Chapter 3

The discrete wavelet transform

3.1 Introduction

In this chapter we consider the discrete wavelet transform (DWT). We will see that when certain criteria are met it is possible to completely reconstruct the original signal using infinite summations of discrete wavelet coefficients rather than continuous integrals (as required for the CWT). This leads to a fast wavelet transform for the rapid computation of the discrete wavelet transform and its inverse. We will then see how to perform a discrete wavelet transform on discrete input signals of finite length: the kind of signal we might be presented with in practice. We will also consider briefly biorthogonal wavelets, which come in pairs, and some space is devoted to two-dimensional discrete wavelet transforms. The chapter ends with wavelet packets: a generalization of the discrete wavelet transform which allows for adaptive partitioning of the time–frequency plane.

3.2 Frames and orthogonal wavelet bases

3.2.1 Frames

In chapter 2, the wavelet function was defined at scale \( a \) and location \( b \) as

\[
\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left( \frac{t - b}{a} \right) \quad (3.1)
\]

In this section the wavelet transform of a continuous time signal, \( x(t) \), is considered where discrete values of the dilation and translation parameters, \( a \) and \( b \), are used.

A natural way to sample the parameters \( a \) and \( b \) is to use a logarithmic discretization of the \( a \) scale and link this, in turn, to the size of steps taken between \( b \) locations. To link \( b \) to \( a \) we move in discrete steps to each location \( b \) which are proportional to the \( a \) scale. This kind of discretization of the wavelet has the form

\[
\psi_{m,n}(t) = \frac{1}{\sqrt{d_0^n}} \psi\left( \frac{t - nb_0a_0^m}{a_0^m} \right) \quad (3.2)
\]

Copyright © 2002 IOP Publishing Ltd.
where the integers \( m \) and \( n \) control the wavelet dilation and translation respectively; \( a_0 \) is a specified fixed dilation step parameter set at a value greater than 1, and \( b_0 \) is the location parameter which must be greater than zero. The control parameters \( m \) and \( n \) are contained in the set of all integers, both positive and negative. It can be seen from the above equation that the size of the translation steps, \( \Delta b = b_0 a_0^m \), is directly proportional to the wavelet scale, \( a_0^m \).

The wavelet transform of a continuous signal, \( x(t) \), using discrete wavelets of the form of equation (3.2) is then

\[
T_{m,n} = \int_{-\infty}^{\infty} x(t) \frac{1}{a_0^{m/2}} \psi(a_0^{-m} t - n b_0) \, dt \tag{3.3a}
\]

which can also be expressed as the inner product

\[
T_{m,n} = \langle x, \psi_{m,n} \rangle \tag{3.3b}
\]

where \( T_{m,n} \) are the discrete wavelet transform values given on a scale–location grid of index \( m, n \). For the discrete wavelet transform, the values \( T_{m,n} \) are known as wavelet coefficients or detail coefficients. These two terms are used interchangeably in this chapter as they are in the general wavelet literature. To determine how ‘good’ the representation of the signal is in wavelet space using this decomposition, we can resort to the theory of wavelet frames which provides a general framework for studying the properties of discrete wavelets. Wavelet frames are constructed by discretely sampling the time and scale parameters of a continuous wavelet transform as we have done above. The family of wavelet functions that constitute a frame are such that the energy of the resulting wavelet coefficients lies within a certain bounded range of the energy of the original signal, i.e.

\[
AE \leq \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} |T_{m,n}|^2 \leq BE \tag{3.4}
\]

where \( T_{m,n} \) are the discrete wavelet coefficients, \( A \) and \( B \) are the frame bounds, and \( E \) is the energy of the original signal given by equation (2.19) in chapter 2: \( E = \int_{-\infty}^{\infty} |x(t)|^2 \, dt = \| x(t) \|_2^2 \), where our signal, \( x(t) \), is defined to have finite energy. The values of the frame bounds \( A \) and \( B \) depend upon both the parameters \( a_0 \) and \( b_0 \) chosen for the analysis and the wavelet function used. (For details of how to determine \( A \) and \( B \) see Daubechies (1992).) If \( A = B \) the frame is known as ‘tight’. Such tight frames have a simple reconstruction formula given by the infinite series

\[
x(t) = \frac{1}{A} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} T_{m,n} \psi_{m,n}(t) \tag{3.5}
\]

A tight frame with \( A (=B) > 1 \) is redundant, with \( A \) being a measure of the redundancy. However, when \( A = B = 1 \) the wavelet family defined by the frame forms an orthonormal basis. If \( A \) is not equal to \( B \) a reconstruction formula can still be written:

\[
x'(t) = \frac{2}{A + B} \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} T_{m,n} \psi_{m,n}(t) \tag{3.6}
\]
where \( x'(t) \) is the reconstruction which differs from the original signal \( x(t) \) by an error which depends on the values of the frame bounds. The error becomes acceptably small for practical purposes when the ratio \( B/A \) is near unity. It has been shown, for the case of the Mexican hat wavelet, that if we use \( a_0 = 2^{1/v} \), where \( v \geq 2 \) and \( b_0 \leq 0.5 \), the frame is nearly tight or 'snug' and for practical purposes it may be considered tight. (This fractional discretization, \( v \), of the power-of-two scale is known as a voice.) For example, setting \( a_0 = 2^{1/2} \) and \( b_0 = 0.5 \) for the Mexican hat leads to \( A = 13.639 \) and \( B = 13.673 \) and the ratio \( B/A \) equals 1.002. The closer this ratio is to unity, the tighter the frame. Thus discretizing a Mexican hat wavelet transform using these scale and location parameters results in a highly redundant representation of the signal but with very little difference between \( x(t) \) and \( x'(t) \). The nearly tight Mexican hat wavelet frame with these parameters \( (a_0 = 2^{1/2} \) and \( b_0 = 0.5 \) is shown in figure 3.1 for two consecutive scales \( m \) and \( m+1 \) and at three consecutive locations, \( n = 0, 1 \) and 2.

### 3.2.2 Dyadic grid scaling and orthonormal wavelet transforms

Common choices for discrete wavelet parameters \( a_0 \) and \( b_0 \) are 2 and 1 respectively. This power-of-two logarithmic scaling of both the dilation and translation steps is known as the dyadic grid arrangement. The dyadic grid is perhaps the simplest and most efficient discretization for practical purposes and lends itself to the construction
of an orthonormal wavelet basis. Substituting $a_0 = 2$ and $b_0 = 1$ into equation (3.2), we see that the dyadic grid wavelet can be written as

$$\psi_{m,n}(t) = \frac{1}{\sqrt{2^m}} \psi \left( \frac{t - n2^n}{2^m} \right)$$  \hspace{1cm} (3.7a)

or, more compactly, as

$$\psi_{m,n}(t) = 2^{-m/2} \psi(2^{-m}t - n)$$  \hspace{1cm} (3.7b)

Note that this has the same notation as the general discrete wavelet given by equation (3.2). From here on in this chapter we will use $\psi_{m,n}(t)$ to denote only dyadic grid scaling with $a_0 = 2$ and $b_0 = 1$.

Discrete dyadic grid wavelets are commonly chosen to be orthonormal. These wavelets are both orthogonal to each other and normalized to have unit energy. This is expressed as

$$\int_{-\infty}^{\infty} \psi_{m,n}(t) \psi_{m',n'}(t) \, dt = \begin{cases} 1 & \text{if } m = m' \text{ and } n = n' \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (3.8)

That is to say, the product of each wavelet with all others in the same dyadic system (i.e. those which are translated and/or dilated versions of each other) are zero. This means that the information stored in a wavelet coefficient $T_{m,n}$ is not repeated elsewhere and allows for the complete regeneration of the original signal without redundancy. In addition to being orthogonal, orthonormal wavelets are normalized to have unit energy. This can be seen from equation (3.8) as, when $m = m'$ and $n = n'$, the integral gives the energy of the wavelet function equal to unity. Orthonormal wavelets have frame bounds $A = B = 1$ and the corresponding wavelet family is an orthonormal basis. (A basis is a set of vectors, a combination of which can completely define the signal, $x(t)$. An orthonormal basis has component vectors which, in addition to being able to completely define the signal, are perpendicular to each other.) The discrete dyadic grid wavelet lends itself to a fast computer algorithm, as we shall see later.

Using the dyadic grid wavelet of equation (3.7a), the discrete wavelet transform (DWT) can be written as:

$$T_{m,n} = \int_{-\infty}^{\infty} x(t) \psi_{m,n}(t) \, dt$$  \hspace{1cm} (3.9)

By choosing an orthonormal wavelet basis, $\psi_{m,n}(t)$, we can reconstruct the original signal in terms of the wavelet coefficients, $T_{m,n}$, using the inverse discrete wavelet transform as follows:

$$x(t) = \sum_{m=\infty}^{\infty} \sum_{n=\infty}^{\infty} T_{m,n} \psi_{m,n}(t)$$  \hspace{1cm} (3.10a)

requiring the summation over all integers $m$ and $n$. This is actually equation (3.5), with $A = 1$ due to the orthonormality of the chosen wavelet. Equation (3.10a) is often seen
written in terms of the inner product
\[ x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \langle x, \psi_{m,n} \rangle \psi_{m,n}(t) \]  
(3.10b)

where the combined decomposition and reconstruction processes are clearly seen: going from \( x(t) \) to \( T_{m,n} \) via the inner product \( \langle x, \psi_{m,n} \rangle \) then back to \( x(t) \) via the infinite summations. In addition, as \( A = B \) and \( A = 1 \), we can see from equation (3.4) that the energy of the signal may be expressed as
\[ \int_{-\infty}^{\infty} |x(t)|^2 \, dt = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} |T_{m,n}|^2 \]  
(3.11)

Before continuing it is important to make clear the distinct difference between the DWT and the discretized approximations of the CWT covered in chapter 2. The discretization of the continuous wavelet transform, required for its practical implementation, involves a discrete approximation of the transform integral (i.e. a summation) computed on a discrete grid of \( a \) scales and \( b \) locations. The inverse continuous wavelet transform is also computed as a discrete approximation. How close an approximation to the original signal is recovered depends mainly on the resolution of the discretization used and, with care, usually a very good approximation can be recovered. On the other hand, for the DWT, as defined in equation (3.9), the transform integral remains continuous but is determined only on a discretized grid of \( a \) scales and \( b \) locations. We can then sum the DWT coefficients (equation (3.10a)) to get the original signal back exactly. We will see later in this chapter how, given an initial discrete input signal, which we treat as an initial approximation to the underlying continuous signal, we can compute the wavelet transform and inverse transform discretely, quickly and without loss of signal information.

3.2.3 The scaling function and the multiresolution representation

Orthonormal dyadic discrete wavelets are associated with scaling functions and their dilation equations. The scaling function is associated with the smoothing of the signal and has the same form as the wavelet, given by
\[ \phi_{m,n}(t) = 2^{-m/2} \phi(2^{-m} t - n) \]  
(3.12)

They have the property
\[ \int_{-\infty}^{\infty} \phi_{0,0}(t) \, dt = 1 \]  
(3.13)

where \( \phi_{0,0}(t) = \phi(t) \) is sometimes referred to as the father scaling function or father wavelet (cf. mother wavelet). (Remember from chapter 2 that the integral of a wavelet function is zero.) The scaling function is orthogonal to translations of itself, but not to dilations of itself. The scaling function can be convolved with the signal to produce approximation coefficients as follows:
\[ S_{m,n} = \int_{-\infty}^{\infty} x(t) \phi_{m,n}(t) \, dt \]  
(3.14)
From the last three equations, we can see that the approximation coefficients are simply weighted averages of the continuous signal factored by $2^{-m/2}$. The approximation coefficients at a specific scale $m$ are collectively known as the *discrete approximation* of the signal at that scale. A *continuous approximation* of the signal at scale $m$ can be generated by summing a sequence of scaling functions at this scale factored by the approximation coefficients as follows:

$$x_m(t) = \sum_{n=-\infty}^{\infty} S_{m,n} \phi_{m,n}(t)$$  \hspace{1cm} (3.15)

where $x_m(t)$ is a smooth, scaling-function-dependent, version of the signal $x(t)$ at scale index $m$. This continuous approximation approaches $x(t)$ at small scales, i.e. as $m \to -\infty$. Figure 3.2(a) shows a simple scaling function, a block pulse, at scale index 0 and location index 0: $\phi_{0,0}(t) = \phi(t)$ — the father function — together with two of its corresponding dilations at that location. It is easy to see that the convolution of the block pulse with a signal (equation (3.14)) results in a local weighted averaging of the signal over the nonzero portion of the pulse. Figure 3.2(b) shows one period of a sine wave, $x(t)$ contained within a window. Figure 3.2(c) shows various approximations of the sine wave generated using equations (3.14) and (3.15) with the scaling function set to a range of widths, $2^0$ to $2^7$. These widths are indicated by the vertical lines and arrows in each plot. Equation (3.14) computes the approximation coefficients $S_{m,n}$ which are, as mentioned above for this simple block scaling function, the weighted average of the signal over the pulse width. The approximation coefficients are then used in equation (3.15) to produce an approximation of the signal which is simply a sequence of scaling functions placed side by side, each factored by their corresponding approximation coefficient. This is obvious from the blocky nature of the signal approximations. The approximation at the scale of the window ($= 2^7$) is simply the average over the whole sine wave which is zero. As the scale decreases, the approximation is seen to approach the original waveform. This simple block pulse scaling function used in this example here is associated with the Haar wavelet, which we will come to shortly.

We can represent a signal $x(t)$ using a combined series expansion using both the approximation coefficients and the wavelet (detail) coefficients as follows:

$$x(t) = \sum_{n=-\infty}^{\infty} S_{m_0,n} \phi_{m_0,n}(t) + \sum_{m=-\infty}^{m_0} \sum_{n=-\infty}^{\infty} T_{m,n} \psi_{m,n}(t)$$  \hspace{1cm} (3.16)

We can see from this equation that the original continuous signal is expressed as a combination of an approximation of itself, at arbitrary scale index $m_0$, added to a succession of signal details from scales $m_0$ down to negative infinity. The *signal detail* at scale $m$ is defined as

$$d_m(t) = \sum_{n=-\infty}^{\infty} T_{m,n} \psi_{m,n}(t)$$  \hspace{1cm} (3.17)

hence we can write equation (3.16) as

$$x(t) = x_{m_0}(t) + \sum_{m=-\infty}^{m_0} d_m(t)$$  \hspace{1cm} (3.18)
From this equation it is easy to show that
\[
\frac{x_m}{C^2_{m_1}}(t) = \frac{x_m}{C^2_{m_{m_1}}}(t) + d_{m}(t)
\]
which tells us that if we add the signal detail at an arbitrary scale (index \(m\)) to the approximation at that scale we get the signal approximation at an increased resolution (i.e. at a smaller scale, index \(m - 1\)). This is called a multiresolution representation.

---

**Figure 3.2. Smooth approximation of a sine wave using a block pulse scaling function.** (a) Simple block scaling function shown at scale 1 (scale index \(m = 0\)) and location \(n = 0\), i.e. \(\phi_{0,0}(t)\) (shown bold), together with its dilations at that location. (b) Sine wave of one period. (c) Selected smooth approximations, \(x_m(t)\), of the sine wave at increasing scales. The width of one of the scaling functions \(\phi_{m,n}(t)\) at scale index \(m\) is depicted in each figure by the arrows. Note that the example has been set up so that one period of the sinusoid corresponds to scale \(2^7\). Note also that at small scales \(x_m(t)\) approaches \(x(t)\).
3.2.4 The scaling equation, scaling coefficients and associated wavelet equation

The scaling equation (or dilation equation) describes the scaling function \( \phi(t) \) in terms of contracted and shifted versions of itself as follows:

\[
\phi(t) = \sum_k c_k \phi(2t - k) \tag{3.20}
\]

where \( \phi(2t - k) \) is a contracted version of \( \phi(t) \) shifted along the time axis by an integer step \( k \) and factored by an associated scaling coefficient, \( c_k \). (Take note of the similar but different terminology—scaling equation and scaling function.) Equation (3.20) basically tells us that we can build a scaling function at one scale from a number of scaling equations at the previous scale. The solution to this two-scale difference equation gives the scaling function \( \phi(t) \). For the sake of simplicity in the rest of the chapter we concern ourselves only with wavelets of compact support. These have sequences of nonzero scaling coefficients which are of finite length. Integrating both sides of the above equation, we can show that the scaling coefficients must satisfy the following constraint:

\[
\sum_k c_k = 2 \tag{3.21}
\]

In addition, in order to create an orthogonal system we require that

\[
\sum_k c_k c_{k+2k'} = \begin{cases} 
2 & \text{if } k' = 0 \\
0 & \text{otherwise} 
\end{cases} \tag{3.22}
\]

This also tells us that the sum of the squares of the scaling coefficients is equal to 2. The same coefficients are used in reverse with alternate signs to produce the differencing of the associated wavelet equation, i.e.

\[
\psi(t) = \sum_k (-1)^k c_{1 - k} \phi(2t - k) \tag{3.23}
\]

This construction ensures that the wavelets and their corresponding scaling functions are orthogonal. This wavelet equation is commonly seen in practice. In this chapter, however, we will consider only wavelets of compact support which have a finite number of scaling coefficients, \( N_k \). For this case we can define the wavelet function as

\[
\psi(t) = \sum_k (-1)^k c_{N_k - 1 - k} \phi(2t - k) \tag{3.24}
\]

This ordering of scaling coefficients used in the wavelet equation allows for our wavelets and their corresponding scaling equations to have support over the same interval \([0, N_k - 1]\). (The ordering of equation (3.23) leads to wavelet and scaling functions displaced from each other, except for the Haar wavelet where \( N_k = 2 \).) Note that, if the number of scaling coefficients is not finite, we cannot use this reordering and must revert back to an ordering of the type given by equation (3.23). We will stick to the ordering specified by equation (3.24) in this text.

