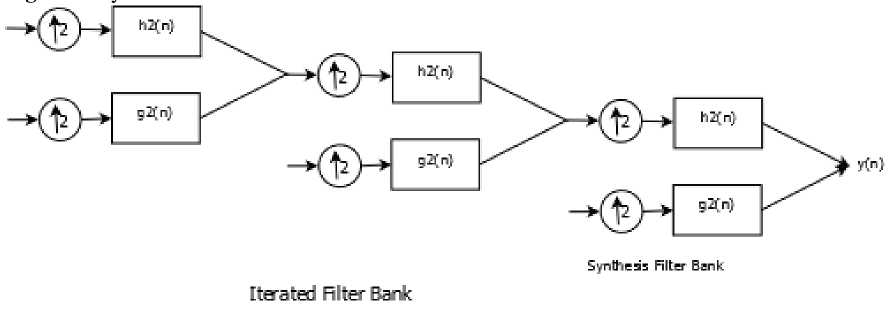


Fig. 1.15 Synthesis Bank



Using Noble Identities it can be shown that iterating over J stages of two-channel analysis filter bank is equivalent to filtering by $\prod_{n=0}^{J-1} H(2^n\omega)$ or in terms of $Z-$transform $\prod_{n=0}^{J-1} H(z^{2^n})$ and then downsampling by 2^J. The process works in reverse in synthesis case.

1.5 Fourier Analysis

Fourier Analysis is rooted in the knowledge that almost all signals can be represented as a sum of cosines and it makes sense to break them down into their component cosines in order to analyze for things like noise, discontinuities etc. in the frequency domain.

1.5.1 Continuous Time Analysis

In continuous domain Fourier Transform analysis equation is given by

$$X(e^{j\omega}) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt$$

and the synthesis equation is

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(e^{j\omega})e^{j\omega t}d\omega$$

This continuous fourier transform exists only and only if signal $x(t)$ is absolutely integrable.

$$\int_{-\infty}^{\infty} x(t)dt < \infty$$

Before we move on to discrete domain it is helpful to point out that Fourier Transform requires knowledge of signal at all times which is really impractical in discrete domain. Continuous Time Fourier Series [CTFS] attempts to address this issue and is given by

$$c_n = \frac{1}{T}\int_0^T x(t)e^{-jn\omega_0 t}dt$$

where c_n is the nth harmonic and ω_0 is the fundamental frequency.
The synthesis equation is given by

$$x(t) = \sum_{n=-\infty}^{\infty} c_n e^{jn\omega_0 t}$$

CTFS seems better from a digital point of view than it actually is. The integral is finite but for synthesis equation we still need to calculate for all values of n which isn't any simpler for computers.

1.5.2 Discrete Time Analysis

Corresponding to continuous time fourier transform, we have Discrete Time Fourier Transform(DTFT).

$$X(e^{j\omega_n}) = \sum_{k=-\infty}^{\infty} x(k)e^{-jk\omega_n}$$

where $\omega_n \in (-\pi,\pi)$ is the normalized frequency.It is normalized with respect to sampling frequency f_s. DTFT exists only if $x(k)$ is absolutely summable.

$$\sum_{k=-\infty}^{\infty} x(k) < \infty$$

Inverse Discrete Time Fourier Transform is similarly given by

$$x(k) = \frac{1}{2\pi}\int_{\omega_n=-\pi}^{\pi} X(e^{j\omega_n})e^{jk\omega_n}d\omega_n$$

One problem that jumps out that periodic signals, eg. sinusoids, are neither absolutely summable nor of finite energy which means that DTFT can not exist for these signals. Discrete Time Fourier Series is defined for periodic signal $x_p(k)$.

$$x_p(k) = \sum_{n=0}^{N-1} c_n e^{\frac{j2\pi nk}{N}}$$

where the complex exponential $e^{\frac{j2\pi nk}{N}}$ is periodic in both k and n with period N. c_n is known as the spectrum of the signal and , because of periodicity of n ,k and Nyquist sampling rate, it is given by

$$c_n = \frac{1}{N}\sum_{k=0} N-1 x_p(k)e^{-\frac{j2\pi nk}{N}}$$

It should be noted that both analysis and synthesis equations of DTFS are finite summations which makes it suitable for digital implementation with only problem being it is only defined for periodic signals which makes its implementation tricky and leads us to Discrete Fourier Transform

1.5.3 Discrete Fourier Transform

Discrete Fourier Transform is a mapping of N-sample time domain signal to N-sample frequency domain signal. Signals in either or both domain can be complex so it is a $C^N \leftrightarrow C^N$ mapping. The Discrete Fourier Transform of a signal $x(k)$ is given by

$$X(n) = \sum_{k=0}^{N-1} x(k)e^{\frac{-j2\pi kn}{N}}$$

for $n \in [0,1,...,N-1]$

The synthesis equation for DFT is known as Inverse Discrete Fourier Transform and is given by

$$x(k) = \frac{1}{N}\sum_{n=0}^{N-1} X(n)e^{\frac{j2\pi kn}{N}}$$

for $k \in [0,1,...,N-1]$

Unlike DTFS, DFT assumes periodicity and we don't have to prove periodicity of a given signal before computing its transform which makes things significantly easier. One of the most important property of DFT is that it is orthogonal. DFT's basis functions $e^{\frac{-j2\pi kn}{N}}$ form a 2D orthogonal matrix. This property is utilized in numerous signal processing and digital communications applications.

1.5.4 Fast Fourier Transform

One issue with DFT is that it is computationally intensive for large values of N.

$$X(n) = \sum_{k=0}^{N-1} x(k)e^{\frac{-j2\pi kn}{N}}$$

If you look at the DFT equation, we need N^2 complex multiplication and approximately N^2 addition operations in order to compute $X(n)$ and this may get unwieldy for large values of N. Fast Fourier Transform is an algorithm (there are many algorithms actually) that makes DFT computation faster. A simple introduction follows. Let us consider a N-point DFT where $N = 2^n$. We can partition $x(k)$ into even and odd terms $x_e(k)$ and $x_o(k)$ of length $N/2$ each. $x_e(k) = x(2k)$ are even terms and $x_o(k) = x(2k+1)$ are odd terms.

Taking DFT of $x(k)$

$$X(n) = \sum_{k=0}^{N-1} x(k) e^{\frac{-j2\pi kn}{N}}$$

$$X(n) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-j2\pi 2kn}{N}} + \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-j2\pi(2k+1)n}{N}}$$

$e^{\frac{-j2\pi 2kn}{N}}$ in equation above are basis functions of a N-point DFT but since each part of the signal is being computed for $N/2$-point DFT, the equations can be rewritten as

$$X(n) = \sum_{k=0}^{N/2-1} x(2k) e^{\frac{-j2\pi kn}{N/2}} + \sum_{k=0}^{N/2-1} x(2k+1) e^{\frac{-j\pi n}{N/2}} e^{\frac{-j2\pi kn}{N/2}}$$

It can be seen that above equation represents sum of two $N/2$ DFTs with second one having a “twiddle” factor of $e^{\frac{-j\pi n}{N/2}}$. The computation needed for this configuration. ,excluding twiddle factor multiplication, is, therefore $2(N/2)^2 = \frac{N^2}{2}$ which is roughly half of original N-point DFT. We can similarly keep halving the signal to the point where only $N \log_2(N)$ computations are needed. As can be seen, for large values of N FFT algorithm is far superior to original DFT and is an essential part of any engineering and mathematical toolbox.

References

1. D M Manolakis, J G Proakis (2006) Digital Signal Processing 4th Edition. Prentice-Hall
2. J Mellot, F J Taylor (1998) Hands-On Digital Signal Processing. McGraw-Hill Professional Publishing
3. P P Vaidyanathan (1993) Multirate systems and filter banks. Prentice-Hall
4. S Mallat (1999) A Wavelet Tour of Signal Processing. Academic Press
5. G Strang, T Nguyen(1996) Wavelets and Filter Banks. Wellesley Cambridge Press
6. M Vetterli, J Kovacevic(1995) Wavelets and Subband Coding. Prentice Hall Signal Processing Series
7. I Daubechies (1992) Ten Lectures on Wavelets. SIAM: Society for Industrial and Applied Mathematics
8. M Weeks (2006) Digital Signal Processing using Matlab and Wavelets. Infinity Science Press
9. V Goyal, M Vetterli, J Kovacevic (2010) Fourier and Wavelet Signal Processing. Manuscript Draft

Chapter 2
Mathematics

Abstract To be completed

2.1 Hilbert Space

2.1.1 Banach Space

Before we define Hilbert space, it is helpful to define normed linear spaces and Banach Spaces.

A vector space X is called normed linear space if for each element $x \in X$ there is a number $\|x\|$ called norm such that following conditions are satisfied

1. $\|x\| \geq 0$ with $\|x\| = 0$ if and only if $x = 0$
2. $\|cx\| = |c|\|x\|$ for a scalar c
3. $\|x+y\| \leq \|x\| + \|y\|$

Convergence and Completeness condition: Let X be a normed linear space

1. A sequence of vectors x_n converges to $x \in X$ if $\lim_{n\to\infty} \|x - x_n\| = 0$. Putting it in a different way, if $\forall \varepsilon > 0$, $\exists N > 0, \forall n \geq N$ $\|x - x_n\| > \varepsilon$
2. Cauchy Sequence: A vector sequence x_n is Cauchy if $\lim_{m,n\to\infty} \|x_m - x_n\| = 0$. Putting it in a different way, if $\forall \varepsilon > 0$, $\exists N > 0, \forall m,n \geq N$ $\|x_m - x_n\| > \varepsilon$
3. We say that X is complete if every Cauchy sequence in (X) is a convergent sequence. A complete normed linear space is called a BANACH SPACE.

2.1.2 Hilbert Space

A vector space H is called a Hilbert space if for each pair (x, y) of elements in the space H there is a unique number called inner product, denoted by $< x, y >$,subject to following three conditions

1. Linearity :

$$< \alpha x + \beta y, z >= \alpha < x, z > + \beta < y, z >$$

where $x, y, z \in H$ and $\alpha, \beta \in C$

2. Conjugated linearity: $< x, y >= \overline{< y, x >}$

3. $(< x, x >) > 0$ for $x \neq 0$

2.1.2.1 Properties of Hilbert Space

1. Norm: For an element $x \in H$, the norm is given by

$$\|x\| = \sqrt{< x, x >}$$

2. Cauchy-Schwartz Inequality holds for any elements $x, y \in H$

$$| < x, y > | \leq \|x\| \|y\|$$

3. Orthogonality: Vectors x and y are said to be orthogonal if $< x, y >= 0$. Orthogonality is written as $x \perp y$

Euclidean Pythagoran theorem holds in Hilbert Space

$$x \perp y \Rightarrow \|x + y\|^2 = \|x\|^2 + \|y\|^2$$

2.1.2.2 Operators on Hilbert Spaces

Adjoint Operator:

A linear operator $U^* : H_2 \to H_1$ is said to be adjoint of the operator $U : H_1 \to H_2$ when

$$< Ux, y >_{H_2} = < x, U^* y >_{H_1}$$

for every $x \in H_1$ and every $y \in H_2$

Self-Adjoint Operator: If $U = U^*$ then U is called self-adjoint operator.

Unitary Operator: $UU^* = U^*U = I$

2.2 Bases and Frames

2.2.1 Bases

Definition: Consider a set of vectors $\{e_k\}_{k=1}^{\infty}$ in H

1. This set is a basis in H if for each value of $f \in H$ there is a set of unique scalars c_k^f such that

$$f = \sum_{k=1}^{\infty} c_k^f e_k$$

2. The basis $\{e_k\}_{k=1}^{\infty}$ is an orthonormal basis if it is an orthonormal system

$$< e_k, e_j >= \delta_{k,j}$$

2.2.1.1 Orthonormal Basis properties

1. $f = \sum_{k=1}^{\infty} < f, e_k > e_k \text{ , } \forall f \in H$
2. $< f, g >= \sum_{k=1}^{\infty} < f, e_k >< e_k, g > \text{ , } \forall f, g \in H$
3. $\overline{span}\{e_k\}_{k=1}^{\infty} = H$
4. $\sum_{k=1}^{\infty} | < f, e_k > |^2 = \|f\|^2, \forall f \in H$
5. If $< f, e_k >= 0$ then $f = 0$

2.2.1.2 Biorthogonal Bases

Definition: Sets of vectors $\{\widetilde{e}_k\}_{k=1}^{\infty}$ and $\{e_k\}_{k=1}^{\infty}$ constitute biorthogonal bases in H if

1. $\forall k, j \in Z, < e_k, \widetilde{e}_j >= \delta_{k,j}$
2. For any $f \in H$ there exist $A, B > 0$ such that

$$A\|f\|^2 \leq \sum_k | < f, e_k > |^2 \leq B\|f\|^2$$

$$\frac{1}{B}\|f\|^2 \leq \sum_k | < f, \widetilde{e}_k > |^2 \leq \frac{1}{A}\|f\|^2$$

Bases that satisfy the above two equations are called Riesz Bases. $f \in H$ can be expanded as

$$f = \sum_{k=1}^{\infty} < f, e_k > \widetilde{e}_k = \sum_{k=1}^{\infty} < f, \widetilde{e}_k > e_k$$

2.2.1.3 Limitations of Bases or why we need Frames

Bases are characterized by their expansion property, ie. any function $f \in H$ can be expressed as a linear combination of basis vectors e_k that span H. However, it is possible to expand a function $f \in H$ as a linear combination of another set of vectors ϕ which may not be a basis in H. The representation in this case may be redundant but it may provide additional flexibility which may be a good given specific applications. Another problem with basis is that they may be difficult to construct and a small change in implementing one vector in the basis may destroy the basis so having extra vectors in a redundant expansion may not be such a bad idea. In other applications, as in image compression, we may not want to keep all the coefficients of signal expansion with a set of given vectors so it is quite possible that frames may be able to do the job instead of bases which need more tedious construction.

2.2.2 Frames

Definition: Consider a set of vectors $\{\phi_k\}_{k=1}^{\infty} \in H$ and that there exist constants $A, B > 0$. ϕ_k is a frame in H if

$$A\|f\|^2 \leq \sum_{k=1}^{\infty} | < f, \phi_k > |^2 \leq B\|f\|^2, \forall f \in H$$

If $A = B$ then the frame is called a Tight Frame.

If vectors $\{\phi_k\}_{k=1}^{\infty} \in H$ are linearly independent then the frame is non-redundant and is called a Riesz Basis.

Let $\{\phi_k\}_{k=1}^{\infty} \in H$ be a frame with frame operator S and bounds A, B then following is true

1. S is bounded, invertible, adjoint and positive.

2. $\{S^{-1}\phi_k\}_{k=1}^{\infty} \in H$ is a frame with frame operator S^{-1} and frame bounds A^{-1}, B^{-1}.

3. If A, B are optimal frame bounds for $\{\phi_k\}_{k=1}^{\infty}$ then A^{-1}, B^{-1} are optimal frame bounds for $\{S^{-1}\phi_k\}_{k=1}^{\infty}$.

4. $f = \sum_{k=1}^{\infty} < f, S^{-1}\phi_k > \phi_k, \forall f \in H$ and
$f = \sum_{k=1}^{\infty} < f, \phi_k > S^{-1}\phi_k, \forall f \in H$

Both converge for all $f \in H$.

Frame Operators and Pseudo-Inverse : Let U be the frame operator associated with the frame $\{\phi_k\}_{k=1}^{\infty} \in H$ and let $U : H \to H$ be bounded with closed range then $\{U\phi_k\}_{k=1}^{\infty}$ is a frame sequence with frame bounds $A\|U^{\dagger}\|^{-2}$ and $B\|U\|^2$. $U^{\dagger}$ is known as the pseudo inverse of the frame operator U and is defined as

$$U^{\dagger} = (U^*U)^{-1}U^*$$

where U^* is the adjoint of U. As can be seen from the definition, $U^{\dagger}$ is the left inverse.

Dual Frames: Assume that $\{\phi_k\}_{k=1}^{\infty} \in H$ is an overcomplete frame then there exist frames $\{\chi_k\}_{k=1}^{\infty} \in H$ for which $f = \sum_{k=1}^{\infty} < f, \chi_k > \phi_k, \forall f \in H$. Following holds for dual frames phi_k and chi_k

1. $f = \sum_{k=1}^{\infty} < f, \chi_k > \phi_k, \forall f \in H$

2. $f = \sum_{k=1}^{\infty} < f, \phi_k > \chi_k, \forall f \in H$

3. $< f, g >= \sum_{k=1}^{\infty} < f, \phi_k >< \chi_k, g >, \forall f, g \in H$

2.3 Approximation Theory

To motivate the concepts of approximation, consider a subspace $V_N \subset H$ and consider a function $f \in H$. An orthogonal projection of f on V_N is given by f_N. Using a set of N-vector biorthogonal basis we can recover the signal f_N in this subspace.

$$f_N(t) = \sum_{n=0}^{N-1} < f, \phi_n > \widetilde{\phi_n}$$

Now the signal constructed using N-vector biorthogonal basis is usually not exactly the same as f. Therefore, we compute the approximation error as $\|f - f_N\|$. To give a formal definition of approximation error, consider an orthonormal basis $\{e_k\}_{k=0}^{\infty}$ in L^2. The function f can, therefore be fully reconstructed using this basis.

$$f = \sum_{k=0}^{\infty} < f, e_k > e_k$$

However, if we are using N-vectors of this orthonormal basis to reconstruct f then the approximation difference is given by

$$f - f_N = \sum_{k=0}^{\infty} < f, e_k > e_k - \sum_{k=0}^{N-1} < f, e_k > e_k = \sum_{k=N}^{\infty} < f, e_k > e_k$$

and the approximation error is

$$e_N = \|f - f_N\|^2 = \sum_{k=N}^{\infty} | < f, e_k > |^2$$

Few Observations

1. As the value of N increases, the error decreases.

$$\lim_{n \to \infty} \|f - f_N\| = 0$$

2. Since $(e_N = \sum_{k=N}^{\infty} | < f, e_k > |^2$, the decay rate of approximation error is proportional to decay rate of $| < f, e_k > |$ as k increases.

3. Approximation error depends on the choice of N basis vectors.

2.3.0.1 Linear Fourier Approximations

$e^{i2\pi mt}$ is an orthonormal basis in $L^2[0,1]$. The signal f can be approximated using N-vector Fourier basis as

$$f_N(t) = \sum_{m=-\frac{N}{2}}^{m=\frac{N}{2}} < f(u), e^{i2\pi mu} > e^{i2\pi mt}$$

Fourier Approximation error is, therefore, given by

$$e_{f,N} = f(t) - f_N(t) = \sum_{m=-\infty}^{m=-\frac{N}{2}} < f(u), e^{i2\pi mu} > e^{i2\pi mt} + \sum_{m=\frac{N}{2}}^{m=\infty} < f(u), e^{i2\pi mu} > e^{i2\pi mt}$$

It should be noted that linear Fourier approximation has a low frequency bias as the approximate signal $f_N(t)$ is constructed using low frequency components. A signal rich in high frequency components will see substantial degradation if Linear Fourier Approximation is used.

2.3.1 Wavelet Approximation Error

Faster the decay of approximation error, fewer basis functions are needed to approximate f. In the case of wavelets, approximation error rate decay depends on the number of low pass filter zeros at $\omega = \pi$. More zeros equate to smoother wavelets, the faster the expansion coefficients decay to zero.

Consider a multiresolution space V_J where $V_J = V_0 \oplus W_0 \oplus W_1 \oplus \ldots\ldots \oplus W_{J-1}$. The space V_J is spanned by scaling functions $\phi(t-k)$ and wavelets $\{\psi(2^j t - k)\}_{j=0}^{J-1}$. The projection of f in this space is given by

$$f_N = \sum_n < f, \phi_{J,n} > \phi_{J,n}$$

or,

$$f_N = \sum_n < f, \phi_{0,n} > \phi_{0,n} + \sum_{j=0}^{J-1} < f, \psi_{j,n} > \psi_{j,n}$$

Now generalizing this equation over entire L^2 space and assuming the smallest scale is 0 , we get

$$f = \sum_n < f, \phi_{0,n} > \phi_{0,n} + \sum_{j=0}^{\infty} < f, \psi_{j,n} > \psi_{j,n}$$

The approximation difference is, therefore,

$$f - f_N = \sum_{j=J}^{\infty} < f, \psi_{j,n} > \psi_{j,n}$$

and linear approximation error is

$$e_{w,N} = \|f - f_N\|^2 = \sum_{j=J}^{\infty} | < f, \psi_{j,n} > |^2$$

2.3.1.1 Linear Approximation Example: Fourier vs Db2 Wavelet

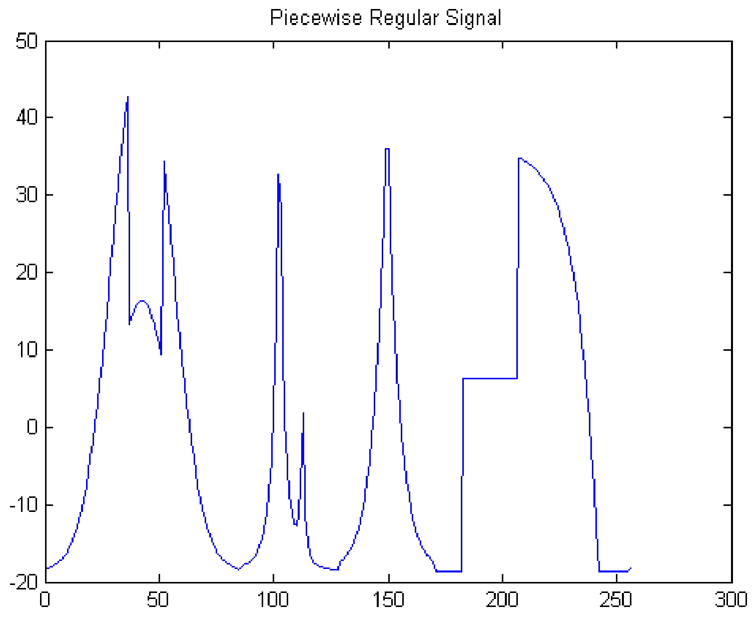
A $N = 256$ length piecewise regular signal is used in these computations. The matlab script is as follows

```
function f=waveapprox

% Linear Fourier and Wavelet Approximations using 100 coefficients out of 256
f = load_signal('Piece-Regular', 256); % Using this Wavelab function to
% generate a piecewise signal
x=f';% Preparing signal for wt.m
% Fourier Linear Aproximation using 100 coefficients
figure(1);plot(x),title('Piecewise Regular Signal');
yf=fft(x);
yf_shift=fftshift(yf);
% plot(abs(yf_shift));
% Signal reconstruction using all coefficients
% oup_f=ifft(yf);
% figure(2)
% plot(real(oup_f));

% Signal Reconstruction using n=100 coefficients

% Linear Fourier Approximation
n=100;
N=length(yf_shift);
yf_appx=[zeros(1,N/2-n/2),yf_shift(N/2-n/2+1:N/2+n/2),zeros(1,N/2-n/2)]
yf_appx=fftshift(yf_appx);
oup_f=ifft(yf_appx);
figure(2)
plot(real(oup_f));title(' Linear Fourier Approximation 100/256 coeffs');

% Linear Wavelet Approximation

% Signal is decomposed to level-4 and first 100 coefficients are chosen.
% Using Db2 wavelets
[lp,hp,lp2,hp2]=wfilters('db2');
J=4;
[cA,cD,dcoeff]=wt_test(x,J,lp,hp);

% Choosing 100 coefficients
% cA-16, dcoeff(4,:)-16, dcoeff(3,:)-32, dcoeff(2,:) - Choose first 36
% coefficients while all coefficients of dcoeff(1,:) are set to zero.
 dcoeff(1,:)=zeros(size(dcoeff(1,:)));
 dcoeff(2,:)=[dcoeff(2,1:36),zeros(1,length(dcoeff(2,:))-36)];
figure(3)
```

```
yw=iwt_test(cA,dcoeff,J,lp2,hp2);
plot(yw),title('Linear Wavelet Approximtion 100/256 coeffs');
coeff2=[cA,dcoeff(4,1:16),dcoeff(3,1:32),dcoeff(2,1:64),dcoeff(1,:)]
```

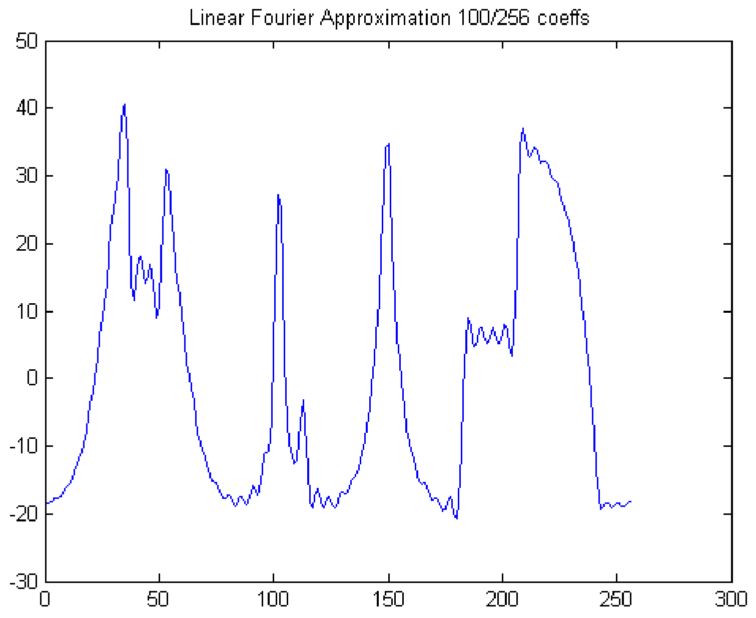
Fourier Linear Approximation computation is pretty straightforward. 256 point FFT is computed and only first 100 coefficients(50 on either side of $\omega = 0$) corresponding to lowest frequencies are retained.We reconstruct the signal (Inverse FFT) using only these 100 coefficients while setting the rest equal to zero.

Fig. 2.1 N=256 Piecewise Regular Signal



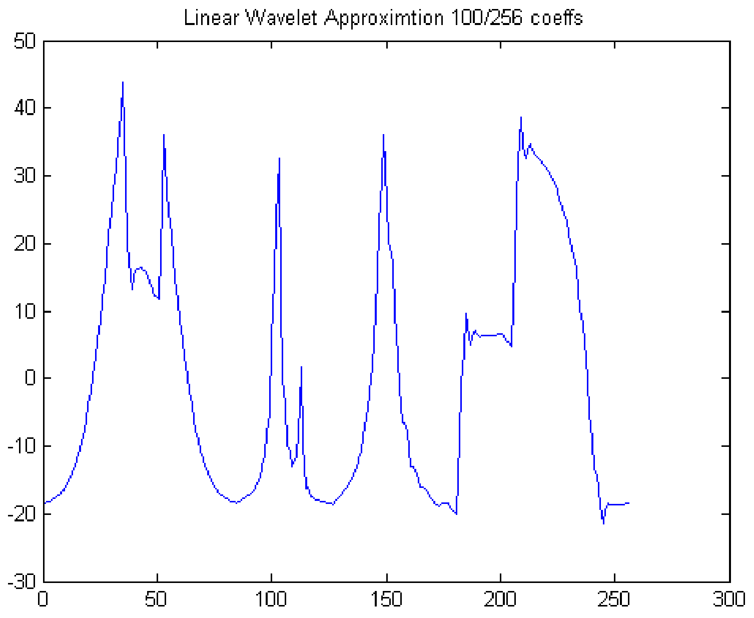
The output signal is plotted below.

Fig. 2.2 100 coefficients Linear Fourier Approximation



Linear Wavelet Approximation is a bit more involved. Firstly, the input signal is decomposed to a scale J.In this demo, J is 4. Then we retain 100 coefficients starting from the coarsest scale, ie., we start with approximation and detail coefficients at the coarsest scale, then we choose detail coefficients from the next finest scale and so on. In this example, approximation and detail coefficients at the coarsest scale have 16 coefficients each. The next finer scale has 32 detail coefficients followed by 64 at the following one and so on. Since we are choosing 100 coefficients , we retain only $36(= 100 - 16 - 16 - 32)$. The rest of the coefficients are set to zero and the inverse DWT is computed in order to reconstruct the signal.The output signal is plotted below

Fig. 2.3 100 coefficients Linear Wavelet Approximation



It can be seen that Wavelet approximation returns a better result with this kind of signal.While linear fourier approximation has a low-frequency bias wavelets tend to capture discontinuities and signal fluctuations at almost every scale which results in better approximation performance.

2.4 Non-Linear Approximation

In data compression applications, a good approximation error rate decay is crucial and it makes sense to utilize wavelet property of returning large coefficients at discontinuities and signal fluctuations. For an orthonormal basis e_k, the approximation error is given by

$$e_N = \|f - f_N\|^2 = \sum_{k=N}^{\infty} | < f, e_k > |^2$$

In linear case, we choose first M coefficients ,eg. in Fourier domain we go with lowest $M/2$ positive and negative frequencies and in wavelet domain we choose first M coefficients starting with the coarsest scale. In Nonlinear case,we go with M largest coefficients by choosing the M largest inner products. So instead of dividing the coefficient space linearly, we do an nonlinear division by allocating coefficients in two sets denoted by ,say, I_m which contains the M largest coefficients and a set which contains all other coefficients.

$$| < f, e_m > | \geq | < f, e_n > |, \forall m \in I_m, \forall n \notin I_m$$

The best nonlinear approximation is ,therefore, given by

$$f_M = \sum_{m \in I_m} < f, e_m > e_m$$

for an orthonormal basis. The approximation error is given as before

$$e_M = \|f - f_M\|^2 = \sum_{m \notin I_m} | < f, e_m > |^2$$

2.4.0.2 Non-Linear Approximation Error Decay

Let us consider Fourier Transform first. If the function is smooth, Fourier transforms do a good job as most energy is concentrated in the lower frequencies and there is not much difference in using Linear and Non-linear Fourier approximation. If the function is piece-wise smooth, ie. it has discontinuities at points t_k, fourier approximation does not give us appreciably better results as largest coefficients are clustered at the lowest frequencies that we pick in linear case anyway.It has been proven that approximation error rate decays as $1/M$ in Fourier case.

Wavelets return large coefficients value at singularities, discontinuities etc. at every scale. If we are using linear approximation, we start with coarse scales and move to the finer scales but as is obvious quite a few large coefficients are ignored using this method. Let us suppose that there are M large coefficients distributed over M scales. If we are using non-linear approximation we can conceivably pick out all of these M coefficients instead of picking M coefficients across only $J < M$ scales. The amplitude of wavelet coefficient is proportional to $2^{-j/2}$ at scale j. So if we stop at scale $J < M$ using linear approximation, the last coefficients picked are of the order $2^{-J/2}$ while the largest coefficient not selected has an amplitude in the range of $2^{-(J+1)/2}$. However, if we are using non-linear approximation, the largest coefficient not picked will have the order $2^{-(M+1)/2}$ so we see that in non-linear case , the error is decaying exponentially.

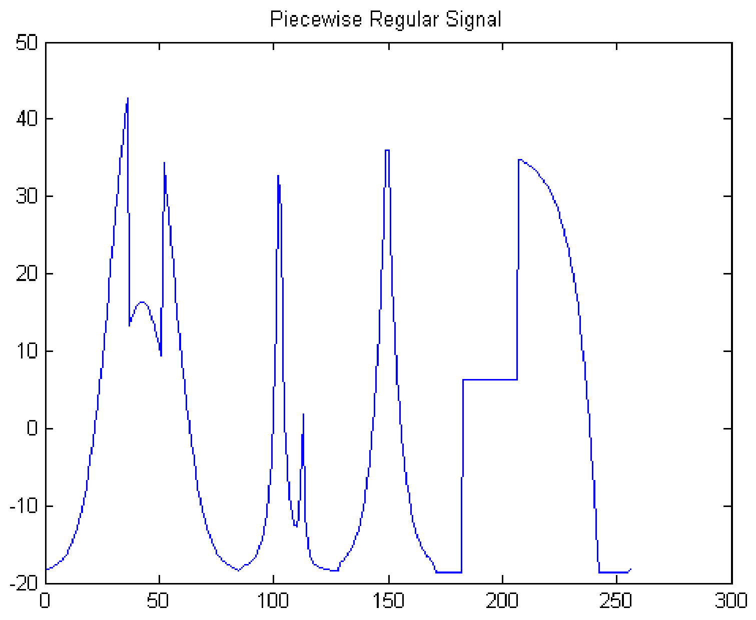
This can be demonstrated in Matlab using the following script

```
 function f=nlapprox

% Linear Fourier and Wavelet Approximations using 100
%coefficients out of 256
f = load_signal('Piece-Regular', 256); % Using this Wavelab
% function to generate a piecewise signal
x=f';% Preparing signal for wt.m
```

```
% Fourier Nonlinear Aproximation using 100 coefficients
figure(1);plot(x),title('Piecewise Regular Signal');
yf=fft(x);
yf_shift=fftshift(yf);
% plot(abs(yf_shift));
% Signal reconstruction using all coeffiecients
% oup_f=ifft(yf);
% figure(2)
% plot(real(oup_f));

% Signal Reconstruction using n=100 coefficients

% Nonlinear Fourier Approximation
n=100;
N=length(yf_shift);
tf=findthresh(yf_shift,n);
yf_shift(abs(yf_shift)<tf)=0;
yf_appx=fftshift(yf_shift);
oup_f=ifft(yf_appx);
figure(2)
plot(real(oup_f));
title(' Nonlinear Fourier Approximation 100/256 coeffs');

% Linear Wavelet Approximation

% Signal is decomposed to level-4 and signal
% is thresholded to choose 100 coefficients
% Using Db2 wavelets
[lp,hp,lp2,hp2]=wfilters('db2');
J=4;
[cA,cD,dcoeff]=wt_test(x,J,lp,hp);

% Choosing 100 coefficients
% All 256 coefficients are processed to find the
%threshold and the largest 100 coefficients are chosen
 % while those below the thresold are set to zero.
coeff2=[cA,dcoeff(4,1:16),dcoeff(3,1:32),....
                    dcoeff(2,1:64),dcoeff(1,1:128)];
tw=findthresh(coeff2,n);
coeff2(abs(coeff2)<tw)=0;
cA=coeff2(1:16);
dcoeff(4,1:16)=coeff2(16+1:2*16);
dcoeff(3,1:32)=coeff2(32+1:2*32);
dcoeff(2,1:64)=coeff2(64+1:2*64);
dcoeff(1,:)=coeff2(128+1:2*128);
```

```
figure(3)
yw=iwt_test(cA,dcoeff,J,lp2,hp2);
plot(yw),title('Nonlinear Wavelet Approximtion 100/256 coeffs');
```

Fig. 2.4 N=256 Piecewise Regular Signal



As can be seen from the program , a threshold is chosen depending on the number of approximation coefficients and all coefficients that fall below this threshold are set to zero. We calculate Inverse FFT and Inverse DWT based on these new values of coefficients. While Nonlinear Fourier Approximation doesn't show much improvement over its linear counterpart but ,as expected Nonlinear Wavelet approximation shows dramatic improvement over linear Wavelet Approximation.