Often the reconfigured coefficients used for the wavelet function are written more compactly as

\[
b_k = (-1)^k c_{N_k - 1 - k} \tag{3.25}
\]
where the sum of all the coefficients $b_k$ is zero. Using this reordering of the coefficients, equation (3.24) can be written as

$$\psi(t) = \sum_{k=0}^{N-1} b_k \phi(2t - k)$$  \hspace{1cm} (3.26)

From equations (3.12) and (3.20) and examining the wavelet at scale index $m + 1$, we can see that for arbitrary integer values of $m$ the following is true:

$$2^{- (m+1)/2} \phi\left(\frac{t}{2^{m+1}} - n\right) = 2^{-m/2} 2^{-1/2} \sum_k c_k \phi\left(\frac{2t}{2 \times 2^m} - 2n - k\right)$$ \hspace{1cm} (3.27a)

which may be written more compactly as

$$\phi_{m+1,n}(t) = \frac{1}{\sqrt{2}} \sum_k c_k \phi_{m,2n+k}(t)$$  \hspace{1cm} (3.27b)

That is, the scaling function at an arbitrary scale is composed of a sequence of shifted scaling functions at the next smaller scale each factored by their respective scaling coefficients. Similarly, for the wavelet function we obtain

$$\psi_{m+1,n}(t) = \frac{1}{\sqrt{2}} \sum_k b_k \phi_{m,2n+k}(t)$$  \hspace{1cm} (3.28)

### 3.2.5 The Haar wavelet

The Haar wavelet is the simplest example of an orthonormal wavelet. Its scaling equation contains only two nonzero scaling coefficients and is given by

$$\phi(t) = \phi(2t) + \phi(2t - 1)$$ \hspace{1cm} (3.29)

that is, its scaling coefficients are $c_0 = c_1 = 1$. We get these coefficient values by solving equations (3.21) and (3.22) simultaneously. (From equation (3.21) we see that $c_0 + c_1 = 2$ and from equation (3.22) $c_0c_0 + c_1c_1 = 2$.) The solution of the Haar scaling equation is the single block pulse shown in figure 3.3(a) and defined as

$$\psi(t) = \begin{cases} 
1 & 0 \leq t < 1 \\
0 & \text{elsewhere}
\end{cases}$$ \hspace{1cm} (3.30)

This is, in fact, the scaling function used earlier in figure 3.2 to generate the signal approximations of the sine wave. Reordering the coefficient sequence according to equation (3.25) we can see that the corresponding Haar wavelet equation is

$$\psi(t) = \phi(2t) - \phi(2t - 1)$$ \hspace{1cm} (3.31)

The Haar wavelet is shown in figure 3.3(b) and defined as

$$\psi(t) = \begin{cases} 
1 & 0 \leq t < \frac{1}{2} \\
-1 & \frac{1}{2} \leq t < 1 \\
0 & \text{elsewhere}
\end{cases}$$ \hspace{1cm} (3.32)
The mother wavelet for the Haar wavelet system, \( \phi(t) \), is formed from two dilated unit block pulses sitting next to each other on the time axis, with one of them inverted. From the mother wavelet we can construct the Haar system of wavelets on a dyadic grid, \( \phi(m, n)(t) \). This is illustrated in figure 3.3(c) for three consecutive scales. The mother wavelet for the Haar wavelet system, \( \psi(t) = \psi_{0,0}(t) \), is formed from two dilated unit block pulses sitting next to each other on the time axis, with one of them inverted. From the mother wavelet we can construct the Haar system of wavelets on a dyadic grid, \( \psi_{m,n}(t) \). This is illustrated in figure 3.3(c) for three consecutive scales.

Figure 3.3. Discrete orthonormal wavelets. (a) The Haar scaling function in terms of shifted and dilated versions of itself. (b) The Haar wavelet in terms of shifted and dilated versions of its scaling function. (c) Three consecutive scales shown for the Haar wavelet family specified on a dyadic grid, e.g. from top to bottom: \( \psi_{m,n}(t) \), \( \psi_{m+1,n}(t) \) and \( \psi_{m+2,n}(t) \). (d) Three Haar wavelets at three consecutive scales on a dyadic grid. (e) Three Haar wavelets at different scales. This time the Haar wavelets are not defined on a dyadic grid and are hence not orthogonal to each other.
orthogonal nature of the family of Haar wavelets in a dyadic grid system is obvious from figure 3.3(d), where it can be seen that the positive and negative parts of the Haar wavelet at any scale coincide with a constant (positive or negative) part of the Haar wavelet at the next larger scale (and all subsequent larger scales). In addition, Haar wavelets at the same scale index \( m \) on a dyadic grid do not overlap. Hence, it is obvious that the convolution of the Haar wavelet with any others in the same dyadic grid gives zero. Figure 3.3(e) shows three Haar wavelets which are not specified on a dyadic grid. The non-orthogonal nature of the Haar wavelets across scales, when specified in this way, is obvious from the plot. In addition, if these wavelets overlap each other along each scale this also destroys orthogonality. (Although this is not the case for other orthonormal wavelets—see later.) Finally, note that the Haar wavelet is of finite width on the time axis; that is, it has compact support. As was stated in the last section, wavelets which have compact support have a finite number of scaling coefficients and these are the type of wavelet we concentrate on in this chapter. Not all orthonormal wavelets have compact support. Figure 3.3(f), for example, shows a Meyer wavelet which is orthonormal with infinite support, although, as with all wavelets, it is localized, decaying relatively quickly from its central peak.

### 3.2.6 Coefficients from coefficients: the fast wavelet transform

From equation (3.14) we can see that the approximation coefficients at scale index \( m + 1 \) are given by

\[ S_{m+1,n} = \int_{-\infty}^{\infty} x(t) \phi_{m+1,n}(t) \, dt \]  (3.33)

Using equation (3.27b) this can be written as

\[ S_{m+1,n} = \int_{-\infty}^{\infty} x(t) \left[ \frac{1}{\sqrt{2}} \sum_k c_k \phi_{m,2n+k}(t) \right] \, dt \]  (3.34)

We can rewrite this as

\[ S_{m+1,n} = \frac{1}{\sqrt{2}} \sum_k c_k \left[ \int_{-\infty}^{\infty} x(t) \phi_{m,2n+k}(t) \, dt \right] \]  (3.35)

Figure 3.3 (continued). (f) A Meyer wavelet and associated scaling function (right).
The integral in brackets gives the approximation coefficients \( S_{m,2n+k} \) for each \( k \). We can therefore write this equation as

\[
S_{m+1,n} = \frac{1}{\sqrt{2}} \sum_k c_k S_{m,2n+k} = \frac{1}{\sqrt{2}} \sum_k c_k - 2n S_{m,k} \quad (3.36)
\]

Hence, using this equation, we can generate the approximation coefficients at scale index \( m + 1 \) using the scaling coefficients at the previous scale.

Similarly the wavelet coefficients can be found from the approximation coefficients at the previous scale using the reordered scaling coefficients \( b_k \) as follows:

\[
T_{m+1,n} = \frac{1}{\sqrt{2}} \sum_k b_k S_{m,2n+k} = \frac{1}{\sqrt{2}} \sum_k b_k - 2n S_{m,k} \quad (3.37)
\]

We can see now that, if we know the approximation coefficients \( S_{m_0,n} \) at a specific scale \( m_0 \) then, through the repeated application of equations (3.36) and (3.37), we can generate the approximation and detail wavelet coefficients at all scales larger than \( m_0 \). Notice that, to do this, we do not even need to know exactly what the underlying continuous signal \( x(t) \) is, only \( S_{m_0,n} \). Equations (3.36) and (3.37) represent the multiresolution decomposition algorithm. The decomposition algorithm is the first half of the fast wavelet transform which allows us to compute the wavelet coefficients in this way, rather than computing them laboriously from the convolution of equation (3.9). Iterating equations (3.36) and (3.37) performs respectively a highpass and lowpass filtering of the input (i.e. the coefficients \( S_{m,2n+k} \)) to get the outputs (\( S_{m+1,n} \) and \( T_{m+1,n} \)). The vectors containing the sequences \((1/\sqrt{2})c_k\) and \((1/\sqrt{2})b_k\) represent the filters: \((1/\sqrt{2})c_k\) is the lowpass filter, letting through low signal frequencies and hence a smoothed version of the signal, and \((1/\sqrt{2})b_k\) is the highpass filter, letting through the high frequencies corresponding to the signal details. We will come back to the filtering process in more detail in later sections of this chapter.

We can go in the opposite direction and reconstruct \( S_{m,n} \) from \( S_{m+1,n} \) and \( T_{m+1,n} \). We know already from equation (3.17) that \( x_{m-1}(t) = x_m(t) + d_m(t) \); we can expand this as

\[
x_{m-1}(t) = \sum_n S_{m,n} \phi_{m,n}(t) + \sum_n T_{m,n} \psi_{m,n}(t) \quad (3.38)
\]

Using equations (3.27b) and (3.28) we can expand this equation in terms of the scaling function at the previous scale as follows:

\[
x_{m-1}(t) = \sum_n S_{m,n} \frac{1}{\sqrt{2}} \sum_k c_k \phi_{m-1,2n+k}(t) + \sum_n T_{m,n} \frac{1}{\sqrt{2}} \sum_k b_k \phi_{m-1,2n+k}(t)
\]

Rearranging the summation indices, we get

\[
x_{m-1}(t) = \sum_n S_{m,n} \frac{1}{\sqrt{2}} \sum_k c_k - 2n \phi_{m-1,k}(t) + \sum_n T_{m,n} \frac{1}{\sqrt{2}} \sum_k b_k - 2n \phi_{m-1,k}(t)
\]

\[
(3.39)
\]

\[
(3.40)
\]
We also know that we can expand \( x_{m-1}(t) \) in terms of the approximation coefficients at scale \( m - 1 \), i.e.

\[
x_{m-1}(t) = \sum_n S_{m-1,n} \phi_{m-1,n}(t)
\]

(Equation 3.41)

Equating the coefficients in equation (3.41) with equation (3.40) we note that the index \( k \) at scale index \( m \) relates to the location index \( n \) at scale index \( m - 1 \). In addition, location index \( n \) in equation (3.40) is not equivalent to location index \( n \) in equation (3.41), as the former corresponds to scale index \( m \), with associated discrete location spacings \( 2^m \), and the latter to scale index \( m - 1 \), with discrete location spacings \( 2^{m-1} \). Hence the \( n \) indices are twice as dense in the latter expression. The simplest way to proceed before equating the two expressions is to swap the indices \( k \) and \( n \) in equation (3.40) which, after some algebra, produces the reconstruction algorithm:

\[
S_{m-1,n} = \frac{1}{\sqrt{2}} \sum_k c_{n-2k} S_{m,k} + \frac{1}{\sqrt{2}} \sum_k b_{n-2k} T_{m,k}
\]

(Equation 3.42)

where we have reused \( k \) as the location index of the transform coefficients at scale index \( m \) to differentiate it from \( n \), the location index at scale \( m - 1 \). Hence, at the smaller scale, \( m - 1 \), the approximation coefficients can be found in terms of a combination of approximation and detail coefficients at the next scale, \( m \). Note that if there are only a finite number of nonzero scaling coefficients (\( = N_K \)), then \( c_{n-2k} \) has nonzero values only when in the range 0 to \( N_K - 1 \). The reconstruction algorithm is the second half of the fast wavelet transform (FWT). Note that in the literature the fast wavelet transform, discrete wavelet transform, decomposition/reconstruction algorithms, fast orthogonal wavelet transform, multiresolution algorithm, pyramid algorithm, tree algorithm and so on, are all used to mean the same thing. It becomes even more confusing when other discretizations of the continuous wavelet transform are referred to as the discrete wavelet transform. Take care!

3.3 Discrete input signals of finite length

3.3.1 Approximations and details

So far we have considered the discrete orthonormal wavelet transform of a continuous function \( x(t) \), where it was shown how the continuous function could be represented as a series expansion of wavelet functions at all scales and locations (equation (3.10a)) or a combined series expansion involving the scaling and wavelet functions (equation (3.16)). In this section, and from here on, we will consider discrete input signals specified at integer spacings. To fit into a wavelet multiresolution framework, the discrete signal input into the multiresolution algorithm should be the signal approximation coefficients at scale index \( m = 0 \), defined by

\[
S_{0,n} = \int_{-\infty}^{\infty} x(t) \phi(t - n) \, dt
\]

(Equation 3.43)
which, as we now know from equations (3.36) and (3.37), will allow us to generate all subsequent approximation and detail coefficients, \( S_{m,n} \) and \( T_{m,n} \), at scale indices greater than \( m = 0 \). In this section we will assume that we have been given \( S_{0,n} \). Section 3.4 considers further the question of discrete input data which may not be \( S_{0,n} \).

In practice our discrete input signal \( S_{0,n} \) is of finite length \( N \), which is an integer power of 2: \( N = 2^M \). Thus the range of scales we can investigate is \( 0 < m < M \). Substituting both \( m = 0 \) and \( m = M \) into equation (3.16), and noting that we have a finite range of \( n \) which halves at each scale, we can see that the signal approximation scale \( m = 0 \) (the input signal) can be written as the smooth signal at scale \( M \) plus a combination of detailed signals as follows:

\[
\sum_{n=0}^{2^M-1} S_{0,n} \phi_{0,n}(t) = S_{M,0} \phi_{M,0}(t) + \sum_{m=1}^{M} \sum_{n=0}^{2^M-1} T_{m,n} \psi_{m,n}(t)
\]

This is the form we use to describe our finite length discrete signal in terms of its discrete wavelet expansion. The covering of a finite length time segment with wavelets is illustrated in figure 3.4 for Daubechies D4 wavelets at two successive scales. The lower scale covers the time window using eight wavelets, and the larger scale uses four wavelets. One of the wavelets in each plot is shown bold for clarity. The wavelets shown which spill over the end of the window have been wrapped around back to the

![Figure 3.4. Covering the time axis with dyadic grid wavelets. (a) Eight Daubechies D4 wavelets covering the time axis at scale \( m \). (b) Four Daubechies D4 wavelets covering the time axis at scale \( m + 1 \). These wavelets are twice the width of those in (a).](image-url)

Copyright © 2002 IOP Publishing Ltd.
beginning. Known as wraparound, it is the simplest and one of the most common treatments of the boundary for a finite length signal and we will employ it throughout the rest of this chapter. However, note that, by employing this method, we assume that the signal segment under investigation represents one period of a periodic signal and we are in effect pasting the end of the signal back on to the beginning. Obviously, if the signal is not periodic, and in practice it usually is not, then we create artificial singularities at the boundary which results in large detail coefficients generated near to the boundary.

We can rewrite equation (3.44) as

$$x_0(t) = x_M(t) + \sum_{m=1}^{M} d_m(t)$$

(3.45)

where the mean signal approximation at scale $M$ is

$$x_M(t) = S_{M,n} \phi_{M,n}(t)$$

(3.46)

As the approximation coefficients are simply factored, weighted averages of the signal then, when wraparound is employed to deal with the boundaries, the single approximation component $S_{M,n}$ is related to the mean of the input signal through the relationship $\overline{S_{0,n}} = S_{M,n}/\sqrt{2^M}$, where the overbar denotes the mean of the sequence $S_{0,n}$. In addition, when wraparound has been used to deal with the boundaries, the mean signal approximation at the largest scale, $x_M(t)$, is a constant valued function equal to the input signal mean. (We will see why this is so later in section 3.5.1.)

The term on the far right of equation (3.45) represents the series expansion of the fluctuating components of the finite length signal at various scales in terms of its detail coefficients. The detail signal approximation corresponding to scale index $m$ is defined for a finite length signal as

$$d_m(t) = \sum_{n=0}^{2^M - m - 1} T_{m,n} \psi_{m,n}(t)$$

(3.47)

As we saw above (equation (3.45)), adding the approximation of the signal at scale index $M$ to the sum of all detail signal components across scales $0 < m < M$ gives the approximation of the original signal at scale index 0. Figure 3.5(a) shows the details of a chirp signal with a short burst of noise added to the middle of it. A Daubechies D20 wavelet was used in the decomposition (see an example of this wavelet later in figure 3.15(e)). The original signal is shown at the top of the plot. Below the signal the details for ten wavelet scales, $d_1(t)$ to $d_{10}(t)$, are shown. The bottom trace is the remaining signal approximation $x_{10}(t)$. Adding together all these details plus the remaining approximation (which is the signal mean) returns the original signal. Two things are noticeable from the plot. First, there is a shift to the left of the large amplitude details with increasing scale, as we would expect as the chirp oscillation increases in frequency from left to right. The second thing to notice is that the high frequency burst of noise is captured at the smallest scales, again as we would expect.