Fig. 2.5 100 coefficients Nonlinear Fourier Approximation

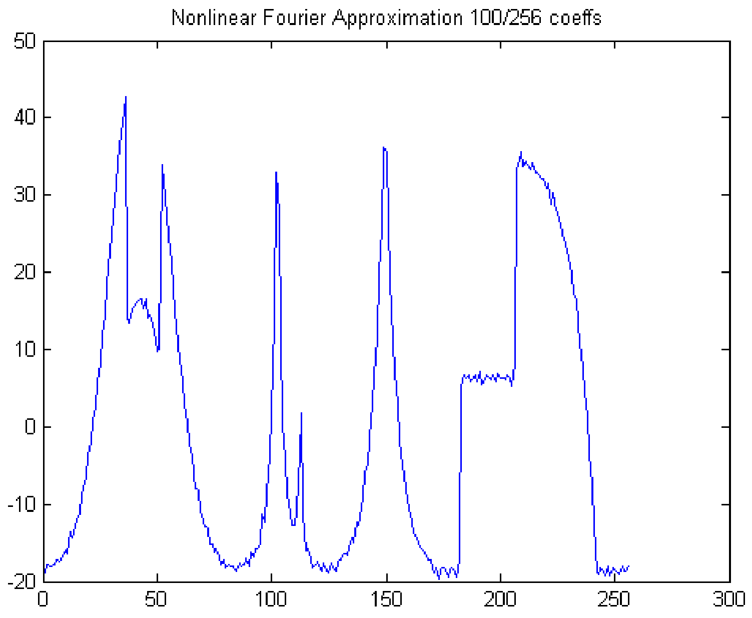
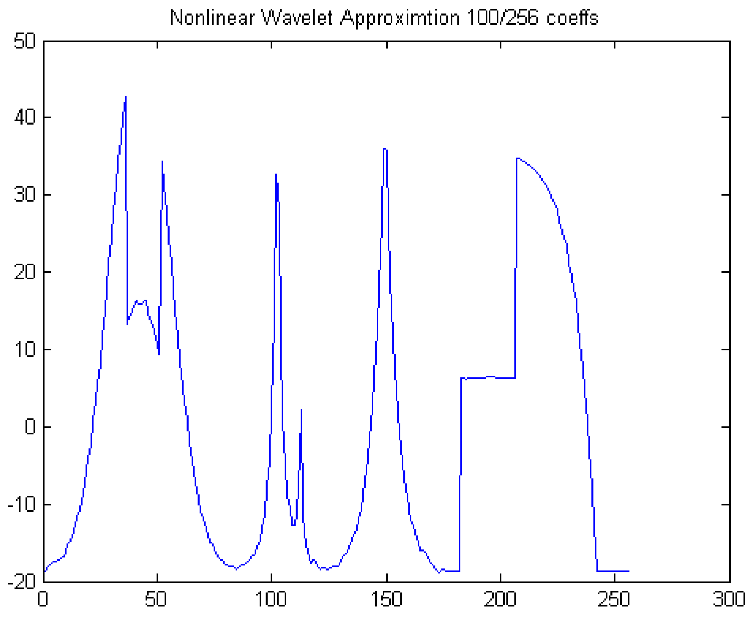


Fig. 2.6 100 coefficients Nonlinear Wavelet Approximation



References

1. O Christensen (2002) An Introduction to Frames and Riesz Bases. Birkhuser Boston
2. G Peyr (2010) Advanced Signal, Image and Surface Processing. Manuscript
3. S Mallat (1999) A Wavelet Tour of Signal Processing. Academic Press
4. G Strang, T Nguyen(1996) Wavelets and Filter Banks. Wellesley Cambridge Press
5. M Vetterli, J Kovacevic(1995) Wavelets and Subband Coding. Prentice Hall Signal Processing Series
6. I Daubechies (1992) Ten Lectures on Wavelets. SIAM: Society for Industrial and Applied Mathematics
7. M Weeks (2006) Digital Signal Processing using Matlab and Wavelets. Infinity Science Press
8. V Goyal, M Vetterli, J Kovacevic (2010) Fourier and Wavelet Signal Processing. Manuscript Draft

Chapter 3
Wavelets and Wavelet Transforms

Abstract Each chapter should be preceded by an abstract (10–15 lines long) that summarizes the content. The abstract will appear *online* at www.SpringerLink.com and be available with unrestricted access. This allows unregistered users to read the abstract as a teaser for the complete chapter. As a general rule the abstracts will not appear in the printed version of your book unless it is the style of your particular book or that of the series to which your book belongs.

Please use the 'starred' version of the new Springer `abstract` command for typesetting the text of the online abstracts (cf. source file of this chapter template `abstract`) and include them with the source files of your manuscript. Use the plain `abstract` command if the abstract is also to appear in the printed version of the book.

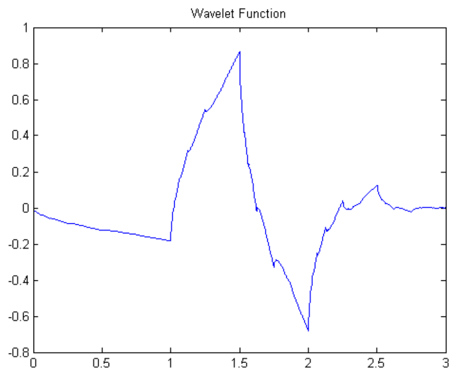
3.1 Introduction

Wavelets are normalized,finite, short-duration, zero mean functions.

$$\int_{-\infty}^{\infty} \psi(t)\, dt = 0$$

$\psi(t)$ is also known as Mother wavelet as it can be dilated and translated to yield Child wavelets. Function to be analyzed is then processed with these Children wavelets to yield wavelet coefficients.An example of a wavelet function is shown below. It is a Daubechies2 wavelet generated using Matlab.

Fig. 3.1 Daubechies2 Wavelet



The Children wavelets are given by $\psi_{k,s}(t)$ where the mother wavelet is scaled by s and translated by k.

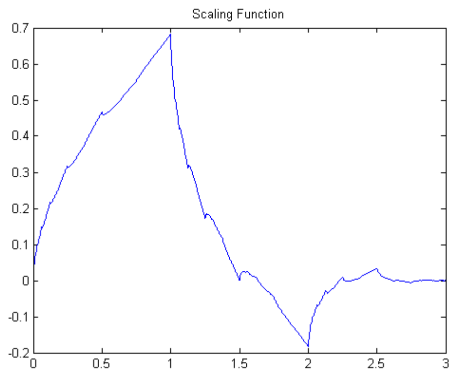
$$\psi_{k,s}(t) = \frac{1}{\sqrt{s}}\psi(\frac{t-k}{s})$$

The Wavelet Transform Wf of a function f(t) is computed by taking the inner product of function f(t) with the translated and dilated versions of mother wavelet.

$$Wf = < f, \psi_{k,s} > = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\psi^*(\frac{t-k}{s})\,dt$$

For small values of s, $\psi_{k,s}$ will be of shorter duration and higher frequency. For large value of s, $\psi_{k,s}$ it will be more spread out in time and will consist of low frequencies. The fact that wavelet functions are bandpass functions ensures that we cannot cover the entire frequency spectrum just with the wavelet functions.To solve this problem, scaling functions $\phi(t)$ are introduced. They are complement of wavelet functions and correspond to low pass filter in signal processing terms.

Fig. 3.2 Daubechies2 Scaling Function



Wavelet Transform, so defined, happens to be computationally unwieldy as translations and dilations can take any value. A better approach is to discretize translation and dilation steps. Mathematically, let $s = a^m$ and $k = a^m n$ where m and n are integers. Below is an example of Daubechies2 wavelet being scaled by $a = 1/2$.The scale a is inversely proportional to frequency. Small scale values [$0 < a < 1$] correspond to high frequency while large scale values [$a > 1$] to low frequencies. The second figure shows the same mother wavelet being shifted by $b = 20$.

Fig. 3.3 Wavelet Scaling Demo

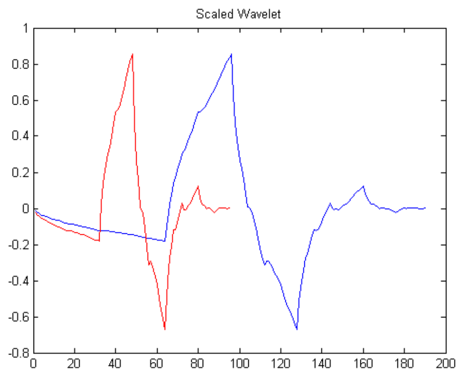
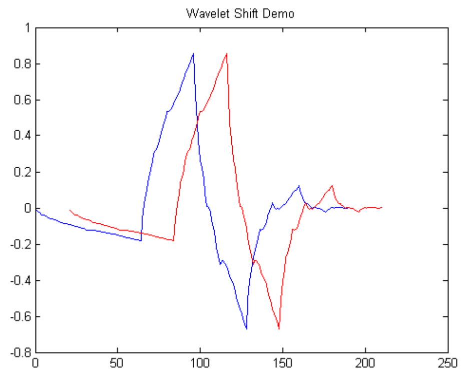


Fig. 3.4 Wavelet Shift Demo



$$\psi_{m,n}(t) = a^{\frac{-m}{2}} \psi(a^{-m}t - n)$$

3.1.1 Wavelet properties

1. Vanishing Moments: A wavelet of n vanishing moments is orthogonal to polynomials of degree $n-1$. All polynomials of degree less than n will get filtered out when they are processed by wavelet $\psi(t)$

$$\int_{-\infty}^{\infty} t^k \psi(t)dt = 0, for(0 \leq k < n)$$

For a given scaling function and wavelet derived from low pass filter $h(n)$, number of vanishing moments depends solely on number of zeros at $\omega = \pi$. In other words, its transfer function and its $n-1$ derivatives in frequency(or z) domain vanish at $\omega = \pi(z = -1)$.

2. Regularity: of a wavelet is related to differentiability of a wavelet in the frequency domain. More regularity corresponds to smoother wavelet and more vanishing moments.Scaling filter is N regular if it has N zeros at $\omega = \pi$. For more on local and global regularity, refer to books by Daubechies and Mallat.

3. Admissibility condition is given by

$$\int_{-\infty}^{\infty} \frac{|\psi(\omega)|^2}{|\omega|} d\omega < \infty$$

Wavelets being band pass functions is an implication of admissibility condition.

4. Compact Support: Wavelets should ideally have compact support. Like other properties, this property too depends on the low pass filter h.

5. Symmetry: Certain wavelets , eg. biorthogonal wavelets , are designed to be symmetric. These are specially useful in image processing applications.

6. Orthogonality: With the exception of Haar wavelets, wavelets can't be both symmetric and orthogonal so the choice usually comes down to specific applications and requirements.

3.2 Discrete Wavelet Transform

Discrete Wavelet Transform was introduced previously with translation and dilation steps being uniformly discretized.

$$\psi_{m,n}(t) = a^{\frac{-m}{2}} \psi(a^{-m}t - n)$$

To make computations simpler and to ensure perfect or near-perfect reconstruction, Dyadic Wavelet Transform is utilized. In the dyadic case $\{a\}$ is chosen to be equal to 2 which yields the following translation- dilation equation.

$$\psi_{j,n}(x) = 2^{\frac{-j}{2}} \psi(2^{-j}x - n)$$

where j gives the level of scale and n gives the translation where $j, n \in Z$. As noted previously, wavelet functions $\psi(t)$ are bandpass so even with dyadic scaling they cannot cover the entire spectrum so low pass scaling functions $\phi(t)$ are introduced. Therefore, wavelet coefficients of a function over scales $j_0 < j < J$ are then given by

$$Wf = < f, \phi_{j_0,n} > + \sum_{j=j_0}^{J} < f, \psi_{j,n} >$$

Let there be functions $\tilde{\phi_{j_0,n}}$ and $\tilde{\psi_{j,n}}$ such that function $f(x)$ can be reconstructed from its wavelet coefficients using following equation

$$f(x) = \sum_n < f, \phi_{j_0,n} > \tilde{\phi_{j_0,n}}(x) + \sum_n \sum_{j=j_0}^{J} < f, \psi_{j,n} > \tilde{\psi_{j,n}}(x)$$

The ease with which dyadic Inverse Discrete Wavelet Transform (IDWT) can be constructed makes it ideal for a number of signal processing and image processing applications where reconstruction is absolutely critical(eg., Image compression). More on Inverse DWT using filter banks in next few chapters when this topic will be revisited.

3.2.1 DWT Implementation

DWT is implemented using decimated filter banks and this implementation is a major reason why DWT has become so popular in DSP and other fields. The concepts behind this implementation will be developed over next few sections but we'll start with the implementation here with results displayed using Matlab.

Fig. 3.5 Iterated Filter Bank(Analysis

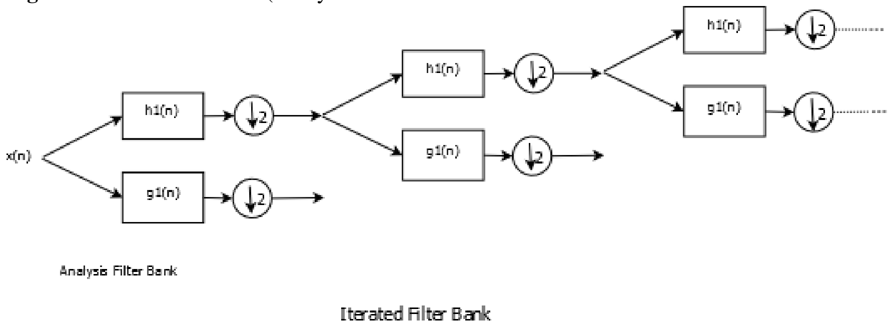
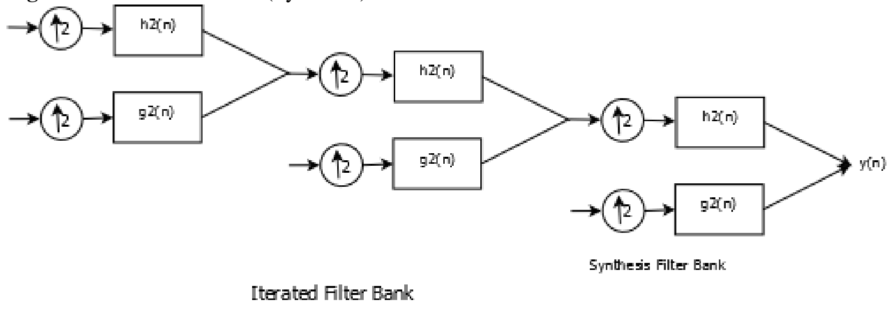
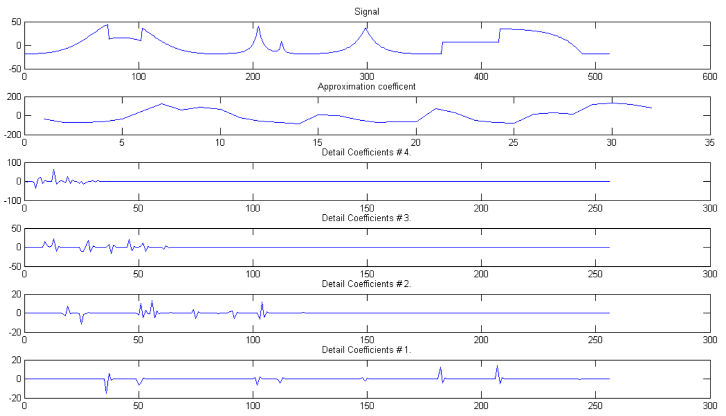


Fig. 3.6 Iterated Filter Bank(Synthesis)



The figures show DWT and Inverse DWT implementations using iterated and decimated filter banks. Consider DWT, at each stage the signal is convolved with low pass and high pass filters and the result is decimated by 2. Decimation is necessary in order to keep the system non-redundant as the number of coefficients will, otherwise, double at every stage. The downside is that decimation introduces shift variance.More on this later. For now, we'll pass a signal through the filter bank shown in the first signal and analyze it at every stage. Let us say that the signal is decomposed through 4-levels of the filter banks. At the fourth stage, we have two sets of coefficients- Approximation coefficients corresponding to low pass filter and Detail coefficients corresponding to high pass filtering. Since the low pass approximation coefficients at stages 1 through 3 are decomposed we'll only consider the detail coefficients at these three stages.At every stage, higher frequency components are peeled off the signal and we are left with a smoother version of the original signal. The analysis is shown in figure.

Fig. 3.7 DWT Analysis of a given signal



3.3 Multiresolution Analysis

Mother Wavelet $\psi(t)$ is an orthonormal wavelet if its translates and dilates are orthonormal under the inner product.

$$< \psi_{j,k}, \psi_{l,m} >= \int_{-\infty}^{\infty} \psi_{j,k}(t)\psi^*_{l,m}(t)dt = \delta_{j,l}\delta_{k,m}$$

where $j,k,l,m \in Z$ and δ is the Kronecker delta function. Such a wavelet system is self-dual,ie., for a given function $f(x)$

$$f(x) = \sum_j \sum_k < f, \psi_{j,k} > \psi_{j,k}(x)$$

The idea of Multiresolution is to decompose a signal $f(t) \in L^2(R)$ such that orthogonal projections of $f(t)$ given by f_j "live" in the space V_j.V_j and W_j are complementary subspaces with W_j being the difference between V_j and V_{j+1}.

$$V_{j+1} = V_j \oplus W_j$$

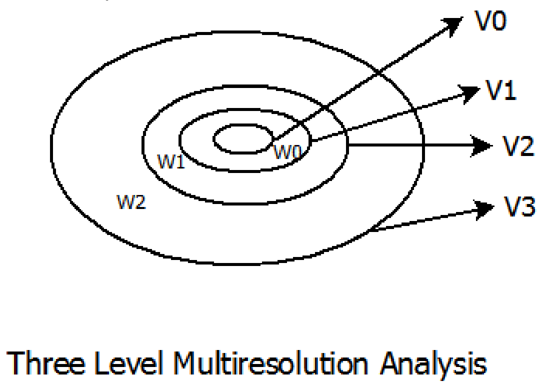
Multiresolution approximation, as defined by Mallat and Meyer, has the following properties-

1. $V_j \subset V_{j+1}$ A function in subspace j is in all the finer subspaces. In other words, if we know a signal $f_j(t)$ at subspace V_j, we can obtain ts coarse approximation using MRA. Think of a signal being decomposed using an iterated chain of complementary low pass and high pass filters. At every step we obtain a low pass and high pass version of the signal from the previous step. However, the low pass signal in step two is contained in the signal from the step one.

2. $f(t) \in V_0 \Leftrightarrow f(t-k) \in V_0$ This is the translation (shift) invariant property of the subspace. A signal in a given subspace , if translated by $k \in Z$ is still in that subspace. This property is valid for all subspaces.

3. $f(t) \in V_j \Leftrightarrow f(2t) \in V_{j+1}$ This is the scale invariant property of the Multiresolution analysis. In frequency domain terms, $f(2t)$ contains $2X$ highest frequency compared to that contained in $f(t)$. Using iterated filter bank example with a low pass filter that halves the frequencies in every step, it becomes clear that moving back one step in each step of the filter chain doubles the highest frequency content.

Fig. 3.8 Multiresolution Analysis



4. $\bigcap_{j\to-\infty} V_j = \{0\}$ As we move to lower subspaces, the space occupied by V_j shrinks until it becomes nearly zero.

5.$\bigcup_{j\to\infty} V_j = L^2(R)$ Union of all subspaces as $j \to \infty$ encompasses the whole $L^2(R)$ space.

3.3.1 Dilation Equation

Let $f_0(t)$ and $f_1(t)$ be the projections of signal $f(t)$ associated with subspaces V_0 and V_1 respectively. The difference of these two projections "lives" in the complementary Wavelet space W_0.

According to 1, $V_0 \subset V_1$. Let $\phi(t)$ and $\psi(t)$ be the scaling and wavelet functions associated with subspaces V_0 and W_0. It follows from 1 that $\phi(t)$ is contained in V_1 and can be expressed in terms of $\phi(2t)$.

$$\phi(t) = 2^{\frac{1}{2}} \sum_k h(k)\phi(2t-k)$$

The term $2^{\frac{1}{2}}$ comes from the definition of basis function at scale $j = 1$. The equation above is known as the dilation equation and forms the bridge between wavelets and filterbanks. $h(k)$ corresponds to the low pass filter that is associated with the scaling function. If $g(k)$ is the complementary high pass filter in this orthogonal filterbank then the Wavelet equation can be expressed as

$$\psi(t) = 2^{\frac{1}{2}} \sum_k g(k)\phi(2t-k)$$

The wavelet equation written above follows from the fact that W_0 also nests inside V_1 and can be obtained by using the similar process as used in the scaling function case.

3.3.2 Wavelet Transform Algorithm

Again from 1, $\psi_{j,k}(t)$ and $\phi_{j,k}(t)$ can be represented by $\phi_{j+1,k(t)}$. Let $S_{j,k}$ and $W_{j,k}$ be scaling and wavelet coefficients at scale j for a given signal $f(t)$.

$$S_{j,k} = \int_{-\infty}^{\infty} f(t)\phi_{j,k}(t)dt$$

$$W_{j,k} = \int_{-\infty}^{\infty} f(t)\psi_{j,k}(t)dt$$

Using MRA properties,

$$\sum S_{j+1,k}\phi_{j+1,k}(t) = \sum W_{j,k}\psi_{j,k}(t) + \sum S_{j,k}\phi_{j,k}(t)$$

as signal projection $f_{j+1}(t) = f_j(t) + \Delta f_j(t)$ where $\Delta f_j(t)$ is the signal projection in the wavelet space. We shift dilation and wavelet equations by v and generalize to scale j.

$$\phi(2^j t - v) = 2^{\frac{1}{2}} \sum_k h(k)\phi(2^{j+1}t - 2v - k)$$

$$\psi(2^j t - v) = 2^{\frac{1}{2}} \sum_k g(k)\phi(2^{j+1}t - 2v - k)$$

Substituting $l = 2v + k$ and integrating by multiplying both equations by $f(t)$ gives

$$\int_{-\infty}^{\infty} f(t)\phi_{j,v}dt = 2^{\frac{1}{2}} \sum_k h(l-2v) \int_{-\infty}^{\infty} f(t)\phi_{j+1,l}(t)dt$$

$$\int_{-\infty}^{\infty} f(t)\psi_{j,v}dt = 2^{\frac{1}{2}} \sum_k g(l-2v) \int_{-\infty}^{\infty} f(t)\phi_{j+1,l}(t)dt$$

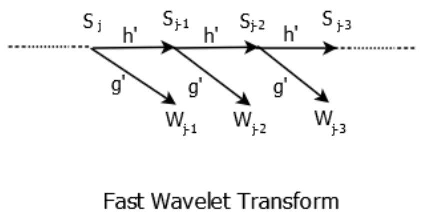
Using values of $S_{j,k}$ and $W_{j,k}$ from above yields

$$S_{j,v} = 2^{\frac{1}{2}} \sum_k h(l-2v) S_{j+1,l}$$

$$W_{j,v} = 2^{\frac{1}{2}} \sum_k g(l-2v) S_{j+1,l}$$

where $h(l-2v)$ and $g(l-2v)$ can be thought of as time-reversed and decimated by 2 low pass and high pass filters. This is the fast wavelet transform algorithm.

Fig. 3.9 Fast Wavelet Transform



The Inverse Fast Wavelet transform is simply inverse of the FWT and can be easily shown using same equations. More on Filterbanks and Wavelets will be covered in next chapter.

3.4 Filter Banks and Wavelets

As mentioned in the previous chapter Discrete Wavelet Transform can be obtained by iterating over low pass filters of a given filter bank.

Fig. 3.10 Iterated Filter Bank(Analysis

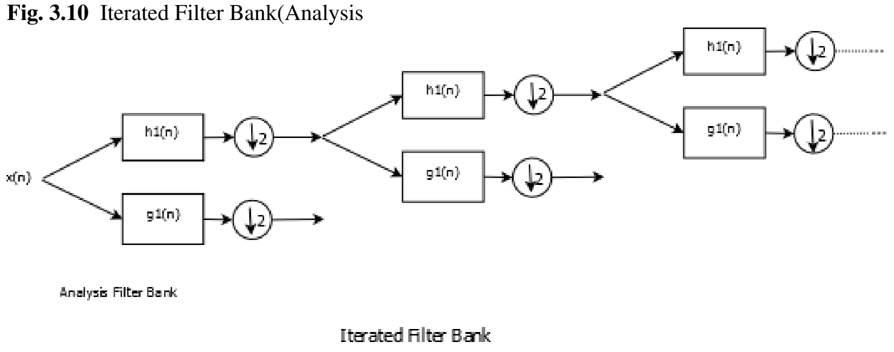
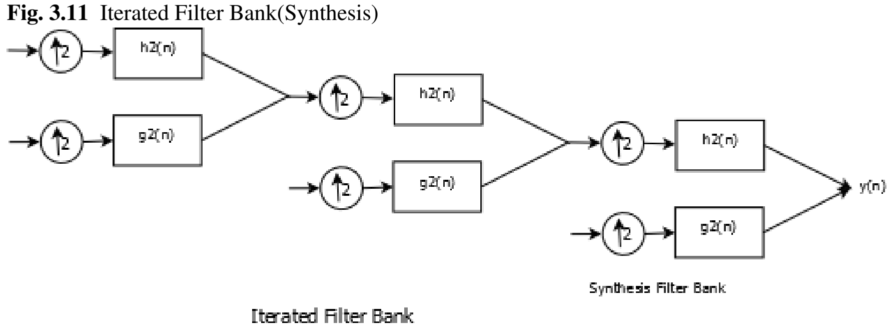


Fig. 3.11 Iterated Filter Bank(Synthesis)



Let us consider one stage filtering and downsampling in the analysis filter bank. Downsampling is needed in order to get rid of redundancy as , say, a 1000 point input sample results in two 1000 point outputs after filtering. We downsample by two in order to keep the signal samples at 1000. Similarly at the synthesis bank, we first upsample the wavelet coefficients by 2 before filtering. Filter have to be designed appropriately in order to ensure that the signal reconstruction is perfect. Filter choices also effect wavelet shape as will be demonstrated shortly.

Recall the dilation and wavelet equations.

$$\phi(t) = 2\sum_k h(k)\phi(2t-k)$$

$$\psi(t) = 2\sum_k g(k)\phi(2t-k)$$

If we iterate over these equations then the iteration can be written as

$$\phi^{(i+1)}(t) = 2\sum_{k} h(k)\phi^{(i)}(2t-k)$$

Keep in mind that both wavelet and dilation equations will end up getting iterated over $\phi(t)$ where $\phi^{(0)}(t)$ is the box function $[1,1]$. If the functions converge then we should be able to get wavelet and scaling functions in few steps. In the example below Daubechies's db2 wavelet and scaling filters were iterated to yield wavelets and scaling functions in five iterations.

Fig. 3.12 Db2 4-tap Low Pass and High pass Filters

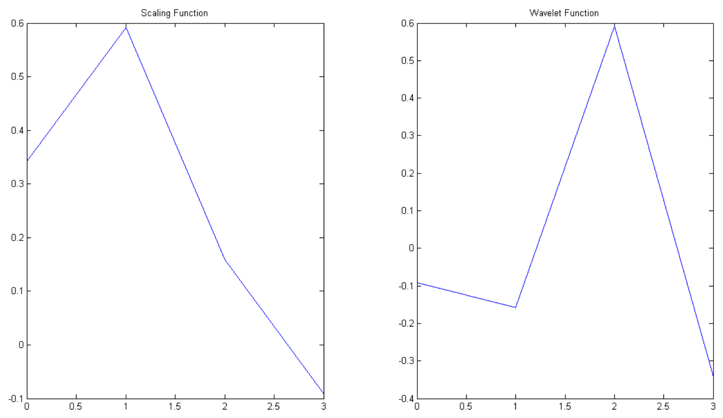


Fig. 3.13 Scaling and Wavelet Function Iteration 1

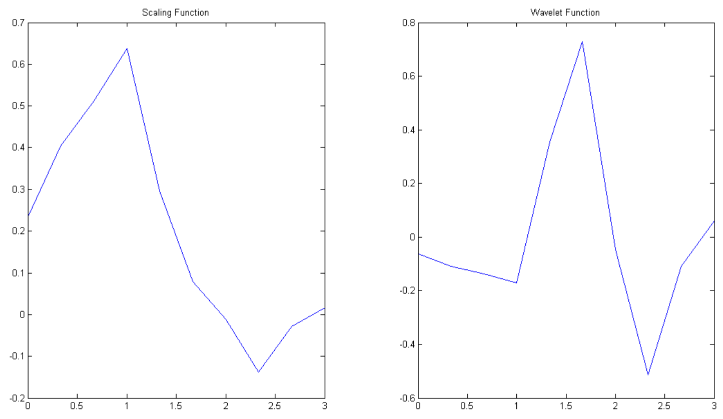


Fig. 3.14 Scaling and Wavelet Function Iteration 2

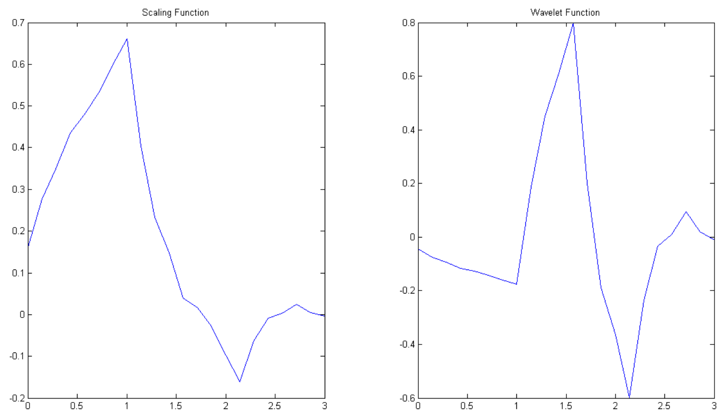


Fig. 3.15 Scaling and Wavelet Function Iteration 3

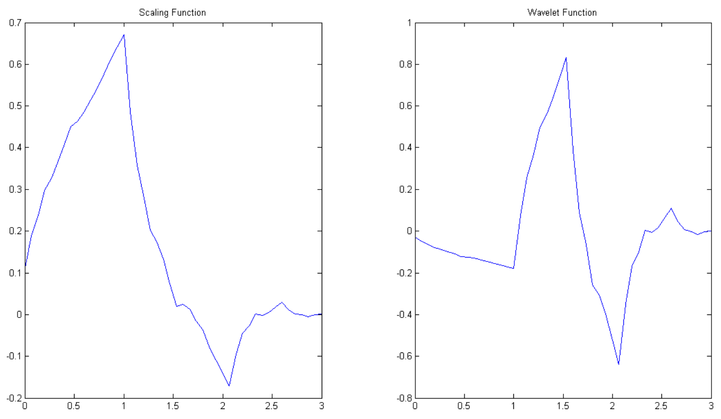


Fig. 3.16 Scaling and Wavelet Function Iteration 4

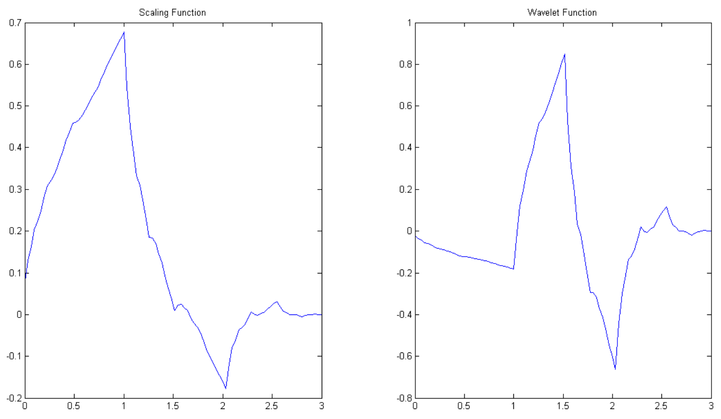
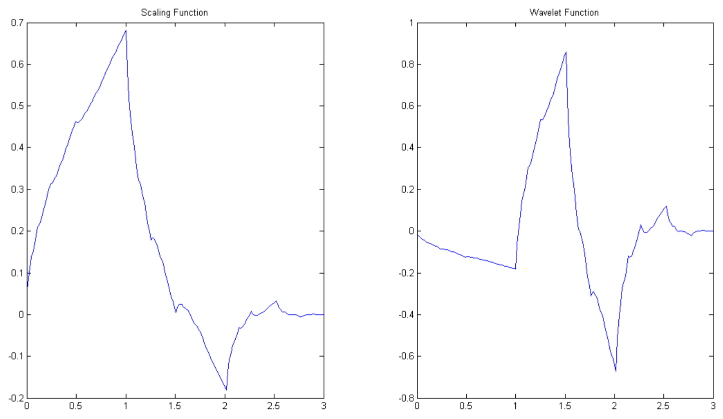


Fig. 3.17 Scaling and Wavelet Function Iteration 5



3.4.1 *Orthonormal Discrete Wavelet Transform*

In terms of scaling and wavelet functions we have three orthonormality equations at a given scale j

$$< \phi_{j,k}, \phi_{j,l} >= \int_{-\infty}^{\infty} 2^{\frac{j}{2}} \phi(2^j t - k) 2^{\frac{j}{2}} \phi^*(2^j t - l) dt = \delta(k-l)$$

$$< \psi_{j,k}, \psi_{j,l} >= \int_{-\infty}^{\infty} 2^{\frac{j}{2}} \psi(2^j t - k) 2^{\frac{j}{2}} \psi^*(2^j t - l) dt = \delta(k-l)$$

$$< \phi_{j,k}, \psi_{j,l} >= \int_{-\infty}^{\infty} 2^{\frac{j}{2}} \phi(2^j t - k) 2^{\frac{j}{2}} \psi^*(2^j t - l) dt = 0$$

Derivation is straightforward. If h_1 and g_1 are low pass and high pass filters respectively then in an orthonormal filter bank, their inner product is zero. Use dilation equations, take inner product of scaling and wavelet functions. Notice that it can be expressed in forms of h_1 and $g_1$and that these two are orthonormal filters satisfying orthonormality properties.