We saw in equation (3.19) that the signal approximation at a specific scale was a combination of the approximation and detail at the next lower scale. If we rewrite this
equation as

\[ x_m(t) = x_{m-1}(t) - d_m(t) \]  

(3.48)

and begin at scale \( m - 1 = 0 \), that of the input signal, we can see that at scale index \( m = 1 \), the signal approximation is given by

\[ x_1(t) = x_0(t) - d_1(t) \]  

(3.49a)

at the next scale (\( m = 2 \)) the signal approximation is given by

\[ x_2(t) = x_0(t) - d_1(t) - d_2(t) \]  

(3.49b)

and at the next scale by

\[ x_3(t) = x_0(t) - d_1(t) - d_2(t) - d_3(t) \]  

(3.49c)

Figure 3.5. Multiresolution decomposition of a chirp signal containing a short burst of noise. (a) Signal details \( d_m(t) \). (The signals have been displaced from each other on the vertical axis to aid clarity.) (b) Signal approximations \( x_m(t) \).
and so on, corresponding to the successive stripping of high frequency information (contained within the \(d_m(t)\)) from the original signal. Figure 3.5(b) contains successive approximations \(x_m(t)\) of the chirp signal. The top trace is the original signal \(x_0(t)\). Subsequent smoothing of the signal takes place throughout the traces from the top to the bottom of the figure. As we saw in equation (3.48), the difference between each of the approximation traces \(x_{m-1}(t)\) and \(x_m(t)\) is the detail component \(d_m(t)\). We can view these differences in the detail component traces of figure 3.5(a).

We have glossed over much of the mathematical detail of multiresolution analysis here. Most mathematical accounts of the subject begin with a discussion of orthogonal nested subspaces and the signal approximations and details, \(x_m(t)\) and \(d_m(t)\), as projections on to these spaces. This tack has not been followed here; see for example Mallat (1998), Blatter (1998), Sarkar et al (1998) or Williams and Armatunga (1994). In this chapter we concentrate on the mechanics, rather than the mathematics, of multiresolution analysis.

### 3.3.2 The multiresolution algorithm—an example

Once we have our discrete input signal \(S_{0,n}\), we can compute \(S_{m,n}\) and \(T_{m,n}\) using the decomposition algorithm given by equations (3.36) and (3.37). This can be done for scale indices \(m > 0\), up to a maximum scale determined by the length of the input signal. To do this, we use an iterative procedure as follows. First we compute \(S_{1,n}\) and \(T_{1,n}\) from the input coefficients \(S_{0,n}\), i.e.

\[
S_{1,n} = \frac{1}{\sqrt{2}} \sum_k c_k S_{0,2n+k}
\]

and

\[
T_{1,n} = \frac{1}{\sqrt{2}} \sum_k b_k S_{0,2n+k}
\]

In the same way, we can then find \(S_{2,n}\) and \(T_{2,n}\) from the approximation coefficients \(S_{1,n}\), i.e.

\[
S_{2,n} = \frac{1}{\sqrt{2}} \sum_k c_k S_{1,2n+k}
\]

and

\[
T_{2,n} = \frac{1}{\sqrt{2}} \sum_k b_k S_{1,2n+k}
\]

Next, we can find \(S_{3,n}\) and \(T_{3,n}\) from the approximation coefficients \(S_{2,n}\), and so on, up to those coefficients at scale index \(M\), where only one approximation and one detail coefficient is computed: \(S_{M,0}\) and \(T_{M,0}\). At scale index \(M\) we have performed a full decomposition of the finite-length, discrete input signal. We are left with an array of coefficients: a single approximation coefficient value, \(S_{M,0}\), plus the detail coefficients, \(T_{m,n}\), corresponding to discrete wavelets of scale \(a = 2^m\) and location \(b = 2^m n\). The finite time series is of length \(N = 2^M\). This gives the ranges of \(m\) and \(n\) for the detail coefficients as respectively \(1 < m < M\) and \(0 < n < 2^{M-m} - 1\).
Notice that the range of $n$ successively halves at each iteration as it is a function of scale index $m$ for a finite length signal. At the smallest wavelet scale, index $m = 1$, $2^M/2^1 = N/2$ coefficients are computed, at the next scale $2^M/2^2 = N/4$ are computed and so on, at larger and larger scales, until the largest scale ($m = M$) where only one ($= 2^M/2^M$) coefficient is computed. The total number of detail coefficients for a discrete time series of length $N = 2^M$ is then, $1 + 2 + 4 + 8 + \cdots + 2^{M-1}$, or $\sum_{m=1}^{M-1} 2^m = 2^M - 1 = N - 1$. In addition to the detail coefficients, the single approximation coefficient $S_{M,0}$ remains. This is related to the signal mean as we saw above, and is required in addition to the detail coefficients to fully represent the discrete signal. Thus a discrete input signal of length $N$ can be broken down into exactly $N$ components without any loss of information using discrete orthonormal wavelets. In addition, no signal information is repeated in the coefficient representation. This is known as zero redundancy.

The decomposition of approximation coefficients into approximation and detail coefficients at subsequent levels can be illustrated schematically thus

\[ S_{0,n} \rightarrow S_{1,n} \rightarrow S_{2,n} \rightarrow S_{3,n} \rightarrow \cdots \rightarrow S_{M,0} \]

\[ T_{1,n} \rightarrow T_{2,n} \rightarrow T_{3,n} \rightarrow \cdots \rightarrow T_{M,0} \]

Figure 3.6 shows an alternative schematic of the same process, illustrating the decomposition and insertion of the approximation and detail coefficients at each scale index. The discrete wavelet transform

\[ \text{original signal} \quad n=0,1,\ldots,N-1 \]

\[ \text{wavelet transform vector at full decomposition} \]

\[ n=0,\ldots,N/32-1 \quad n=0,1,\ldots,N/8-1 \quad n=0,\ldots,N/2-1 \]

\[ n=0,1,\ldots,N/16-1 \quad n=0,1,\ldots,N/4-1 \]

Figure 3.6. Schematic diagram of the Haar filtering algorithm. $\rightarrow S_{m,n}$ used to derive $S_{m+1,n}$; $\rightarrow S_{m,n}$ used to derive $T_{m+1,n}$; $\rightarrow T_{m,n}$ taken to next level as $T_{m,n}$, i.e. no further manipulation.

Copyright © 2002 IOP Publishing Ltd.
iteration within the wavelet transform vector for an arbitrary input signal vector containing 32 components. The wavelet transform vector after the full decomposition has the form $W^{(M)} = (S_M, T_M, T_{M-1}, \ldots, T_m, \ldots, T_2, T_1)$, where $T_m$ represents the sub-vector containing the coefficients $T_{m,n}$ at scale index $m$ (where $n$ is in the range 0 to $2^M - m - 1$). We can halt the transformation process before the full decomposition. If we do this, say at an arbitrary level $m_0$, the transform vector has the form $W^{(m_0)} = (S_{m_0}, T_{m_0}, T_{m_0-1}, \ldots, T_2, T_1)$, where $m_0$ can take the range $1 \leq m_0 \leq M - 1$. In this case, the transform vector does not contain a single approximation component but rather the sequence of approximation components $S_{m_0,n}$. However, the transform vector always contains $N = 2^M$ components. For example, we can see from figure 3.6 that stopping the algorithm at $m = 2$ results in $W^{(2)} = (S_2, T_2, T_1)$. Remember, the range of $n$ is a function of scale index $m$, hence this vector contains eight $S_{2,n}$ components, eight $T_{2,n}$ components and 16 $T_{1,n}$ components, matching the 32 components of the original input signal vector. The range of $n$ is indicated below the full decomposition vector in figure 3.6. Notice also that we can express the original input signal as the transform vector at scale index zero, i.e. $W^{(0)}$.

An example of a wavelet decomposition using a discrete wavelet is shown in figure 3.7. The input signal is composed of a section of a sine wave, some noise and a flatline. The signal is decomposed using a Daubechies D6 wavelet. A member of this family is shown in figure 3.7(b). (We will look more closely at the Daubechies wavelet family later in this chapter.) The discrete transform plot is shown in figure 3.7(c), where the dyadic grid arrangement may be seen clearly. This plot is simply a discretized dyadic map of the detail coefficients, $T_{m,n}$, where the coefficients at larger scales have correspondingly longer boxes (as the wavelets cover larger segments of the input signal). In addition to the detail coefficients, $T_{m,n}$, the remaining approximation coefficient $S_{0,0}$ is added to the bottom of the plot. As we would expect it covers the whole time axis. We can see from the transform plot that the dominant oscillation is picked up at scale index $m = 6$ and the high frequency noise is picked up within the middle segment of the transform plot at the smaller scales. We can use the reconstruction algorithm (equation (3.42)) to get back the original input signal $S_{0,n}$ from the array of detail coefficients shown in figure 3.7(c). Alternatively, as with the continuous transform, we can reconstruct a modified version of the input signal by using only selected coefficients in the reconstruction. This is shown in figures 3.7(d) and (e), where only the coefficients corresponding to scales $m = 5$ to 8 are kept (the others are set to zero) and the signal is reconstructed. This has removed a significant amount of the noise from the signal although the sinusoidal waveform is less smooth than the original in figure 3.7(a). We will come across more sophisticated ways to remove noise and other signal artefacts later, in section 3.4.2 of this chapter.

3.3.3 Wavelet energy

After a full decomposition, the energy contained within the coefficients at each scale is given by

$$E_m = \sum_{n=0}^{2^M - m - 1} (T_{m,n})^2$$  \hspace{1cm} (3.52)
A wavelet-based power spectrum of the signal may be produced using these scale-dependent energies. To do so, we require a frequency measure which is a reciprocal of the wavelet dilation, e.g. the passband centre of the power spectrum of the wavelet. A wavelet power spectrum can then be produced for the signal which is directly comparable with both its Fourier and continuous wavelet counterparts (see chapter 2, section 2.9). This topic is dealt with in detail in chapter 4, section 4.2.1, in connection with the statistical measures used to analyse turbulent fluid signals.

The total energy of the discrete input signal

\[ E = \sum_{n=0}^{N-1} (S_{0,n})^2 \]  

(3.53)
is equal to the sum of the squared detail coefficients over all scales plus the square of the remaining approximation coefficient, $S_{M,0}$, as follows:

$$E = (S_{M,0})^2 + \sum_{m=1}^{M} \sum_{n=0}^{2^{M-m}-1} (T_{m,n})^2$$

(3.54)

In fact, the energy contained within the transform vector at all stages of the multi-resolution decomposition remains constant. We can, therefore, write the conservation of energy more generally as

$$E = \sum_{i=0}^{N-1} (W_{i}^{(m)})^2$$

(3.55)

where $W_{i}^{(m)}$ are the individual components of the transform vector $W^{(m)}$ ordered as described in the previous section. When $m = 0$, this equation corresponds to the summation of the component energies of the input signal (equation (3.53)) and when $m = M$ it corresponds to the summation of the energies within the coefficients at full decomposition (equation (3.54)).

### 3.3.4 Alternative indexing of dyadic grid coefficients

There are three main methods used in practice to index the coefficients resulting from the discrete wavelet transform. It is worth discussing these now. All three methods are popular in the scientific literature and appear often in many of the examples of the practical application of wavelet analysis in the subsequent chapters of this book. We will use the full decomposition of a 32 component input signal as an illustration.

**Scale indexing:** The scale indexing system $(m, n)$ corresponding to an input signal of length $N = 2^M$ is shown schematically on a dyadic grid in figure 3.8(b) for the discrete signal shown in figure 3.8(a). The lowest scale on the grid $m = 1$ corresponds to a spacing of $2^1 = 2$ on the data set. The discrete input signal is at scale index $m = 0$. We have already come across this grid structure in the plot of the transform coefficients in figure 3.7(c). Such plots give a good visual indication of the covering of the timescale plane by the wavelets and their relative importance in making up the signal.

**Sequential indexing:** Again, we have already come across this form of indexing (figure 3.6). Figure 3.8(c) contains a schematic of the transform coefficients given in sequential format. We know that a signal $N$ samples long produces $N$ wavelet vector components. It makes sense, therefore, to find an ordering of these coefficients to fit into a vector of length $N$. The discrete time series, $S_{0,n}, n = 0, 1, 2, \ldots, N-1$, can be decomposed into $N-1$ detail coefficients plus one approximation coefficient where $N$ is an integer power of 2: $N = 2^M$. After a full decomposition these transform coefficients can be resequenced from the two-dimensional dyadic array $T_{m,n}$ to the transform vector $W^{(M)}$, where the components $W_{i}^{(M)}$ have the same range as the original signal $(i = 0, 1, 2, \ldots, N-1)$. The vector component index $i$ is found from the dyadic grid indices $m$ and $n$ through the relationship $i = 2^{M-m} + n$. In addition, the superscript $M$ in parentheses denotes a full decomposition of the signal over $M$ scales. The transform vector components, $W_{i}^{(M)}$ are plotted in figure 3.8(c).
half of the series in the figure represents the coefficients corresponding to the smallest wavelet scale (index $m = 1$). The next quarter, working backwards, represents the coefficients corresponding to scale index $m = 2$, the next eighth to the $m = 3$ coefficients, and so on, back to $W_1^{(M)}$, which is the single coefficient for scale index $m = M$. The remaining component $W_0^{(M)}$ is the single approximation coefficient ($S_{M,0}$). As
mentioned in section 3.3.2, if we halt the decomposition algorithm at scale \( m_0 \) (before full decomposition) this results in an intermediate transform vector \( W^{(m_0)} \) which contains a number of approximation coefficients, \( S_{m_0,n} \), at its beginning.

**Level indexing**: Level indices, \( l \), are often used instead of scale indices, \( m \). The level index \( l \) is simply equal to \( M - m \). In this case, the number of wavelets used to cover the signal at each level at a specific level is \( 2^l \), e.g. level \( l = 0 \) corresponds to a single wavelet and the scale of the whole time series, level \( l = 1 \), corresponds to two wavelets, and the scale of one half of the time series, and so on. The number of wavelets used at a level is simply \( 2^l \). In addition, it is standard in this nomenclature to denote the remaining approximation coefficient at level \( l = -1 \). The wavelet array becomes \( T_{l,n} \). Figure 3.8(d) shows the \( l,n \) indexing system. Level indexing is used when the signal is specified over the unit interval and the analysis is set up in terms of resolution rather than scale (i.e. the lowest resolution \( l = 0 \) corresponds to the length of the whole signal, whereas the smallest scale \( m = 0 \) corresponds to the distance between each signal point).

### 3.3.5 A simple worked example: the Haar wavelet transform

Now we will illustrate the methods described above using a Haar wavelet in the decomposition of a discrete input signal, \( S_{0,n} \): \( n = 0, 1, 2, \ldots, N - 1 \). To do so, we employ the decomposition algorithm given by equations (3.36) and (3.37). The Haar wavelet has two scaling coefficients, \( c_0 = 1 \) and \( c_1 = 1 \). Substituting these into equation (3.36) we can obtain the approximation coefficients at the next scale through the relationship

\[
S_{m+1,n} = \frac{1}{\sqrt{2}} [S_{m,2n} + S_{m,2n+1}]
\]  
(3.56)

Similarly, through equation (3.37) we can obtain the detail coefficients at subsequent scales using

\[
T_{m+1,n} = \frac{1}{\sqrt{2}} [S_{m,2n} - S_{m,2n+1}]
\]  
(3.57)

Using these equations, we will perform the Haar wavelet decomposition of the (very) simple discrete signal, \( (1, 2, 3, 4) \). As the signal contains only four data points, we can only perform two iterations of the Haar decomposition algorithm given by equations (3.56) and (3.57). After two iterations we expect to obtain four transform coefficients: three wavelet coefficients \( T_{m,n} \)—two at scale index \( m = 1 \), \( (T_{1,0}, T_{1,1}) \), and one at scale index \( m = 2 \), \( (T_{2,0}) \), plus a signal mean coefficient at scale index \( m = 2 \), \( (S_{2,0}) \). This is illustrated through the iteration of the transform vector in figure 3.9 and also through a schematic of the coefficients in figure 3.10.

The first iteration of the decomposition algorithm gives

\[
T_{1,0} = \frac{1}{\sqrt{2}} [1 - 2] = -\frac{1}{\sqrt{2}} \quad T_{1,1} = \frac{1}{\sqrt{2}} [3 - 4] = -\frac{1}{\sqrt{2}}
\]

\[
S_{1,0} = \frac{1}{\sqrt{2}} [0 + 3] = \frac{3}{\sqrt{2}} \quad S_{1,1} = \frac{1}{\sqrt{2}} [3 + 4] = \frac{7}{\sqrt{2}}
\]
The transform coefficient vector, after this first iteration, is then

\[ \mathbf{W}(1) = (S_{1,0}, S_{1,1}, T_{1,0}, T_{1,1}) = \left( \frac{3}{\sqrt{2}}, \frac{7}{\sqrt{2}}, - \frac{1}{\sqrt{2}}, - \frac{1}{\sqrt{2}} \right) \]

The second iteration (only involving the remaining approximation coefficients \( S_{1,0} \) and \( S_{1,1} \)) yields

\[ T_{2,0} = \frac{1}{\sqrt{2}} \left[ \frac{3}{\sqrt{2}} - \frac{7}{\sqrt{2}} \right] = -2 \quad \text{and} \quad S_{2,0} = \frac{1}{\sqrt{2}} \left[ \frac{3}{\sqrt{2}} + \frac{7}{\sqrt{2}} \right] = 5 \]

The transform coefficient vector after this second iteration is now

\[ \mathbf{W}(2) = (S_{2,0}, T_{2,0}, T_{1,0}, T_{1,1}) = \left( 5, -2, - \frac{1}{\sqrt{2}}, - \frac{1}{\sqrt{2}} \right) \]

The signal mean is found from the remaining approximation coefficient, i.e. \( S_{2,0}/(\sqrt{2})^2 = 2.5 \). We can also see that the energy of the original discrete signal \( (1^2 + 2^2 + 3^2 + 4^2 = 30) \) is equal to the energy of the transform coefficient vector after the first iteration

\[ \left( \frac{3}{\sqrt{2}} \right)^2 + \left( \frac{7}{\sqrt{2}} \right)^2 + \left( - \frac{1}{\sqrt{2}} \right)^2 + \left( - \frac{1}{\sqrt{2}} \right)^2 = 30 \]

and the second iteration, giving the full decomposition

\[ 5^2 + (-2)^2 + \left( - \frac{1}{\sqrt{2}} \right)^2 + \left( - \frac{1}{\sqrt{2}} \right)^2 = 30 \]

which is a result we expect from the conservation of energy condition expressed in equations (3.52) to (3.55).

To return from the transform coefficients to the original discrete signal we work in reverse. The inverse Haar transform can be written simply as: at even locations \( 2n \):

\[ S_{m,2n} = \frac{1}{\sqrt{2}} [S_{m+1,n} + T_{m+1,n}] \quad (3.58) \]
and at odd locations $2n + 1$:

$$S_{m,2n+1} \equiv \frac{1}{\sqrt{2}} [S_{m+1,n} - T_{m+1,n}]$$  \hspace{1cm} (3.59)

For the Haar wavelet transform, we can derive these reconstruction equations directly from equations (3.56) and (3.57). Alternatively, we could derive them from the reconstruction algorithm of equation (3.42).
The discrete wavelet transform

We will perform the reconstruction on the transform vector we have just computed:

\[ W^{(2)} = (S_{2,0}, T_{2,0}, T_{1,0}, T_{1,1}) = \left( 5, -2, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}} \right) \]

The first iteration using the reconstruction pair (equations (3.58) and (3.59)) yields

\[ S_{1,0} = \frac{1}{\sqrt{2}} (S_{2,0} + T_{2,0}) = \frac{3}{\sqrt{2}} \quad S_{1,1} = \frac{1}{\sqrt{2}} (S_{2,0} - T_{2,0}) = \frac{7}{\sqrt{2}} \]

which results in

\[ (S_{1,0}, S_{1,1}, T_{1,0}, T_{1,1}) = \left( \frac{3}{\sqrt{2}}, \frac{7}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}} \right) \]

Iterating again gives

\[ S_{0,0} = \frac{1}{\sqrt{2}} (S_{1,0} + T_{1,0}) = 1 \quad S_{0,1} = \frac{1}{\sqrt{2}} (S_{1,0} - T_{1,0}) = 2 \]

\[ S_{0,2} = \frac{1}{\sqrt{2}} (S_{1,1} + T_{1,1}) = 3 \quad S_{0,3} = \frac{1}{\sqrt{2}} (S_{1,1} - T_{1,1}) = 4 \]

hence we get back the original signal \((S_{0,0}, S_{0,1}, S_{0,2}, S_{0,3}) = (1, 2, 3, 4)\).

Figure 3.10 attempts to show visually the decomposition of the signal into the Haar wavelet components. Figure 3.10(a) contains the original signal, figure 3.10(b) plots the coefficients after the first iteration of the decomposition algorithm and figure 3.10(c) plots the coefficients after the second iteration of the algorithm. The coefficients contained in figure 3.10(c) correspond in turn to a single scaling function at scale index \(m = 2\), a wavelet at scale index \(m = 2\) and two wavelets at scale index \(m = 1\). These are shown respectively in figures 10(d)–(f). The transform coefficient plot for this signal is equally simplistic, consisting of four coefficient values partitioned as shown in figure 3.10(g).

We can find the corresponding approximation and details of the signal by taking the inverse transform of the coefficients at each scale. First the approximation coefficient is used to determine the signal approximation at the largest scale:

\[ x_2(t) = S_{2,0} \phi_{2,0}(t) \]  

(3.60)

where the scaling function \(\phi_{2,0}(t)\), for the Haar wavelet, is simply a block pulse of length 4 and amplitude \(1/(\sqrt{2})^2 = \frac{1}{2}\), beginning at \(t = 0\). Thus \(x_2(t)\) is simply a block pulse of length 4 and magnitude \(S_{2,0} \times \frac{1}{2} = 2.5\), which is, in fact, the signal mean. A plot of \(x_2(t)\) is shown in figure 3.10(d). Similarly, we can obtain the signal detail component at scale index 2 as follows:

\[ d_2(t) = T_{2,0} \psi_{2,0}(t) \]  

(3.61)

This detail component (shown in figure 3.10(e)) is simply a single Haar wavelet spanning the data set with coefficient value \(T_{2,0} = -2\), hence magnitude \(-2/(\sqrt{2})^2 = -1\). Next the detail signal component at scale index 1 is found from

\[ d_1(t) = \sum_{n=0}^{1} T_{1,n} \psi_{1,n}(t) = T_{1,0} \psi_{1,0}(t) + T_{1,1} \psi_{1,1}(t) \]  

(3.62)
which is simply two Haar wavelets at scale index 1, side by side, with amplitudes given by
\[ T_{1,0}/(\sqrt{2}) = (-1/\sqrt{2})/\sqrt{2} = -0.5 \quad \text{and} \quad T_{1,1}/(\sqrt{2}) = (-1/\sqrt{2})/\sqrt{2} = -0.5. \]
The detail component \( d_1(t) \) is plotted in figure 3.10(f). We already know from equation (3.45) that the approximation of the signal at scale index 0 can be found by adding together all the detail components plus the signal approximation at scale index \( M \), i.e.

\[ x_0(t) = x_M(t) + \sum_{m=1}^{M} d_m(t) \quad (3.63a) \]

We have already seen an example of this partitioning of the signal into approximation and detail components in figure 3.5 for a chirp signal. For the Haar case considered here \( M = 2 \), hence

\[ x_0(t) = x_2(t) + d_2(t) + d_1(t) \quad (3.63b) \]

This is simply the addition of the approximation and detail components shown in figures 3.10(d)–(f). This is shown schematically in figure 3.10(h).

### 3.4 Everything discrete

#### 3.4.1 Discrete experimental input signals

We will now consider the case where we have a discrete experimental signal collected, say, using some piece of experimental apparatus, and we want to perform a wavelet decomposition of it using the multiresolution algorithm. In addition, we would also like to represent the approximation and detail signal components discretely at the resolution of the input signal. This discrete signal, which we will denote \( x_i \), is of finite length \( N \), i.e. \( i = 0, 1, \ldots, N - 1 \). It has been acquired at discrete time intervals \( \Delta t \) (the sampling interval) to give the discrete time signal \( x(t_i) \); \( i = 0, 1, 2, \ldots, N - 1 \). The sampling of the signal provides a finite resolution to the acquired signal. This discretization of the continuous signal is then mapped on to a discrete signal \( x_i \), where the sampling interval has been normalized to 1. In doing so we must remember \( \Delta t \) and add it when required, for example in real applications when computing signal frequencies.

Common practice is to input the discretely sampled experimental signal, \( x_i \), directly as the approximation coefficients at scale \( m = 0 \), and begin the multiresolution analysis from there. However, it is not correct to use the sampled time series \( x_i \) directly in this way. We should really use the approximation coefficients \( S_{0,n} \) obtained from the original continuous signal at scale \( m = 0 \), as defined by equation (3.43). In practice we usually do not know exactly what \( x(t) \) is. As \( S_{0,n} \) is a weighted average of \( x(t) \) in the vicinity of \( n \), then it is usually reasonable to input \( x_i \) as \( S_{0,n} \) if our signal is slowly varying between samples at this scale. That is we simply set

\[ S_{0,n} = x_n \quad (3.64) \]
where, at scale index \( m = 0 \), both the coefficient location index \( n \) and signal discretization index \( i \) have the same range (0 to \( N - 1 \)) and are equal to each other. In the rest of this chapter we will assume that the sampled experimental signal has been input directly as \( S_{0,n} \). The literature is full of studies of real data where this has been done and, if anything, it is the rule rather than the exception. Obviously, it does not make it any more correct! In fact, Strang and Nguyen (1996) call the use of the sampled signal directly within the transform a ‘wavelet crime’ and suggest various ways to preprocess the sampled signal prior to performing the analysis.

In practice, continuous approximations, \( x_m(t) \), and details, \( d_m(t) \), of the signal are not constructed from the multiresolution analysis, especially when inputting the signal \( x_i \) as the detail coefficients at scale index \( m = 0 \). Instead, either the approximation and detail coefficients, \( S_{m,n} \) and \( T_{m,n} \), are displayed at their respective scales or, alternatively, they are used to construct representations of the signal at the scale of the input signal \((m = 0)\). The latter is sometimes preferable as the scalings of the displays are visually comparable. As an example, let us consider the Haar wavelet decomposition of a simple discrete signal containing eight components: \( x_i = (4, 5, -4, -1, 0, 1, 2, 0) \). The signal is shown in figure 3.11(a). We can see that the first half of the signal has a marked step from 5 to \(-4\), whereas the second half of the signal appears much smoother. The transform vector after a full decomposition is

\[
W(3) = (S_{3,0}, T_{3,0}, T_{2,0}, T_{2,1}, T_{1,0}, T_{1,1}, T_{1,2}, T_{1,3})
\]

\[
= (2.475, 0.354, 7.000, -0.500, -0.707, -2.121, -0.707, 1.141)
\]

\[
= \left( \frac{7}{\sqrt{2}^3}, \frac{1}{\sqrt{2}^3}, \frac{14}{\sqrt{2}^3}, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{3}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{2}{\sqrt{2}} \right)
\]

(You can find these values for yourself using the procedure described in section 3.3.5.) This vector is shown schematically in figure 3.11(b). Performing the inverse transform on this vector leads us back to the original discrete input signal. Referring back to the schematic of the wavelet decomposition given in section 3.3.2, we can similarly represent the reconstruction from the transform vector, \( W(3) \), as

\[
\text{S}_{3,n} \longrightarrow \text{S}_{2,n} \longrightarrow \text{S}_{1,n} \longrightarrow \text{S}_{0,n}
\]

\[
\text{T}_{3,n} \quad \text{T}_{2,n} \quad \text{T}_{1,n}
\]

Remember that the original signal \( x_i \) was input into the multiresolution algorithm as \( S_{0,n} \).

Let us now look at what happens to the reconstructed signal when we remove the smallest-scale (i.e. highest-frequency) detail coefficients in turn (figures 3.11(c)–(e)). Removing the components, \( T_{1,n} \), the modified transform vector becomes \((2.475, 0.354, 7.000, -0.500, 0, 0, 0, 0)\), where the coefficients set to zero are shown bold. Performing the inverse transform on this vector with the coefficients at the smallest wavelet scale set to zero removes the high-frequency details at this scale and returns a smoother version of the input signal. We will denote this smooth version of the
input signal as \( x_{1,i} \). This operation is shown schematically as

\[
\begin{align*}
\mathbf{S}_{3,n} & \longrightarrow \mathbf{S}_{2,n} \longrightarrow \mathbf{S}_{1,n} \longrightarrow x_{1,i} \\
\mathbf{T}_{3,n} & \quad \mathbf{T}_{2,n} \quad \text{zeros}
\end{align*}
\]

The reconstructed discrete signal becomes \( x_{1,i} = (4.5, 4.5, -2.5, -2.5, 0.5, 0.5, 1, 1) \). That is, the signal is smoothed by the averaging of each pair of signal values as shown in figure 3.11(c). Note that we use the nomenclature \( x_{1,i} \) for the approximation signal \( x_1(t) \) expressed in terms of discrete coefficients at the scale of the original input signal (i.e. \( m = 0 \)). Remember from section 3.2.3 that the approximation coefficients, \( S_{m,n} \),
provide a discrete approximation of the signal at scale index \( m \). By passing the coefficients through the reconstruction filter we can express the contributions of these discrete approximations at the scale of the original signal (scale index 0). In the rest of this section, we will see how to express all the discrete approximations and details at the scale of the input signal with index \( m = 0 \).

Now we remove the detail coefficients associated with the next smallest scales from the transform vector to get (2.475, 0.354, 0, 0, 0, 0, 0). Reconstructing the signal using this modified transform vector, we get \( x_{2,i} = (1, 1, 1, 1, 0.75, 0.75, 0.75, 0.75) \) shown in Figure 3.11(e). Again we can show this schematically as

\[
\begin{array}{cccc}
S_3,n & \rightarrow & S_2,n & \rightarrow & S_1,n & \rightarrow & x_{2,i} \\
T_3,n & \rightarrow & \text{zeros} & \rightarrow & \text{zeros} \\
\end{array}
\]

where we have put the prime on \( S_1,n \) to indicate that it has different valued components from \( S_1,n \) above in the full reconstruction as it is computed with the \( T_2,n \) components set to zero. We can see that the nomenclature is beginning to get a bit awkward here as \( x_{2,i} \) is in effect \( S_0,n \): a modified version of the input signal. In \( x_{2,i} \), the subscript 2 relates to the largest scale of the original approximation coefficients used at the beginning of the reconstruction, and the index \( i \) reminds us that we are back at the input resolution of the original signal (\( m = 0 \)) with \( N \) components.

Removing the last detail coefficient, the modified transform vector becomes (2.475, 0, 0, 0, 0, 0, 0) which, when used to reconstruct the signal, produces \( x_{3,i} = (0.875, 0.875, 0.875, 0.875, 0.875, 0.875, 0.875, 0.875) \). This can be shown as

\[
\begin{array}{cccc}
S_3,n & \rightarrow & S_2,n & \rightarrow & S_1,n & \rightarrow & x_{3,i} \\
\text{zeros} & \rightarrow & \text{zeros} & \rightarrow & \text{zeros} \\
\end{array}
\]

Actually, we have now removed all signal detail and all the components of vector \( x_{3,i} \) are of constant value. The signal has been progressively smoothed until all that is left is a row of constant values equal to the signal mean (Figure 3.11(g)).

We can by see comparing Figures 3.11(c), (e) and (g) that the high-frequency information has been successively stripped from the discrete input signal (Figure 3.11(a)) by successively removing the discrete details at scale indices \( m_0 = 1, 2 \) and 3 in turn. These discrete signal details are shown in Figures 3.11(d), (f) and (h). If we want to generate the details of the discrete signal at any scale using the multiresolution algorithm, we perform the inverse transform using only the detail coefficients at that scale (we could also subtract two successive discrete approximations). For example, to compute the detail at the lowest wavelet scale (index \( m = 1 \)), which we will denote \( d_{1,i} \), the following multiresolution operation is required:

\[
\begin{array}{cccc}
zeros & \rightarrow & \text{zeros} & \rightarrow & \text{zeros} & \rightarrow & d_{1,i} \\
zeros & \rightarrow & \text{zeros} & \rightarrow & T_1,n \\
\end{array}
\]


\[ d_{1,i} \text{ is in fact the contribution of the wavelets at scale index } m = 1 \text{ (i.e. } \psi_{1,0}(t), \psi_{1,1}(t), \psi_{1,2}(t) \text{ and } \psi_{1,3}(t)) \text{ to the original signal } x_i \text{ expressed at discrete points at the scale of the input function } m = 0 \text{ at locations } i = 0, 1, \ldots, N - 1. \] 

Similarly, we can calculate the contribution \( d_{2,i} \) using the following operation:

\[
\begin{array}{cccc}
\text{zeros} & \rightarrow & \text{zeros} & \rightarrow \quad S_{1,n} & \rightarrow \quad d_{2,i} \\
\text{zeros} & \rightarrow & T_{2,n} & \rightarrow & \text{zeros}
\end{array}
\]

where, obviously, the \( S_{1,n} \) vector has all of its components equal to zero as it comes from \( S_{3,n} \) and \( T_{3,n} \) with elements set to zero. Then \( d_{2,i} \) is the contribution of the wavelets at scale index \( m = 2 \) (i.e. which for this example come from the two wavelets \( \psi_{2,0}(t) \) and \( \psi_{2,1}(t) \)) to the original signal \( x_i \) given at discrete points at the scale of the input function \( m = 0 \), i.e. at locations \( i = 0 \) to \( N - 1 \). Similarly, beginning with \( T_{3,n} \) we can calculate the contribution \( d_{3,i} \).