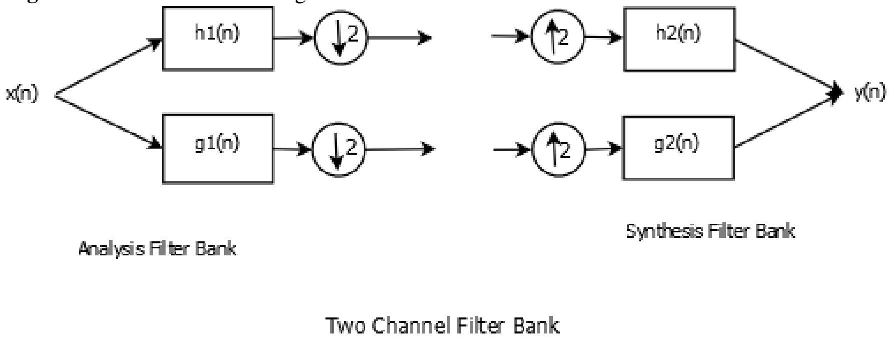
Discrete Wavelet Transform requires that four filters be designed- two for analysis bank and the other two for synthesis bank. However, perfect reconstruction imposes conditions on filters such that no more than two filters are enough to design a DWT Multiresolution Analysis. In case of Orthogonal DWT, only one filter is sufficient as the other three filters can be obtained from the prototypical low pass filter. More on this will follow after discussion of Biorthogonal wavelets. The analysis high pass filter is the mirror image (alternate flip) of low pass filter while synthesis filters are transposes of analysis filters.For a given $N+1$-tap low pass analysis filter $H_1(z)$, the other three filters are

High Pass Analysis Filter $H_2(z) = -z^{-N}H_1(z)$
Low Pass Synthesis Filter $G_1(z) = H_2(-z)$
High pass Synthesis Filter $G_2(z) = -H_1(-z)$

3.4.2 Biorthogonal Discrete Wavelet Transform

We get biorthogonal wavelets from biorthogonal filters. Biorthogonal analysis filters are not orthogonal to each other but the analysis filters and synthesis filters are orthogonal with respect to each other. Consider a two-channel biorthogonal filter bank.

Fig. 3.18 Two Channel Biorthogonal Filter Bank



In case of orthonormal filter banks we had three orthonormality equations. In case of biorthogonal filter banks, we have four equations. In filter bank terms, these four equations are

$$< h_1(n), h_2(2k-n) >= \delta(k)$$

$$< g_1(n), g_2(2k-n) >= \delta(k)$$

$$< h_1(n), g_2(2k-n) >= 0$$

$$< g_1(n), h_2(2k-n) >= 0$$

3.4.3 Biorthogonal Multiresolution Analysis

In biorthogonal case, $V_{j+1} = V_j \oplus W_j$ is still valid but V_j and W_j are not orthogonal complements. This necessitates introduction of another complementary multiresolution space. It is usually denoted by $\widetilde{V}_j$ and $\widetilde{W}_j$ such that

$$\widetilde{V}_{j+1} = \widetilde{V}_j + \widetilde{W}_j$$

The relationship between "normal" MRA and "Tilde" MRA is crucial as it defines biorthogonality

$$\widetilde{V}_j \perp W_j$$

$$V_j \perp \widetilde{W}_j$$

"Tilde" MRA is usually associated with analysis filter bank while the "normal" MRA is associated with synthesis filter bank. Therefore $\widetilde{\phi}$ and $\widetilde{\psi}$ are analysis scaling and wavelet functions while ϕ and ψ are synthesis scaling and wavelet functions, respectively. We also have two sets of dilation equations in this case.

Analysis Dilation Equations

$$\widetilde{\phi}(t) = 2\sum_k \widetilde{h}(k)\widetilde{\phi}(2t-k)$$

$$\widetilde{\psi}(t) = 2\sum_k \widetilde{g}(k)\widetilde{\phi}(2t-k)$$

Synthesis Dilation Equations

$$\phi(t) = 2\sum_k h(k)\phi(2t-k)$$

$$\psi(t) = 2\sum_k g(k)\phi(2t-k)$$

In terms of scaling and wavelet functions we have four biorthogonality equations at a given scale j

$$<\phi_{j,k}, \widetilde{\phi}_{j,l}> = \int_{-\infty}^{\infty} 2^{\frac{j}{2}}\phi(2^j t-k)2^{\frac{j}{2}}\widetilde{\phi}^*(2^j t-l)dt = \delta(k-l)$$

$$<\psi_{j,k}, \widetilde{\psi}_{j,l}> = \int_{-\infty}^{\infty} 2^{\frac{j}{2}}\psi(2^j t-k)2^{\frac{j}{2}}\widetilde{\psi}^*(2^j t-l)dt = \delta(k-l)$$

$$<\widetilde{\phi}_{j,k}, \psi_{j,l}> = \int_{-\infty}^{\infty} 2^{\frac{j}{2}}\widetilde{\phi}(2^j t-k)2^{\frac{j}{2}}\psi^*(2^j t-l)dt = 0$$

$$<\widetilde{\psi}_{j,k}, \phi_{j,l}> = \int_{-\infty}^{\infty} 2^{\frac{j}{2}}\widetilde{\psi}(2^j t-k)2^{\frac{j}{2}}\phi^*(2^j t-l)dt = 0$$

Thee four equations can be proved by plugging dilation equations into the left hand side (inner products). The decomposition and reconstruction proof is similar to the orthogonal MRA case.

3.5 Filter Design for Wavelets

For two channel filter banks, perfect reconstruction equations are given by

$$\frac{1}{2}(H_1(z)H_2(z)+G_1(z)G_2(z)) = z^{-L}$$

and

$$\frac{1}{2}(H_1(-z)H_2(z)+G_1(-z)G_2(z)) = 0$$

First equation is called the no-distortion equation while second one is no aliasing condition. For filter design we focus mainly on the first equation. $H_1(z)$ and $H_2(z)$ are analysis and synthesis low pass filters while $G_1(z)$ and $G_2(z)$ are high pass filters. For perfect reconstruction, it is sufficient to design low pass filters and obtain high pass filters from them.

Let $P(z)$ be the product filter of the two low pass filters.

$$P(z) = H_1(z)H_2(z)$$

Three steps are needed to design the perfect reconstruction filter bank

1. Design Low Pass Filter $P(z)$.
2. Factorize $P(z)$ to find $H_1(z)$ and $H_2(z)$.
3. Use PR conditions to obtain high pass filters from low pass filters.

Step 3 is trivial once the low pass filters are obtained. Choose $H_2(z) = G_1(-z)$ and $G_2(z) = -H_1(-z)$ for alias cancellation. Plugging this in first PR equation we get

$$\frac{1}{2}(H_1(z)H_2(z)-H_2(-z)H_1(-z)) = z^{-L}$$

$$P(z)-P(-z) = 2z^{-L}$$

It can be seen from the equation above that all odd powers of $P(z)$ will cancel out. If we redefine $P(z) := z^{-L}P(z)$, the equation changes to

$$z^{-L}P(z)-(-z)^{-L}P(-z) = 2z^{-L}$$

or,

$$P(z)+P(-z) = 2$$

This is done to center the equation so that the coefficient of the term z^0 is non-zero. Filter $P(z)$ is known as halfband filter. All of its even terms, with the exception of z^0 are zero. When we add $P(z)$ with $P(-z)$ we are left with a constant 2.[Remember that only odd terms will cancel out with the addition so if there is an even term present it will appear in the output].Furthermore, $P(z)$ is a product of two low pass filters.

In orthogonal case,

$$P(z) = H_1(z)H_1(z^{-1}) = H_1(e^{j\omega})H_1(e^{-j\omega})$$

which can be seen as an autocorrelation function because in time domain, the two filters are time reversed versions of each other. eg., $1 + az^{-1} + bz^{-2} + cz^{-3}$ and $1 + az^1 + bz^2 + cz^3$ have coefficients $[1, a, b, c]$ and $[c, b, a, 1]$.

In the case where the two low pass filters are "different" (one cannot be obtained from other just by reversing the filter order) ,as in biorthogonal filter bank, the product $P(z)$ can be seen as a cross-correlation function. It will still be a low pass filter in both cases.

There are a number of halfband filter design and factorization methods. Daubechies and Meyer's methods are more commonly used.

Daubechies Method: Daubechies defined P as $2(1-y)^p B_p(y)$ where $B_p(y)$ is a truncated Binomial polynomial of degree $(p-1)$ and p coefficients.

$$B_p(y) = (1-y)^{-p} = 1 + py + \frac{p(p+1)}{2}y^2 + ... + \binom{2p-2}{p-1}y^{p-1}$$

The coefficient of y^k is $\binom{p+k-1}{k}$. The factor $(1-y)^p$ has p zeros at $y = 1$. In Low Pass filter case, we want zeros at $z = -1$ or $\omega = \pi$. Let

$$y = \frac{(1-e^{-j\omega})}{2}\frac{(1+e^{-j\omega})}{2}$$

in order to maintain symmetry

$$y = \left(\frac{1-cos\omega}{2}\right)$$

which gives

$$P(\omega) = 2\left(\frac{1+cos\omega}{2}\right)^p \sum_{k=0}^{p-1} \binom{p+k-1}{k} \left(\frac{1-cos\omega}{2}\right)^k$$

In z-domain this can be written as

$$P(z) = 2\left(\frac{1+z}{2}\right)^p \left(\frac{1+z^{-1}}{2}\right)^p \sum_{k=0}^{p-1} \binom{p+k-1}{k} \left(\frac{1-z}{2}\right)^k \left(\frac{1-z^{-1}}{2}\right)^k$$

$P(z)$ can be obtained for different values of p. Once we have P(z), the next step is to factorize it in order to obtain the low pass filters.

Meyer's Method: It consists of integrating $P'(\omega)$ and choosing a value c such that $P(\pi) = 0$.It is given by

$$P(\omega) = 2 - c\int_0^{\omega} (sin(\omega))^{2p-1} d\omega$$

Plots of $P(\omega)$ for values of p=1,2,3

Fig. 3.19 $P(\omega)$ for p=1

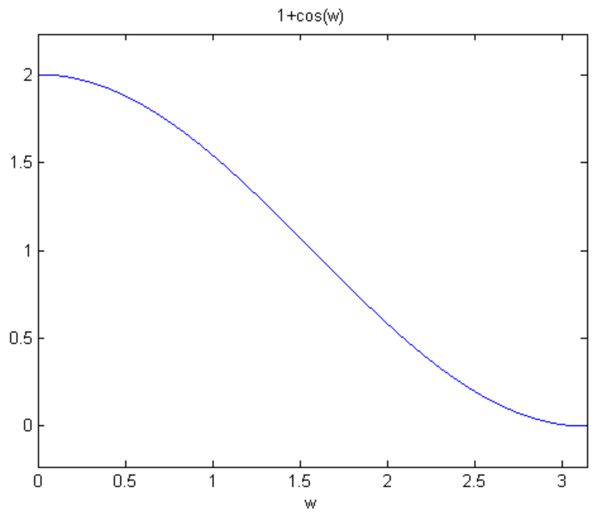


Fig. 3.20 $P(\omega)$ for p=2

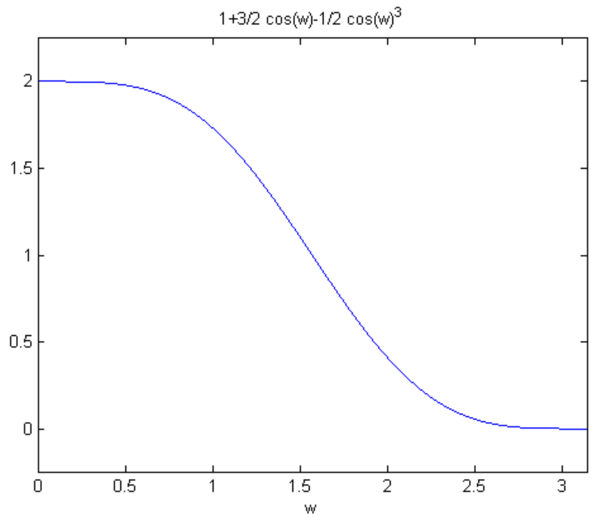
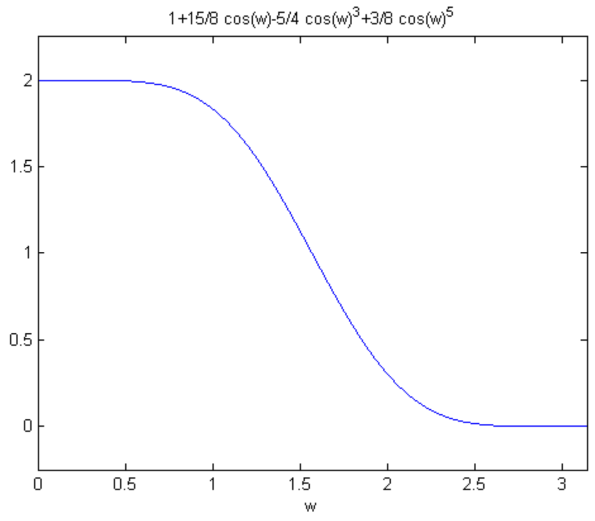


Fig. 3.21 $P(\omega)$ for p=3



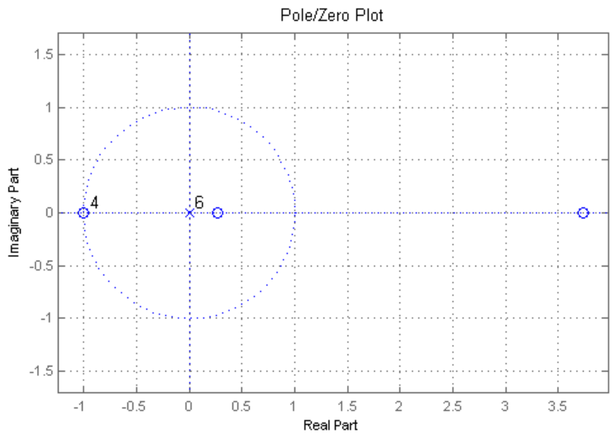
3.5.1 Spectral factorization

Once we have $P(z)$, the next step is to find low pass filters. In the orthogonal case $P(z) = H(z)H(z^{-1})$. As seen above $P(z)$ can be re-written as

$$P(z) = [(1+z)^p(1+z^{-1})^p Q(z)]$$

The value of $Q(z)$ comes from binomial expansion and , looking at these equations, it makes sense to factorize $Q(z)$ as $Q(z) = R(z)R(z^{-1})$. Daubechies suggested that $H(z)$ be chosen such that it is minimum phase which will make $H(z^{-1}$ maximum phase. Therefore, $R(z)$ must be chosen to be causal and with all its zeros inside the unit circle. For example, consider the product filter for p=2. Its pole-zero plot as obtained using Matlab is shown below in figure 11.

Fig. 3.22 Pole zero plot of $P(\omega)$ for p=2



There are four zeros at $z = -1$. For orthogonal filters, we need to assign them equally to $H(z)$ and $H(z^{-1})$.

$$P(z) = (\frac{-1}{16}z^3 + \frac{9}{16}z^1 + 1 + \frac{9}{16}z^{-1} + \frac{9}{16}z^{-3})$$

This can be expressed as

$$P(z) = \frac{1}{16}(1+z)^2(1+z^{-1})^2(-z+4-z^{-1})$$

As can be seen from the equation and pole-zero plot. This has 4 zeros at $z = -1$. The other two zeros are at $2-\sqrt{3}$ and $2+\sqrt{3}$. $H(z)$ is assigned the zero inside the unit circle while $H(z^{-1})$. Calculating coefficients using roots function of matlab gives us four coefficients for analysis low pass filter $h(n) = [0.3415, 0.5915, 0.1585, -0.0915]$.

The synthesis low pass filter is time reversed $h(-n) = [-0.0915, 0.1585, 0.5915, 0.3415]$. The analysis and synthesis high pass filters can be calculated by alternate flipping the low pass coefficients and they are found to be $[-0.0915, -0.1585, 0.5915, -0.3415]$ and $[0.3415, -0.5915, 0.1585, 0.0915]$. The scaling and wavelet functions can be calculated using iteration method. They are shown in figures 12 and 13.

Fig. 3.23 Scaling Function for Db2 wavelets(p=2)

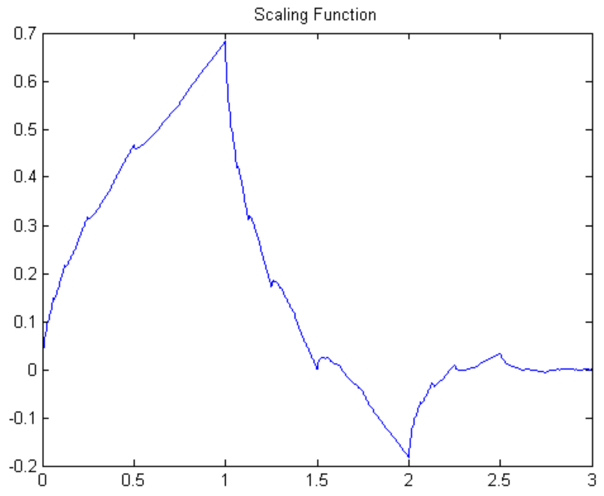
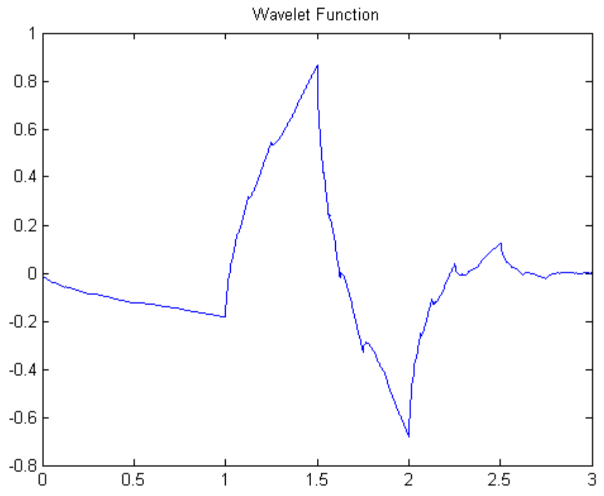


Fig. 3.24 Wavelet Function for Db2 wavelets(p=2)



3.5.2 *Filters for Biorthogonal Wavelets*

Recall that we'll need two filters in this case and one of the requirement is that filters be linear phase. We can use same product filter $P(z)$ in previous example and factor it accordingly to suit our goals. Obviously $P(z)$ can be factored in a number of ways but one of the more commonly used factorization is designing one filter with two zeros at $z = -1$ and the other filter contains other four zeros.

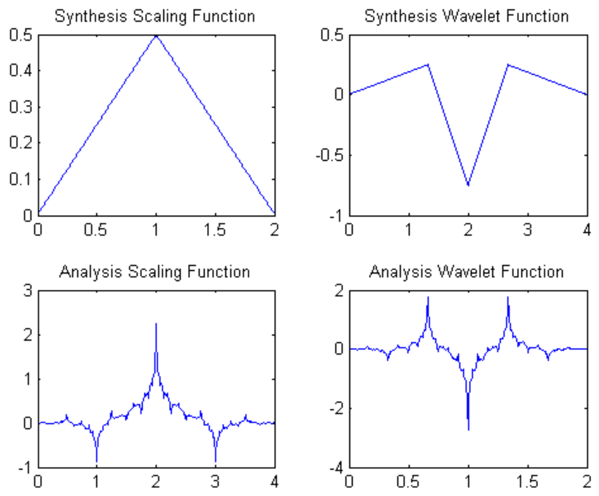
$$H_1(z) = \frac{1}{4}(1+z^{-1})^2$$

and

$$H_2(z) = -\frac{1}{4}(1+z^{-1})^2(2+\sqrt(3)-z^{-1})(2-\sqrt(3)-z^{-1})$$

Looking at it from discrete time-domain angle, the first filter is a "hat function", a symmetric filter with coefficients $[1,2,1]/4$ and the other filter is a convolution between $[1,2,1]/4$ and $[-14-1]/2$. The second filter has the coefficients $[-0.1250, 0.2500, 0.7500, 0.2500, -0.1250]$. It is easy to see that both filters are symmetric, linear phase filters. We can obtain the two corresponding high pass filters using perfect Reconstruction condition. Using Matlab, they are found to be $[0.2500, -0.5000, 0.2500]$ and $[0.1250, 0.2500, -0.7500, 0.2500, 0.1250]$ This biorthogonal filter bank is usually known as $5/3$ filter bank because of the length of th two filters (3 and 5 taps, respectively). The scaling and wavelet functions associated with these filters are calculated by iterating these filters over their respective dilation equations and they are plotted in figure 14.

Fig. 3.25 Wavelet and Scaling Functions for 5/3 Biorthogonal Filters (p=2)



3.5.3 *Lifting and Haar Decomposition*

Haar Decomposition is the most basic form of wavelet decomposition which essentially decomposes a signal into its average(Low Pass) and difference(High Pass).The signal $s_{j+1,k}$ at scale $j+1$ is decomposed into approximated signal $s_{j,k}$ and a detail part $d_{j,k}$ which is stripped away from the original signal.

$$s_{j,k} = \frac{s_{j+1,2k} + s_{j+1,2k+1}}{2}$$

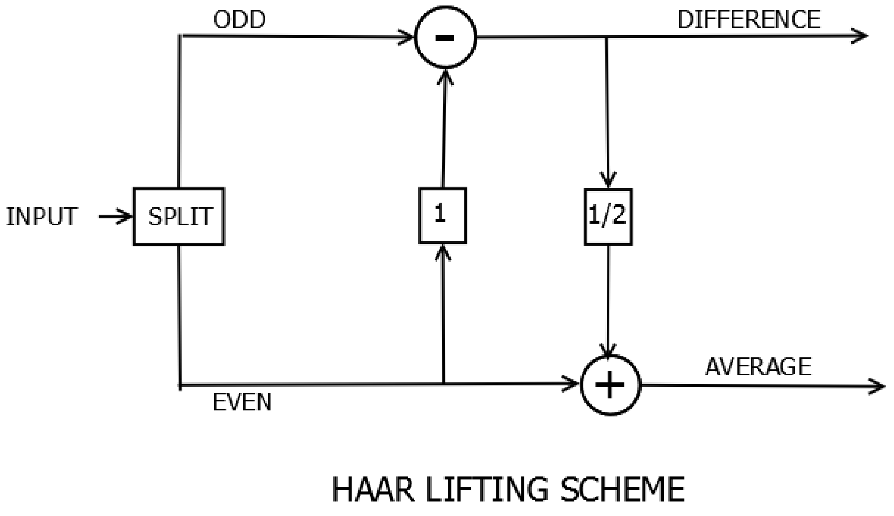
$$d_{j,k} = s_{j+1,2k+1} - s_{j+1,2k}$$

or the average $s_{j,k}$ can be written as

$$s_{j,k} = s_{j+1,2k} + \frac{d_{j,k}}{2}$$

which is the even value of the signal added to half the difference between even and odd values of signal. In other words, if we calculate the difference first, we can calculate the average using only even values of the signal.

Fig. 3.26 Haar Decomposition using Lifting Scheme



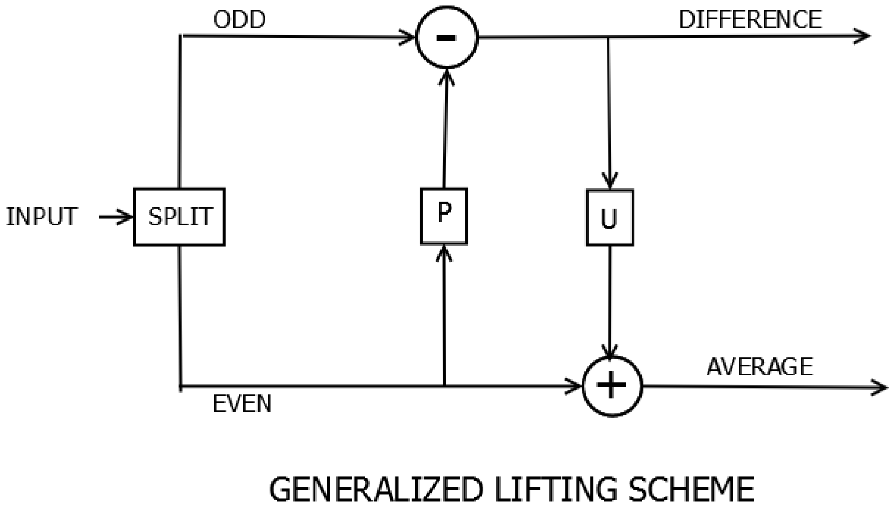
Looking at this scheme from a slightly different angles, we are trying to predict the average using only even terms. This scheme can be generalized by using three steps

1. Split: Signal is split into even and odd components.

2. Predict: We use predict matrices P that work only on one half (even) of the signal and replace odd part with the difference or detail of the signal. In other words, we are trying to predict odd values using even values.

3. Update: The update matrices U update the even part by processing the difference part with U. Update is needed to preserve certain scalar quantities like average.

Fig. 3.27 Generalized Lifting Scheme

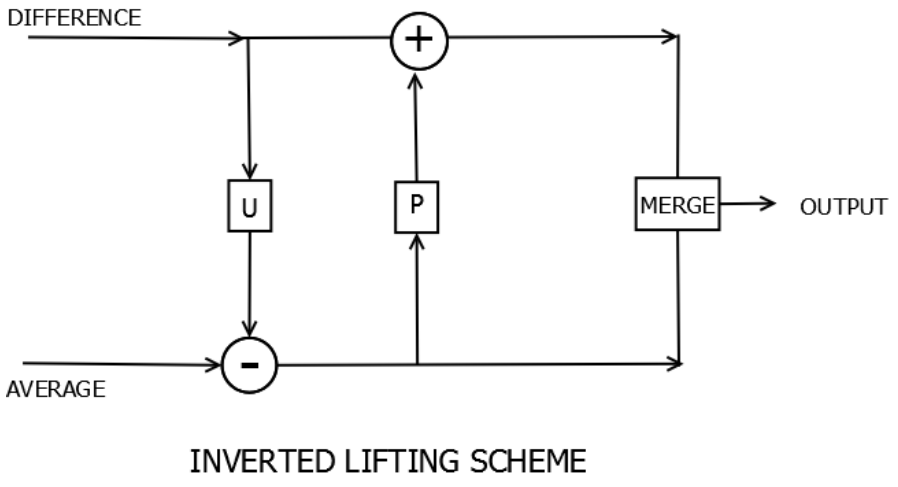


3.5.3.1 Properties of Lifting Scheme

1. In-Place calculations: As can be seen from block diagrams, there is no reason to keep both even and odd components of the signal. We can simply replace odd values with the output after prediction. This idea can be extended when dealing with multi-stage lifting scheme. We can keep replacing inputs of any given branch with outputs and compute next stages as we go.

2. Invertibility: This is, of course, the most important property which makes lifting scheme useful in calculating wavelet transforms. As can be seen from the Haar example, the odd values of the input signal can be obtained by using even values and he difference values. In multistage lifting scheme inversion, we start from the last stage and keep inverting systematically until we recover the input. Inverted Lifting Scheme is shown below.

Fig. 3.28 Generalized Inverted Lifting Scheme



3.5.3.2 Constructing Biorthogonal Wavelets using Lifting Scheme

Consider a Biorthogonal Filter Bank with
Low Pass Analysis Filter $\widetilde{H_j}$
High Pass Analysis Filter $\widetilde{G_j}$
Low Pass Synthesis Filter H_j
High Pass Synthesis Filter G_j
With lifting, the new filters are given by

$$H_j^{new} = H_j$$

$$\widetilde{H_j^{new}} = \widetilde{H_j} + S_j\widetilde{G_j}$$

$$G_j^{new} = G_j - S_j^* H_j$$

$$\widetilde{G_j^{new}} = \widetilde{G_j}$$

where S_j is a lifting operator which can be seen as equivalent to update operator U.

For these new filters to work, they must satisfy perfect reconstruction property.

$$\widetilde{H_j^{new}}(z)H_j^{new}(z) + \widetilde{G_j^{new}}(z)G_j^{new}(z) = 2$$

and

$$\widetilde{H_j^{new}}(-z)H_j^{new}(z) + \widetilde{G_j^{new}}(-z)G_j^{new}(z) = 0$$

In Matrix notation,

$\begin{pmatrix} \widetilde{H_j^{new}} \\ \widetilde{G_j^{new}} \end{pmatrix} = \begin{pmatrix} 1 & S_j \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \widetilde{H_j} \\ \widetilde{G_j} \end{pmatrix}$

$\begin{pmatrix} H_j^{new} \\ G_j^{new} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -S_j^* & 1 \end{pmatrix} \begin{pmatrix} H_j \\ G_j \end{pmatrix}$

Multiplying first equation on both sides with conjugate transpose of the second equation on both sides. The product of two S_j matrices is, therefore

$\begin{pmatrix} 1 & S_j \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & -S_j \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Plugging values of new filters in terms of old filters we get

These two happen to be perfect reconstruction condition for the original filters so lifting a biorthogonal filer bank by S as shown in figure below(see Forward and Inverse Lifted Wavelet Transform) will satisfy perfect reconstruction property.

In time-domain, these four equations can be written as

$$h_j^{new}(n) = h_j(n)$$

$$\widetilde{h_j^{new}}(n) = \widetilde{h}_j + \sum_k s_j(k)\widetilde{g}_j(n-k)$$

$$g_j^{new}(n) = g_j - \sum_k s_j^*(k)h_j(n-k)$$

$$\widetilde{g_j^{new}}(n) = \widetilde{g}_j(n)$$

Tt can be seen from the equations that $\widetilde{g}_j(n)$ and $h_j^{new}(n)$ are unchanged. In order to find new wavelets and new scaling function we'll utilize dilation equations.

Analysis Dilation Equations

$$\widetilde{\phi}(t) = 2\sum_k \widetilde{h}(k)\widetilde{\phi}(2t-k)$$

$$\widetilde{\psi}(t) = 2\sum_k \widetilde{g}(k)\widetilde{\phi}(2t-k)$$

Synthesis Dilation Equations

$$\phi(t) = 2\sum_k h(k)\phi(2t-k)$$

$$\psi(t) = 2\sum_k g(k)\phi(2t-k)$$

Since low pass synthesis filter is unchanged, the new scaling function associated will also be unchanged.

$$\phi^{new}(t) = \phi(t)$$

However, new wavelet function for synthesis side is as follows

$$\psi^{new}(t) = 2\sum g^{new}(n)\phi(2t-n)$$

Plugging value of $g^{new}(n)$ gives us the following

$$\psi^{new}(t) = 2\sum g(n)\phi(2t-n) - \sum_k \sum s^*(k)h(n'-k))\phi(2t-n)$$

or,

$$\psi^{new}(t) = \psi(t) - \sum_k s^*\phi(t-k)$$

Computing similarly we'll get $\widetilde{\phi^{new}(t)}$ and $\widetilde{\psi^{new}(t)}$.