In the above example we were effectively taking the information stored in the approximation and detail coefficients \( S_{m,n} \) and \( T_{m,n} \) and expressing it at scale index \( m = 0 \). The discrete approximation and detail contributions \( x_{m,i} \) and \( d_{m,i} \) are related to their continuous counterparts \( x_m(t) \) and \( d_m(t) \) through the scaling equation at scale \( m = 0 \) as follows:

\[
x_m(t) = \sum_{i=0}^{N-1} x_{m,i} \phi_{0,i}(t) = \sum_{n=0}^{2^{M-m}-1} S_{m,n} \phi_{m,n}(t) \tag{3.65}
\]

and

\[
d_m(t) = \sum_{i=0}^{N-1} d_{m,i} \phi_{0,i}(t) = \sum_{n=0}^{2^{M-m}-1} T_{m,n} \psi_{m,n}(t) \tag{3.66}
\]

Notice that the scaling function \( \phi_{0,i}(t) \) is not only used to compute the continuous approximation from \( x_{m,i} \) but also the detail components from \( d_{m,i} \). This is because the scaling coefficient sequence for the wavelet, \( b_k \), has already been used in the initial stages of the reconstruction sequence to take \( T_{m,n} \) to \( S'_{m-1,n} \). Thereafter the contributions can be expressed in terms of scaling functions, i.e. from \( S'_{m-1,n} \) to \( S'_{m-2,n} \) and so on. You can verify equations (3.65) and (3.66) using the reconstruction algorithm of equation (3.42) (noting that one of its terms will be zero) and the scaling function relationship given by equation (3.27b).

Figure 3.11(c) shows \( x_{1,i} \), i.e. where only the detail signal \( d_{1,i} \) corresponding to the smallest scale wavelets \( (m = 1) \) has been removed (figure 3.11(d)). Figure 3.11(e) shows \( x_{2,i} \), where both detail signals \( d_{1,i} \) and \( d_{2,i} \), corresponding to the smallest and next smallest scale wavelets, have been removed from the original discrete input signal. \( d_{2,i} \) is shown in figure 3.11(f). Figure 3.11(g) shows \( x_{3,i} \), where the detail signals \( d_{m,i} \) corresponding to all the wavelet scales \( (m = 1, 2 \text{ and } 3) \) have been stripped from the original signal leaving only the signal mean component. This can be written as

\[
x_{3,i} = x_{0,i} - \sum_{m=1}^{3} d_{m,i} \tag{3.67a}
\]
or, in general, over $M$ scales as

$$x_{M,i} = x_{0,i} - \sum_{m=1}^{M} d_{m,i} \quad (3.67b)$$

where $x_{0,i}$ is simply the input signal $x_i$. This equation stems directly from its continuous counterpart of equation (3.45) using equations (3.65) and (3.66). We have now seen how we can express everything discretely. We can use a suitable discrete input signal within the multiresolution algorithm, and we can express the approximation and details of this signal at discrete points at the input resolution; and, as we will see later in section 3.5.1, even the wavelet and scaling functions are expressed discretely within the multiresolution algorithm, built up by the repeated iteration of the scaling coefficient vectors.

### 3.4.2 Smoothing, thresholding and denoising

Let us look again at the discrete input signal used in the previous section (figure 3.12(a)). Once we have performed the full decomposition, we are free to alter any of the coefficients in the transform vector before performing the inverse. We can set groups of components to zero, as we did in the last section (figure 3.11), or set selected individual components to zero. We can reduce the magnitudes of some components rather than set them to zero. In fact, we can manipulate the components in a variety of ways depending on what we want to achieve. Notice, however, that the transform vector contains a range of values: some large and some small. Let us see what happens if we throw away the smallest valued coefficients in turn and perform the inverse transforms. We start with the smallest valued coefficient 0.354. Setting it to zero, the modified transform vector becomes $(2.475, 0, 7.000, -0.500, -0.707, -2.121, -0.707, 1.414)$. Performing the inverse on this vector gives the reconstructed discrete signal $(3.875, 4.875, 0, 4.125, 1.125, 0.875, 1.875, -0.125)$, as shown in figure 3.12(b). There is no discernible difference between the original discrete signal and the reconstruction. We now remove the next smallest valued coefficient (smallest in an absolute sense), the $-0.5$ coefficient. The transform vector becomes $(2.475, 0, 7.000, 0, -0.707, -2.121, 0, 1.414)$ and the reconstructed signal is $(3.875, 4.875, -4.125, -1.125, 0.375, 1.375, 1.875, -0.125)$. Again we have to look carefully at figure 3.12(c) to discern any difference between the original discrete signal and the reconstruction. Next we remove the two $-0.707$ components to get $(2.475, 0, 7.000, 0, 0, -2.121, 0, 1.414)$. Reconstructing we get $(4.375, 4.375, -4.125, -1.125, 0.875, 0.875, 1.875, -0.125)$. This reconstruction is shown in figure 3.12(d) where we can now notice obvious smoothing between both the first and second signal point and the fifth and sixth. Next setting the last transform vector component to zero, we reconstruct to get $(4.375, 4.375, -4.125, -1.125, 0.875, 0.875, 0.875, 0.875)$. This is shown in figure 3.12(e). Removing the $-2.121$ component leaves the coefficient vector as $(2.475, 0, 7.000, 0, 0, 0, 0)$. We can now reconstruct the coarsest approximation signal $(4.375, 4.375, -2.625, -2.625, 0.875, 0.875, 0.875, 0.875)$ as shown in figure 3.12(f). Removing the 2.475 component (the signal mean component) leaves the coefficient vector with only a single component: $(0, 0, 7.000, 0, 0, 0, 0)$. Reconstructing using this...
component leaves a single (wavelet shaped) fluctuation contained within the first half of the signal (figure 3.12(g)).

Can you see what has happened? The least significant components have been smoothed out first, leaving the more significant fluctuating parts of the signal intact. What we have actually done is to threshold the wavelet coefficients at increasing magnitudes. First, all the components whose magnitude was equal to or below 0.354 were removed; i.e. 0.354 was the threshold. Then 0.5 was set as the threshold, then 0.707 and so on. By doing this we removed the least significant influences on the signal first. Hence, the shape of the reconstructed signal resembles the original even with a large number of coefficients set to zero. Compare this magnitude thresholding of the coefficients to the scale-dependent smoothing of figure 3.11, which

Figure 3.12. Signal reconstruction using thresholded wavelet coefficients. The coefficient vectors used in the reconstructions are given below each reconstructed signal. Note that (a) is the original signal as it uses all of the original eight wavelet coefficients.

Copyright © 2002 IOP Publishing Ltd.
removed the components at each scale in turn, beginning with the smallest scales. Note that in practice we would normally deal with the signal mean coefficient separately, either retaining it regardless of magnitude or removing it at the beginning of the thresholding process.

We can define the scale-dependent smoothing of the wavelet coefficients as

\[
T_{m,n}^{\text{scale}} = \begin{cases} 
0 & m \geq m^* \\
T_{m,n} & m < m^* 
\end{cases} \quad (3.68)
\]

where \(m^*\) is the index of the threshold scale, or the transform vector. Figure 3.13(a) shows a schematic diagram of scale-dependent smoothing for sequentially indexed coefficients \(W_i\). (Assuming a full decomposition, we have dropped the \(M\) superscript from \(W_i^M\).) In this case the thresholding criterion is defined as

\[
W_i^{\text{scale}} = \begin{cases} 
0 & i \geq 2^M - m^* \\
W_i & i < 2^M - m^* 
\end{cases} \quad (3.69)
\]

where the range of the sequential index \(i\) is from 0 to \(N - 1\) and \(N\) is the length of the original signal; hence \(i = 2^M - m^*\) is the first location index within the transform vector where the coefficients are set to zero. (Remember that the smallest scale coefficients are placed at the right-hand end of the transform vector and hence have the highest index values.) Note that in practice the coefficients at the very largest scales are sometimes also set to zero to remove drift effects from the signal. The reconstructed signal using the scale thresholded coefficient vector with scale threshold \(m^*\) is simply the smooth approximation of the signal at scale \(m^*\) expressed at the scale of the input signal, \(m = 0\), i.e. \(x_m^{m^*}\). (See previous section.)

Magnitude thresholding is normally carried out to remove noise from a signal, to partition signals into two or more (and not necessarily noisy) components, or simply to smooth the data. It involves the reduction or complete removal of selected wavelet coefficients in order to separate out the behaviour of interest from within the signal. There are many methods for selecting and modifying the coefficients. The two most popular are hard and soft thresholding. Unlike scale-dependent smoothing, which removes all small scale coefficients below the scale index \(m^*\) regardless of amplitude, hard and soft thresholding remove, or reduce, the smallest amplitude coefficients regardless of scale. This is shown schematically in figure 3.13(b). To hard threshold the coefficients, a threshold, \(\lambda\), is set which is related to some mean value of the wavelet coefficients at each scale, e.g. standard deviation, mean absolute deviation, etc. Those coefficients above the threshold are deemed to correspond to the coherent part of the signal, and those below the threshold are deemed to correspond to the random or noisy part of the signal. Hard thresholding is of the form:

\[
W_i^{\text{hard}} = \begin{cases} 
0 & |W_i| < \lambda \\
W_i & |W_i| \geq \lambda 
\end{cases} \quad (3.70)
\]

Hard thresholding makes a decision simply to keep or remove the coefficients. (Hard thresholding was performed in the example of figure 3.12.) Soft thresholding
Figure 3.13. Scale-dependent smoothing and coefficient thresholding. (a) Scale-dependent smoothing. (b) Amplitude thresholding. (c) The relationship between the original coefficients and hard (left) and soft (right) thresholded coefficients.
recognizes that the coefficients contain both signal and noise, and attempts to isolate the signal by removing the noisy part from all coefficients. Soft thresholding is of the form

\[
W_i^{\text{soft}} = \begin{cases} 
0 & |W_i| < \lambda \\
\text{sign}(W_i)(|W_i| - \lambda) & |W_i| \geq \lambda 
\end{cases}
\]

(3.71)

where all coefficients below the threshold, \( \lambda \), are set to zero and all the coefficients whose magnitude is greater than \( \lambda \) are shrunk towards zero by an amount \( \lambda \). Figure 3.13(c) shows a schematic diagram of the relationship between the original and thresholded coefficients for both hard and soft thresholding. The retained coefficients shown in the figure are kept as they are when hard thresholding is employed and their magnitude is reduced by \( \lambda \) when soft thresholding is employed.

Figure 3.14 shows examples of both hard and soft thresholding of a test signal composed of two sinusoids plus Gaussian white noise (figures 3.14(a) and (b)). The wavelet coefficients obtained from a Daubechies D10 decomposition are shown in figure 3.14(c). These have been hard thresholded for thresholds set at \( \lambda = 2, 4, 6 \) and 8 respectively. The reconstructions corresponding to each of the thresholds are shown in figures 3.14(d)–(g). We can see that for low thresholds some of the high frequency noise is retained, whereas for high thresholds the signal is excessively smoothed. From visual inspection, an optimum threshold would seem to lie somewhere in the region of \( \lambda = 4 \). One commonly used measure of the optimum reconstruction is the mean square error between the reconstructed signal and the original signal and, in fact, it is found to be minimum near to this value of the threshold. The corresponding soft thresholded reconstructions are shown in figures 3.14(h) and (i) for \( \lambda = 2 \) and 4 respectively.

Often we do not know the exact form of either the underlying signal or the corrupting noise. The choice of threshold is therefore non-trivial and we can apply a number of signal and/or noise based criteria in producing a value pertinent to the problem under consideration. The threshold can, for example, be a constant value applied to the coefficients across all scales, some of the scales, or its value can vary according to scale. One of the most popular and simplest thresholds in use is the universal threshold defined as

\[
\lambda_U = (2 \ln N)^{1/2} \sigma
\]

(3.72)

where \( (2 \ln N)^{1/2} \) is the expected maximum value of a white noise sequence of length \( N \) and unit standard deviation, and \( \sigma \) is the standard deviation of the noise in the signal under consideration. Thus if the underlying coherent part of the signal is zero (i.e. the signal contains only noise), using \( \lambda_U \) as the threshold gives a high probability of setting all the coefficients to zero. For large samples the universal threshold will remove, with high probability, all the noise in the reconstruction, but part of the underlying function might also be lost, hence the universal threshold tends to over-smooth in practice. (This method is often known as the VISUSHRINK method as the resulting smooth signal estimate is visually more appealing.) In addition, in practice it is usual to leave the coefficients at the largest scales untouched even if they do not pass the universal threshold. The example of the sinusoids plus
noise shown in figure 3.14 contains Gaussian noise with $\sigma = 1$ and $N = 1024$, the universal threshold for this data is then $\lambda_U = 3.723$. Within a wide range of practical data sizes ($2^6$–$2^{19}$), only in about one tenth of the realizations will any pure noise variable exceed the threshold. In this sense universal thresholding method gives a ‘noise-free’ reconstruction.

Figure 3.14. Hard and soft thresholding. (a) Original time series composed of two sinusoids, both with unit amplitude, one twice the periodicity of the other. (b) Time series in (a) with added Gaussian noise (zero mean and unit standard deviation). (c) Wavelet coefficients in sequential format derived using the Daubechies D10 shown. (d) Hard thresholded coefficients, $\lambda = 2$ (top), and corresponding reconstructed time series (bottom). (The original time series of figure (a) is shown dashed.)
Figure 3.14 (continued). (e) Hard thresholded coefficients, $\lambda = 4$ (top), and corresponding reconstructed time series (bottom). (The original time series of figure (a) is shown dashed.) (f) Hard thresholded coefficients, $\lambda = 6$ (top), and corresponding reconstructed time series (bottom). (The original time series of figure (a) is shown dashed.) (g) Hard thresholded coefficients, $\lambda = 8$ (top), and corresponding reconstructed time series (bottom). (The original time series of figure (a) is shown dashed.)
Universal soft thresholding usually produces a signal reconstruction with less energy than that for hard thresholding using the same threshold value, as the retained coefficients are all shrunk towards zero. This can be seen by comparing figure 3.14(e) with 3.14(i) for $\lambda = 4$. Hence, it is often the case in practice that the universal threshold used for hard thresholding is divided by about 2 when employed as a soft threshold. Another problem encountered when implementing the universal threshold in practice is that we do not know the value of $\sigma$ for our signal. In this case, a robust estimate $\hat{\sigma}$ is used, typically set to the median of absolute deviation (MAD) of the wavelet coefficients at the smallest scale divided by 0.6745 to calibrate with the standard deviation of a Gaussian distribution. Thus the universal threshold becomes

\[
\lambda_U = \frac{(2 \ln N)^{1/2} \text{MAD}}{0.6745} = (2 \ln N)^{1/2} \hat{\sigma}
\]  

(3.73)
Many other thresholding methods have been proposed including the minimax method, the SURE method, the HYBRID method, cross-validation methods, the Lorentz method and various Bayesian approaches. Some of these produce a global threshold of constant value across scales and others provide a scale-dependent threshold. We do not consider these in detail here. More information is provided at the end of the chapter.

3.5 Daubechies wavelets

As we saw with the Haar wavelet transform earlier in section 3.3.5, the coefficients are ordered in two distinct sequences: one acts as a smoothing filter for the data, the other extracts the signal detail at each scale. The Haar wavelet is extremely simple in that it has only two scaling coefficients and both are equal to unity. In this section we will look at a family of discrete wavelets of which the Haar wavelet is the simplest member—the Daubechies wavelets. The scaling functions associated with these wavelets satisfy the conditions given in section 3.2.4, as all orthogonal wavelets do. In addition to satisfying these criteria, Daubechies required that her wavelets had compact support (i.e. a finite number, \( N_k \), of scaling coefficients) and were smooth to some degree. The smoothness of the wavelet is associated with a moment condition which can be expressed in terms of the scaling coefficients as

\[
\sum_{k=0}^{N_k-1} (-1)^k c_k k^m = 0
\]

(3.74)

for integers \( m = 0, 1, 2, \ldots, N_k/2 - 1 \). These wavelets have \( N_k/2 \) vanishing moments, which means that they can suppress parts of the signal which are polynomial up to degree \( N_k/2 - 1 \). Or, to put it another way, Daubechies wavelets are very good at representing polynomial behaviour within the signal. Some examples of Daubechies wavelets, their scaling functions and associated energy spectra are given in figure 3.15. The support lengths of the Daubechies wavelets are \( N_k - 1 \), i.e. the D2 (Haar) wavelet, as we already know, has a support length of 1, the D4 wavelet has support length of 3, the D6 a support length of 5 and so on. We can see from figure 3.15 that the scaling function lets through the lower frequencies and hence acts as a lowpass filter, and the associated wavelet lets through the higher frequencies and hence acts as a highpass filter. In addition, we can see that the spectra are oscillatory in nature, with bumps decreasing in amplitude from the lower to higher frequencies. The magnitudes of the secondary bumps in the spectra reduce as the number of scaling coefficients, and hence the number of vanishing moments of the wavelet, increase.

In this chapter we have already looked in detail at the Daubechies wavelet with only two scaling coefficients (the D2 or Haar wavelet). Let us now look at the Daubechies wavelet which has four scaling coefficients, the D4. The ‘D’ represents this particular family of Daubechies wavelet and the ‘4’ represents the number of nonzero scaling coefficients, \( N_k \). (Note that Daubechies wavelets are also often defined by the number of zero moments they have, which equals \( Nk/2 \), in which case the sequence runs D1, D2, D3, D4, \ldots, etc.)
Figure 3.15. A selection of Daubechies wavelets and scaling functions with their energy spectra. All wavelets and scaling functions are shown only over their respective support—outside their support they have zero value. Note also that only the positive part of the spectrum is given; an identical mirror image is present at negative frequencies.
From equation (3.20), we know that the scaling equation for a four-coefficient wavelet is
\[ \phi(t) = c_0 \phi(2t) + c_1 \phi(2t - 1) + c_2 \phi(2t - 2) + c_3 \phi(2t - 3) \quad (3.75) \]
and, from equation (3.23), that the corresponding wavelet function is
\[ \psi(t) = c_3 \phi(2t) - c_2 \phi(2t - 1) + c_1 \phi(2t - 2) - c_0 \phi(2t - 3) \quad (3.76) \]
To find the values of the scaling coefficients for the D4 wavelet we use equations (3.21), (3.22) and (3.74). From equation (3.21) we get
\[ c_0 + c_1 + c_2 + c_3 = 2 \quad (3.77) \]
from equation (3.22) we get
\[ c_0^2 + c_1^2 + c_2^2 + c_3^2 = 2 \quad (3.78) \]
and from equation (3.74) with \( m = 0 \) we get
\[ c_0 - c_1 + c_2 - c_3 = 0 \quad (3.79) \]
and again using equation (3.74), this time setting \( m = 1 \), we get
\[ -1c_1 + 2c_2 - 3c_3 = 0 \quad (3.80) \]
Four scaling coefficients which satisfy the above four equations are
\[ c_0 = \frac{1 + \sqrt{3}}{4}, \quad c_1 = \frac{3 + \sqrt{3}}{4}, \quad c_2 = \frac{3 - \sqrt{3}}{4}, \quad c_3 = \frac{1 - \sqrt{3}}{4} \]
and so do
\[ c_0 = \frac{1 - \sqrt{3}}{4}, \quad c_1 = \frac{3 - \sqrt{3}}{4}, \quad c_2 = \frac{3 + \sqrt{3}}{4}, \quad c_3 = \frac{1 + \sqrt{3}}{4} \]
One set leads to \( \phi(t) \) and the other to \( \phi(-t) \). We will adopt the first set, which are \( c_0 = 0.6830127, \quad c_1 = 1.1830127, \quad c_2 = 0.3169873 \) and \( c_3 = -0.1830127 \) respectively. The scaling coefficients for the Daubechies wavelet system for larger numbers of coefficients are found by numerical computation. The coefficients for Daubechies wavelets up to D20 are given in table 3.1.

We can compute the scaling function from the D4 coefficients using equation (3.75). To do this we must rewrite it as
\[ \phi_j(t) = c_0 \phi_{j-1}(2t) + c_1 \phi_{j-1}(2t - 1) + c_2 \phi_{j-1}(2t - 2) + c_3 \phi_{j-1}(2t - 3) \quad (3.81) \]
where the subscript \( j \) is the iteration number. Then choosing an arbitrary initial shape for the scaling function \( \phi_0(t) \) we find \( \phi_1(t) \), then using \( \phi_1(t) \) we find \( \phi_2(t) \), and so on, iterating until \( \phi_j(t) = \phi_{j-1}(t) \) (or at least until \( \phi_j(t) \) is close enough to \( \phi_{j-1}(t) \) for our purposes). Once we have an approximation for \( \phi(t) \) we could define the wavelet directly using equation (3.76). However, to perform the D4 wavelet transform of a discrete signal in practice we are not required to compute the wavelet or scaling functions directly; rather we simply employ the scaling coefficients within the multi-resolution algorithm (equations (3.36) and (3.37)) in the same manner as we did for
Table 3.1. Daubechies wavelet coefficients D2 to D20

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2</td>
<td>-0.34265671</td>
</tr>
<tr>
<td>1</td>
<td>-0.04560113</td>
</tr>
<tr>
<td>1</td>
<td>0.10970265</td>
</tr>
<tr>
<td>-0.00882680</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>-0.01779187</td>
</tr>
<tr>
<td>0.6830127</td>
<td></td>
</tr>
<tr>
<td>1.1830127</td>
<td></td>
</tr>
<tr>
<td>0.3169873</td>
<td></td>
</tr>
<tr>
<td>-0.1830127</td>
<td></td>
</tr>
<tr>
<td>D6</td>
<td>0.47046721</td>
</tr>
<tr>
<td>0.14111692</td>
<td></td>
</tr>
<tr>
<td>0.650365</td>
<td></td>
</tr>
<tr>
<td>-0.19093442</td>
<td></td>
</tr>
<tr>
<td>-0.12083221</td>
<td></td>
</tr>
<tr>
<td>0.0498175</td>
<td></td>
</tr>
<tr>
<td>D8</td>
<td>0.32580343</td>
</tr>
<tr>
<td>1.01094572</td>
<td></td>
</tr>
<tr>
<td>0.89220014</td>
<td></td>
</tr>
<tr>
<td>-0.03957503</td>
<td></td>
</tr>
<tr>
<td>-0.26450717</td>
<td></td>
</tr>
<tr>
<td>0.0436163</td>
<td></td>
</tr>
<tr>
<td>0.0465036</td>
<td></td>
</tr>
<tr>
<td>-0.01498969</td>
<td></td>
</tr>
<tr>
<td>D10</td>
<td>0.22641898</td>
</tr>
<tr>
<td>0.85394354</td>
<td></td>
</tr>
<tr>
<td>1.02432694</td>
<td></td>
</tr>
<tr>
<td>0.19576966</td>
<td></td>
</tr>
<tr>
<td>D12</td>
<td>0.15774243</td>
</tr>
<tr>
<td>0.69950381</td>
<td></td>
</tr>
<tr>
<td>1.06226376</td>
<td></td>
</tr>
<tr>
<td>-0.04466375</td>
<td></td>
</tr>
<tr>
<td>D14</td>
<td>0.11009943</td>
</tr>
<tr>
<td>0.56079128</td>
<td></td>
</tr>
<tr>
<td>0.103114849</td>
<td></td>
</tr>
<tr>
<td>-0.20351382</td>
<td></td>
</tr>
<tr>
<td>D16</td>
<td>0.07695562</td>
</tr>
<tr>
<td>0.44246725</td>
<td></td>
</tr>
<tr>
<td>0.95548615</td>
<td></td>
</tr>
<tr>
<td>0.082781653</td>
<td></td>
</tr>
<tr>
<td>D18</td>
<td>0.05385035</td>
</tr>
<tr>
<td>0.34483430</td>
<td></td>
</tr>
<tr>
<td>0.85534906</td>
<td></td>
</tr>
<tr>
<td>0.18836955</td>
<td></td>
</tr>
<tr>
<td>D20</td>
<td>0.03771716</td>
</tr>
<tr>
<td>0.26612218</td>
<td></td>
</tr>
<tr>
<td>0.74557507</td>
<td></td>
</tr>
<tr>
<td>0.97362811</td>
<td></td>
</tr>
<tr>
<td>0.39763774</td>
<td></td>
</tr>
</tbody>
</table>

For the Haar wavelet. In this case, the approximation coefficients are computed using

\[
S_{m+1,n} = \frac{1}{\sqrt{2}} \sum_{k=0}^{N_k-1} c_k S_{m,2n+k}
\]

\[
= \frac{1}{\sqrt{2}} [c_0 S_{m,2n} + c_1 S_{m,2n+1} + c_2 S_{m,2n+2} + c_3 S_{m,2n+3}]
\]

\[
= 0.483S_{m,2n} + 0.837S_{m,2n+1} + 0.224S_{m,2n+2} - 0.129S_{m,2n+3} \quad (3.82a)
\]

We therefore take the product of a four-digit sequence of the signal by the scaling coefficient vector \((1/\sqrt{2})c_k\) (the lowpass filter) to generate the approximation component at the first scale. The mechanics of this are shown schematically in figure 3.16. The four-digit sequence, \((1/\sqrt{2})c_k\), slides across the signal at scale index \(m\) in jumps of two generating, at each jump, an approximation component at
Figure 3.16. Filtering of the signal: decomposition. The original signal at scale index \( m = 0 \) (i.e. \( S_{0,n} \)) is filtered to produce the approximation coefficients \( S_{1,n} \). This is done by sliding the lowpass filter along the signal one step at a time. Subsampling removes every second value. The diagram shows only the retained values, i.e. effectively the filter coefficient vector jumps two steps at each iteration. Next, the sequence \( S_{1,n} \) is filtered in the same way to get \( S_{2,n} \) and so on. The detail coefficients are found by using the same method but employing highpass wavelet filter coefficients.
the next scale \((m + 1)\). To generate the corresponding coefficients for the wavelet we use the reconfigured scaling coefficient sequence \((1/\sqrt{2})b_k\) (the highpass filter), where \(b_k = (-1)^k c_{N_k-1-k}\), as follows:

\[
T_{m+1,n} = \frac{1}{\sqrt{2}} \sum_{k=0}^{N_t-1} b_k S_{m,2n+k}
\]

\[
= \frac{1}{\sqrt{2}} \left[ c_3 S_{m,2n} - c_2 S_{m,2n+1} + c_1 S_{m,2n+2} - c_0 S_{m,2n+3} \right]
\]

\[
= -0.129 S_{m,2n} - 0.224 S_{m,2n+1} + 0.837 S_{m,2n+2} - 0.483 S_{m,2n+3}
\]

As an example, we will now consider the D4 wavelet decomposition of the signal \(x_i = (1, 0, 0, 0, 0, 0, 0, 0)\) which we input into the multiresolution algorithm as \(S_{0,n}\). It has \(N = 8\) components, so we iterate the transform decomposition algorithm three times. After the first iteration the transform vector becomes \((0.483, 0.000, 0.000, 0.224, -0.129, 0.000, 0.000, 0.837)\) containing four approximation components followed by four detailed coefficients, i.e. \((S_{1,0}, S_{1,1}, S_{1,2}, S_{1,3}, T_{1,0}, T_{1,1}, T_{1,2}, T_{1,3})\). Remember that we employ wraparound to deal with the edges of the signal. When either of the filter pairs is placed at \(2n = 6\), the \(2n + 2\) and \(2n + 3\) locations are not contained within the signal vector \((i = 0, 1, \ldots, N - 1)\). Thus, the last two filter coefficients of 0.224 and –0.129 (approximation) or 0.837 and –0.483 (detail) are placed back at the start of the signal vector, the first of each coinciding with the unit value at the beginning of the signal, hence producing respectively the values 0.224 or 0.837. These values are placed at location \(S_{1,3}\) and \(T_{1,3}\) respectively. The next iteration is performed on the remaining four approximation components \(S_{1,0}, S_{1,1}, S_{1,2}, S_{1,3}\) producing two approximation components followed by two detailed wavelet components. These are added to the four detailed coefficients obtained at the previous iteration to give the vector \((0.204, 0.296, -0.171, 0.354, -0.129, 0.000, 0.000, 0.837)\). The third and final iteration produces \((0.354, -0.065, -0.171, 0.354, -0.129, 0.000, 0.000, 0.837)\). This vector has the form \((S_{3,0}, T_{3,0}, T_{2,0}, T_{2,1}, T_{1,0}, T_{1,1}, T_{1,2}, T_{1,3})\), where the signal mean can be found from the first coefficient, i.e. \(S_{3,0}/2^{3/2} = 0.354/2^{3/2} = 0.125\).

We know that the Daubechies D4 wavelet has two vanishing moments, thus it can suppress both constant signals and linear signals. This means that for an infinitely long constant or linear signal all the coefficients are zero. However, end effects occur if the signal is of finite length. We will use the linear ramp signal \((0, 1, 2, \ldots, 31)\), shown in figure 3.17(a), as an example. The first iteration produces the transform vector

\[
W^{(1)} = (S_{1,0}, S_{1,1}, S_{1,2}, \ldots, S_{1,15}, T_{1,0}, T_{1,1}, T_{1,2}, \ldots, T_{1,15})
\]

\[
= (0.897, 3.725, 6.553, 9.382, 12.21, 15.039, 17.867, 20.696, 23.524, 26.352, 29.181, 32.009, 34.838, 37.666, 40.495, 40.291, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\]

Copyright © 2002 IOP Publishing Ltd.
where all the detail coefficients at the lowest scale, $T_{1,n}$, are zero except for the end coefficient $T_{1,15}$. This nonzero coefficient is caused by the signal wraparound where the wavelet filter vector $\left(1/\sqrt{2}\right)b_k$ placed at the end of the signal encounters the sequence $30, 31, 0, 1$. We can see that the computation for the corresponding detail component is

$$T_{1,15} = -0.129 \times 30 - 0.224 \times 31 + 0.837 \times 0 - 0.483 \times 1 = -11.314$$

A histogram of the coefficients is shown in figure 3.17(b). Notice also the edge effect in the approximation coefficients where they all increase linearly in value except the last

Figure 3.17. The multiresolution decomposition of a ramp signal using Daubechies wavelets. (a) Original signal. (b) Decomposition 1 (D4 wavelet). (c) Decomposition 2 (D4 wavelet). (d) Decomposition 3 (D4 wavelet). (e) Decomposition 4 (D4 wavelet). (f) Decomposition 5, full decomposition (D4 wavelet). (g) Full decomposition using D6 wavelets. (h) Full decomposition using D12 wavelets. (Note change in scale of the vertical axes.)
one due to the wraparound with the scaling filter. The second iteration produces the transform vector

\[
W^{(2)} = (3.804, 11.804, 19.804, 27.804, 35.804, 43.80, 52.196, 52.981, \\
0, 0, 0, 0, 0, 0, 1.464, -15.321, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11.314)
\]

where we see that the next set of detail coefficients (\(T_{2,0}\) to \(T_{2,7}\)) are zero except those at the end (figure 3.17(e)). There are two nonzero coefficients now due to the edge effect in the smooth coefficients at the previous scale. If we repeat the decomposition to the largest scale we get the fully decomposed signal

\[
W^{(5)} = (87.681, -31.460, 6.405, -29.530, 0, 0, 3.624, -21.149, \\
0, 0, 0, 0, 0, 0, 1.464, -15.321, 0, 0, 0, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 11.314)
\]

As we get ‘edge’ coefficients at all scales, we get no zero coefficients at scales \(m = 4\) and \(m = 5\) where there are respectively only two (\(T_{4,0}, T_{4,1}\)) and one (\(T_{5,0}\)) coefficient.

The full D2 (Haar) decomposition of the ramp signal is

\[
W^{(5)} = (87.7, -45.3, -16.0, -16.0, -5.7, -5.7, -5.7, -5.7, \\
-2.0, -2.0, -2.0, -2.0, -2.0, -0.7, -0.7, -0.7, -0.7, -0.7, -0.7, -0.7, -0.7)
\]

Notice that there are no zero coefficients, as we would expect, since the Haar wavelet has only one vanishing moment and therefore does not suppress linear signals.

On the other hand, all Daubechies wavelets with more than four scaling coefficients have more than two vanishing moments and all can therefore suppress linear signals. For example, the D6 wavelet can suppress mean, linear and quadratic signals. The D6 decomposition of the ramp signal, which employs the scaling coefficients \((c_0, c_1, c_2, c_3, c_4, c_5) = (0.470, 1.141, 0.650, -0.191, -0.121, 0.0498)\), is

\[
W^{(5)} = (87.681, -2.259, -29.201, 25.344, 0.149, 9.909, \\
-22.224, 6.2484, 0, 0, 0, 0, 0, 4.22, -17.636, 4.766, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -15.175, 3.861)
\]

Notice that this time the two end coefficients are nonzero. This is because the filter is longer. It now has six components and hence two end locations of the filter on the discrete signal overlap and wraparound to the beginning of the signal. A D8 decomposition would generate three end coefficients, a D10 would generate four and so on. The full decompositions of the ramp signal using both the D6 and D12 wavelet are shown in figures 3.17(f) and (g) for comparison. The more scaling coefficients the wavelet has, the higher the number of its vanishing moments and hence the higher the degree of polynomial it can suppress. However, the more scaling coefficients that a wavelet has, the larger its support length and hence the less compact it becomes.
This makes it less localized in the time domain and hence less able to isolate singularities in the signal (including edge effects). This is the trade-off which must be considered when selecting the best wavelet for the data analysis task.

### 3.5.1 Filtering

Let us revisit the filtering process described in the last section and shown in figure 3.16. In signal processing, the approximation coefficients at resolution $m$, $S_{m,n}$, are convolved with the lowpass filter. This done by moving the filter along the signal one step at a time. The approximation coefficients are then subsampled (or down-sampled) where every second value is chosen to give the approximation coefficient vector at scale $m + 1$. The approximation coefficients at resolution $m$, $S_{m,n}$, are also convolved with the highpass filter and subsampled in the same way to give the detail signal coefficients at scale $m + 1$. The detail components $T_{m+1,n}$ are kept and the approximation components $S_{m+1,n}$ are again passed through the lowpass and highpass filters to give components $T_{m+2,n}$ and $S_{m+2,n}$. This process is repeated over all scales to give the full decomposition of the signal. Each step in the decomposition filtering process is shown schematically in figure 3.18(a). The sequences $(1/\sqrt{2})c_k$ and $(1/\sqrt{2})b_k$ are contained respectively within the lowpass and highpass filter vectors used within the wavelet decomposition algorithm. The filter coefficients are sometimes referred to as ‘taps’.

For signal reconstruction (equation (3.42)), the filtering process is simply reversed, whereby the components at the larger scales are fed back through the filters.