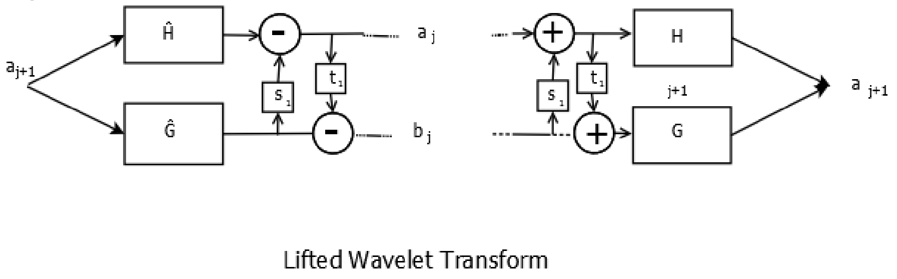
$$\widetilde{\phi^{new}(t)} = 2\sum \widetilde{h}(n)\widetilde{\phi^{new}}(2t-n) + 2\sum s(k)\widetilde{\psi^{new}}(t-k)$$

and

$$\widetilde{\psi^{new}(t)} = 2\sum \widetilde{g}(n)\widetilde{\phi^{new}}(2t-n)$$

The Lifted Wavelet Transform along with its inverse is shown in figure. The transform consists of primal lifting and dual lifting.

Fig. 3.29 Forward and Reverse Lifted Wavelet Transform



S corresponds to update stage(primal lifting) while t corresponds to prediction(dual lifting). Next we express this transform in its polyphase terms. Polyphase of a filter bank consisting of two filters $H(z)$ and $G(z)$ can be written as

$$P(z) = \begin{pmatrix} H_e(z) & G_e(z) \\ H_o(z) & G_o(z) \end{pmatrix}$$

where $H(z) = H_e(z^2) + z^{-1}H_o(z^2)$ with even and odd parts separated and $G(z) = G_e(z^2) + z^{-1}G_o(z^2)$

If new filter is given by $G^{new}(z) = G(z) + S(z^2)H(z)$ then $S(z^2)H(z)$ can be given by polyphase $H_e(z)S(z)$ and $H_o(z)S(z)$. Therefore, the new polyphase matrix is

$$P^{new}(z) = P(z)\begin{pmatrix} 1 & S(z) \\ 0 & 1 \end{pmatrix}$$

For dual lifting, the equation is $H^{new}(z) = H(z) + T(z^2)G(z)$. Proceeding as above, new filter polyphase matrix is given just for dual lifting is

$P^{new}(z) = P(z)\begin{pmatrix} 1 & 0 \\ T(z) & 1 \end{pmatrix}$

Since we are using both primal and dual lifting in the same circuit, the new polyphase on the analysis side will be

$\widetilde{P^{new}}(z) = \widetilde{P(z)}\begin{pmatrix} 1 & S(z) \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ T(z) & 1 \end{pmatrix}$

Next assume that we are using m-level cascade of primal and dual lifting then the new polyphase will be a product of m primal and dual matrices. Wim Sweldens and Ingrid Daubechies proved that using this method and starting with "Lazy wavelet" every finite wavelet transform can be obtained this way. A Lazy wavelet filter bank is one with $H(z) = 1$ and $G(z) = z^{-1}$.

$\widetilde{P^{new}}(z) = \begin{pmatrix} K_1 & 0 \\ 0 & K_2 \end{pmatrix}\prod_i \begin{pmatrix} 1 & S_i(z) \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ T_i(z) & 1 \end{pmatrix}$

3.6 Two Dimensional Wavelet transform

Two dimensional wavelets and filter banks are used extensively in image processing and compression applications. It is easy to extend 1D ideas to 2D. We'll start with dilation equations.

$$\phi(x_1, x_2) = \sum_{n_1}\sum_{n_2} h_0(n_1, n_2)\phi(2x_1 - n_1, 2x_2 - n_2)$$

where h_0 is a 2D low pass filter while ϕ is a 2D scaling function. As is obvious , we are interested in $L^2(R^2)$ spaces and in $L^2(R^2)$ space filter implementations we have two options. Either we can design 2D filters or we can use 2 1D filters to create one 2D filter. Wavelets bases obtained from former are called nonseparable wavelet bases while latter yields separable bases.

Let h be a 1D low pass filter while g be the corresponding high pass filter. The scaling dilation equation can be written as

$$\phi(x_1, x_2) = \sum_{n_1}\sum_{n_2} h(n_1)h(n_2)\phi(2x_1 - n_1, 2x_2 - n_2)$$

We'll have three, not one, wavelet dilation equations and three mother wavelets.

$$\psi_{hg}(x_1, x_2) = \sum_{n_1}\sum_{n_2} h(n_1)g(n_2)\phi(2x_1 - n_1, 2x_2 - n_2)$$

$$\psi_{gh}(x_1, x_2) = \sum_{n_1}\sum_{n_2} g(n_1)h(n_2)\phi(2x_1 - n_1, 2x_2 - n_2)$$

$$\psi_{gg}(x_1, x_2) = \sum_{n_1}\sum_{n_2} g(n_1)g(n_2)\phi(2x_1 - n_1, 2x_2 - n_2)$$

Fig. 3.30 1D Low Pass Filter h

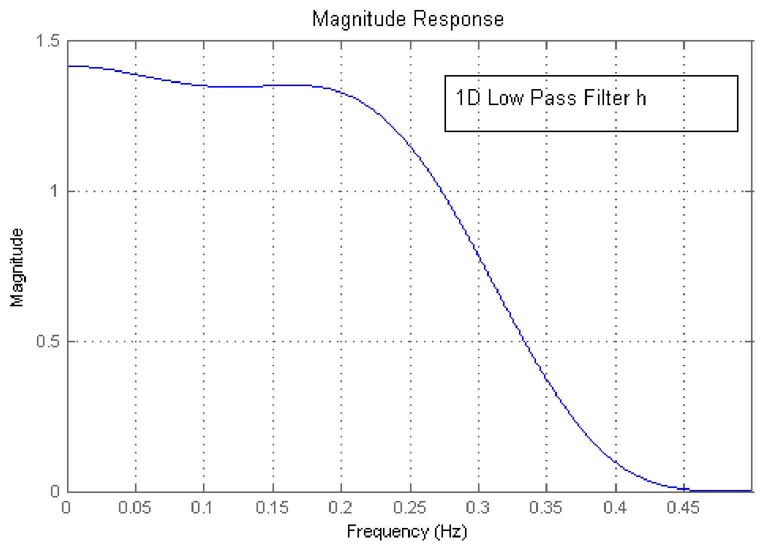


Fig. 3.31 1D High pass filter g

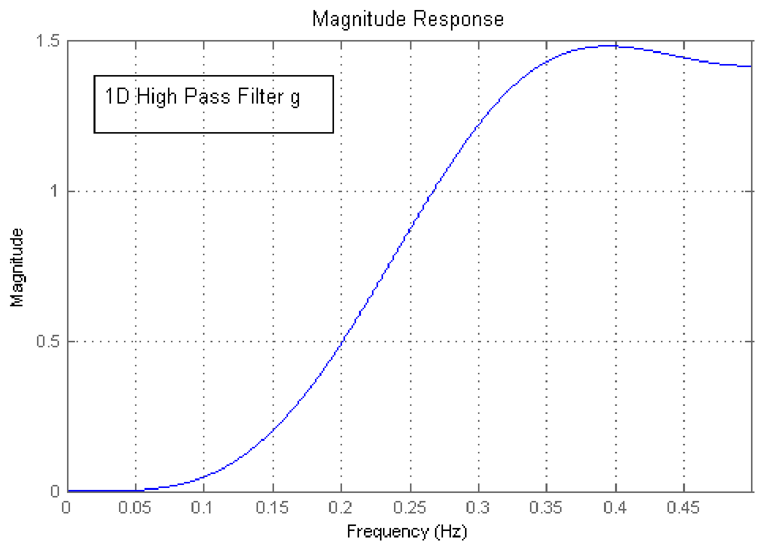
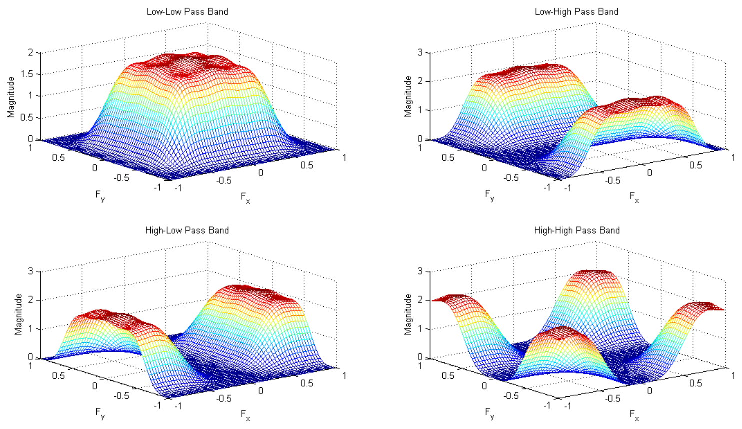


Fig. 3.32 2D Filters obtained from h and g



This idea will be made clearer shortly.

3.6.1 Two-Dimensional Separable Multiresolutions

All 1D ideas can be directly applied to 2D separable Multiresolution case. The spaces are usually represented as V_j^2 and W_j^2 where V_j^2 is a tensor product of two 1D spaces.

$$V_j^2 = V_j \otimes V_j$$

while nesting is similar to 1D case as well

$$V_{j+1}^2 = V_j^2 \oplus W_j^2$$

The two dimensional wavelets and scaling functions can also be represented in terms of their 1D counterparts.Consider, $L^2(R^2)$ space with 1D scaling functions $\phi(x_1)$ and $\phi(x_2)$ corresponding to each dimension.The wavelets are given by $\psi(x_1)$ and $\psi(x_2)$. We are using a one-to-one correspondence with filters h and g here with ϕ being associated with low pass filter and ψ with high pass filter. Corresponding to four different filter configurations seen above, we have one scaling function and three wavelets.

$$\phi(x_1,x_2) = \phi(x_1)\phi(x_2)$$

$$\psi^1(x_1,x_2) = \phi(x_1)\psi(x_2)$$

$$\psi^2(x_1,x_2) = \psi(x_1)\phi(x_2)$$

$$\psi^3(x_1,x_2) = \psi(x_1)\psi(x_2)$$

If ϕ and ψ are orthonormal basis functions in a MRA spanned by V_j and W_j then the corresponding 2D space is also orthonormal. The three wavelets $\psi^1_{j,n}$, $\psi^2_{j,n}$ and $\psi^3_{j,n}$ form an orthonormal basis. Corresponding to these three wavelets we have three wavelet spaces that are given by

$$W_j^2 = (V_j \otimes W_j) \oplus (W_j \otimes V_j) \oplus (W_j \otimes W_j)$$

We can extend orthonormal MRA idea to biorthogonal MRA which is actually a set of two MRAs- one "normal" MRA and other "tilde" MRA. Common notation is to denote analysis filter banks as "tilde" MRA.

3.6.2 2D DWT Algorithm

2D DWT algorithm is developed the same way as the 1D algorithm. Let $S_{j,k}$ and $W^k_{j,k}$ be scaling and wavelet coefficients at scale j for a given signal $f(t)$ where $k = 1,2,3$.We'll be working with separable orthonormal filters so 2D filters can be expressed as a product between low pass filter h and high pass filter g. Proceeding as before, the coefficients at scale j can be obtained from coefficients at scale $j+1$

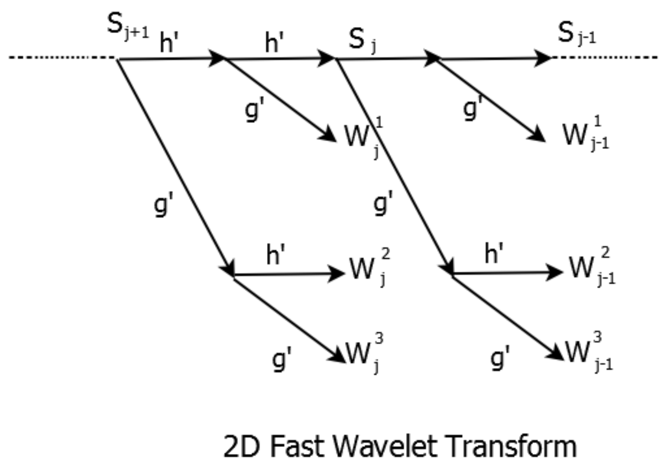
$$S_{j,v} = 2^{\frac{1}{2}} \sum_k hh(l-2v) S_{j+1,l}$$

$$W^1_{j,v} = 2^{\frac{1}{2}} \sum_k hg(l-2v) S_{j+1,l}$$

$$W^2_{j,v} = 2^{\frac{1}{2}} \sum_k gh(l-2v) S_{j+1,l}$$

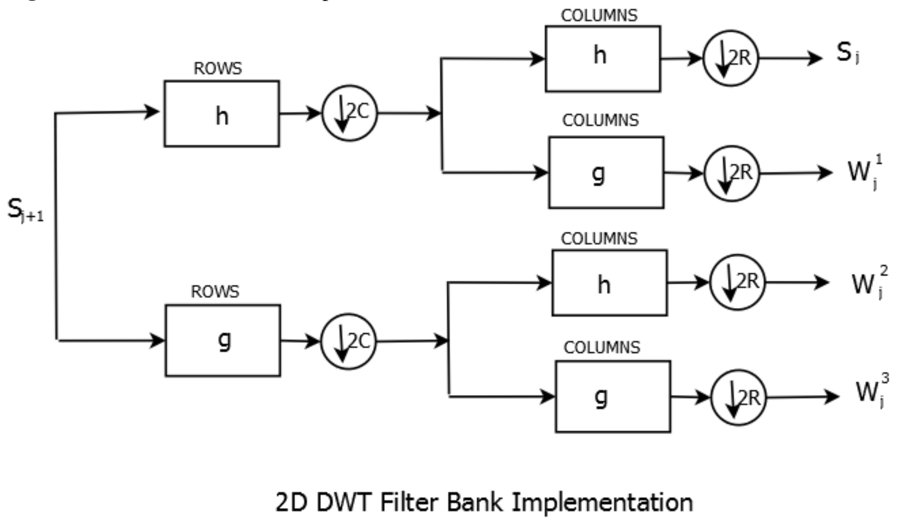
$$W^3_{j,v} = 2^{\frac{1}{2}} \sum_k gg(l-2v) S_{j+1,l}$$

Fig. 3.33 2D Fast Wavelet Transform



Looking at 2D Fast Wavelet transform diagram, 2D filters are developed using two 1D filters in each branch. Just as in 1D case, these filters are time-reversed and decimated by 2. To implement this filter bank, we use two-stage filter banks. In the first stage, rows of two dimensional signal are convolved with h,g filters and then we downsample columns by 2 (eg., we keep only even indexed columns). In the next stage, columns are convolved with the filters h,g and we keep only even indexed rows. In other words, a $N * N$ image is transformed into two $N * (N/2)$ images after first stage and four $(N/2) * (N/2)$ images after the second stage.

Fig. 3.34 2D DWT Filter Bank Implementation



Following is a 1-level and 2-level decomposition of "Lena" image using the 2D filter bank.

Fig. 3.35 1 level Decomposition of Lena Image

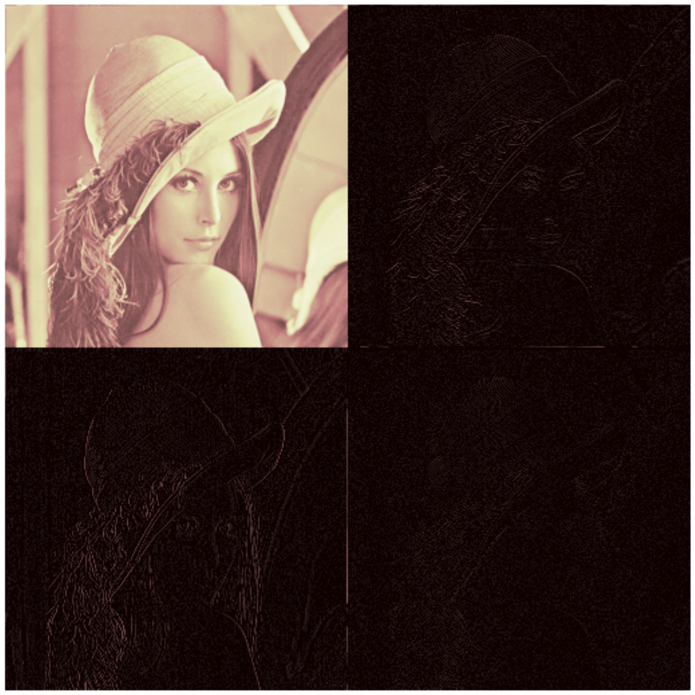
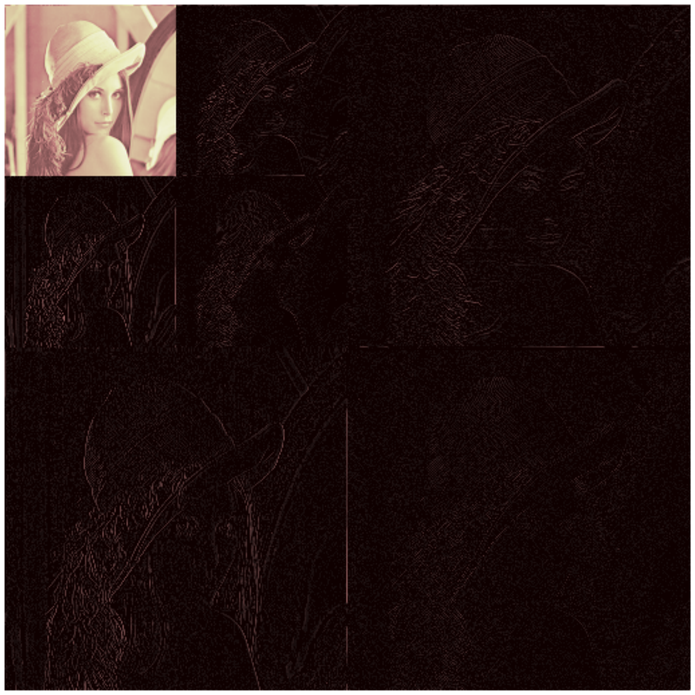


Fig. 3.36 2 level Decomposition of Lena Image



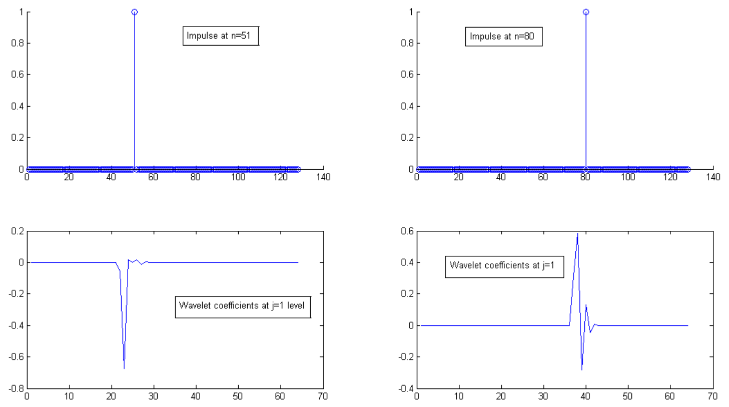
To understand the decomposition process, we observe that the decomposition consists of 4 equal sized blocks in 1-level decomposition.Starting from top left and moving clockwise, they correspond to low-low band, low-high band, high-high band and high-low band. Low-low band gives us low pass filtered image as can be seen in the top left block. Low-high band consists of low horizontal frequencies and high vertical frequencies. The wavelet associated with high vertical frequency is orthogonal to all vertical frequency components so they are cancelled out and we see horizontal edges dominate. The high-low band has high horizontal frequencies and the wavelet associated with this band is orthogonal to horizontal frequency components of the image and , therefore, vertical edges are accentuated. The high-high band accentuates corners as it isn't directional in either the vertical or horizontal directions.

3.7 Complex Wavelet Transform

3.7.1 Motivation for Complex Wavelet Transform

Discrete Wavelet Transform has many advantages over Fourier Transform with main advantage that it can do localized analysis but there are other areas where DWT lags behind Fourier Transform. Shift Variance is one of those areas. Fourier Transform is shift invariant which is a desirable property but the use of downsampling makes DWT prone to shift variance. It can be seen in the figure below. An impulse response at $n = 51$ results in a difference response than the same shift at $n = 80$ at level one of decomposition with a Daubechies8 wavelet DWT filter bank. Not only the two wavelet coefficients are different, the second set of coefficients contain nearly 20% more energy than the first signal.Keep in mind, that DWT can be made nearly shift invariant by removing downsamplers but it makes DWT "significantly" redundant.

Fig. 3.37 Shift Variance Demo



Directionality is another issue with regular DWT. As seen with 2D DWT, it is only good in horizontal and vertical direction. While DWT can isolate diagonals with the High-High band it cannot distinguish between signal elements aligned at 45°and 135°.

Some other issues with real-valued DWT are absence of any phase information and aliasing that may be present because of quantization errors etc. which may not allow perfect reconstruction of signal.

3.7.2 Hilbert Transform and Analytic Signal

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} d\omega$$

Fourier Transform consists of two 90°out of phase sine and cosine terms. It can represented as

$$e^{-j\omega t} = cos(\omega t) + jsin(\omega t)$$

In order to develop complex wavelets that are analogous to Fourier Transform, it is important to take a look at Analytic signals and Hilbert Transform.

An Analytic signal $x_a(t)$ is defined as

$$x_a(t) = g(t) + j\widehat{g}(t)$$

where $g(t)$ is a real valued signal while $\widehat{g}(t)$ is the 90°out of phase version of $g(t)$. We use Hilbert transform to generate $\widehat{g}(t)$. Hilbert transform of a signal $g(t)$ is given by

$$\widehat{g}(t) = H(g(t)) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{g(t)}{t-\tau} d\tau$$

Taking Fourier Transform, we get

$$\widehat{G}(\omega) = -jSgn(\omega)G(\omega)$$

where

$$sgn(\omega) = 1, \omega > 0$$

$$sgn(\omega) = -1, \omega < 0$$

Since $x_a(t)$ is a complex signal, its magnitude and angle are given by

$$\|x_a(t)\| = \sqrt{g(t)^2 + \widehat{g}(t)^2}$$

$$\angle x_a(t) = tan^{-1} \left[\frac{\widehat{g}(t)}{g(t)} \right]$$

Another very important property of analytic signal is that it only exists for positive frequencies.

$$X_a(\omega) = (1 + Sgn(\omega))G(\omega)$$

$$X_a(\omega) = 2G(\omega), \omega > 0$$

$$X_a(\omega) = G(\omega), \omega = 0$$

$$X_a(\omega) = 0, \omega < 0$$

This helps reduce bandwidth use by the signal and reduces aliasing.

3.7.3 Complex Valued Wavelet Coefficients

Let $\psi_A(t)$ be the analytic wavelet and let $\psi_R(t)$ and $\psi_I(t)$ be its real and imaginary terms where $\psi_I(t) = H(\psi_R(t))$.

$$\psi_A(t) = \psi_R(t) + j\psi_I(t)$$

The wavelet coefficients at level j can be given by

$$W_a(j,n) = W_r(j,n) + jW_i(j,n)$$

The magnitude and angles are, therefore,

$$\|W_a(j,n)\| = \sqrt{W_r(j,n)^2 + W_i(j,n)^2}$$

$$\angle W_a(j,n) = tan^{-1}\left[\frac{W_i(j,n)}{W_r(j,n)}\right]$$

As can be seen, the wavelet coefficients(as well as scaling coefficients) are complex but, like Fourier transform, they can be used to analyze complex as well as real signals. Complex wavelets can be implemented with filter banks but ,in this case, the implementation will involve two filter banks , with one corresponding to the real wavelet while the other corresponding to complex. Compared to decimated DWT this Dual-Tree Complex wavelet Transform is redundant.

3.7.4 Dual-Tree Complex Wavelet Transform

As mentioned,the idea is to have two sets of analysis and synthesis filter banks in quadrature with each other.

Fig. 3.38 Complex DWT Analysis Bank

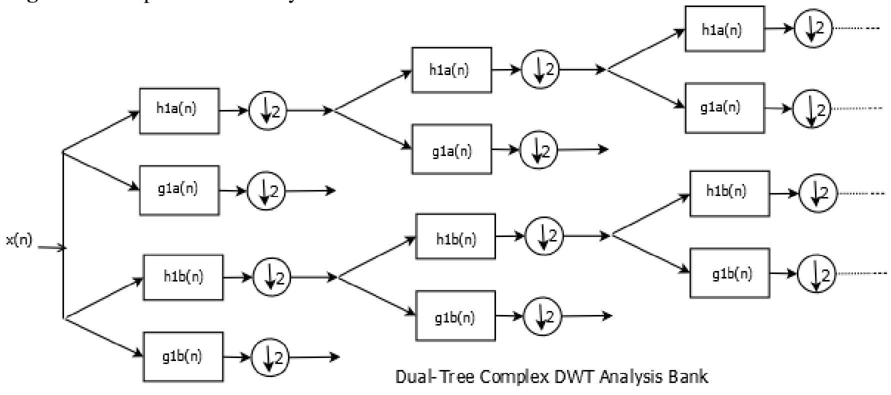
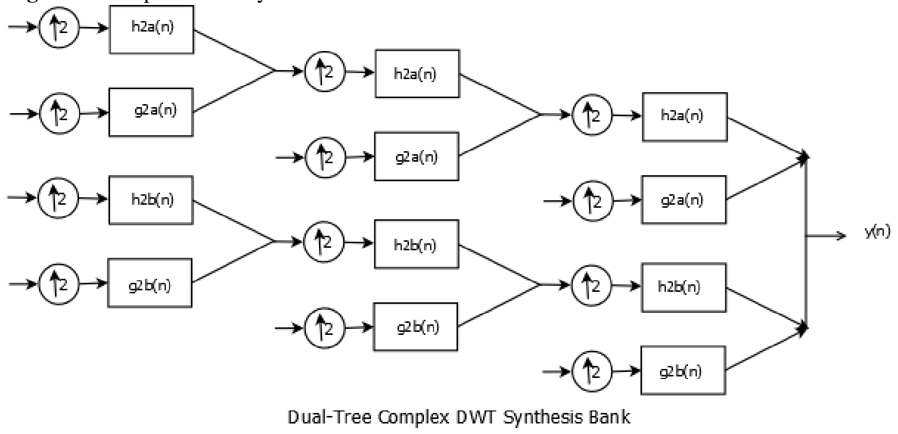


Fig. 3.39 Complex DWT Synthesis Bank



On the analysis side, filters h_{1a},h_{1b},g_{1a} and g_{1b} are all real filters but they are designed so that h_{1a} and g_{1a} are in quadrature. Same is true for high pass filters h_{1b} and g_{1b}. Let $\psi_a(t)$ and $\psi_b(t)$ be the wavelets associated with the two branches then they need to be designed so that

$$\psi_b(t) = H[\psi_a(t)]$$

This is in addition to the perfect reconstruction conditions the filter banks have to fulfill. The additional Hilbert Transform means that normal DWT implementations (eg., Daubechies's construction) won't work and filters need to be designed from scratch.

3.7.5 Hilbert Transform condition

Using dilation equations, the scaling function and wavelet for top branch (real) can be written as

$$\phi_a(t) = 2\sum_n h_{1a}(n)\phi(2t-n)$$

$$\psi_a(t) = 2\sum_n g_{1a}(n)\phi(2t-n)$$

Assuming that these two filters are orthonormal, the high pass filter can be expressed as alternating flipped version of low pass filter.

$$g_{1a}(n) = (-1)^n h_{1a}(d-n)$$

Furthermore, Selesnick showed that if the two low pass filters in the analysis filter bank are half sample shifted version of other, the wavelet functions obtained by iterating the related high pass filters will be Hilbert transform pair. In other words,

$$h_{1b}(n) \approx h_{1a}(n-0.5) \Rightarrow \psi_b(t) \approx H[\psi_a(t)]$$

In Fourier Domain, it translates to a difference of 0.5ω while the magnitudes of two filters are equal. A true half-sample shifted system isn't practically realizable so $\psi_b(t)$ and $\psi_a(t)$ are only approximately realizable.

3.7.6 Filter Design Methods

1. Linear Phase Biorthogonal Filters: Low pass filter $h_{1a}(n)$ is chosen to be even length N symmetric filter while the other low pass filter(complex branch) $h_{1b}(n)$is chosen to be odd length $N-1$ symmetric filter. It is seen that the phase difference between the two filters is $-0.5N\omega-(-0.5(N-1)\omega) = -0.5\omega$ which satisfies the half sample-shifted phase condition but magnitude isn't necessarily equal. We can however design filters so that the magnitudes of two low pass filters are roughly equal.

2. Q-Shift Design: Using quarter-shift method, the low pass filters are related as following

$$h_{1b}(n) = h_{1a}(N-1-n)$$

where $h_{1a}(n)$ filter length N is even. As can be see that this low pass filter configuration has both filters having same magnitude but they don't automatically satisfy the half sample-shifted phase condition. Fourier domain frequency response of $H_{1b}(\omega)$ can be written as

$$H_{1b}(\omega) = H_{1a}^{*}(\omega)e^{-j(N-1)\omega}$$

which corresponds to following phase response

$$\angle H_{1b}(\omega) = -\angle H_{1a}(\omega) - (N-1)\omega$$

Since the two filters are supposed to satisfy the half sample-shifted condition

$$\angle H_{1b}(\omega) \approx \angle H_{1a}(\omega) - 0.5\omega$$

Solving for $H_{1a}(\omega)$, we get

$$\angle H_{1a}(\omega) \approx -0.5(N-1)\omega + 0.25\omega$$

Two observations:
i. Low Pass Filter $H_{1a}(\omega)$ is approximately symmetric.
ii. There is a quarter-shift element(It is shifted by 0.25 from the natural symmetric point) present that makes the equation not fully symmetric.

The analytic wavelets in this case have real and imaginary values that are time reversed version of each other.Since the filters are even and almost symmetric, orthonormal solutions are possible unlike in the first case.

3. Selesnick's Method: In his 2002 paper, Selesnick suggested following design for the two low pass filters.

$$H_{1a}(z) = F(z)D(z)$$

$$H_{1b}(z) = F(z)z^{-L}D(z^{-1})$$

where $D(z)$ is chosen such that the half sample shifted condition is satisfied.

$$H_{1b}(z) = H_{1a}(z)\frac{z^{-L}D(z^{-1})}{D(z)}$$

Observe that

$$A(z) = \frac{z^{-L}D(z^{-1})}{D(z)}$$

is an all pass filter which makes the magnitudes of $H_{1a}(z)$ and $H_{1b}(z)$ equal. $A(z)$ needs to be designed so that the two low pass filters are half sample shifted. For more on these filters and their implementations, see the references.

References

1. S Mallat (1999) A Wavelet Tour of Signal Processing. Academic Press
2. G Strang, T Nguyen(1996) Wavelets and Filter Banks. Wellesley Cambridge Press
3. M Vetterli, J Kovacevic(1995) Wavelets and Subband Coding. Prentice Hall Signal Processing Series
4. I Daubechies (1992) Ten Lectures on Wavelets. SIAM: Society for Industrial and Applied Mathematics

5. M Weeks (2006) Digital Signal Processing using Matlab and Wavelets. Infinity Science Press
6. V Goyal, M Vetterli, J Kovacevic (2010) Fourier and Wavelet Signal Processing. Manuscript Draft
7. M Marcellin, D Taubman (2001) JPEG2000: Image Compression Fundamentals, Standards and Practice. Springer

Chapter 4
Wavelet Applications

Abstract To be completed

4.1 Wavelet Applications in Signal processing

4.1.1 Sharp Transition/ Discontinuities Detection

Wavelets are short duration mathematical functions that can be dilated and translated along a given signal and thus have the capability to analyze a signal at different scales. This makes them ideally suited for detecting short duration rapid variations in the signal in the form of discontinuities and singularities.

Discontinuities can be detected using either continuous or discrete wavelet transforms. Example using continuous and discrete transforms utilizing db2 wavelet follows.

Fig. 4.1 N=256 Piecewise Regular Signal

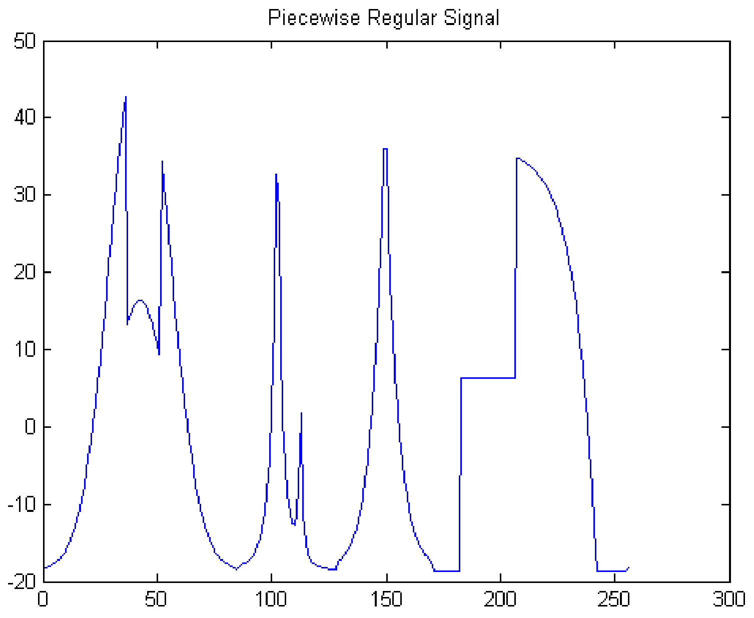
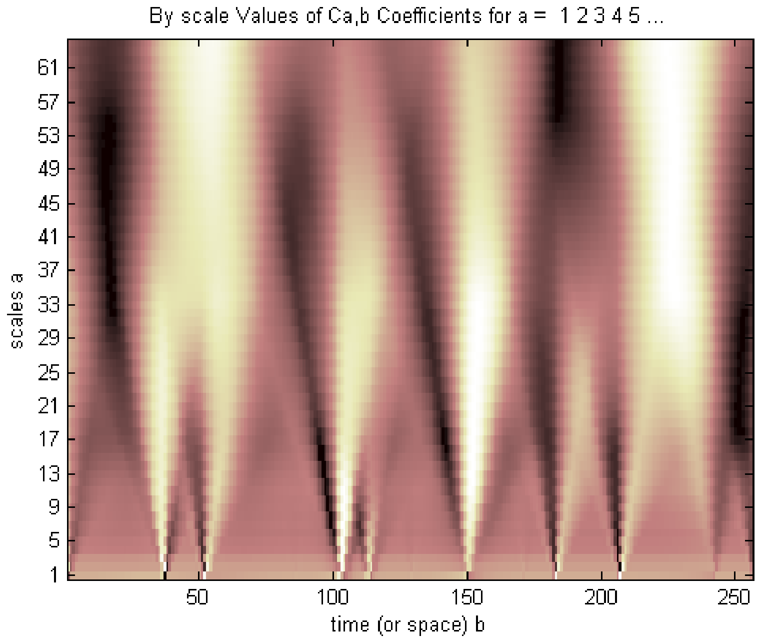


Fig. 4.2 Continuous Wavelet Transform of signal using Db2 wavelet



First, let us consider continuous transform. In this example 64 scale levels are utilized. Since scales are inversely proportional to frequency, the scales at the lowest end of the figure correspond to highest frequencies and vice-versa.The lighter colors correspond to higher coefficients. We see high coefficients value near sharp transitions and the effect is much more pronounced at lower scales (higher frequencies) as these high frequencies match with sharp transitions. The inner product between scaled/dilated wavelet and a portion of signal yields the largest coefficients where the signal has similar spectral components as the wavelet. Coefficients are calculated as following

$$Wf = < f, \psi_{k,s} > = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{s}} \psi^{*}(\frac{t-k}{s})\, dt$$

The mother wavelet (Db2) in this case is scaled and for that scale value it is translated along the signal length and coefficients are calculated at every point.The scale value is changed and the process is repeated. The practical implementation involves using discrete values of translation steps and scaling values but large enough step values approximates CWT well.

A 3*D* display of coefficients distribution along scales and signal length is shown in the figure below. It can be seen that at higher frequencies CWT does a very good job of isolating sharp transitions.

Fig. 4.3 Continuous Wavelet Transform of signal using Db2 wavelet(3D View)

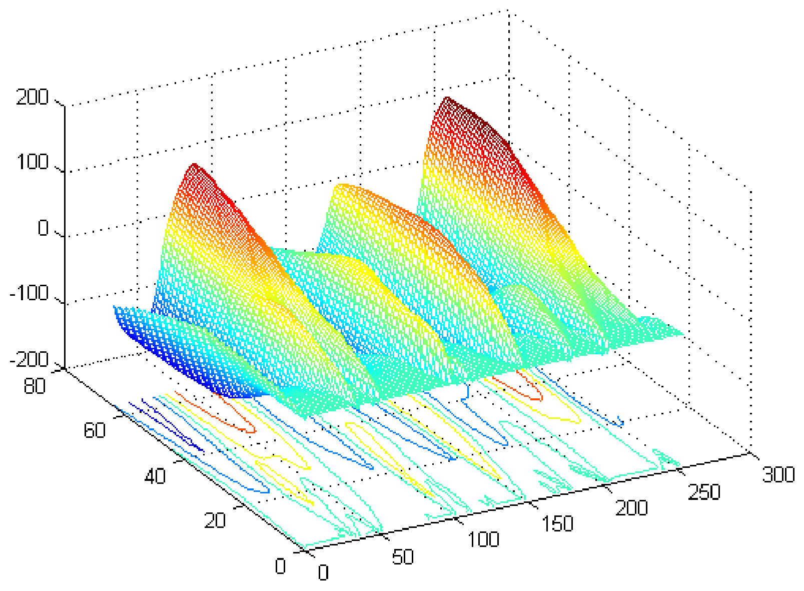


Fig. 4.4 Undecimated Filter Bank

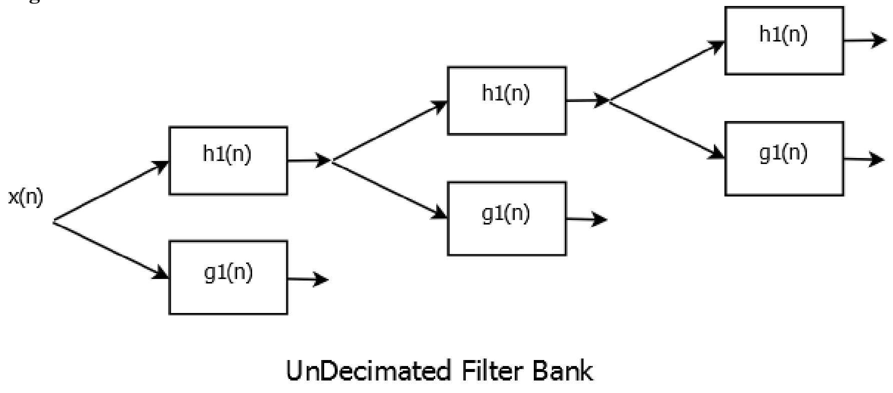
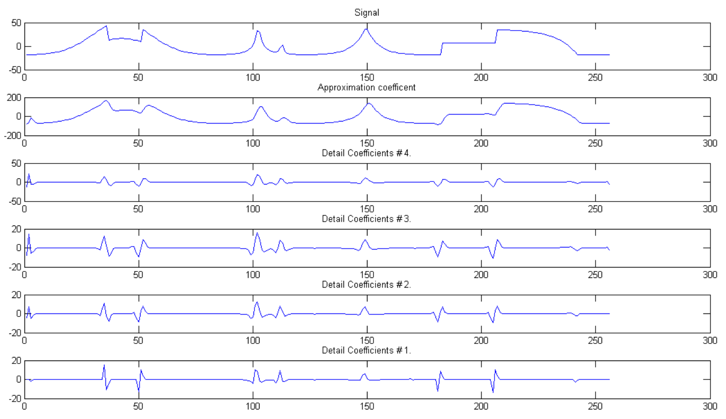


Fig. 4.5 4-level Discrete Wavelet Transform of signal using Db2 wavelet



Discrete Wavelet Transform is computed a little differently considering we use filter banks and dyadic scale values but it still does a good job of isolating sharp transitions. We usually use undecimated filter banks in determining presence and location of discontinuities/transitions as coefficients storage for signal reconstruction is not a crucial issue. We get detail coefficients after convolving with high pass filters and approximation coefficients after filtering with low pass filters. If we are resolving the signal to J=4 scales, we have one approximation set of coefficients at 4th scale while four sets of detail coefficients. In the DWT display of signal, approximation and detail coefficients we move from coarser scales down to finer scales (higher frequencies). As can be seen from the figures, sharp transitions are isolated best at the highest frequencies.

4.1.2 Signal Denoising

DWT is implemented by iterating along the low pass channel of designed filter banks and we take out high frequency components at every stage. This kind of set-up is specially useful when dealing with noise as it usually consists of higher frequency components. In the following figures, an example of denoising is shown. The signal is a doppler with frequencies decreasing in time. DWT filters out higher frequencies from the signal in every successive stage. After 4 stages of filtering we have an approximation of the original signal. It is seen that DWT does a good job with lower frequencies while higher frequencies of the signal (at the beginning of the signal) get chopped down along with the noise. DWT performs substantially better with smoother noise-contaminated signals.

Fig. 4.6 Original Doppler Signal

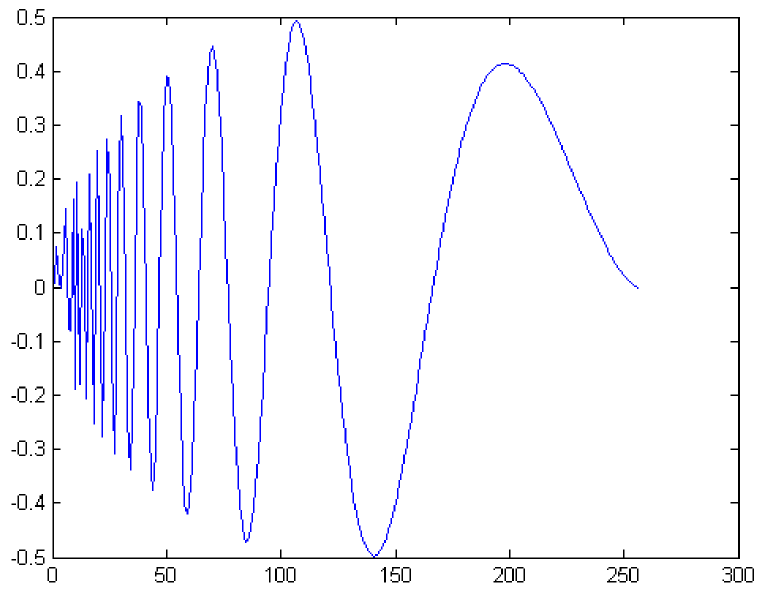


Fig. 4.7 Doppler signal with AWGN

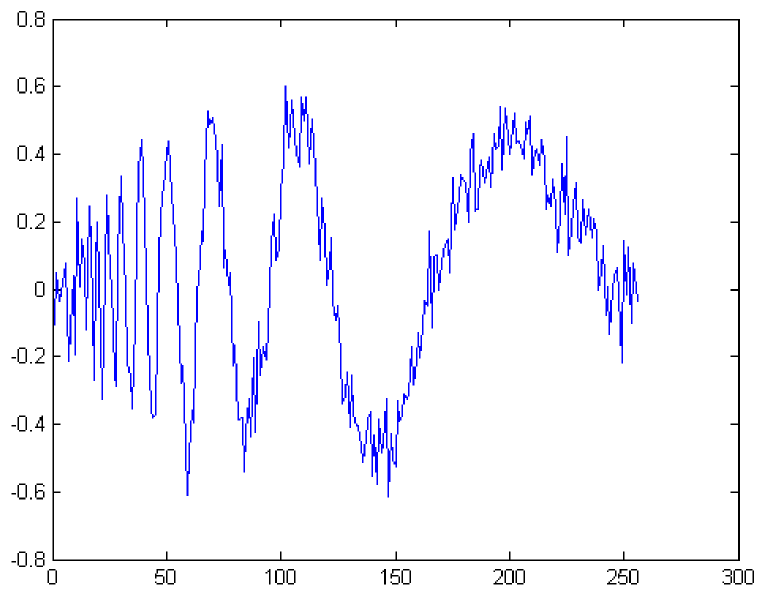


Fig. 4.8 4-level Discrete Wavelet Transform of Noisy signal using Db2 wavelet

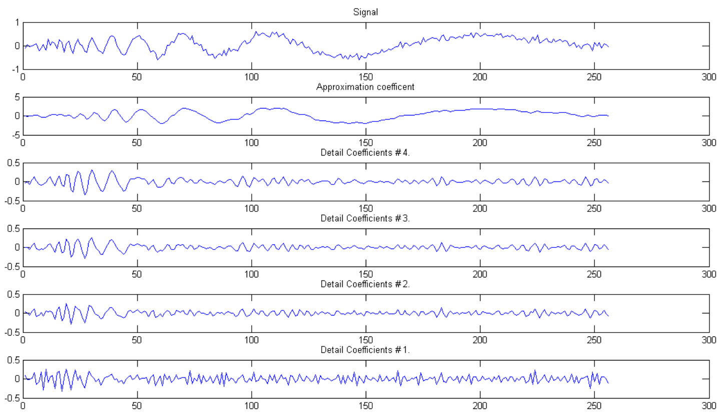
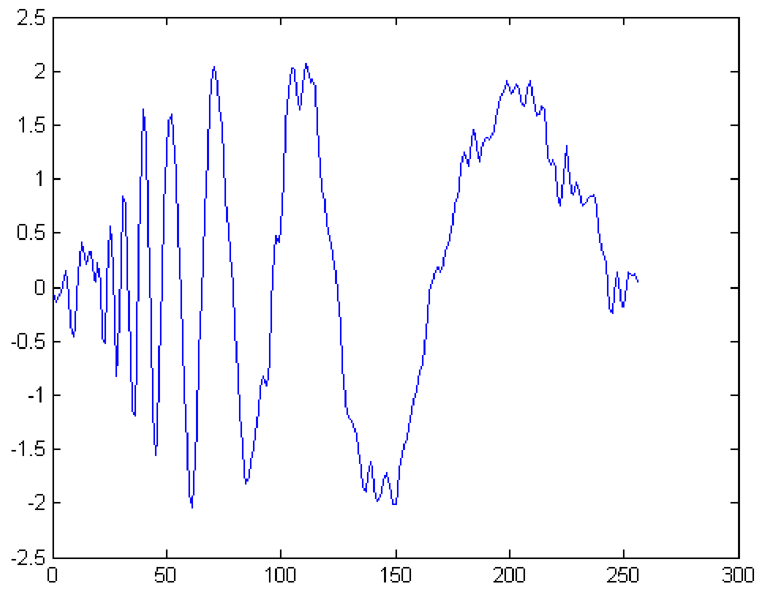


Fig. 4.9 Denoised Signal



4.2 Image Processing

4.2.1 Image Denoising

$2D$ Discrete Wavelet works similarly to $1D$ transform and $1D$ denoising principles can be used to denoise $2D$ signals.$2D$ decomposition transforms an image into 4 subimages, each a quarter size of the original, at each level.Of these *subimages*, three are detail images and one is the approximation image. Denoising methods focus on these three detail images as they are the ones that contain high frequencies in at least one direction. A generic denoising algorithm is given as

1. Calculate DWT of a noisy image.

2. Use thresholding in all the detail images to get rid of high spatial-frequency components.

3. Take Inverse DWT of thresholded signal.

There are different thresholding algorithms involving different trade-offs as a strict threshold results in loss of data while a more lax threshold leaves noise in the image.

There are a number of algorithms(see references) that are used to denoise images. One of those method is Visushrink , which isn't a really efficient method but is good to illustrate the denoising process as it uses global thresholding. The threshold is chosen to be $T = \sigma\sqrt{2\log M}$ where M is the number of pixels in the detail subimages.Hard thresholding is pretty straightforward as all coefficients below the chosen threshold are set to zero and results in a relatively poor performance as too many coefficients are set to zero.

Fig. 4.10 Noisy Lena Image

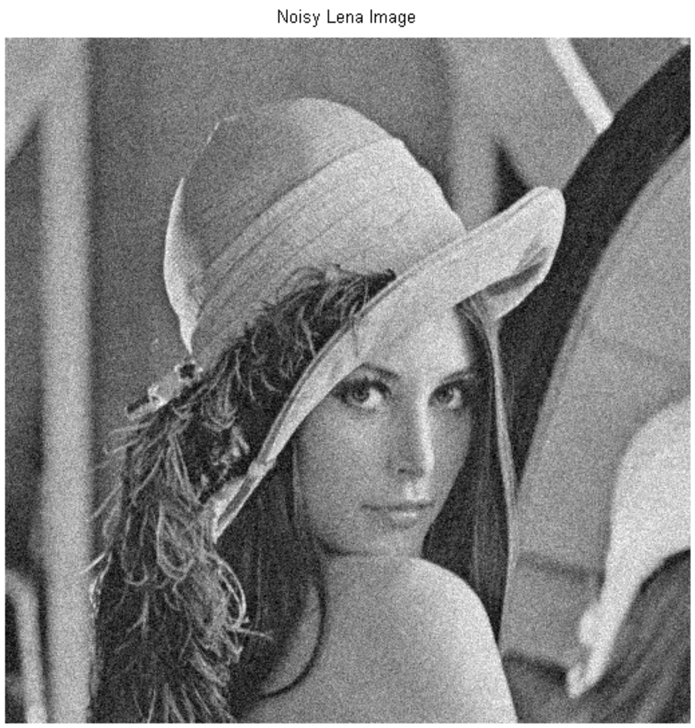
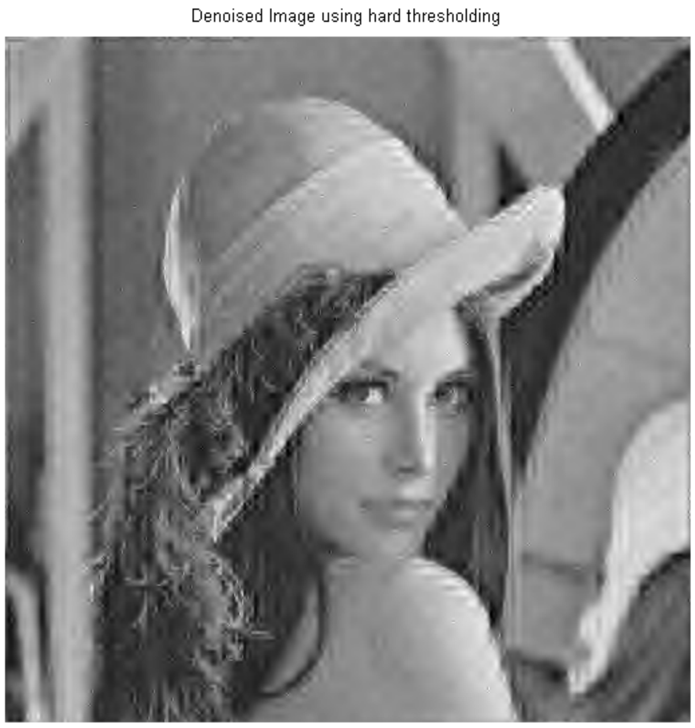


Fig. 4.11 Denoised Image using hard thresholding



Soft thresholding function is given by $f_s(y,T) = sign(y)max(|y| - T, 0)$.

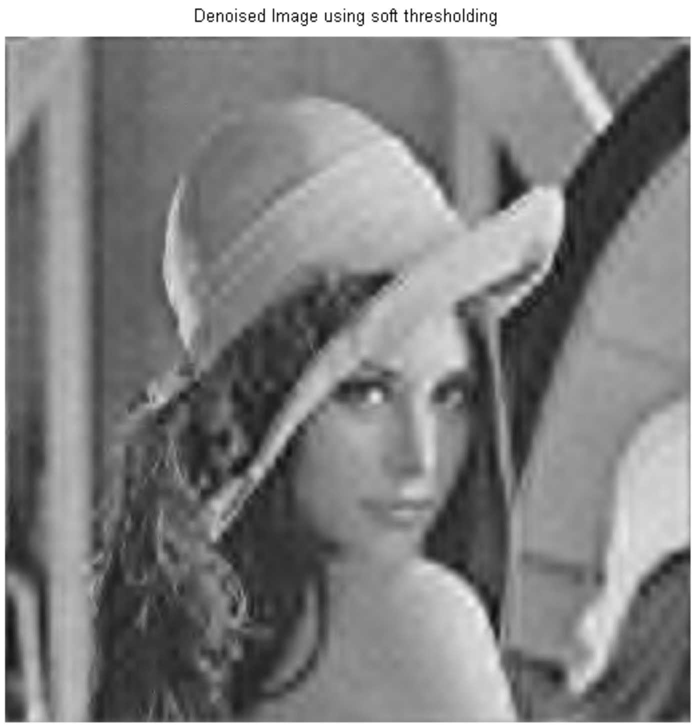


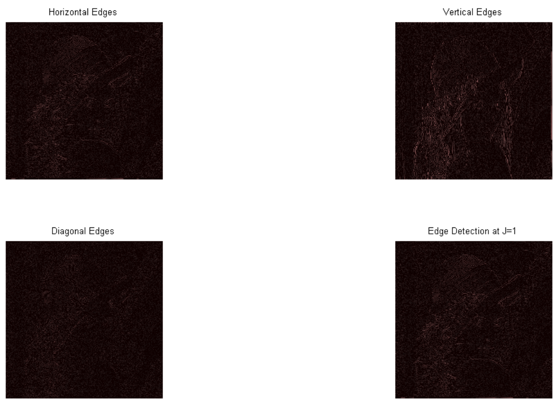
Fig. 4.12 Denoised Image using soft thresholding

Soft thresholding gives better results. For more and better algorithms, including those utilizing local thresholds, on image denoising see references.

4.2.2 Wavelet Edge Detection

Wavelets perform well as singularity/discontinuity detector in one dimension so it makes sense to extend the same ideas to two dimensions and edge detection. We know that one level of separable wavelet decomposition yields four subimages and three of these four subimages correspond to horizontal , vertical and diagonal edges. For a straightforward wavelet edge detector implementation, all three detail subimages can be linearly combined to give an edge detector at a given scale as shown in the figure below. The edge detector is implemented using “Db4” orthogonal wavelets and it isn’t thresholded. Thresholding can be applied to yield better results but it is obvious from the figure that this is not an optimal solution. It is too noisy and some true edges are missed while false edges are being detected. One of the big issue of using high frequency response is that it contains noise and most edge detectors used in computer vision and image processing literature use smoothing gaussian filters to get rid of noise. It has been suggested that applying Gaussian pre-processing to standard wavelet transforms will yield better results. A better approach is to use wavelets that are derived from these isotropic smoothing functions like Gaussians and splines.

Fig. 4.13 Edge Detection at scale J=1



4.2.2.1 Mallat's Multiscale Edge Detector

Let $\theta(x)$ be a smoothing function,ie. $\int_{-\infty}^{\infty}\theta(x)dx = 1$, then the wavelet $\psi(x)$ can be defined as

$$\psi(x) = \frac{d\theta}{dx}$$

In his 1992 paper, Mallat designed wavelets and filters using a smoothing function with Fourier Transform

$$\theta(\omega) = \left[\frac{\sin(\omega/4)}{\omega/4}\right]^{(2n+2)}$$

The scaling and wavelet functions are given by

$$\phi(x) = \left[\frac{\sin(\omega/2)}{\omega/2}\right]^{(2n+1)}$$

$$\psi(x) = i\omega \left[\frac{\sin(\omega/4)}{\omega/4}\right]^{(2n+2)}$$

Filters are designed for $n = 1$ in the example below. For 2d implementation, we use two smoothing functions $\theta_1(x)$ and $\theta_2(x)$ and we assume that they are approximately equal. The two wavelets are then given by

$$\psi_1(x,y) = \frac{d\theta_1(x,y)}{dx}$$

$$\psi_2(x,y) = \frac{d\theta_2(x,y)}{dy}$$

They can be further defined by as separable computations of one dimensional functions.

$$\psi_1(x,y) = 2\psi(x)\phi(2y)$$

$$\psi_2(x,y) = 2\psi(y)\phi(2x)$$

Each signal $s^d_{2^{j+1}}(f)$ is decomposed into three signals - a low pass smoothed signal $s^d_{2^j}(f)$ and two high pass components that correspond to horizontal and vertical edges if the input signal is an image, namely $w^{1,d}_{2^j}(f)$ and $w^{2,d}_{2^j}(f)$. The modulus M of the signal is computed as

$$M_{2^j}(f) = \sqrt{|w^{1,d}_{2^j}(f)|^2 + |w^{2,d}_{2^j}(f)|^2}$$

at each point (x,y). An example of this Multiscale edge detector is shown below. The treatment of edge detector on this page is incomplete as of now so please see references for proofs and methods to compute Wavelet Transform Modulus Maxima.

Fig. 4.14 Lena Image at scales J=1,2,3,4

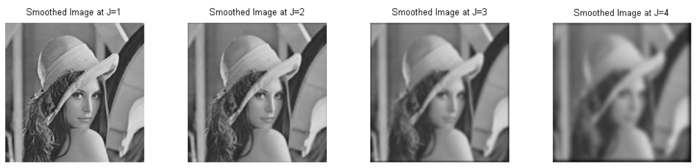


Fig. 4.15 Horizontal Edge Detection at scales J=1,2,3,4

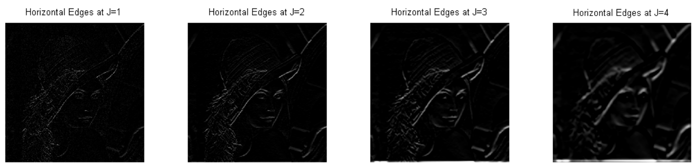


Fig. 4.16 Vertical Edge Detection at scales J=1,2,3,4

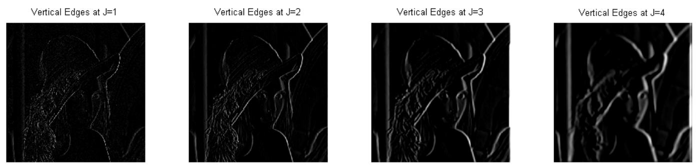
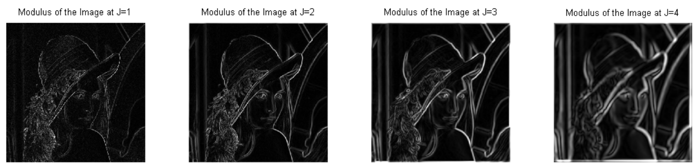


Fig. 4.17 Image Modulus at scales J=1,2,3,4


Modulus of the Image at J=1
Modulus of the Image at J=2
Modulus of the Image at J=3
Modulus of the Image at J=4

4.3 Wavelet Applications in Communications

4.3.1 OFDM Review

OFDM or Orthogonal Frequency Division Multiplexing is a data transmission scheme that consists of transmitting a high rate data stream by using lower rate data streams modulated over orthogonal subcarriers. An OFDM transmission system consists of a constellation encoder followed by a serial-to-parallel converter and a FFT encoder which modulates the signal symbols over an orthogonal bank of subcarriers. The stream is then converted from parallel to serial and a redundant guard band is added to prevent Inter Symbol Interference caused by multipath propagation delays in wireless and other channels.Block diagrams of OFDM Transmitters and Receivers are shown in figures below.

Fig. 4.18 OFDM Transmitter

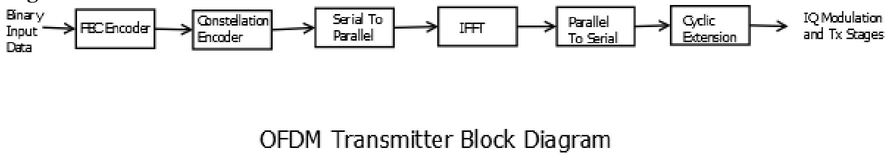
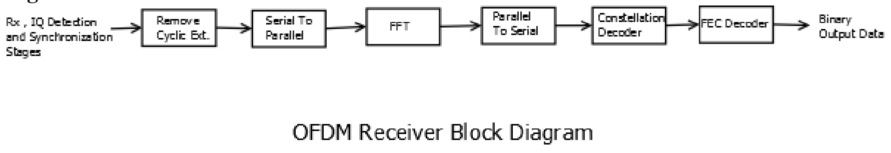


Fig. 4.19 OFDM Receiver



4.3.1.1 OFDM Stages

On the transmitter side, an OFDM system consists of

1. FEC Encoder : Wireless channels suffer from frequency selective fading which may result in wiping out of certain subcarriers.However, using a combination of error correction coding and interleaving, we can counteract some of the effects of this fading phenomena.IEEE 802.11a standard utilizes convolution coding in conjunction with Viterbi decoders at the receiver. Convolutional Encoders with encoders with bit rates of $1/2$,$2/3$ and $3/4$ are typically used.

2. Constellation Encoders: In OFDM systems PSK (Phase Shift Keying) and QAM (Quadrature Amplitude Modulation) are used to convert FEC-encoded bits to symbols.

3. Serial to parallel Converters: are used to convert serial stream into N parallel streams.

4. IFFT : IFFT stage is used to modulate these parallel streams into N orthogonal channels.

5. Serial to Parallel Converters : convert these parallel streams to serial stream

6. Cyclic Extension: One of the biggest issues with wireless channel happens to be multipath delay. Any transmitted signal is prone to get reflected by obstacles in the path(buildings, trees etc.) and instead of receiver getting one copy of the signal, it may receive multiple delayed copies. OFDM uses lower rate subcarriers which helps with delay spread but in order to almost completely remove multipath delay cyclical extension is used in which last L subcarriers out of N $[L < N]$ are folded back in front of each OFDM symbol in order to form one large $L+N$ orthogonal symbol. As long as channel multipath delay is smaller than L all delayed copies of the given symbol continue to be orthogonal to the original symbol and they don't overlap with the next symbol and are , therefore cancelled out.

OFDM receiver stages are essentially the reverse of the transmission stages as can be seen from the block diagram.

Mathematically,an OFDM signal $x[k]$ can be seen as a series of ,say, s OFDM symbols each consisting of M orthogonally modulated waveforms.

$$x[k] = \sum_{s} \sum_{m=0}^{M-1} a_{s,m} \phi_m [k - sM]$$

where $\phi_m[k]$ are M orthogonal waveforms such that

$$< \phi_m[k], \phi_n[k] >= \delta[m-n]$$

In the case of regular OFDM case, we use FFTs to implement orthogonal channels, such that

$$\phi_m[k] = e^{\frac{j2\pi mkT}{M}}$$

4.3.2 Advantages of OFDM

1. Orthogonality between subcarriers allows channel overlap and , thus, more efficient use of Bandwidth.

2. Orthogonality also helps in eliminating ICI (Inter Channel Interference) as subcarriers are orthogonal to each other.

3. ISI (Inter Symbol Interference) can be dealt with by using a guard band or cyclic prefix. Essentially, we don't use all subcarriers for data transmission. Instead some of the subcarriers use redundant information as data of the last L subcarriers

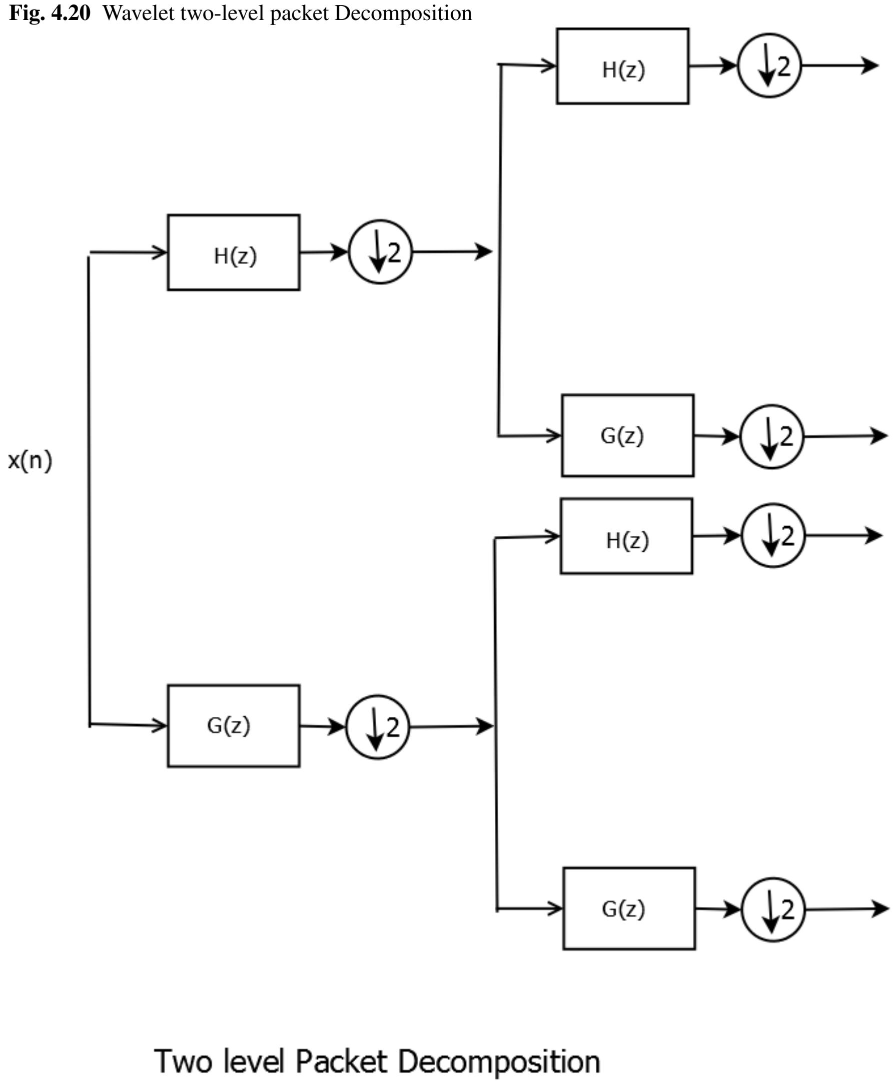
is folded back and added as a prefix to the symbol. This cyclic prefix is discarded at the receiver and helps guard against things such as multipath effects where delayed copies of same symbol is received because of inefficiencies in wireless channel.

4. Use of lower-rate multiple subcarriers helps in dealing with noise by using proper channel coding. A loss of handful of subcarriers due to noise can be compensated by using error correction codes while noise in high rate single carrier systems may result in loss of entire stream.