![Figure 3.18](https://example.com/figure3.18.png)

**Figure 3.18. Schematic of the signal filtering.** (a) Schematic diagram of the filtering of the approximation coefficients to produce the approximation and detail coefficients at successive scales. The subsample symbol $\downarrow$ means take every second value of the filtered signal. HP and LP are, respectively, the highpass and lowpass filters. In practice the filter is moved along one location at a time on the signal, hence filtering plus subsampling corresponds to skipping every second value as shown in figure 3.16. (b) Schematic diagram of the filtering of the approximation and detail coefficients to produce the approximation coefficients at successively smaller scales. The upsample symbol $\uparrow$ means add a zero between every second value of the input vector. The coefficients in the HP' and LP' filters are in reverse order to their counterparts used in the decomposition shown in (a). Figures (a) and (b) represent a subband coding scheme.
This is shown in figure 3.18(b). The approximation and detail components at scale index \( m + 1 \) are first upsampled (zeros are inserted between their values) and then passed through the lowpass and highpass filters respectively. This time, however, the filter coefficients are reversed in order and are shifted back along the signal. An example of the reconstruction filtering is shown in figure 3.19 where the component \( S_{m,5} \) is found from the sequences \( S_{m+1,0} \) and \( T_{m+1,0} \). Note that the leading (right-hand) filter coefficient defines the location of the \( S_{m,n} \) coefficient. Hence the computation of the \( S_{m,0}, S_{m,1} \) and \( S_{m,2} \) coefficients involve components of the filter vector lying off the left-hand end of the upsampled signal at scale \( m + 1 \). These are simply wrapped back around to the right-hand end of the signal.

We can produce a discrete approximation of the scaling function at successive scales if we set all the values of the transform vector to zero except the first one and pass this vector repeatedly back up through the lowpass filter. This is illustrated in the sequence of plots contained in figure 3.20. Figure 3.20(a) shows the initial transform vector, 64 components long, with only the first component set to unity, the rest set to zero, i.e. \( (1, 0, 0, 0, \ldots, 0) \). After the first iteration, four components can be seen. These are equal to \( (c_0/\sqrt{2}, c_1/\sqrt{2}, c_2/\sqrt{2}, c_3/\sqrt{2}, 0, 0, 0, \ldots, 0) \) (notice that this is the decomposition filter). The next iteration (figure 3.20(c)) produces 10 nonzero values in the transform vector, the next, 22 nonzero values, and so on. The transform vector components take on the appearance of the scaling function quite quickly. What we are actually doing is reconstructing \( S_{0,0} \) based on the initial vector \( (1, 0, 0, 0, \ldots, 0) \) equal to the following transform vectors: \( (S_{1,0}, S_{1,1}, S_{1,2}, \ldots, T_{1,0}, T_{1,1}, T_{1,2}, \ldots) \) for figure 3.20(b); \( (S_{2,0}, S_{2,1}, S_{2,2}, \ldots, S_{1,0}, S_{1,1}, S_{1,2}, \ldots, T_{1,0}, T_{1,1}, T_{1,2}, \ldots) \) for figure 3.20(c); \( (S_{3,0}, S_{3,1}, S_{3,2}, \ldots, S_{2,0}, S_{2,1}, S_{2,2}, \ldots, S_{1,0}, S_{1,1}, S_{1,2}, \ldots, T_{1,0}, T_{1,1}, T_{1,2}, \ldots) \) for figure 3.18(d), and so on. Notice that, as we employ wraparound to deal with the edge effects, the final signal shown in figure 3.20(g) is a constant value equal to the signal mean. Figure 3.20(h) shows a high-resolution approximation of the signal.
Figure 3.20. The reconstruction of a Daubechies D4 scaling function. (a) The transform vector (1, 0, 0, 0, 0, ...). (b) One iteration, i.e. $S_{1,0} = 1$. (c) Two iterations, i.e. $S_{2,0} = 1$. (d) Three iterations, i.e. $S_{3,0} = 1$. (e) Four iterations, i.e. $S_{4,0} = 1$. (f) Five iterations, i.e. $S_{5,0} = 1$. (g) Six iterations of the transform vector with 64 components, i.e. $S_{6,0} = 1$. This results in a constant signal equal to the mean due to wraparound. (h) After nine iterations of a transform vector (1, 0, 0, 0, ...) with 4096 components.

Copyright © 2002 IOP Publishing Ltd.
representation of a Daubechies scaling function generated by iterating nine times with an initial transform vector containing 4096 components.

You can verify that the constant signal of figure 3.20(g) stems from the wrap-around employed when reconstructing from the full decomposition wavelet vector with $S_{M,0}$ set to unity. In this case, for the first iteration, adding a zero to get $(S_{M,0},0)$ and running the reconstruction filter over both components leads to two approximation coefficients $(S_{M-1,0}, S_{M-1,1})$ equal to $((c_0 + c_2)/\sqrt{2}, (c_1 + c_3)/\sqrt{2}) = (0.707, 0.707)$. Adding zeros to this vector and iterating again produces four approximation coefficients $(0.5, 0.5, 0.5, 0.5)$ and so on until we obtain the full reconstruction after the sixth iteration which contains constant values of $S_{0,0}$ equal to $(1/\sqrt{2})^6 = 0.125$. We can see this is true for the general case as, from equation (3.74) with $m = 0$ together with equation (3.21), it can be shown that

$$\sum_{k \text{ even}} c_k = \sum_{k \text{ odd}} c_k = 1 \quad (3.83)$$

Hence, the combined effect of wraparound plus the zeros inserted in the reconstruction process causes $S_{M,0}$ to be multiplied by both $(1/\sqrt{2}) \sum_{k \text{ even}} c_k$ and $(1/\sqrt{2}) \sum_{k \text{ odd}} c_k$ in turn to find $S_{M-1,0}$ and $S_{M-1,1}$, which both have the same value simply equal to $S_{M,0}/\sqrt{2}$. This transform vector now has two constant values (the rest zero). The next iteration produces four constant nonzero values and so on. It is easy to prove to yourself that using Daubechies wavelets with different numbers of nonzero scaling coefficients, $N_k$, will always result in constant valued transform vector components.

If we repeat the reconstruction filtering, this time setting the first detail coefficient to unity, we can generate an approximation to a wavelet. Notice that, in doing so, only the first iteration of the reconstruction algorithm requires the reordered scaling coefficient sequence for the wavelet, $b_k$, and subsequent reconstruction filtering then uses the scaling coefficient sequence $c_k$. Figure 3.21 shows the wavelet reconstruction process. From these examples, where a single component was set to unity, we can see that the transform vector components, when passed back through the filters during the reconstruction process, increasingly spread their influence (information) over the transform vector in terms of either discrete scaling functions (if the original component was an approximation coefficient) or discrete wavelet function (if the original component was a detail coefficient). Figures 3.20 and 3.21 illustrate the discrete approximations of the wavelet and scaling functions at various scales implicit within the multiresolution algorithm. The discrete scaling and wavelet functions at scale 1 are, in fact, the filter coefficient vectors $(1/\sqrt{2})b_k$ and $(1/\sqrt{2})c_k$ respectively.

### 3.5.2 Symmlets and Coiflets

Looking back at figure 3.15 we can see that Daubechies wavelets are quite asymmetric. To improve symmetry while retaining simplicity, Daubechies proposed Symmlets as a modification to her original wavelets (also spelled symlets). Symmetric wavelet filters are desirable in some applications, e.g. image coding, as it is argued that our visual system is more tolerant of symmetric errors. In addition, it makes it easier to deal with image boundaries. Daubechies came up with Symmlets by ‘juggling with their phase’ during their construction. Figure 3.22 contains two examples of
Symmlets together with their scaling functions. They have $N_k/2 - 1$ vanishing moments, support length $N_k - 1$ and filter length $N_k$. However, true symmetry (or antisymmetry) cannot be achieved for orthonormal wavelet bases with compact support with one exception: the Haar wavelet which is antisymmetric. Coiflets are

![Figure 3.21. The reconstruction of a Daubechies D4 wavelet. (a) One iteration with only $T_{1,0}$ initially set equal to 1. (b) Two iterations with only $T_{2,0}$ initially set equal to 1. (c) Three iterations with only $T_{3,0}$ initially set equal to 1. (d) Four iterations with only $T_{4,0}$ initially set equal to 1.](image)

![Figure 3.22. Symmlets and their associated scaling functions. (a) S6. (b) S10. (The scaling functions are shown dotted.)](image)

Copyright © 2002 IOP Publishing Ltd.
another wavelet family found by Daubechies. They are also nearly symmetrical and
have vanishing moments for both the scaling function and wavelet: the wavelet has
$N_k/3$ moments equal to zero and the scaling function has $N_k/3 - 1$ moments equal
to zero. They have support length $N_k - 1$ and filter length $N_k$. Three examples of
Coiflet wavelets are shown in figure 3.23. The number of coefficients, $N_k$, used to
define Coiflets increase in multiples of six.

3.6 Translation invariance

Translation invariance (or shift invariance) is an important property associated with
some wavelet transforms but not others. It simply means that if you shift along the
signal all your transform coefficients simply move along by the same amount.
However, for the dyadic grid structure of the discrete wavelet transform this is clearly
not the case: only if you shift along by the grid spacing at that scale do the coefficients
become translation invariant at that scale and below. Even for the discretization of
the continuous wavelet transform, the transform values are translation invariant
only if shifted by any integer multiple of the discrete time steps. This is illustrated
in figure 3.24 where a simple sinusoidal signal is decomposed using the discrete
orthonormal Haar wavelet and a continuous Mexican hat wavelet discretized at
each time step. The original signal shown on the left-hand side of figure 3.24(a) is
composed of 10 cycles of a sinusoid made up of 1024 discrete data points. The
middle plot in figure 3.24(a) contains the Haar transform coefficients of the signal
plotted in their dyadic grid formation. The right-hand plot contains the continuous
wavelet transform using the Mexican hat wavelet. The coarse structure of the
dyadic grid is evident when compared with the smooth, high resolution Mexican
hat transform plot.

We will assume the signal continues indefinitely and shift the signal window
along to see what happens to the transform plots. The variation in the coefficients
of the Haar transform with displacement of the time window along the signal can
be seen by looking down the central plots in the figure. Shifting the window forward
(or back) along the signal by a scale of $2^m$ produces the same coefficients in the
transform plot at and below that scale shifted in time by $2^m$. This can be observed
in figures 3.24(b)–(d) where the size of the shift is indicated in the transform plot of
The coefficient level at and below which the coefficients remain the same as the original is indicated by the arrows at the left-hand side of each transform plot. Shifting the signal by an arbitrary scale (not a power of two) leads to a completely different set of coefficients, as can be seen in figure 3.24(e), where the signal is shifted by an eighth of a sinusoidal cycle (12.8 data points). The translation invariance of the

![Figure 3.24. Translation invariance. (a) Original signal, 1024 data points. (b) Shift by 64 data points. (c) Shift by 128 data points. (d) Shift by 256 data points. (e) Shift by 1/8 cycle.](image-url)
continuous transform plots is obvious by looking down the right-hand figures where, for any arbitrary shift in the signal, the wavelet transform values are simply translated with the shift.

We can make a translation invariant version of the discrete wavelet transform. Known by a variety of names including the redundant, stationary, translation invariant, maximal overlap or non-decimated wavelet transform, we simply skip the subsampling part of the filtering process described in section 3.5.1. This results in the same number of wavelet coefficients generated at each step, equal to the number of signal components $N$. The decomposition is the same as that shown in figure 3.16 except that every value is retained as the filter moves one step at a time along the signal. In addition, the filters have to be stretched through the addition of zeros between coefficients, hence this algorithm is called the ‘à trous’ algorithm from the French ‘with holes’. An average basis inverse can be performed which gives the average of all possible discrete wavelet transform reconstructions over all possible choices of time origin in the signal. This is sometimes useful for statistical applications including denoising. In addition to being translation invariant, the redundant discrete wavelet transform is extendable to signals of arbitrary length. We do not consider the redundant discrete wavelet transform herein, rather the reader is referred elsewhere at the end of this chapter.

### 3.7 Biorthogonal wavelets

For certain applications real, symmetric wavelets are required. One way to obtain symmetric wavelets is to construct two sets of biorthogonal wavelets: $\psi_{m,n}$ and its dual, $\tilde{\psi}_{m,n}$. One set is used to decompose the signal and the other to reconstruct it. For example, we can decompose the signal using $\psi_{m,n}$ as follows:

$$T_{m,n} = \int_{-\infty}^{\infty} x(t) \psi_{m,n}(t) \, dt$$

and perform the inverse transform using $\tilde{\psi}_{m,n}$,

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} T_{m,n} \tilde{\psi}_{m,n}(t)$$

Alternatively, we can decompose the signal using

$$\tilde{T}_{m,n} = \int_{-\infty}^{\infty} x(t) \tilde{\psi}_{m,n}(t) \, dt$$

and reconstruct using

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \tilde{T}_{m,n} \psi(t)$$

Copyright © 2002 IOP Publishing Ltd.
Biorthogonal wavelets satisfy the biorthogonality condition:

\[
\int_{-\infty}^{\infty} \psi_{m,n}(t) \tilde{\psi}_{m',n'}(t) \, dt = \begin{cases} 
1 & \text{if } m = m' \text{ and } n = n' \\
0 & \text{otherwise}
\end{cases} \tag{3.88}
\]

Using biorthogonal wavelets allows us to have perfectly symmetric and antisymmetric wavelets. Further, they allow certain desirable properties to be incorporated separately within the decomposition wavelet and the reconstruction wavelet. For example, \(\psi_{m,n}\) and \(\tilde{\psi}_{m,n}\) can have different numbers of vanishing moments. If \(\psi_{m,n}\) has more vanishing moments than \(\tilde{\psi}_{m,n}\), then decomposition using \(\psi_{m,n}\) suppresses higher order polynomials and aids data compression. Reconstruction with the wavelets \(\tilde{\psi}_{m,n}\) with fewer vanishing moments leads to a smoother reconstruction. This can sometimes be a useful property, for example in image processing. Figure 3.25 shows three examples of compactly supported biorthogonal spline wavelets and their duals commonly used in practice, together with their associated scaling equations.

![Figure 3.25. Biorthogonal spline wavelets.](image)

(a) Biorthogonal (1,5) spline wavelets: \(\phi(t)\), \(\psi(t)\) (left) and \(\tilde{\phi}(t)\), \(\tilde{\psi}(t)\) (right). (b) Biorthogonal (2,4) spline wavelets: \(\phi(t)\), \(\psi(t)\) (left) and \(\tilde{\phi}(t)\), \(\tilde{\psi}(t)\) (right). (c) Biorthogonal (3,7) spline wavelets: \(\phi(t)\), \(\psi(t)\) (left) and \(\tilde{\phi}(t)\), \(\tilde{\psi}(t)\) (right). (Scaling functions shown dotted.)
3.8 Two-dimensional wavelet transforms

In many applications the data set is in the form of a two-dimensional array, for example the heights of a machined surface or natural topography, or perhaps the intensities of an array of pixels making up an image. We may want to perform a wavelet decomposition of such arrays either to compress the data or to carry out a wavelet-based parametric characterization of the data. To perform a discrete wavelet decomposition of a two-dimensional array we must use two-dimensional discrete wavelet transforms. We can simply generate these from tensor products of their one-dimensional orthonormal counterparts. The most common (and simplest) arrangement is to use the same scaling in the horizontal and vertical directions. These give square transforms. (Other forms are possible, e.g. rectangular transforms where the horizontal and vertical scaling vary independently and also two-dimensional transforms which are not simply tensor products. These are outside the scope of this text.) The two-dimensional Haar scaling and wavelet functions are:

- two-dimensional scaling function
  \[ \phi(t_1, t_2) = \phi(t_1)\phi(t_2) \]  
  \[ \psi^h(t_1, t_2) = \phi(t_1)\psi(t_2) \]  
  \[ \psi^v(t_1, t_2) = \psi(t_1)\phi(t_2) \]  
  \[ \psi^d(t_1, t_2) = \psi(t_1)\psi(t_2) \]

Remember that in both the last chapter and this we have been using \( t \) as our independent variable, either temporal or (less common) spatial. In this section \( t_1 \) and \( t_2 \) represent spatial coordinates.