4.3.3 Wavelet Packet Modulation

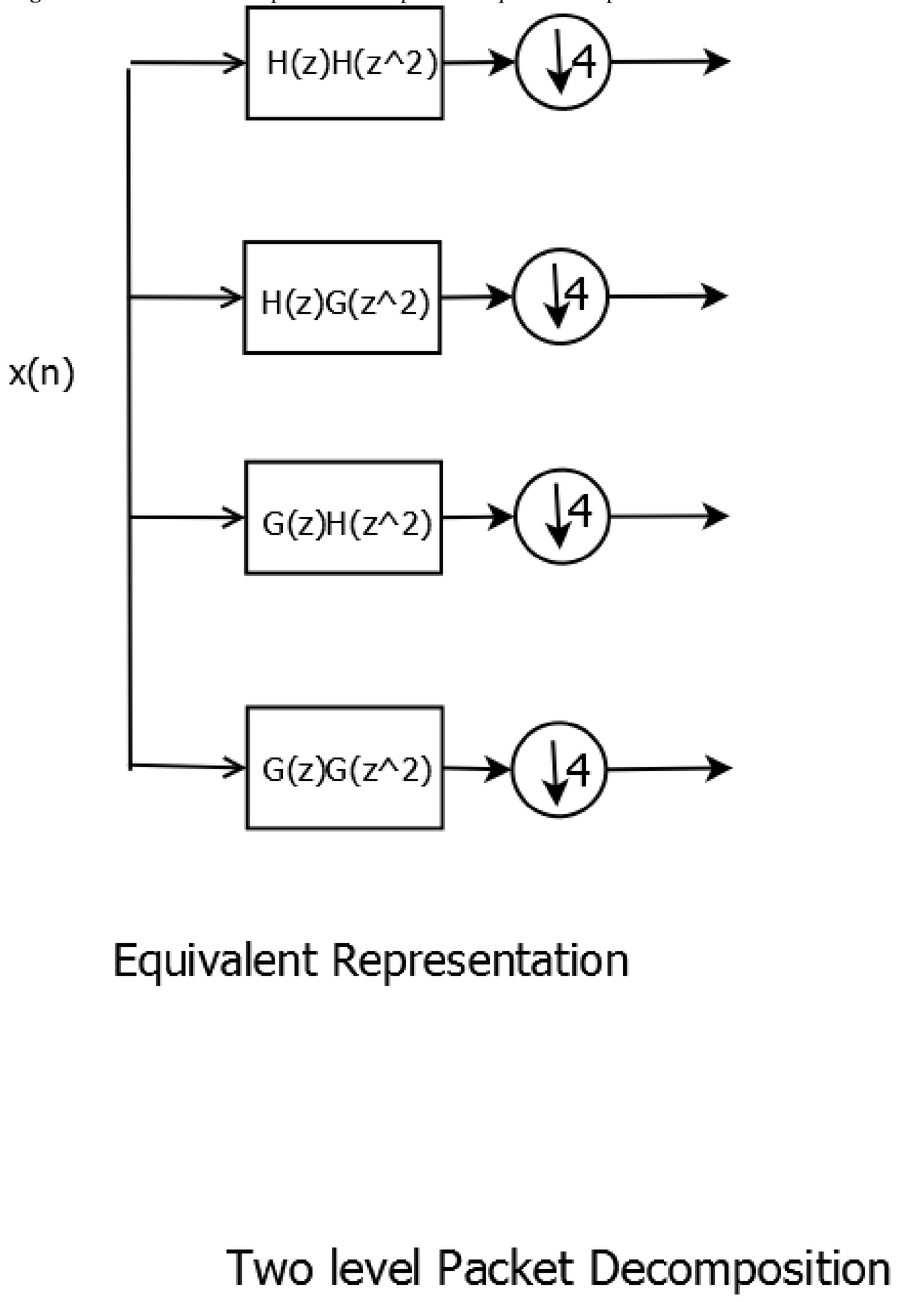
Since OFDM channels consist of M orthogonal subcarriers, we need M-band wavelet implementation if we want to substitute wavelet transform in place of DFTs. We achieve this using wavelet packet decomposition and reconstruction. Instead of dyadic iterations across only low pass filters, we dyadically decompose both high and low pass filters at every stage as show in the figure for two level wavelet packet tree decomposition.

Fig. 4.20 Wavelet two-level packet Decomposition



This wavelet tree is equivalent to one stage 4 band wavelet filter bank as shown in the figure.

Fig. 4.21 Wavelet two-level packet Decomposition Equivalent Representation



Decomposition is done similarly for any $M = 2^n$. The filters of M-band wavelet transform are obtained using noble identities and iterated filter bank properties. They are respectively $H(z)H(z^2)$,$H(z)G(z^2)$,$G(z)H(z^2)$ and $G(z)G(z^2)$. Wavelets corresponding to these filters are computed using scaling and wavelet dilation equations. The dilation equations are different in the M-band case:

$$\phi(t) := \sum_k hh(k)\phi(Mt-k)$$

and

$$\psi(t) := \sum_k gg(k)\phi(Mt-k)$$

where hh and gg filters are iterated bandpass filters obtained using noble identities.

An M-band wavelet system has one scaling function corresponding to the low pass filter and $M-1$ wavelet functions corresponding to each bandpass filter. For a three level wavelet packet decomposition using Daubechies orthonormal wavelets, the filters and $M = 8$ wavelets[unnormalized] are plotted in Matlab.

Fig. 4.22 Equivalent Filter Banks for 3 level wavelet packet decomposition tree

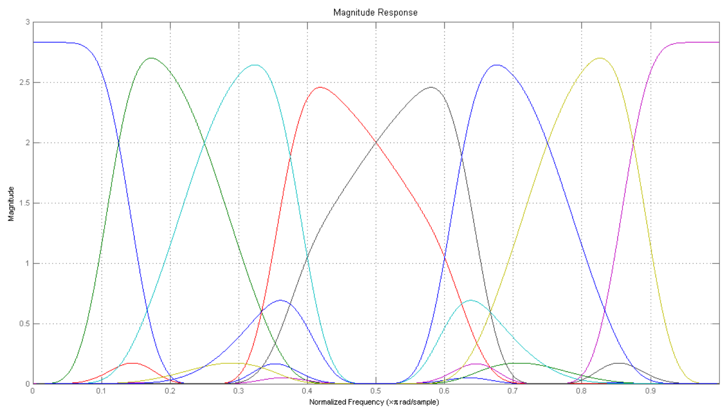
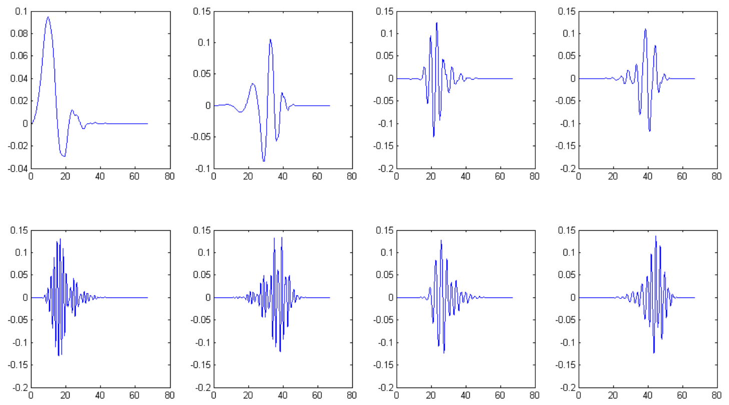


Fig. 4.23 Wavelets for 3 level wavelet packet decomposition tree



4.3.4 *Wavelet Packet Based OFDM systems*

Wavelet based OFDM systems are similar to regular OFDM with orthogonal subcarriers generated by iterated wavelet filter banks instead of DFTs but there are several crucial differences. Wavelet-OFDMs have better ISI and ICI performance. In addition, wavelet symbols generated are longer than DFT symbols. The length of iterated wavelet filters for M level wavelet packet decomposition is equal to $(M-1)(L-1)+1$ where L is the length of original filters. For higher level decompositions, iterated wavelet filters are significantly larger which results in longer subcarrier symbols. Guard Band or cyclical prefix is usually not needed in Wavelet OFDM case. A block diagram of Wavelet OFDM can, therefore, be given by

Fig. 4.24 Wavelet OFDM Transmitter

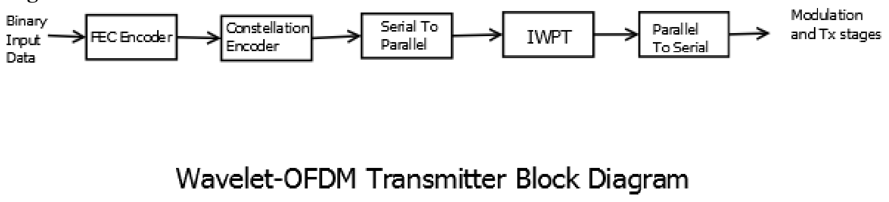
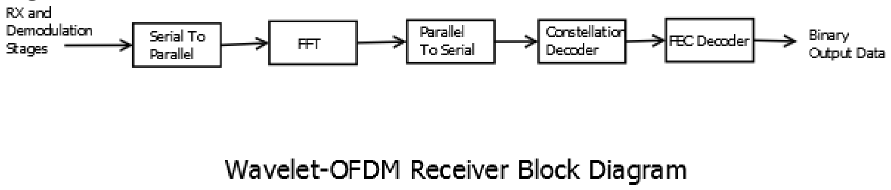


Fig. 4.25 Wavelet OFDM Receiver



Some properties of W-OFDM are following

1. Wavelet-OFDM has much better power spectrum characteristics than conventional OFDM which results in significantly improved band-rejection properties.For more on this , visit HD-PLC alliance webpage at http://www.hd-plc.org/modules/about/original.html .

2. Variable symbol lengths and variable subcarrier bandwidths can be easily obtained if we choose not to fully decompose each branch. This can be especially useful in wireless channels and other channels that suffer from frequency selective fading as it makes sense to not transfer same amount of data on each subcarrier.

3. Guard bands (cyclical prefix) is usually not needed (the WOFDM symbols are longer and they overlap which makes an effective cyclical prefix impossible to implement anyway) which results in more efficient data transmission. Additionally, pilot symbols(used for estimation and synchronization purposes in conventional OFDM) are not needed which adds to bandwidth efficiency.

4. WOFDM is entirely dependent on scaling filter (all other filters can be obtained from it) so it is much more flexible and we can design the system according to our needs by choosing the correct type of orthogonal filters.

5. It has been shown that wavelet systems give better Inter Carrier Interference (ICI) and Inter Symbol Interference (ISI) performance thanks to a more robust orthogonal construction.

For more information on WOFDM applications in industry, most prominently in wireless and powerline communications, see HD-PLC Panasonic, IEEE 802.11 standard resources(for information about conventional OFDM) and IEEE P1901 powerline standards/working groups websites. Check the references.

4.4 Wavelets and Seismology

4.5 Biomedical Engineering Applications

4.6 Image Compression

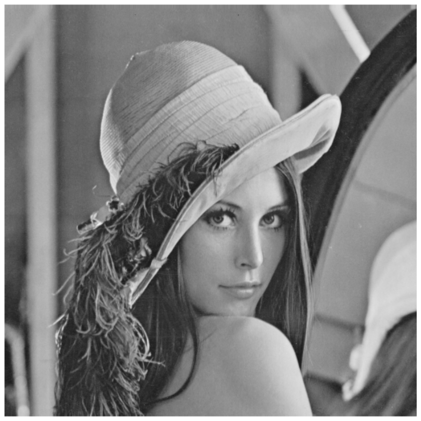
4.6.1 Image Approximation

The ideas for one dimensional signal approximation using fewer coefficients were developed previously. These ideas can be extended to $2-D$ signals, ie. images. It was also shown that Nonlinear Approximation results in better performance than linear approximation.Recall that Nonlinear approximation consists of selecting M largest coefficients instead of the first M coefficients as in Linear case. Nonlinear approximation performance for wavelets is significantly better than in the linear case as quite a few large coefficients exist at every scale of signal decomposition. In the images case, we'll stick with nonlinear approximation owing to its superior performance.

4.6.1.1 Approximation Example: Lena Image Approximation using $1/50$ coefficients

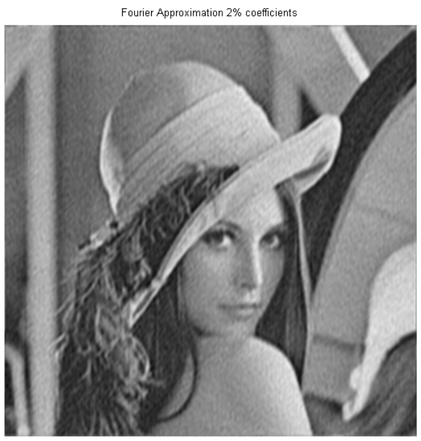
As an example of approximation , consider the Lena image approximation using Fourier Transform and Wavelet Transform. Algorithms are same as in the $1-D$ case.

Fig. 4.26 Original 512X512 Lena Image



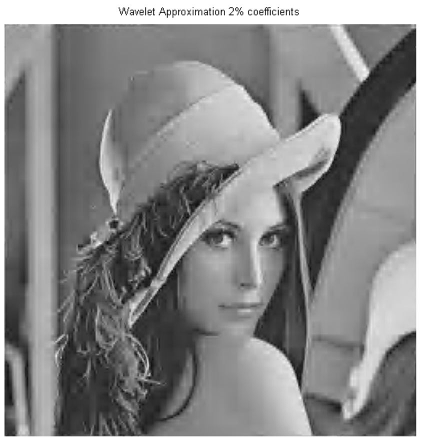
In Fourier case, we take the fourier transform of the image and retain only the largest one out of fifty coefficients while setting the rest equal to zero and then we reconstruct the image using only largest 2% coefficients . The reconstructed image is shown below.

Fig. 4.27 Lena Image Fourier Approximation using only largest 2% coefficients



Next we repeat the same approximation process with Wavelet Method. We take $J = 4$ stage Discrete Wavelet Transform (derived from Db2 wavelets) of the Lena image, choose the largest $\frac{1}{50}$ coefficients while setting the rest equal to zero. Then the Inverse Discrete Wavelet Transform is taken using these new coefficients. The resulting Image is shown below and is seen to be superior reconstruction compared to the Fourier case.

Fig. 4.28 Lena Image Wavelet Approximation using only largest 2% coefficients



4.6.2 Information Theory Background

Let us assume that system source is a discrete and finite consisting of N symbols and is given by S also known as alphabet.

$$S = [s_0, s_1, ..., s_{N-1}]$$

Let $X = [x_0, x_1, ..., x_i, ...$ be a collection of output sequence where each output x_i is taken from the set S. The probability that this system outputs the kthsymbol s_k is given by p_k where

$$\sum_{k=0}^{N-1} p_k = 1$$

as there are only N symbols.

Self-Information: We define information content I_k of each symbol based on its probability of being the output. It is given by

$$I_k = -\log_2(p_k)$$

and is measured in bits. If $p_k = 1$ or the symbol occurs every time then the information content of such a certain event is zero. On the other hand if the event is impossible $p_k = 0$ then the information content of such an event is ∞. More realistically, the likelier an event , less information is needed to define the event. On the other hand, unlikley events carry more information.

Source Entropy: is defined as expected value of self-information.

$$H(S) = E\{I_k\} = -\sum_{k=0}^{N-1} p_k \log_2(p_k)$$

$H(S)$ is measured in bits per symbol and is maximal for flat probability distribution, ie., if each symbol is equally probable $\frac{1}{N}$ then $H(S) = \log_2(N)$. Smaller source entropy can be obtained if probability distribution isn't flat.

Average Bit rate R_x: Let l_k be the length of the code associated with kth symbol with output probability p_k then the Average Bit Rate R_x is given by

$$R_x = \sum_{k=0}^{N-1} p_k l_k$$

The objective of good coding is to minimize the average bit rate for a given information source.Also, $H(S)$ sets the lower bound on average bit rate so R_x for a particular coding scheme can be measured against source entropy.

Prefix Code: One important condition for any good code is that no codeword can be a prefix of another codeword. As an example consider a code with following codewords

$$c_1 = 1, c_2 = 11, c_3 = 10, c_4 = 101$$

Consider a sequence 1011. It can be written as both c_4c_1 and c_3c_2 which will not work at the decoder stage. The solution is to generate codes using prefix binary tree.Each node of the tree has only one input branch and no more than two output branches with left branch labeled as 0 branch and right branch as 1. The binary tree ends in K leaves each corresponding to a unique codeword. In this case, length l_k of each codeword is given by the number of branches from the node to the kth leaf.

4.6.2.1 Huffman Coding

Premise of Huffman coding is that more probable a symbol, the shorter its length should be. If $P(s_i) < P(s_j)$ then $l_j >= l_i$. Huffman Codes are constructed in 4 steps as following.

1. Arrange symbol probabilities in decreasing order. $p_0 >= p_1 >= .... >= p_{N-1}$. The symbols s_k form the kth leaf nodes in binary tree T.

2. Combine the two nodes with lowest probability to form a new "parent" node. The new node will have the probability.

$$p_{12} = p_1 + p_2$$

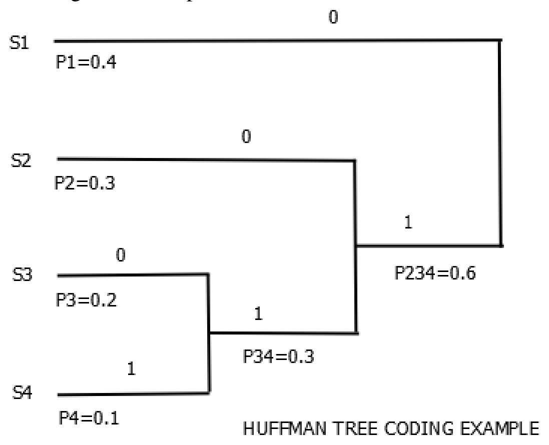
where p_1 and p_2 are the probabilities of the two “children” nodes while p_{12} is the probability of the new “parent” node.We assign 0 value to the branch connecting parent node to child node with probability p_1 and 1 value to the branch connecting parent node to child node with probability p_2.

3. Update binary tree T by replacing the two children nodes with the parent node. If the tree has more than one node after updating repeat step 2. If there is only one node left, it is the root node.

4. Codeword of each s_k can be obtained by traversing from root node to the kth leaf node.

Example:

Fig. 4.29 Huffman Coding Tree Example



The codeword can be obtained as in 4. We get

$$s_1 = 0, s_2 = 10, s_3 = 110, s_4 = 111$$

The average bit rate is found by using

$$R_x = \sum_{k=0}^{N-1} p_k l_k = 0.4*1 + 0.3*2 + 0.2*3 + 0.1*3 = 1.9$$

The source entropy is calculated using

$$H(S) = E\{I_k\} = -\sum_{k=0}^{N-1} p_k \log_2(p_k)$$

$$H(S) = -[0.4*log_2(0.4) + 0.3*log_2(0.3) + 0.2*log_2(0.2) + 0.1*log_2(0.1)] = 1.846$$

The average bit rate is a bit higher than source entropy but still gives better results than usual bit coding which corresponds to 2 bits per symbol for this example.

4.6.2.2 Arithmetic Coding

Drawbacks of Huffman Coding

1. It assigns integer length to each symbol.

2. It depends on the source probabilities. If the system is adaptive with probabilities that change, the code has to be reset each time.

Basis of Arithmetic Coding: Consider a binary sequence $B = 100111011$. We can convert it into a decimal number $0 \leq v \leq 1$ by expressing it as 0.100111011_2, which is equal to $v = 0.615234375_{10}$. In other words, regardless of how large a number is, it can be mapped to $[0, 1)$. In arithmetic coding, we create a sequence of nested intervals

$$\phi_k(S) = [a_k, b_k)$$

where a_k and b_k are real numbers and

$$0 \leq a_k \leq a_{k+1}, b_{k+1} \leq b_k \leq 1$$

Alternatively, we can represent each interval in form of base (lowest value b) and the length of the interval l.

$$|b, l>$$

Arithmetic coding Algorithm

1. Find probability distribution and cumulative distribution of the source inputs.

2. Find $|b_0, l_0>$ which is the initial value. Typically, initial values correspond to $|0, 1>$ or the entire length of the probability distribution.

3. Corresponding to the first input value, find $|b_1, l_1>$. These two values are calculated as follows

$$|b_k, l_k> = |b_{k-1} + c(s_k)l_{k-1}, p(s_k)l_{k-1}>$$

where the kth input corresponds to one value in the set of source symbol and $p(s_k)$ and $c(s_k)$ are respectively the probability and cumulative density functions associated with the source symbol.

4. Corresponding to every new input value we update the base sequence and its length by using 3.

5. Once the last input value, say nth value is read, we have a sequence $|b_n, l_n>$. This interval is used to encode a binary number choosing a value that corresponds to shortest binary code. This value is the arithmetic encoded value.

Example: Let us consider a source consisting of four symbols with probability and cumulative distribution given as

$$p = [0.2, 0.5, 0.2, 0.1]$$

$$c = [0, 0.2, 0.7, 0.9, 1]$$

Let us consider an input sequence $S = [2, 1]$.

Corresponding to first value 2, we have $p(s_1) = 0.2$ and $c(s_1) = 0.7$. Calculating

$$|b_1, l_1 >= |b_0 + c((s_1)l_0, p(s_1 l_0 >= |0.7, 0.2 >$$

Corresponding to second value 1, we have $p(s_2) = 0.5$ and $c(s_2) = 0.2$. Calculating

$$|b_2, l_2 >= |b_1 + c((s_2)l_1, p(s_2 l_1 >= |0.74, 0.10 >$$

If we are transmitting only two values, this encoding gives us a choice to generate shortest binary number in the range $0.74 - 0.75$. Minimum number of bits needed for encoding purposes are given by

$$B_m = -\log_2(l_n)$$

B_m is rounded up to the nearest integer. This coding is especially effective for a large chain of input values as all inputs are encoded together as a block with trade off being a more complex implementation is needed to generate the codeword.

Decoding

Although some overhead is needed, decoding depends solely on binary string v. The decoding is done as follows

1. Define $v_1 = v$.

2. The output value is given as follows:

$$s_k^o = s : c(s) \leq v_k \leq c(s+1)$$

Essentially, we map s_k^o on the symbol interval based on value of v_k and $c(s)$ is the cdf of the symbol value corresponding to s. As is obvious , the decoder needs the probability distributions (pdf and cdf) of the source symbols in order to decode the binary string. This adds to the overhead but it amounts to small value in case of large scale data transmission.

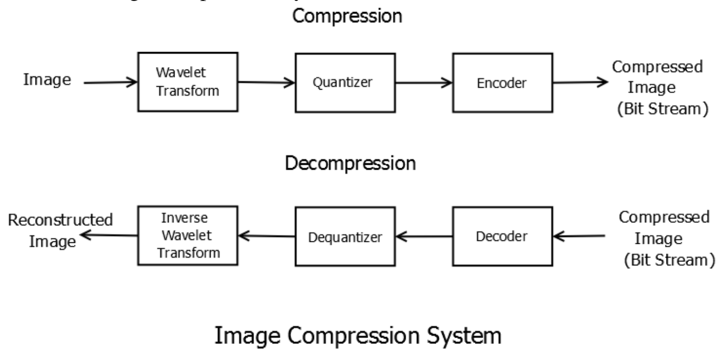
3. Update Value of v_{k+1} as following:

$$v_{k+1} = \frac{v_k - c(s_k)}{p(s_k)}$$

4. One issue with arithmetic coding is that the recursion formula in 3 will keep going on regardless of the number of input values so some overhead is needed to tell the decoder where to stop. Typically, overhead can consist of number of input values N along with the binary string v and probability distributions. The decoder can then decode for N output values. If $k == (N+1)$, the decoder stops else we go back to step 2.

4.6.3 Wavelet Image Compression System

Fig. 4.30 Wavelet Image Compression System



Wavelet Compresser broadly consists of three stages.

1. Wavelet Transform Stage: The advantage of wavelet transform stage can be seen in the image approximation example shown earlier. The image is transformed into a set of coefficients most of which are close to zero and can be eliminated which results in substantial reduction in amount of data that needs to be encoded. Thresholding is done in order to further reduce the number of significant coefficients. Various algorithms will be covered in the next section.

2. Quantizer: Quantization step is usually skipped if one is aiming for lossless compression, eg. medical imaging systems, as quantization is non-invertible. In case of lossy compression, quantization is done to reduce precision of the values of wavelet transform coefficients so that fewer bits are needed to code the image. For example if the transform coefficients are 64- bit floating point numbers while a compression of the order of 8 bits per pixel is required then quantization is necessary.

3. Encoding: Encoding was covered in previous chapter. Basically, quantized wavelet transform coefficients are coded so that fewest number of bits are required. Huffman coding, arithmetic coding and their variants are usually used in these systems.

4.6.4 Wavelet Transform Stage

Image Decomposition in terms of two level filtering can be shown as in the next figure. LL_d represents Low-Low pass filtered image at the decomposition level d. LH_d, HL_d and HH_d represent Low-High pass filtered image, High-Low Pass filtered image and High-High pass filtered image respectively at decomposition level d. The $2D$ filter bank is also shown here.

Fig. 4.31 Image Decomposition

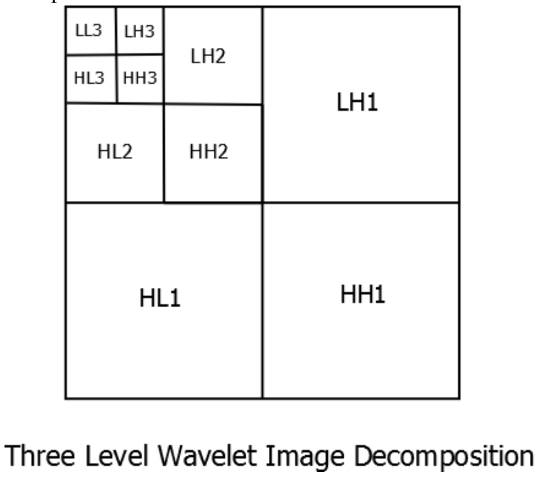
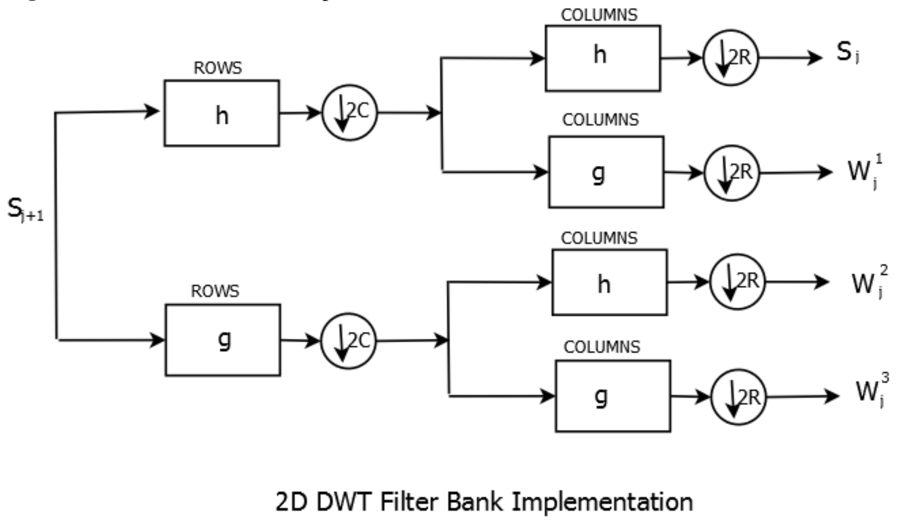
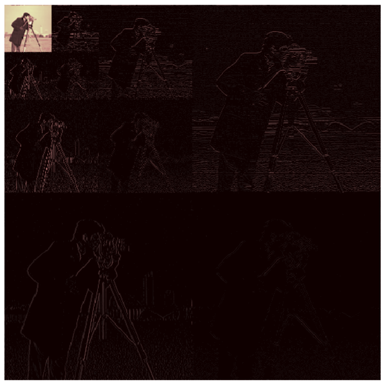


Fig. 4.32 2D DWT Filter Bank Implementation



Three level decomposition of an image (cameraman) is given in the figure below.

Fig. 4.33 Image Decomposition



In the figure above, large coefficients are shown by color pink while negligible coefficients are depicted by black and dark shades. The LL_3 image is the one containing most significant coefficients while higher frequency image segments have very few large coefficients as most of their coefficients are either zero or nearly zero. Additionally, it can be seen that there is a spatial relationship between coefficients at different decomposition levels.

4.6.5 Zero-Tree Coding

If we look at coefficients in ,say, LH_d subimage and compare them to coefficients in LH_{d-1}, it is seen that there is a spatial relationship across scales. For example, there are a bunch of zero coefficients near the cameraman's shoulder (high coefficient region) and it is replicated across scales in LH subimages. Same is true for HL_d and HH_d subimages. This multiscale dependency is the basis of zero-tree coding. For every coefficient at coarse scale there are 4 "children" at the next finer scale, ie. one HL_d coefficient is the "parent" of four coefficients in HL_{d-1} and "grand parent" of 16 coefficients in HL_{d-2}. The finest scale coefficients don't have any children.

4.6.5.1 Embedded Zero-Tree Wavelet(EZW) Coding

J.M. Shapiro introduced EZW coding in his groundbreaking 1993 paper. EZW makes use of zero-trees and is a progressive encoding system, ie., as more bits are added the system becomes more accurate.Coding is done in multiple passes. Large coefficients are most important but the way zero-tree deals with smaller coefficients is what makes EZW so effective.

Algorithm

Initial threshold T_0 is chosen based on the largest coefficient $|y|_{max}$

$$\frac{|y|_{max}}{2} < T_0 < |y|_{max}$$

We also define T_k where $T_k = T_0 2^{-k}$, $k = 0, 1, ..., K-1$ and is used to update threshold value with every pass.

Let us assume that we are compressing an NXN image. This image is coded using $K+1$ binary arrays(bit planes). First bit plane consists of sign bit, the second represents the Most Significant Bit of each coefficient and so on. This process can be terminated at any bit plane $n < K$ but larger the number of bits that are encoded, more accurate the coding is.

Significance Pass: Coefficients are scanned from LL_d down to HH_1 in order and it is ascertained whether they are significant or insignificant. An example of a significance pass and labeling can be as following-

We can use 4 labels to define coefficients

1. POS : Positive Significant. The absolute value of coefficient is greater than the threshold and coefficient has positive sign.

2. NEG : Negative Significant. The absolute value of coefficient is greater than the threshold and coefficient has negative sign.

3. ZTR : Zero Tree Root. The coefficient value is 0 and so are the values of each of its descendants.

4. IZ : Isolated Zero. The coefficient value is 0 but the descendant values are not all zero.

Refinement Pass : Once we have labeled coefficients, the next pass is used to code the bit values. Each significant bit is compared with updated threshold value T_k, ie. , $T_1 = T_0/2$ and so on in successive passes. If it is greater than T_k then a bit value 1 is the output, otherwise 0 is the output.

Full Algorithm

1a. Set wavelet transform mean to zero by subtracting the mean from each sample value. We should end up with positive and negative values instead of just positive values.

1b. Sign bits are assigned based on whether the coefficient is greater than or less than zero. These sign bits are transmitted in addition to bit plane encoded bits.

2. Choose Initial threshold T_0. It is usually chosen as

$$th = floor(\log_2(|y|_{max}))$$

$$T_0 = 2^{th}$$

where $(|y|_{max})$ is the largest coefficient value as mentioned earlier.

3. Significance Pass: In which we find the significant bits in each iteration. They are POS (P) or NEG (N). Insignificant bits are either ZTR(Z) or IZ(I).

4. Refinement Pass: All coefficients are set to zero and then we systematically assign bit values only to significant bits. This is done by comparing the coefficient values of POS (P) and NEG (N) with the threshold. If they are greater than the threshold, the output bit is 1 and we subtract the threshold value from the absolute coefficient value and the updated coefficient value is stored in the coefficient map.

5. Threshold is updated $T_k = T_0 2^{-k}$ for $k = 1, ..., K$

6. Repeat steps 3-5 until either all K bit planes are finished or the process is terminated early if less accuracy is needed.

4.6.6 JPEG2000 Wavelet Standard

JPEG2000 uses two wavelet transforms. Both are biorthogonal but only one is "reversible" as it uses only integer coefficients.

1. Biorthogonal $9/7$ Wavelet Transform: It introduces quantization error as coefficients are non-integer.

2. Biorthogonal $5/3$ Wavelet Transform.

4.6.6.1 JPEG2000 Coder Basics

1. Color Component Transformation and Tiling : RGB image is transformed into either YC_BC_R or YUV. this transformed image is then divided into blocks(tiles), each of which is processed separately.

2. Wavelet Transform:N level dyadic transform is performed on each tile using either $9/7$ or $5/3$ biorthogonal wavelets.

3. Scalar Quantization and Partition: Scalar Quantization is used to reduce complexity at the cost of some loss of quality. Packet partitioning is done to transform each quantized subband into a set of non-overlapping rectangles.

4. Block Coding: Code blocks are formed using non-overlapping rectangles. Each code block is encoded separately using arithmetic coding of bit planes. Bit plane coding is done in three passes- significance, refinement and clean up. In case of lossless coding, all bit planes need to be coded and transmitted. These bit planes are then encoded using arithmetic coder.

For more details on JPEG2000, see the references.

References

1. Jamin A, Mahonen P (2005) Wavelet Packet Modulation for Wireless Communications. Wireless Communications and Mobile Computing Journal,Vol. 5, Issue 2: 123–137
2. Yang W., et.al (1997) A Multirate Wireless Transmission System Using Wavelet Packet Modulation. IEEE 47th Vehicular Technology Conf., Phoenix, AZ, USA, 1: 368–372

3. Galli S., Koga H, Kodama N (2008) Advanced Signal Processing for PLCs: Wavelet-OFDM. IEEE Int. Symp. on Power Line Communications and Its Applications: 187–192
4. Fisal N, Hosseini H,Syed-Yosef (2010) Wavelet Packet based Multicarrier Modulation for Cognitive UWB Systems.Signal Processing: An International Journal (SPIJ) Volume: 4 Issue: 2:75–84
5. Huffman, D (1952) A Method for the Construction of Minimum-Redundancy Codes. Proceedings of the IRE Volume: 40 Issue: 9: 1098 - 1101
6. Marcellin M , Bilgin A, Gormish M, Boliek M (2000) An Overview of JPEG-2000.Proceedings of the Conference on Data Compression (DCC'00) : pp. 523
7. IEEE 802.11a-1999 Standard
http://www.ieee802.org/11/
8. RM Wavelet Based (WOFDM) PHY Proposal for 802.16.3, Rev.0.0
http://www.ieee802.org/16/tg3/contrib/802163c-01_12.pdf
9. Antini A L, Orthogonal Frequency Division Multiplexing for Wireless Networks
http://www.create.ucsb.edu/ATON/01.01/OFDM.pdf
10. Said A, Introduction to Arithmetic Coding- Theory and Practice
http://www.hpl.hp.com/techreports/2004/HPL-2004-76.pdf
11. S Mallat (1999) A Wavelet Tour of Signal Processing. Academic Press
12. G Strang, T Nguyen(1996) Wavelets and Filter Banks. Wellesley Cambridge Press
13. M Vetterli, J Kovacevic(1995) Wavelets and Subband Coding. Prentice Hall Signal Processing Series
14. I Daubechies (1992) Ten Lectures on Wavelets. SIAM: Society for Industrial and Applied Mathematics
15. M Weeks (2006) Digital Signal Processing using Matlab and Wavelets. Infinity Science Press
16. V Goyal, M Vetterli, J Kovacevic (2010) Fourier and Wavelet Signal Processing. Manuscript Draft
17. Gonzalez, Woods (2008) Digital Image Processing, 3rd edition. Prentice Hall
18. J Lim (1989) Two-Dimensional Signal and Image Processing . Prentice Hall
19. M Marcellin, D Taubman (2001) JPEG2000: Image Compression Fundamentals, Standards and Practice . Springer

Chapter 5
Advanced Topics

Abstract To be completed

5.1 Directional Wavelets

5.1.1 2D Continuous Wavelet Transform

All geometric operations in $2D$ space can be seen as a combination of three operations.

1. Translation: Translation in $2D$ corresponds to translation in $1D$ space. A translation of vector $\bar{b}$ is represented as $\bar{x}-\bar{b}$. $\forall \bar{x}, \bar{b} \in R^2$

2. Dilation : Dilation by a scalar a is given by $a\bar{x}$.

3. Rotation : Rotation by θ which is given by $r_\theta(\bar{x})$ where r_θ is the usual rotation matrix given by
$\begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}$

Let $s(\bar{x})$ be a $2D$ signal then the three operations on this signal are given by

$$s_{\bar{b},\theta,a}(\bar{x}) = \frac{1}{a} s(\frac{r_{-\theta}(\bar{x}-\bar{b})}{a})$$

where $\forall \bar{x}, \bar{b} \in R^2, \forall \theta \in [0, 2\pi], \forall a > 0$

If the signal s is rotation invariant then such a signal is given by

$$s_{\bar{b},a}(\bar{x}) = \frac{1}{a} s(\frac{\bar{x}-\bar{b}}{a})$$

which is $2D$ equivalent of a $1D$ translated and dilated signal.

Definition: A $2D$ continuous wavelet transform of a $2D$ signal with respect to a $2D$ wavelet $\psi_{\bar{b},\theta,a}$ is given by the inner product

$$S_{\bar{b},\theta,a} = <\psi_{\bar{b},\theta,a}, s>$$

$$S_{\bar{b},\theta,a} = \int_{R^2} s(\bar{x})\psi^*_{\bar{b},\theta,a} d^2\bar{x}$$

$$S_{\bar{b},\theta,a} = \frac{1}{a}\int_{R^2} s(\bar{x})\psi^*(\frac{r_{-\theta}(\bar{x}-\bar{b})}{a})d^2\bar{x}$$

The Fourier Transform is given by

$$\widehat{S}_{\bar{b},\theta,a} = a\int_{R^2} e^{i\bar{b}\bar{\omega}}\widehat{s}(\bar{\omega})\widehat{\psi}^*(ar_{-\theta}(\bar{\omega}))d^2\bar{\omega}$$

5.1.1.1 *2D* CWT Visualization

$2D$ CWT consists of four variables - two spatial variables represented by $\bar{b}$, scale parameter a and rotation parameter θ. Given that four dimensional visualization is really difficult, the solution is to eliminate one or more variables and then plotting $S_{\bar{b},\theta,a}$, the $2D$ wavelet transform.

1. Position Representation : The scale and rotation parameter are fixed and CWT is plotted as a function of $2D$ space.

2. Scale-Angle Representation : The position parameters are fixed while we vary scale and angles to plot CWT.

Using Polar Co-ordinates : Instead of two-dimensional space parameter $\bar{b}$ we can use polar representation $(|\bar{b}|, \alpha)$ where $|\bar{b}| = \sqrt{b_x^2 + b_y^2}$ known as range and $\alpha = \arctan b_y/b_x$ which is also known as perception angle. We can use these four parameters to come up other sets of visualizations, namely

3. Perception Angle-Scale Representation : We fix range and rotation angle while using perception angle and scale.

4. Range-Rotation Angle Representation: Range and Rotation angles are used while fixing scale and perception angle.

5. Scale-Range Representation: Scale and Range are used while fixing both angles.

6. Angle-Angle Representation: Angles are used while fixing range and scale.

5.1.1.2 Wavelet Discretization

The issue with $2D$ continuous transform is same as in the $1D$ case. It contains a lot of redundant information and a comprehensive continuous transform will be impossibly computationally intensive. We deal with it the same way by discretizing this transform. We choose a grid δ to discretize as following

1. Discretized Scale Parameter is given as

$$a_j = a_0\lambda^{-j}$$

where $\lambda > 1$. If we choose dyadic scales then $\lambda = 2$.

2. Discretized Rotation Parameter is uniformly discretized over the entire rotation space $[0, 2\pi)$ as following

$$\theta_l = \frac{2l\pi}{L}$$

where $l = [0, 1, ..., L-1]$,$\forall L \in N$,$\forall l \in Z$

3. Translation Parameters are discretized using scale and rotation parameters.

$$\bar{b}_{jlm_x m_y} = a_0 \lambda^{-j} r_{\theta_l}(m_x \beta_x, m_y \beta_y)$$

$$\bar{b}_m = \bar{b}_{jlm_x m_y}$$

Wavelet Transform is discretized as following by putting $a_0 = 1$ and using discretized $\bar{b}_m$

$$S_{\bar{b}_m, \theta_l, \lambda^{-j}} = \lambda^j \int_{R^2} s(\bar{x}) \psi^*(\lambda^j r_{-\theta_l}(\bar{x} - \bar{b}_m)) d^2\bar{x}$$

The Fourier Transform is similarly given by
$\widehat{S}_{\bar{b}_m, \theta_l, \lambda^{-j}} = \lambda^{-j} \int_{R^2} e^{i\bar{b}_m \bar{\omega}} \widehat{s}(\bar{\omega}) \widehat{\psi}^*(\lambda^{-j} r_{-\theta_l}(\bar{\omega})) d^2\bar{\omega}$

5.1.2 Isotropic vs Anisotropic Wavelets

Isotropic Wavelets are rotation invariant wavelets whereas Anisotropic wavelets are directional wavelets. One example of wavelet family that can be used to generate isotropic and anisotropic wavelets is $2D$ Mexican Hat Wavelet.Anisotropic wavelets can be obtained by stretching isotropic wavelets in a desired direction. This is accomplished by using Anisotropy matrix A

$$X_a = AX_{iso}$$

where $A = diag[\varepsilon^{-\frac{1}{2}}, 1]$ is a $2X2$ matrix for $\varepsilon \geq 1$

Mexican Hat Wavelets are Laplacian of Gaussian functions which are given by

$$\psi_m(x,y) = [2 - (x^2 + y^2/\varepsilon)]e^{(-\frac{(x^2+y^2/\varepsilon)}{2})}$$

$\varepsilon = 1$ gives Isotropic wavelet which is plotted in figure below.

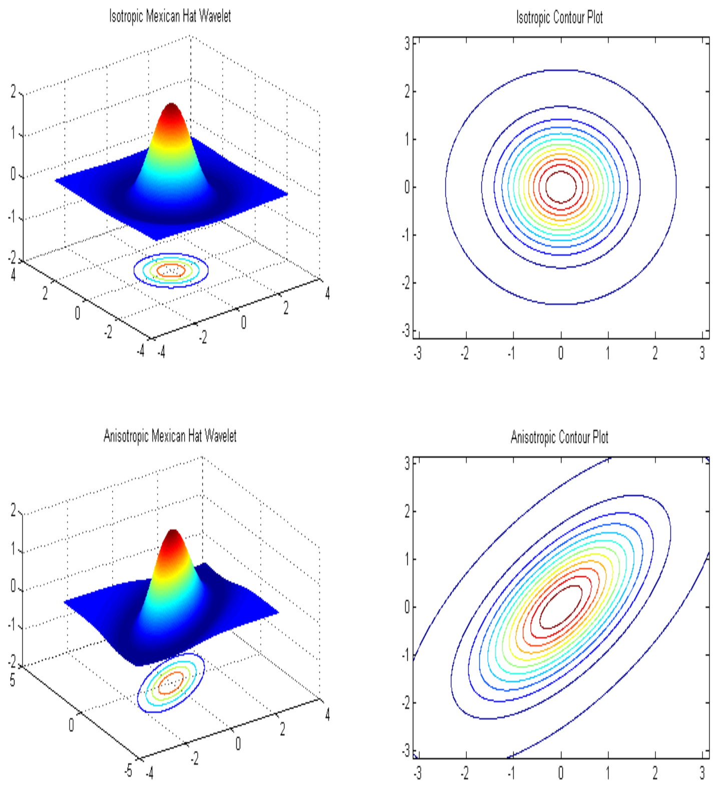
$$\psi_{miso}(x,y) = [2 - (x^2 + y^2)]e^{(-\frac{(x^2+y^2)}{2})}$$

Anisotropic (Rotation Variant) Wavelet is given by using the formula introduced above.

$$\psi_{\bar{b},\theta,a}(\bar{x}) = \frac{1}{a}\psi(\frac{r_{-\theta}(\bar{x}-\bar{b})}{a})$$

Different values of θ and scaling parameters a generate different Mexican hat wavelets. For this example, we use values of $a = 0.8$, $\theta = 3\pi/4$ and $\varepsilon = 5$. It should be noted,however, that these "stretched" wavelets only have limited directional selectivity.

Fig. 5.1 Isotropic and Anisotropic Mexican Hat Wavelets



5.1.3 Directional Wavelet Examples

A wavelet is said to be directional wavelet if its fourier transform is directionally contained in a convex cone in frequency space ω with apex at the origin. Thus, anisotropic Mexican Hat wavelet even with its directional selectivity is not a Directional wavelet as it is centered around the origin.

$2D$ Morlet Wavelet is given by

$$\psi_M(x) = e^{i\bar{k}_0\bar{x}} e^{-\frac{1}{2}|A\bar{x}|^2} - e^{-\frac{1}{2}|A^{-1}\bar{k}_0|^2} e^{-\frac{1}{2}|A\bar{x}|^2}$$

where $A = diag[\varepsilon^{-\frac{1}{2}}, 1]$ is a $2X2$ matrix for $\varepsilon \geq 1$ and $\bar{k}_0$ is a $n-$ dimensional vector corresponding to $n-$ dimensional $\bar{x}$ vector. Morlet wavelet is complex and assuming we take values of $\varepsilon = 3$ and $\bar{k}_0 = (0, 6)$, real and imaginary parts of Morlet can be plotted as shown below.

Fig. 5.2 Real Part of Morlet Wavelet

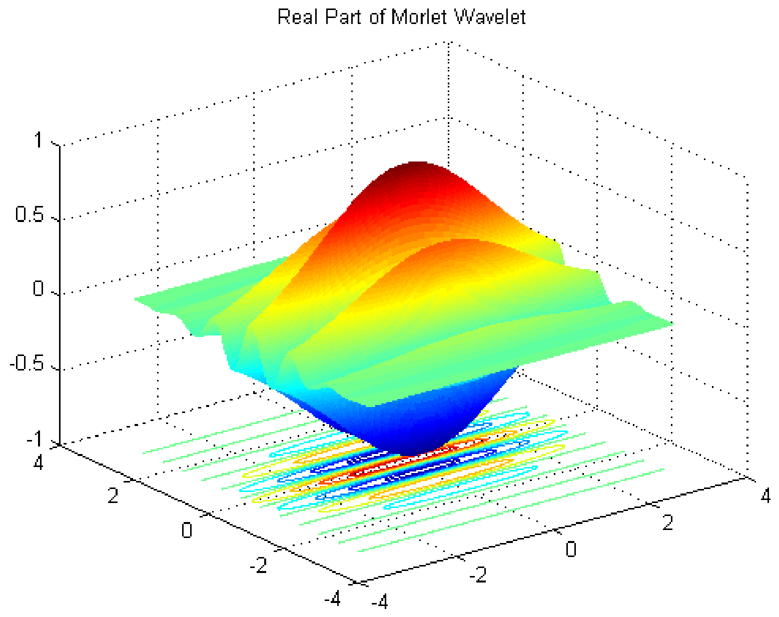
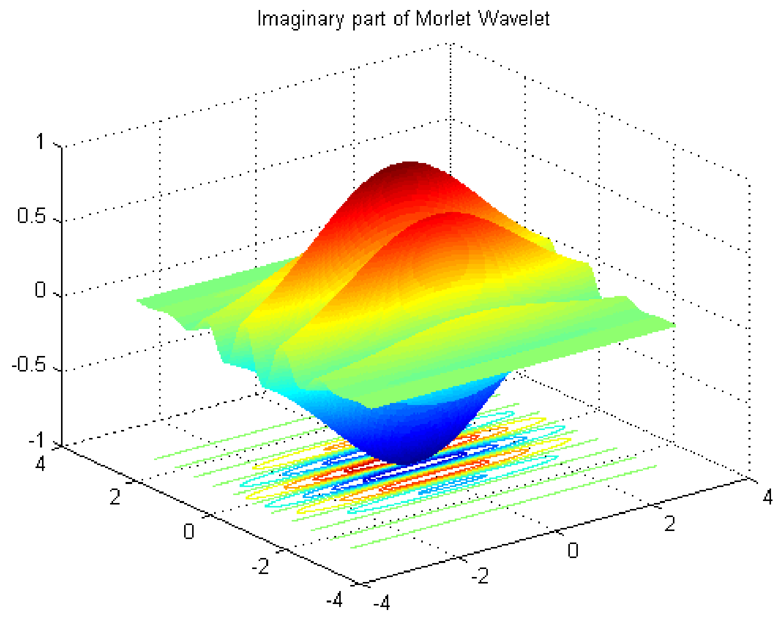
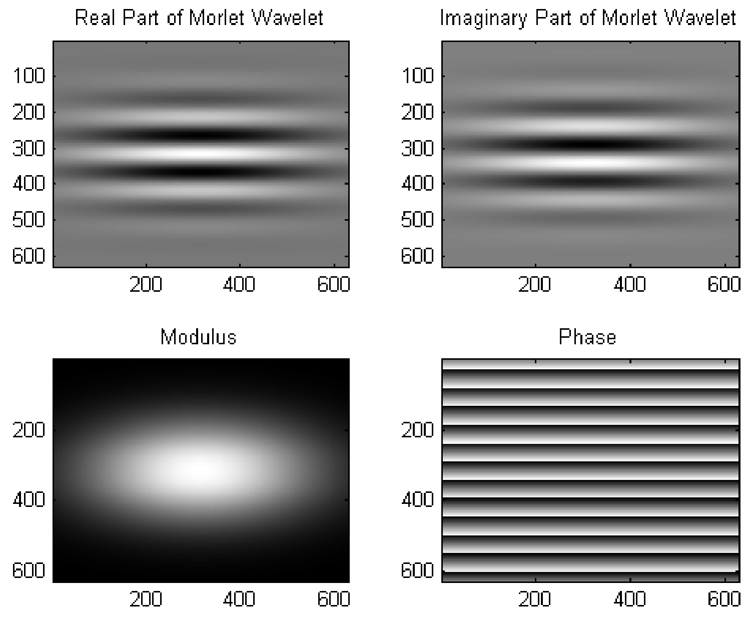


Fig. 5.3 Imaginary Part of Morlet Wavelet



2D plots of real and imaginary parts along with phase and modulus are also shown. Color white corresponds to highest amplitudes while black corresponds to lowest amplitudes. Phase plot is the most interesting one as it is a collection of straight lines whose intensity varies linearly and periodically in perpendicular direction(to $\bar{k}_0$).

Fig. 5.4 Morlet Wavelet Directionality Properties

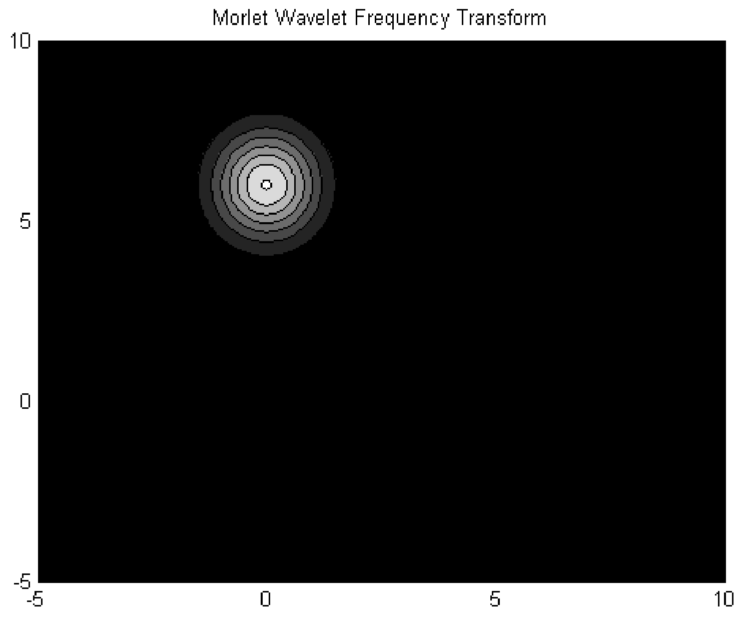


In spatial frequency domain, the fourier transform $\widehat{\psi}_M(\omega)$ is centered at $\bar{k}_0$ instead of origin and its directionality depends on anisotropic matrix ,specifically on value of ε. The Fourier Transform is elongated in ω_y direction in this case. The shape of cone depends on ε value as greater anisotropy corresponds to narrower cone.

$$\widehat{\psi}_M(\omega) = \sqrt{\varepsilon}\left(e^{-\frac{1}{2}|A^{-1}(\bar{\omega}-\bar{k}_0|^2} - e^{-\frac{1}{2}|A^{-1}\bar{k}_0|^2} e^{-\frac{1}{2}|A^{-1}\bar{\omega}|^2}\right)$$

The second term is usually small for large values of $\bar{k}_0$ and is therefore ignored.

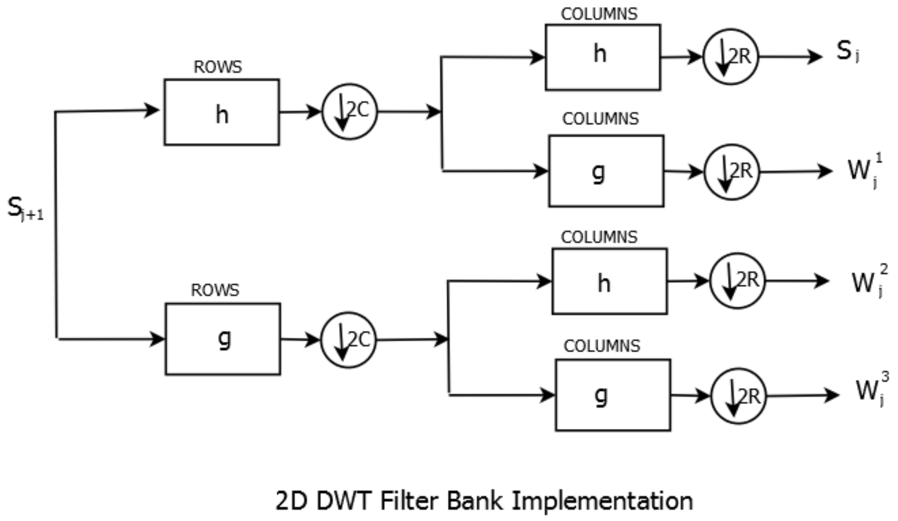
Fig. 5.5 Morlet Wavelet Frequency Transform



5.1.4 *Directionality and Separable Filter Banks*

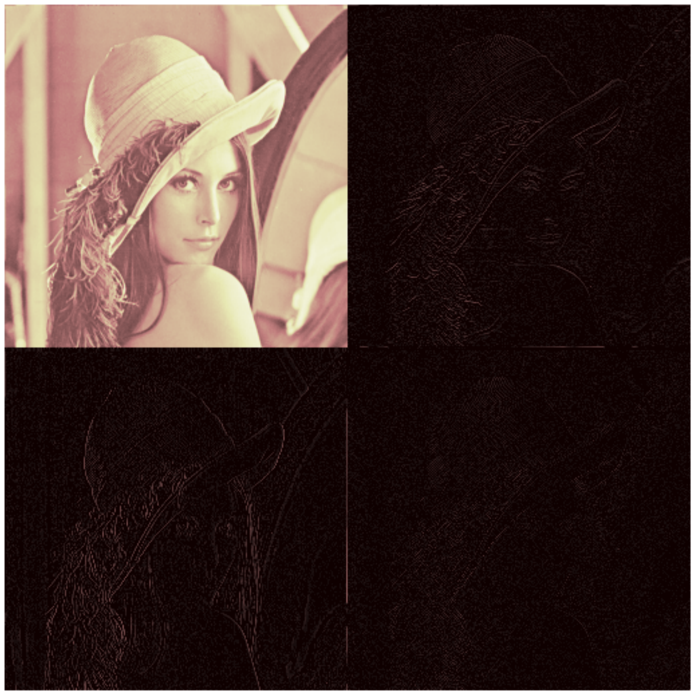
Standard 2*D* DWT has limited directionality as it resolves directions in only three directions- horizontal, vertical and diagonal. 2*D* Filter Bank Implementation of 2*D* DWT is shown below

Fig. 5.6 2D Discrete Wavelet Transform



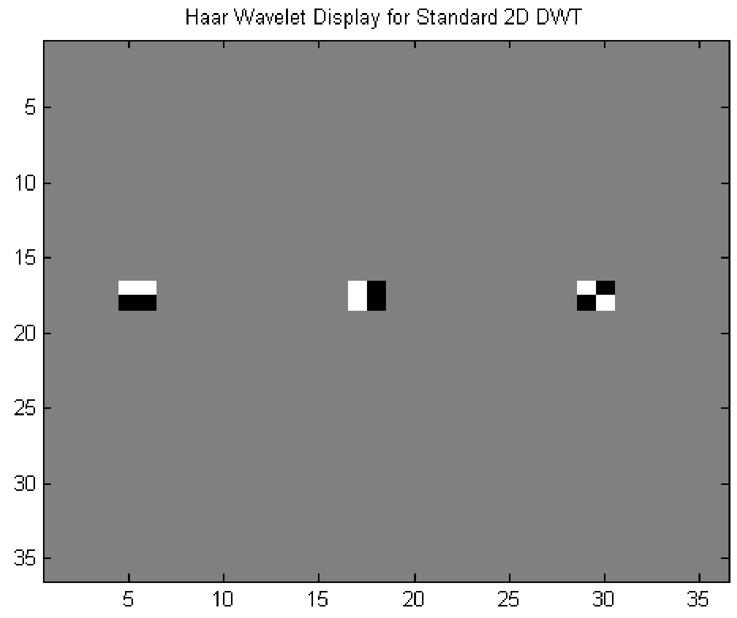
The directional decomposition of Lena image using this filter bank is three-directional with large coefficients in spatial band pass frequency domain aligned in vertical, horizontal and diagonal directions.

Fig. 5.7 1-level DWT decomposition of Lena Image



Standard 2*D* DWT is isotropic as filtering and sampling operations are performed identically in both horizontal and vertical directions. As we know, there are three wavelets corresponding to three bandpass frequency domains - the low-high band, high-low band and high-high band and as can be seen they have limited directional properties.However, low complexity and ease of implementation means that isotropic separable filter banks are used extensively in practice. The three wavelets corresponding to Haar filters are plotted below.

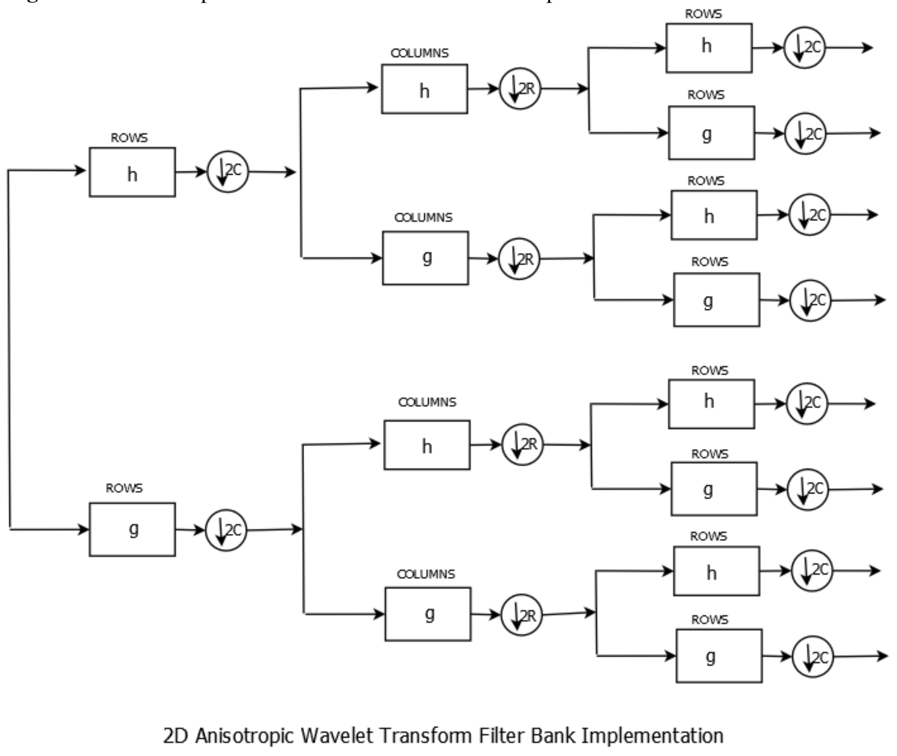
Fig. 5.8 Three Haar Wavelets for 1-level 2D DWT



5.1.4.1 Anisotropic DWT using Separable Filter Banks

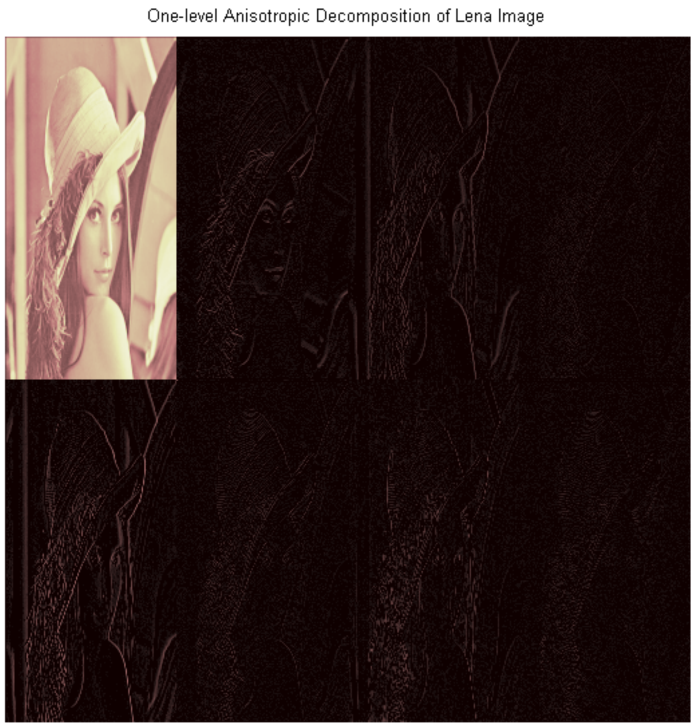
In certain $2D$ applications it is desirable to detect edges and contours that are aligned at various angles. Standard DWT gives sub-optimal performance in these type of cases.One straightforward and slightly better method is to use anisotropic DWTs that are constructed by using unequal filtering and sampling operations in horizontal and vertical directions. As mentioned this is not an optimal approach but may result in better performance in certain cases. One such example is anisotropic filtering is shown below.

Fig. 5.9 2D Anisotropic Discrete Wavelet Transform Example



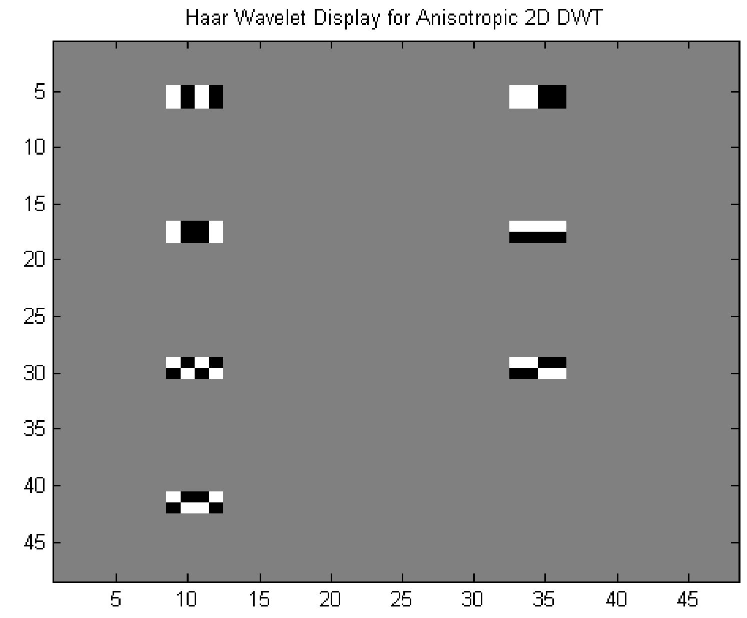
It uses three-levels of filtering and subsampling operations at the analysis and synthesis(not shown) side which means higher complexity than the standard case which uses two levels of operations. In this particular case, horizontal filtering and subsampling is followed by vertical filtering(and subsampling) and the third level consists of horizontal filtering(and downsampling) so for every stage of DWT operations, a 2*D* signal is divided into 8 subimages- one low pass subimage and 7 band pass subimages. This can be seen for “Lena” Image.

Fig. 5.10 1-Stage Anisotropic DWT decomposition of Lena Image



As can be seen the 7 bandpass images accentuate different directional elements of the image. This directionality property of this particular Anisotropic DWT can be further studied by plotting wavelets associated with each of the seven bandpass subimages for one stage of DWT decomposition. As in the standard DWT case, these seven wavelets are plotted corresponding to Haar wavelet filters.

Fig. 5.11 Seven Haar Wavelets for 1-stage 2D Anisotropic DWT



As can be seen from the wavelet plots , while the directional properties are better in this case, such a configuration will still struggle to isolate edges and contours arbitrary aligned in the 2D space.

5.2 Steerable Pyramid

5.2.1 Theory of Steerable Filters

Steerable Filters are a class of oriented filters that can be expressed as a linear combination of a set of basis filters. As an example, let us consider isotropic Gaussian filter $G(x,y)$.

$$G(x,y) = e^{-(x^2+y^2)}$$

First derivative of $G(x,y)$ is given by G_1 and let G_1^{θ} be the first derivative rotated by an angle θ about the origin. In x direction the angle $\theta = 0°$ and in y direction, $\theta = 90°$.

$$G_1^{0^\circ} = \frac{\partial G}{\partial x}$$

$$G_1^{0^\circ} = -2xe^{-(x^2+y^2)}$$

and

$$G_1^{90^\circ} = \frac{\partial G}{\partial y}$$

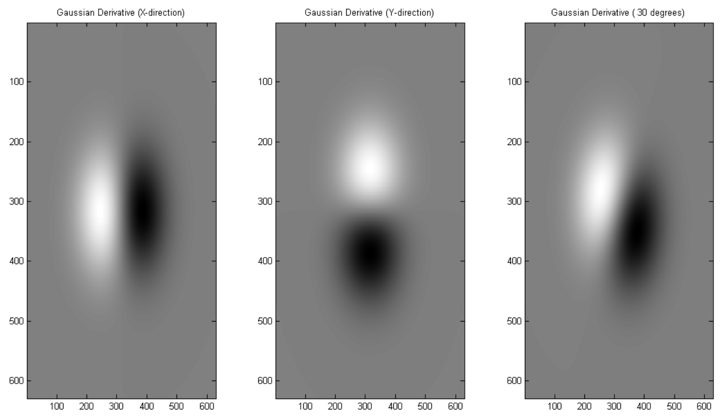
$$G_1^{90^\circ} = -2ye^{-(x^2+y^2)}$$

In $2D$ space $G_1^{0^\circ}$ and $G_1^{90^\circ}$ are seen to span the entire space and are the basis filters. An arbitrarily oriented first derivative filter can be expressed as a linear combination of these two filters.

$$G_1^{\theta^\circ} = G_1^{0^\circ} cos(\theta) + G_1^{90^\circ} sin(\theta)$$

For $\theta = 30^\circ$, the three filters can be plotted as below

Fig. 5.12 Gaussian First Derivative Filters oriented a) in x-direction b) in y-direction and c) at 30°about the origin



The first derivative of Gaussian is also an edge detection filter which smooths the $2D$ signal and then computes the gradient. Having an oriented version of this filter helps in isolating oriented edges and contours. Convolution of an Image I with an oriented Gaussian first derivative filter $G_1^{\theta^\circ}$ can be given by

$$I_o^{\theta^\circ} = G_1^{\theta^\circ} * I$$

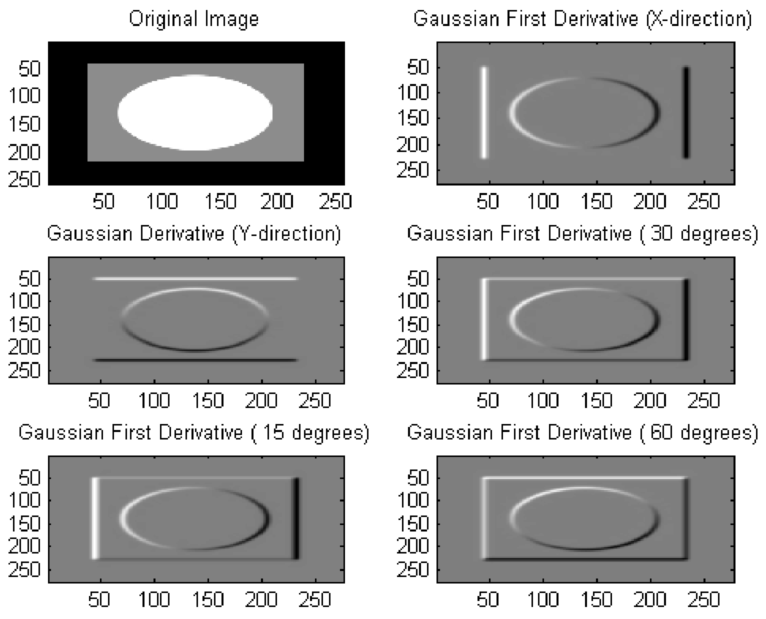
$$I_o^{\theta^\circ} = (G_1^{0^\circ} * I)cos(\theta) + (G_1^{90^\circ} * I)sin(\theta)$$

or,

$$I_o^{\theta^\circ} = I_o^{0^\circ} cos(\theta) + I_o^{90^\circ} sin(\theta)$$

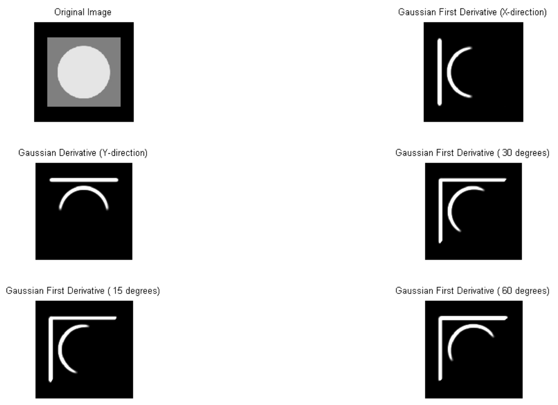
Convolution of a simple symmetric image with different values of θ is shown below.

Fig. 5.13 An Image convolved with oriented Gaussian first derivative filters



The same plot with an edge detection(gradient computation) point of view is also shown below.

Fig. 5.14 Edge Detection:Image convolved with oriented Gaussian first derivative filters a) original Image, oriented b) in x-direction, c) in y-direction, d) at 30°, e) 15°, f) 60°


Original Image
Gaussian First Derivative (X-direction)
Gaussian Derivative (Y-direction)
Gaussian First Derivative (30 degrees)
Gaussian First Derivative (15 degrees)
Gaussian First Derivative (60 degrees)

5.2.1.1 Adelson and Freeman Steerability Theorems

Adelson and Freeman proved three theorems in their 1991 paper and these theorems form the basis of Steerable filters and wavelets. A function $f(x,y)$ steers given the steering condition

$$f^{\theta}(x,y) = \sum_{j=1}^{M} k_j(\theta) f^{\theta_j}(x,y)$$

where $f^{\theta_j}(x,y)$ are M basis functions and $k_j(\theta)$ are steer coefficients for a given orientation θ. $f(x,y)$ is usually represented in polar coordinates as $f(r,\phi)$ where $r = \sqrt{x^2+y^2}$ and $\phi = tan^{-1}(y/x)$.

Theorem 1 : Steering condition holds if and only if $k_j(\theta)$ are solutions of following equations

$$\begin{pmatrix} 1 \\ e^{i\theta} \\ . \\ . \\ e^{iN\theta} \end{pmatrix} = \begin{bmatrix} 1 & 1 & . & 1 \\ e^{i\theta_1} & e^{i\theta_2} & . & e^{i\theta_M} \\ . & . & . & . \\ . & . & . & . \\ e^{iN\theta_1} & e^{iN\theta_2} & . & e^{iN\theta_M} \end{bmatrix} \begin{pmatrix} k_1(\theta) \\ k_2(\theta) \\ . \\ . \\ k_M(\theta) \end{pmatrix}$$

The polar coordinate representation of steerable function can be expanded as

$$f(r,\phi) = \sum_{n=-N}^{N} a_n(r) e^{in\phi}$$

Theorem 2 : Let T be the number of positive or negative frequencies for which $a_n(r) \neq 0$ then the number of basis filters M must be $>= T$.

Example- let us consider the first derivative of a Gaussian. It can be written in polar coordinates form as

$$G_1^{0^\circ} = -2rcos(\phi)e^{-r^2}$$

or,

$$G_1^{0^\circ} = -re^{-r^2}(e^{i\phi} + e^{-i\phi})$$

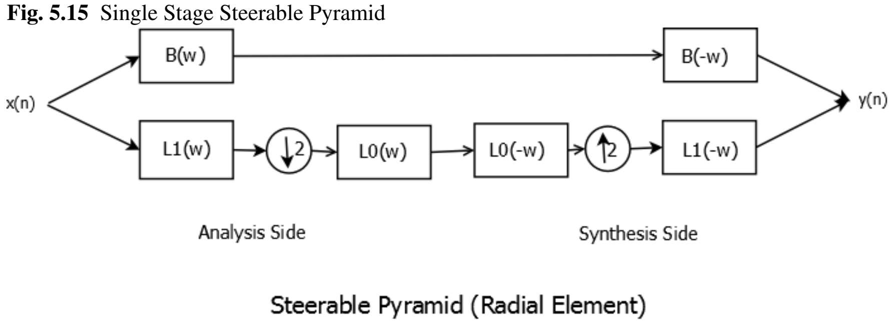
which means that $T = 2$ and we need $M \geq 2$ basis filters to steer the Gaussian first derivative.

Theorem 3 : Let $f(x,y) = W(r)P_N(x,y)$, where $W(r)$ is any windowing function and $P_N(x,y)$ is an Nth order polynomial in x and y then $f(x,y)$ can be steered in any direction using $2N+1$ basis filters.

5.2.2 Steerable Pyramid

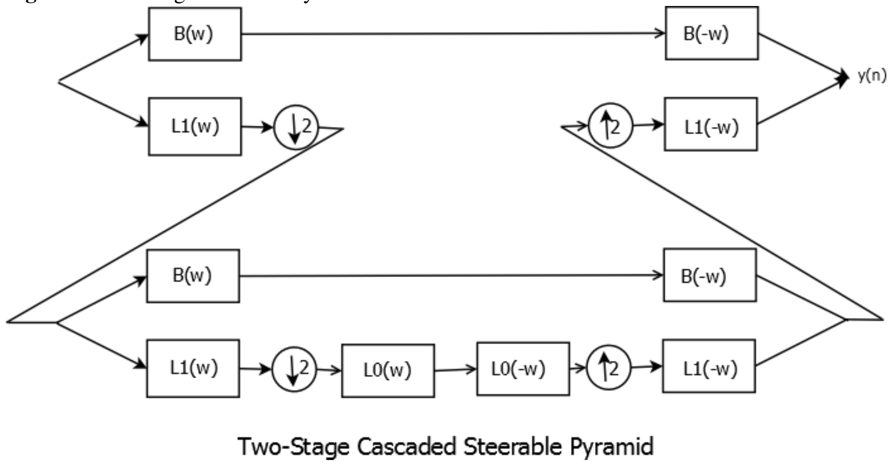
Steerable filter banks are implemented as pyramids. The implementation is done in two steps- the radial element(Pyramid) and the angular implementation which adds orientation to band pass filters. The implementation is in some ways similar to wavelet filter bank implementation with certain key differences. We resolve pyramid into a pass band and a cascade of low pass band filters with subsampling present only in the low pass cascade. The overall pyramid response is low pass. A single-stage Steerable Pyramid is shown below.

Fig. 5.15 Single Stage Steerable Pyramid



$B(\omega)$ is the band pass filter while $L_1(\omega)$ and $L_0(\omega)$ are the low pass filters. Band Pass filters are not subsampled in order to prevent aliasing while low pass filters are designed so that there is no aliasing ($|L(\omega)| = 0, \forall\omega \geq \pi/2$). Angular frequency computations depend on number of orientation bands needed. For example, if four orientation bands are used then the band pass filter is designed and rotated in four directions and each orientation θ is essentially a weighted linear combination of four basis filters. A two-stage Pyramid implementation is shown below. Decomposition is carried out on the low pass branch. It is assumed that system response (which is low pass) is actually $L_0(\omega)$ which helps in designing low pass and band pass filters.

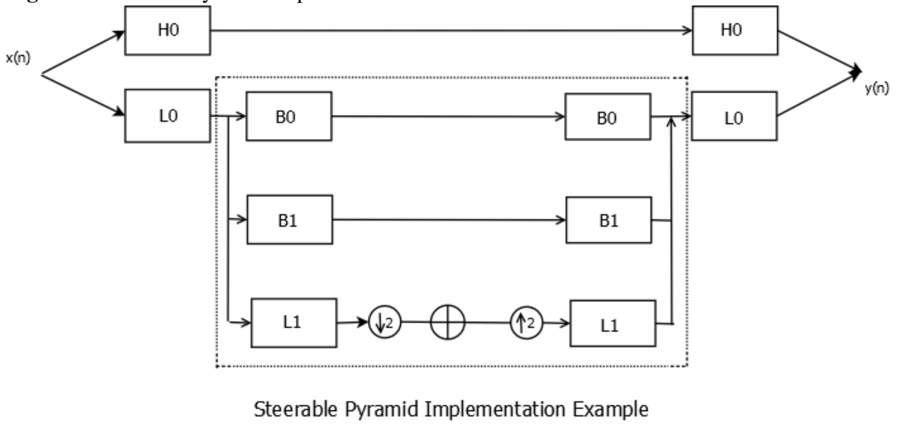
Fig. 5.16 Two-Stage Steerable Pyramid



5.2.3 Steerable Pyramid Implementation

As an example of Steerable Pyramid implementation we will consider the pyramid shown below which was proposed by Simoncelli, et al. An image is pre-processed by filtering it along two channels - one high pass and the other low pass. The decomposition is done on the low pass image by further filtering it with oriented band pass filters. The number of band pass filters k is determined by the desired number of orientation bands. The cascaded pyramid structure is implemented by replacing $\oplus$ in the low pass band by the entire structure contained within the dotted lines.

Fig. 5.17 Steerable Pyramid Implementation



5.2.3.1 Filter Constraints

1. Perfect Reconstruction

a) Unity System Response Amplitude

$$|L_0(\overline{\varpi})|^2[|L_1(\overline{\varpi})|^2+\sum_{n=0}^{k-1}|B_n(\overline{\varpi})|^2]+|H_0(\overline{\varpi})|^2=1$$

b) Cascaded Structure must preserve the prefect reconstruction condition and the low pass nature of the lower branch which gives us following equation

$$|L_1(\overline{\omega/2})|^2[|L_1(\overline{\varpi})|^2+\sum_{n=0}^{k-1}|B_n(\overline{\varpi})|^2]=|L_1(\overline{\omega/2})|^2$$

c) As mentioned earlier, low pass filters must be designed to prevent aliasing as we use subsampling in low pass branch.

$$L_0(\overline{\varpi})=0$$

for all $|\omega|>\pi/2$

2. Orientation : Oriented Band Pass filters are given by

$$B_n(\overline{\varpi})=B(\overline{\varpi})(-jcos(\theta-\theta_n))^{k-1}$$

where $\theta=\arg(\overline{\varpi})$, $\theta_n=\frac{\pi n}{k}$ and $n=0,1,...,k-1$

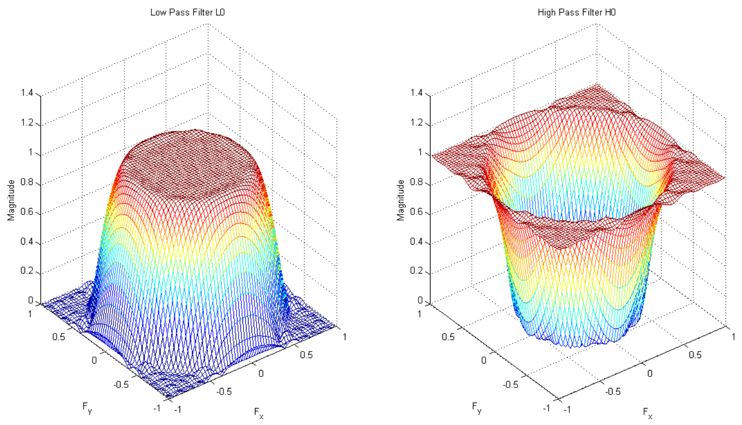
Filters Design

$2D$ Filters for Steerable Pyramid can designed using techniques proposed by Karasaridis et al. and then by Castleman et al.$L_1(\overline{\varpi})$ is designed as a $1D$ raised cosine filter. We then choose $L_0(\overline{\varpi})=L_1(\overline{\omega/2})$ and band pass filters are obtained using perfect reconstruction and angular constraints. These filters are converted to $2D$ filters using McClellan Frequency Transformation outlined in Jae Lim's book. The chosen filters are all $17X17$ filters.

5.2.4 Steerable Pyramid for $k = 2$ Orientations

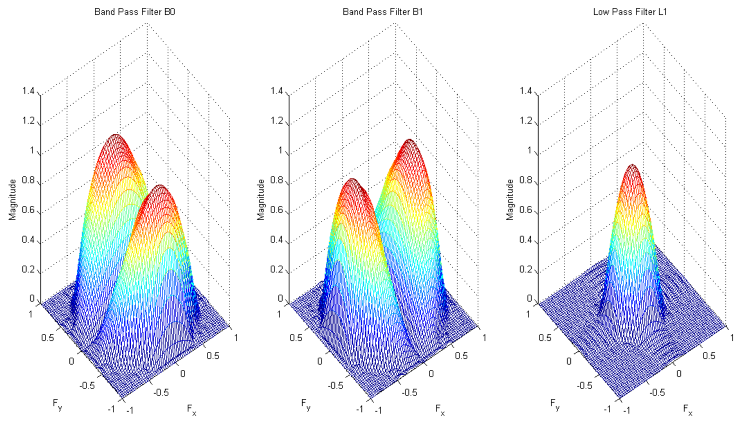
As can be seen from the Steerable Pyramid figure we need 5 filters to implement $k = 2$ orientation pyramid. The first two filters L_0 and H_0 are shown below. These two are also known as pre-processing filters(or post-processing filters at the reconstruction stage).

Fig. 5.18 Steerable Pyramid Pre-Processing Filters L_0 and H_0



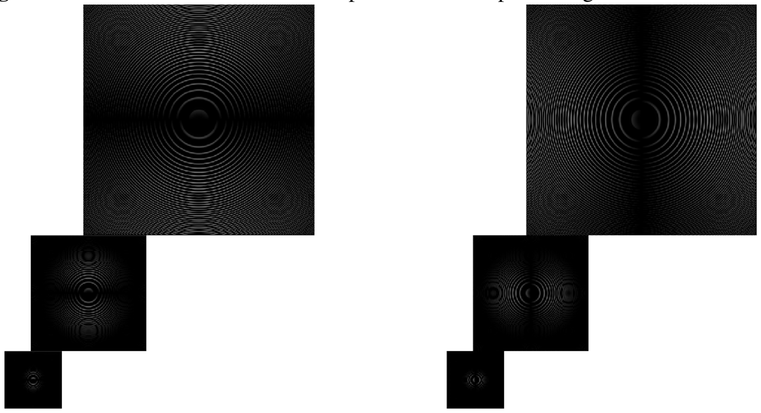
The next set of filters are known also known as Iterated Filters as they are part of the cascaded stage of the Pyramid. We add N cascades for N level of decomposition. $N = 3$ is used in this example. The three filters are the low pass filter L_1 and two oriented band pass filters B_0 and B_1.

Fig. 5.19 Steerable Pyramid Iterated Filters B_0, B_1 and L_1



This pyramid is used to decompose "Zoneplate" image at three scales. It can be seen that the images at all three scales are oriented in two different directions.

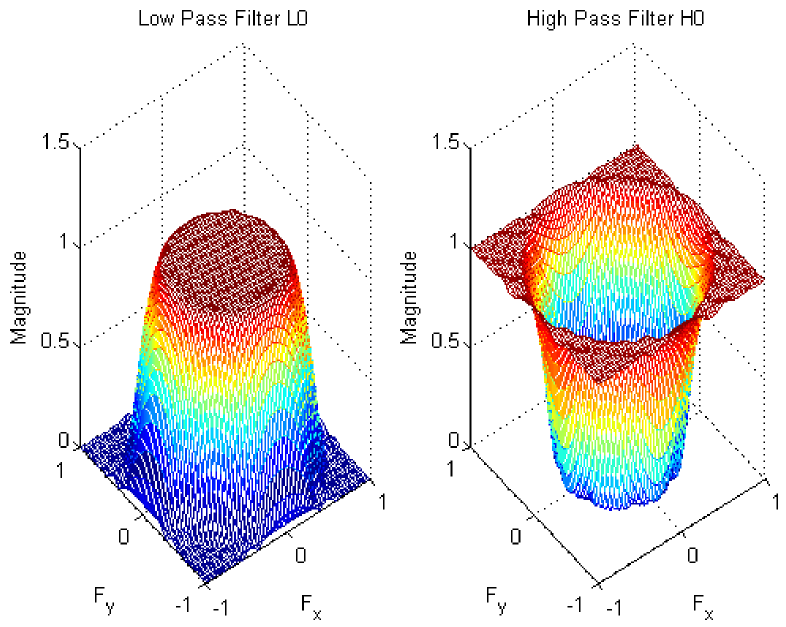
Fig. 5.20 3 Level $k = 2$ Oriented Decomposition of Zoneplate Image.



5.2.5 Steerable Pyramid for $k = 3$ *Orientations*

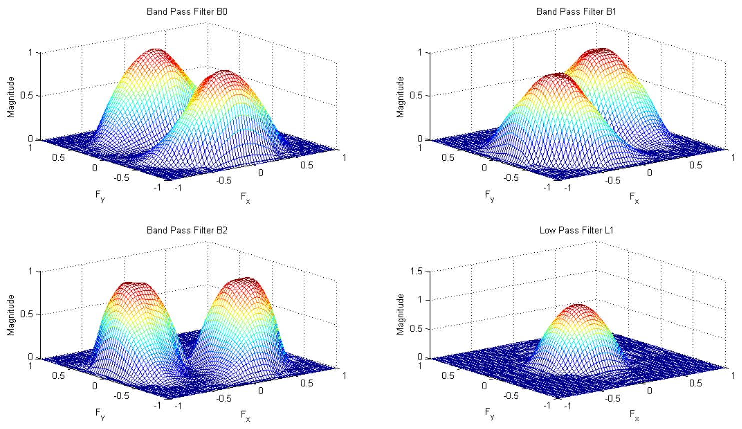
The only difference between $k = 2$ and $k = 3$ or ,indeed, any other value of k is that we have to design k different abnd pass filters alongside two low pass and one high pass filters. Pre-processing and post-processing filters L_0 and H_0 filters are designed the same way and are shown below.

Fig. 5.21 Steerable Pyramid Pre-Processing Filters L_0 and H_0



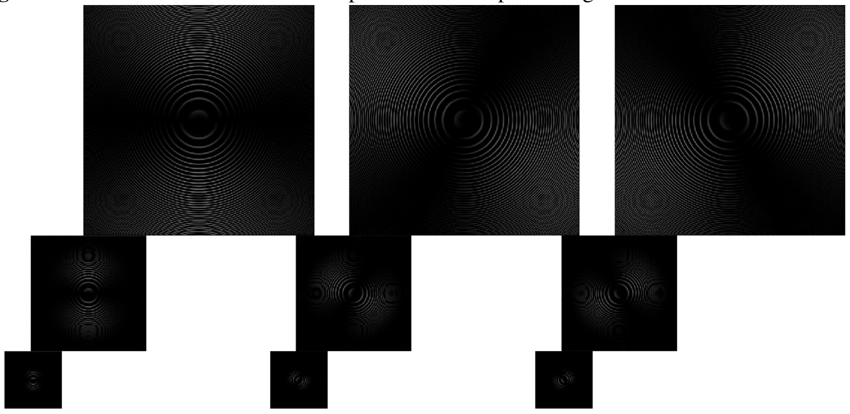
As mentioned earlier, we will have to design three oriented band pass filters in this case. These three filters along with the iterated low pass filter L_0 are shown in the following figure

Fig. 5.22 Steerable Pyramid Iterated Filters B_0, B_1, B_2 and L_1



This Steerable Pyramid is used to decompose "Zoneplate" image at three scales. It can be seen that the images at all three scales are oriented in three different directions.

Fig. 5.23 3 Level $k = 3$ Oriented Decomposition of Zoneplate Image.



5.3 Multiwavelets

5.3.1 MultiWavelet Dilation Equation

Multiwavelet Dilation(Refinement) Equation is given by

$$\overline{\phi}(x) = \sqrt{M} \sum_k H_k \overline{\phi}(Mx - k)$$

$k \in Z$ where $\overline{\phi}(x)$ is a $rX1$ vector given by

$$\overline{\phi}(x) = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ . \\ . \\ \phi_r(x) \end{bmatrix}$$

and H_k is a rXr matrix. Additionally, r is known as Multiplicity and M is known as Dilation Factor. Scalar wavelets can be seen as a special case of Mutiwavelets with $M = 2$ and $r = 1$.

5.3.1.1 Orthogonality Condition

The scaling vector $\overline{\phi}(x)$ is orthogonal if

$$< \overline{\phi}(x), \overline{\phi}(x-k) >= \int \overline{\phi}(x) \overline{\phi}^*(x-k) dx = \delta_{0,k} I$$

where I is a rXr identity matrix.

Or, equivalently the orthogonality can be given in terms of H_k matrix

$$\sum_k H_k H_{2l+k}^T = \delta_{0,l} I$$

The equation above can be proved by substituting dilation equation into the inner product equation.

5.3.2 DGHM Multiscaling Functions

Donovan, Geronimo, Hardin and Massopust proposed DGHM wavelets that are defined by the following recursion coefficients (H_k matrices)

$$H_0 = \frac{1}{20\sqrt{2}} \begin{bmatrix} 12 & 16\sqrt{2} \\ -\sqrt{2} & -6 \end{bmatrix}$$

$$H_1 = \frac{1}{20\sqrt{2}} \begin{bmatrix} 12 & 0 \\ 9\sqrt{2} & 20 \end{bmatrix}$$

$$H_2 = \frac{1}{20\sqrt{2}} \begin{bmatrix} 0 & 0 \\ 9\sqrt{2} & -6 \end{bmatrix}$$

$$H_3 = \frac{1}{20\sqrt{2}} \begin{bmatrix} 0 & 0 \\ -\sqrt{2} & 0 \end{bmatrix}$$

DGHM Multiscaling functions are orthogonal as

$$\sum_{k=0}^{3} H_k H_{2l+k}^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

5.3.3 MultiWavelet Multiresolution Analysis

Orthogonal MultiWavelet MRA is identical to scalar wavelets MRA with following properties.

1. $V_j \subset V_{j+1}$ A function in subspace j is in all the finer subspaces. In other words, if we know a signal $f_j(x)$ at subspace V_j, we can obtain ts coarse approximation using MRA. Think of a signal being decomposed using an iterated chain of complementary low pass and high pass filters. At every step we obtain a low pass and high pass version of the signal from the previous step. However, the low pass signal in step two is contained in the signal from the step one.

2. $f(t) \in V_0 \Leftrightarrow f(x-k) \in V_0$ This is the translation (shift) invariant property of the subspace. A signal in a given subspace , if translated by $k \in Z$ is still in that subspace. This property is valid for all subspaces.

3. $f(x) \in V_j \Leftrightarrow f(2x) \in V_{j+1}$ This is the scale invariant property of the Multiresolution analysis. In frequency domain terms, $f(2t)$ contains $2X$ highest frequency compared to that contained in $f(t)$. Using iterated filter bank example with a low pass filter that halves the frequencies in every step, it becomes clear that moving back one step in each step of the filter chain doubles the highest frequency content.

4. $\bigcap_{j\to-\infty} V_j = \{0\}$ As we move to lower subspaces, the space occupied by V_j shrinks until it becomes nearly zero.

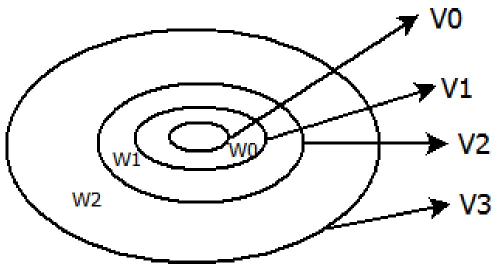
5.$\bigcup_{j\to\infty} V_j = L^2(R)$ Union of all subspaces as $j \to \infty$ encompasses the whole $L^2(R)$ space.

6. Multiwavelets: We define functions $\overline{\psi}^s(x)$ such that they are orthogonal to $\overline{\phi}(x)$ and to each other where $\overline{\psi}^s \in L^2$ and $s = 1, ..., M-1$. Given $k \in Z$, $\overline{\psi}^s(x-k)$ forms a stable basis of W_0.

Since each $\overline{\psi}^s \in V_1$, it can be written as

$$\overline{\psi}^s(x) = \sqrt{M} \sum_k G_k^s \overline{\phi}(Mx-k)$$

The functions $\overline{\psi}^s$ are called multiwavelet functions.



Three Level Multiresolution Analysis

Orthonormality Conditions

$$\sum G_k^s G_{Ml+k}^{t*} = \delta_{0l}\delta_{st} I$$

$$\sum G_k^s H_{Ml+k}^{*} = \sum H_k G_{Ml+k}^{s*} = 0$$

5.3.4 Discrete MultiWavelet Transform

Let Q_k give detail at level k while P_k give approximation at that level.

$$L^2 = \oplus W_n$$

Therefore, a function f can be given as summation of details at all levels.

$$f = \sum_{-\infty}^{\infty} Q_k f$$

$$L^2 = V_J \oplus W_j$$

where $j = J, ..., \infty$
or,

$$L^2 = \oplus_{-\infty}^{\infty} W_j$$

as

$$V_n = \oplus_{k=-\infty}^{n-1} W_k$$

We observe that $P_l f$ converges to f as $l \to \infty$. Let $P_n f$ be the projection of f in the space V_n and $Q_n f$ be the projection of f in the space W_n.

$$V_n + W_n = V_{n+1}$$

or,

$$V_n + W_n + W_{n+1} + \ldots + W_{n+l} = V_{n+1+l}$$

Therefore, $f = P_n f + \sum_{k=n}^{l} Q_k f$ for some $l > n$ is the Discrete MultiWavelet Transform (DMWT) and it is identical to DWT.

$$f = P_n f + \sum_{k=n}^{l} Q_k f$$

$$f = \sum_j a_{n,j} \overline{\phi}_{n,j} + \sum_j \sum_{s=1}^{M-1} d_{n,j}^s \overline{\psi}_{n,j}^s$$

where $a_{n,j}$ and $d_{n,j}^s$ are approximation and detail coefficients and identical to the DWT case.

5.3.5 Biorthogonal MRAs and MultiWavelets

As is the case with scalar wavelets, $\overline{\phi}$ and $\widetilde{\overline{\phi}}$ are scaling function vectors in the Synthesis and Analysis domain respectively.

5.3.5.1 Biorthogonality Condition

$$< \overline{\phi}(x-k), \widetilde{\overline{\phi}}(x-l) >= \delta_{lk} I$$

In terms of low pass symbol

$$\sum H_k \widetilde{H}_{k+Ml}^* = \delta_{0l} I$$

The two scaling equations are given by

$$\overline{\phi}(x) = \sqrt{M} \sum_k H_k \overline{\phi}(Mx - k)$$

$$\widetilde{\overline{\phi}}(x) = \sqrt{M} \sum_k \widetilde{H}_k \widetilde{\overline{\phi}}(Mx - k)$$

Projections $P_n f$ and $\widetilde{P_n f}$ of a L^2 function f onto V_n and $\widetilde{V_n}$ are given as

$$P_n f = \sum_j \widetilde{a_{n,j}} \overline{\phi}_{n,j}$$

and

$$\widetilde{P_n} f = \sum_j a_{n,j} \widetilde{\overline{\phi}}_{n,j}$$

The approximation coefficients can be calculated as

$$\widetilde{a_{n,j}} =< f, \widetilde{\overline{\phi}}_{n,j} >= \int f \widetilde{\overline{\phi}}^{*}_{n,j} dx$$

$$a_{n,j} =< f, \overline{\phi}_{n,j} >= \int f \overline{\phi}^{*}_{n,j} dx$$

Therefore,

$$P_n f =< f, \widetilde{\overline{\phi}}_{n,j} > \overline{\phi}_{n,j}$$

and

$$\widetilde{P}_n f =< f, \overline{\phi}_{n,j} > \widetilde{\overline{\phi}}_{n,j}$$

The projections $Q_n f$ and $\widetilde{Q_n} f$ are defined as

$$Q_n f = P_{n+1} f - P + nf$$

$$\widetilde{Q_n} f = \widetilde{P_{n+1}} f - \widetilde{P}_n f$$

and the subspaces are non-orthogonal direct sums given by

$$V_n + W_n = V_{n+1}$$

$$\widetilde{V}_n + \widetilde{W}_n = \widetilde{V}_{n+1}$$

Biorthogonality Conditions as applied to subspaces in multiwavelet case are same as in the scalar case.

$$V_n \perp \widetilde{V}_n$$

$$V_n \perp \widetilde{W}_n$$

$$W_n \perp \widetilde{V}_n$$

and

$$W_n \perp \widetilde{W}_n$$

if $k \neq n$

Discrete MultiWavelet Transform (DWMT) in this case is same as that in scalar biorthogonal DWT case. Biorthogonalty conditions in terms of low pass and high pass symbols are given as

$$\sum H_k \widetilde{H}^{*}_{k+Ml} = \delta_{0l} I$$

$$\sum G_k^s \widetilde{G}_{Ml+k}^{t*} = \delta_{0l} \delta_{st} I$$

$$\sum G_k^s \widetilde{H}_{Ml+k}^{*} = \sum H_k \widetilde{G}_{Ml+k}^{s*} = 0$$

5.3.6 Pre-processing and Post-processing

Discrete MultiWavelet Transform (DMWT) is given by

$$f = \sum_j a_{n,j} \overline{\phi}_{n,j} + \sum_j \sum_{s=1}^{M-1} d_{n,j}^s \overline{\psi}_{n,j}^s$$

Like DWT, DMWT requires samples $a_{n+1,j}$ to start decomposition which we don't have. Instead , we have sampled versions $f(x)$ of signal $f(t)$.

Preprocessing is the process of getting $a_{n+1,j}$ from the samples f.

$$a_{n+1,j} = < f, \phi_{n+1,j} >$$

Similarly, postprocessing is needed at the reconstruction stage in order to obtain reconstructed signal from the IDMWT outputs.

5.4 Wavelet Meshes

5.5 Monogenic Wavelet Transform

5.6 More Topics

References

1. S Mallat (1999) A Wavelet Tour of Signal Processing. Academic Press
2. G Strang, T Nguyen(1996) Wavelets and Filter Banks. Wellesley Cambridge Press
3. M Vetterli, J Kovacevic(1995) Wavelets and Subband Coding. Prentice Hall Signal Processing Series
4. I Daubechies (1992) Ten Lectures on Wavelets. SIAM: Society for Industrial and Applied Mathematics
5. M Weeks (2006) Digital Signal Processing using Matlab and Wavelets. Infinity Science Press
6. V Goyal, M Vetterli, J Kovacevic (2010) Fourier and Wavelet Signal Processing. Manuscript Draft

Appendix A
Software and Scripts

All's well that ends well

A.1 Matlab Codes

A.2 Octave Implementations

A.2.1 About Octave

A.2.2 Octave Scripts

Glossary

Currently Left Blank