The multiresolution decomposition of the two-dimensional coefficient matrices can be expressed as

- scaling coefficient matrix
  \[ S_{m+1,n_1,n_2} = \frac{1}{2} \sum_{k_1} \sum_{k_2} c_{k_1} c_{k_2} S_{m,n_1+k_1,n_2+k_2} \]  
  \[ T_{m+1,n_1,n_2}^h = \frac{1}{2} \sum_{k_1} \sum_{k_2} b_{k_1} c_{k_2} S_{m,n_1+k_1,n_2+k_2} \]  
  \[ T_{m+1,n_1,n_2}^v = \frac{1}{2} \sum_{k_1} \sum_{k_2} c_{k_1} b_{k_2} S_{m,n_1+k_1,n_2+k_2} \]  
  \[ T_{m+1,n_1,n_2}^d = \frac{1}{2} \sum_{k_1} \sum_{k_2} b_{k_1} b_{k_2} S_{m,n_1+k_1,n_2+k_2} \]

where \( k_1, k_2 \) are the scaling coefficient indices and \( n_1, n_2 \) are the location indices at scale \( m+1 \). (Compare with the one-dimensional case of equations (3.36) and (3.37).) We can simply use discrete versions of the two-dimensional wavelets at scale index 1 to perform...
the multiresolution analysis of the array. The discrete two-dimensional Haar scaling
and wavelet functions in matrix form at scale index $m = 1$ are

$$
\begin{align*}
\text{scaling function} & \quad \text{vertical wavelet} \\
\downarrow & \quad \downarrow \\
\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} & \quad \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \\
\text{horizontal wavelet} & \quad \text{diagonal wavelet} \\
\downarrow & \quad \downarrow \\
\frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} & \quad \frac{1}{2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}
\end{align*}
$$

As these are at scale index 1, they are simply tensor products of the scaling and wavelet filter coefficients $c_k/\sqrt{2}$ and $b_k/\sqrt{2}$. These discrete functions are shown schematically in figure 3.26. Note the $\frac{1}{2}$ factor before each matrix is simply the square of the $1/\sqrt{2}$ factor preceding the corresponding one-dimensional functions. For discrete scaling and wavelet functions at larger scales, this normalization factor becomes $1/2^m$. These four $2 \times 2$ matrices are required for the Haar wavelet decomposition of the two-dimensional array. The general idea of the two-dimensional wavelet decomposition is shown schematically in figure 3.27. The original input array is represented by $X_0$ defined at scale index $m = 0$. As with the one-dimensional case, its components are input as the approximation coefficients at scale index 0, i.e. the matrix $S_0$. After the first wavelet decomposition, a decomposition array is formed at scale index 1 which is split into four distinct submatrices: the vertical detailed

Figure 3.26. The two-dimensional discrete Haar wavelet at scale index 1. (a) Scaling function. (b) Horizontal wavelet. (c) Vertical wavelet. (d) Diagonal wavelet.
components $T^v_1$, the horizontal detailed components $T^h_1$, the diagonal detailed components $T^d_1$ and the remaining approximation components $S_1$. The decomposition array is the same size as the original array. As with the discrete wavelet decomposition of one-dimensional signals, the detail coefficients are subsequently left untouched and the next iteration further decomposes only the approximation components in the

Figure 3.27. Schematic diagram of the matrix manipulation required to perform wavelet decomposition on a two-dimensional grid.
submatrix $S_1$. This results in detail coefficients contained within submatrices $T_v^2$, $T_h^2$ and $T_d^2$ at scale index 2 and approximation coefficients within submatrix $S_2$. This procedure may be repeated $M$ times for a $2^M \times 2^M$ array to get a number of coefficient submatrices $T_v^m$, $T_h^m$ and $T_d^m$ of size $2^{M-m} \times 2^{M-m}$, where $m = 1, 2, \ldots, M$.

Let us look at a very simple example, a matrix with a single nonzero component

$$X_0 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The next iteration uses only the approximation components in the top left-hand quadrant (shown within the dotted box). The four component values of this array

$$W^{(1)} = \begin{bmatrix} 0.5 & 0 & -0.5 & 0 \\ 0 & 0 & 0 & 0 \\ -0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

This array could represent a single pixel turned on in a display or perhaps a spike protruding from a flat surface. We can visualize it in figure 3.28. The two-dimensional Haar multi-resolution decomposition of this array is carried out by scanning over it with each of the four discrete wavelet matrices in turn. This is shown schematically in figure 3.29. On the first iteration (from scale $m = 0$ to 1), the scaling function $2 \times 2$ matrix is scanned over the input array. We require four of these matrices to cover this $4 \times 4$ array. The components of the original array are multiplied in turn by the scaling function array to give the resulting matrix product $S_1$ which is placed in the top right-hand quadrant of the first iteration matrix. Similarly, the vertical wavelet $2 \times 2$ wavelet matrix is scanned over the array and each value is placed in the top right-hand quadrant of the first iteration array. This is illustrated in the $T_v^1$ matrix marked on the right-hand of the figure. The procedure is repeated for the horizontal and diagonal wavelet components, placing the products in the bottom left-hand and bottom right-hand quadrants respectively. The normalization factor of $1/2$ has been left out of the matrix manipulation until the end to aid clarity. Hence, after the first iteration, the resulting wavelet transform matrix is
are then interrogated by each of the scaling and wavelet matrices in turn. This produces the second iteration matrix

\[
W^{(2)} = \begin{bmatrix}
0.25 & 0.25 & -0.5 & 0 \\
0.25 & 0.25 & 0 & 0 \\
-0.5 & 0 & 0.5 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\] (3.93)

This is the second and last iteration we can perform in the decomposition of the 4 × 4 (i.e. \( M = 2 \)) array. This matrix is made up of \( T_1^u, T_1^d, T_2^u, T_2^d \) and, in the top left-hand corner, the approximation component \( S_2 \) which is related to the mean of the original array. Notice that, at each iteration, the resulting matrix has the same energy as the original matrix where the energy is the sum of the squared components.
of the matrix, i.e.

\[ E = \sum_{i=0}^{2^M-1} \sum_{j=0}^{2^M-1} (X_{0,i,j})^2 = \sum_{i=0}^{2^M-1} \sum_{j=0}^{2^M-1} (W_{i,j}^{(m)})^2 \]  

(3.94)

where \(X_{0,i,j}\) and \(W_{i,j}^{(m)}\) are, respectively, the elements of the input and wavelet decomposition matrices located on row \(i\) and column \(j\). \(E\) is equal to 1 for the array considered above. We can easily verify that \(E\) is equal to 1 for the intermediate decomposition matrix as well as the full decomposition matrix. You can confirm this conservation of energy property in all the examples which follow.

Let us look at another simple example, the step shown in figure 3.30(a) and given in matrix form as

\[
X_0 = \begin{bmatrix}
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
\]  

(3.95)

This \(8 \times 8\) array can be decomposed through three iterations using the Haar wavelet as follows:

first decomposition \(W^{(1)}\)

\[
\begin{bmatrix}
4 & 4 & 4 & 4 & 0 & 0 & 0 & 0 \\
4 & 4 & 4 & 4 & 0 & 0 & 0 & 0 \\
3 & 3 & 3 & 3 & 0 & 0 & 0 & 0 \\
2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]  

(3.96a)

Copyright © 2002 IOP Publishing Ltd.
second decomposition $W^{(2)}$

\[
\begin{pmatrix}
8 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\
5 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

(3.96b)

third (full) decomposition $W^{(3)}$

\[
\begin{pmatrix}
13 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

(3.96c)

Note that, after the three iterations, only the approximation component and the components in regions corresponding to the horizontal wavelet decomposition (labelled $T_h^1$, $T_h^2$ and $T_h^3$) contain nonzero elements. This is because the original array only contains a single horizontal feature—the horizontal discontinuity or step. Another thing to notice is that the 64 nonzero elements of the original array have been transformed into only eight nonzero elements, i.e. a compression of the data has taken place. The original arrays, together with the three levels of decomposition, are shown as histograms in figure 3.30.

We can now use the detail coefficients to reconstruct the array at different scales. This is shown in figure 3.31 and is performed using the detail coefficients at each scale of interest. In the same way as we did for the one-dimensional signals earlier in section 3.4, discrete detail arrays $D_h^m$, $D_d^m$ and $D_v^m$ can be reconstructed at the scale of the input array using each of the detail coefficients, $T_h^m$, $T_d^m$ and $T_v^m$. A combined detail can be found at scale index $m$, given by

\[
D_m = D_h^m + D_d^m + D_v^m
\]

(3.97)

As with the one-dimensional case, the array can be represented as a sum of the discrete details plus an array mean.

\[
X_0 = X_M + \sum_{m=1}^{M} D_m
\]

(3.98)

where $X_M$ is the smooth version of the input array at the largest scale index, $M$, expressed at the scale of the input array $m = 0$. For example, to reconstruct the
Figure 3.31. Schematic diagram of the matrix manipulation required to derive the detail matrices at scale index $m = 1$. (a) The three detail coefficient submatrices at scale index 1 for the step. (b) Computing the horizontal discrete detail from the detail coefficients at scale index 1.
These three $4 \times 4$ coefficient submatrices are each expanded into an $8 \times 8$ array corresponding to the horizontal, diagonal and vertical details of the original array using the corresponding discrete wavelets. This is shown schematically in figure 3.31(b) for the horizontal detail at scale index 1, i.e. $D^h_1$. The expansion of the transform coefficients through the discrete Haar wavelet into the detailed components is shown explicitly for two of the $T^h_1$ components in the figure. We can see that the detail matrices $D^h_1$ and $D^v_1$ will have all elements equal to zero as both $T^h_1$ and $T^v_1$ contain only zero elements. Hence the combined detail component matrix is

$$D_1 = D^h_1 + D^d_1 + D^v_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 & -0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(3.99)

Similarly, we can get the detail matrices at scale index 2, $D^h_2$, $D^d_2$ and $D^v_2$, using the scale 2 coefficients, respectively $T^h_2$, $T^d_2$ and $T^v_2$ and the discrete wavelet at scale index 2. This is shown for $D^h_2$ in figure 3.32. Note that the normalization factor at scale 2 is $1/2^2$. The scale 3 coefficient submatrices, $T^h_3$, $T^d_3$ and $T^v_3$, are simply [3], [0] and [0] respectively. It is easily seen that this gives the detail at the largest wavelet

Copyright © 2002 IOP Publishing Ltd.
scale equal to

$$D_3 = \begin{bmatrix}
0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\
0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\
0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\
0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 & 0.375 \\
-0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 \\
-0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 \\
-0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 \\
-0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 & -0.375 \\
\end{bmatrix}$$

(3.100)

where each component is either $3/2^3$ or $-3/2^3$. In fact, this matrix is simply a single large horizontal discrete Haar wavelet scaled by $3/2^3$. The approximation coefficient ($=13$) leads to a matrix of values all equal to the mean of the original array ($=13/2^3$), i.e.

$$X_3 = \begin{bmatrix}
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 & 1.625 \\
\end{bmatrix}$$

(3.101)

Figure 3.33 shows the reconstructions at each scale together with the mean component. From the figure we can see that if we add them all together, $D_1 + D_2 + D_3 + X_3$, we get back the original input array $X_0$.

Figure 3.33. The details of the step array. (a) Original step array $X_0$. (b) $D_1$. (c) $D_2$. (d) $D_3$. (e) $X_3$. (f) Sum of (b), (c), (d) and (e) giving original signal (a).
Figure 3.34. An image and its first two decomposition matrices. (a) Original array $S_0$. (b) Transform array $V^{(1)}$ containing submatrices $S_1$, $T_{v1}$, $T_{h1}$ and $T_{d1}$. (c) Transform array $V^{(2)}$ containing submatrices $S_2$, $T_{v1}$, $T_{h1}$, $T_{d1}$, $T_{v2}$, $T_{h2}$ and $T_{d2}$.

Figure 3.35. Haar decomposition of a surface data set. (a) Original data set 128 x 128 array of surface heights taken from a river bed. (b) Scale index $m = 1$ discrete detail $D_1$. (c) Scale index $m = 2$ discrete detail $D_2$. (d) Scale index $m = 3$ discrete detail $D_3$. (e) Scale index $m = 4$ discrete detail $D_4$. (f) Scale index $m = 5$ discrete detail $D_5$. (g) Scale index $m = 6$ discrete detail $D_6$. (h) Scale index $m = 7$ discrete detail $D_7$. (i) Sum of first three discrete details $D_1 + D_2 + D_3$ (= (a) + (b) + (c) + (d)). (j) Scale index $m = 3$ discrete approximation $X_3 = X_0 - (D_1 + D_2 + D_3)$ (= (a) − (b) − (c) − (d)). (Greyscale used in all images: maximum = white, minimum = black.)
Figures 3.34 and 3.35 show examples of much larger data sets, both using the Haar wavelet transform. Figure 3.34 shows an image (‘Lena’) together with its first two decomposition matrices, where the approximation and detail coefficient sub-matrices can be clearly seen. Figure 3.35 shows an example of the Haar decomposition of a more irregular array. The $128 \times 128$ array shown in figure 3.35(a) contains the heights of a measured rough surface. The details of this array, at scale indices $m = 1–7$, are shown in figures 3.35(b)–(h). Figure 3.35(i) contains the summation of the first three details and figure 3.35(j) shows the resulting approximation at scale index $m = 3$ when these details are subtracted from the original array. The blocky nature of the Haar decomposition is noticeable from the plots. The two-dimensional Haar wavelet is very simple in its form and, as with their one-dimensional counterparts, there are more complex forms of Daubechies wavelets and, of course, many other wavelet families to choose from. Some examples of these wavelets are given in figure 3.36. Using these wavelets will result in overlap of the wavelet at the array edge and therefore would require the use of wraparound or other methods to deal with the data edges.

![Figure 3.36. Examples of two-dimensional orthonormal wavelets: Daubechies, Symmlets and Coiflets.](image)

(a) D2 Haar wavelet. (b) D4. (c) D8. (d) D12. (e) D12, another perspective. (f) D20. (g) Symmlet S8. (h) Coiflet C6. (Wavelets shown viewed at various angles and elevations.)
3.9 Adaptive transforms: wavelet packets

As we saw in chapter 2 (figure 2.29) the resolution of the wavelet transform is not uniform in the time–frequency plane. The Heisenberg boxes expand in frequency and contract in time as fluctuations at smaller and smaller scales are explored. The short term Fourier transform (STFT), on the other hand, covers the time–frequency plane with tiles of constant aspect ratio (figure 2.30). We also looked briefly at matching pursuits which offer another way of extracting time–frequency information. In this section we will consider another method which can adapt to the signal and hence allows for more flexibility in the partitioning of the time–frequency plane: the wavelet packet transform.

Wavelet packet (WP) transforms are a generalization of the discrete wavelet transform. Wavelet packets involve particular linear combinations of wavelets and the wavelet packet decomposition of a signal is performed in a manner similar to the multiresolution algorithm given earlier for the discrete wavelet transform. The difference is that, in the WP signal decomposition, both the approximation and detailed coefficients are further decomposed at each level. This leads to the decomposition tree structure depicted at the top of figure 3.37. Compare this with the schematic of the wavelet decomposition given in figure 3.6. At each stage in the

![Figure 3.37. Schematic diagram of wavelet packet decomposition.](image-url)
wavelet algorithm, the detailed coefficients are simply transferred down, unchanged, to the next level. However, in the wavelet packet algorithm, all the coefficients at each stage are further decomposed. In this way, we end up with an array of wavelet packet coefficients with $M$ levels each with $N$ coefficients. A total of $N$ coefficients from this $M \times N$ array can then be selected to represent the signal. The standard wavelet transform decomposition coefficients are contained within the WP array, shown by the bold boxes in figure 3.37. A new nomenclature is employed in the figure to indicate the operations that have been performed on each set of coefficients. $S$ produces the approximation components of the previous set of coefficients by lowpass filtering, and $T$ the detail components through highpass filtering. We simply add the letter $S$ or $T$ to the left-hand end of the coefficient name to indicate the most recent filtering procedure. $SST_n$, for example, corresponds to the original signal lowpass filtered, then highpass filtered then passed twice through the lowpass filter. Notice also that the subscript contains only the location index $n$. The scaling index $m$ is omitted as it is obviously equal to the number of letters $S$ and $T$ in the coefficient name. As with the original wavelet transform, the number of coefficients at each scale depends upon the scale, with one coefficient in each coefficient group at the largest scale $M$ and $N/2$ coefficients at the smallest scale $m = 1$. Hence, the coefficient index spans $n = 0, 1, \ldots, 2^M - m - 1$.

At each stage in the decomposition, the wavelet packet algorithm partitions the time–frequency plane into rectangles of constant aspect ratio. These become wider (in time) and narrower (in frequency) as the decomposition proceeds. This is shown schematically at the bottom of figure 3.37 for each scale. A variety of tilings of the time–frequency plane is possible using the wavelet packet coefficients. For example, we could keep all the coefficients at a level and discard all the others. This would tile the plane in boxes of constant shape, just like one of those shown at the bottom of figure 3.37. Other tilings are possible, some examples of these being shown in figure 3.38. The standard wavelet transform is just one of all the possible tiling patterns. Figure 3.39 contains the wavelet packet coefficient selections corresponding to the tiling patterns of figure 3.38.

![Figure 3.38. Schematic diagrams of allowable wavelet packet tiling of the time–frequency plane. The right-hand tiling is that used in the wavelet transform algorithm and corresponds to the components contained in the bold boxes shown in the previous figure.](image)

Copyright © 2002 IOP Publishing Ltd.
Figure 3.39. The choice of wavelet packet components leading to the tilings shown in figure 3.38.
The optimal or ‘best’ coefficient selection (hence tiling arrangement) is chosen to represent the signal based on some predefined criterion. This criterion is normally based on an information cost function which aims to retain as much information in a few coefficients as possible. The most common measure of information used is the *Shannon entropy measure*. This is defined for a discrete distribution \( p_i \), \( i = 0, 1, \ldots, N - 1 \), as

\[
S(p) = -\sum_i p_i \log(p_i)
\]  

(3.102)

where, for this case, \( p_i \) are the normalized energies (i.e. squared magnitudes) of the wavelet packet coefficients under consideration. Low entropies occur when the larger coefficient energies are concentrated at only a few discrete locations. The minimum possible entropy of zero occurs when \( p_i = 1 \) for only one value of \( i \), the other probabilities being zero. In this case all the information needed to represent the signal is condensed within a single coefficient. The maximum entropy occurs when there is an equal distribution of coefficient energies. In this case \( p_i = 1/N \) and the signal information is evenly spread throughout all the coefficients. We can see that \( p_i \) acts as a discrete probability distribution of the energies. (More information on the Shannon entropy measure together with an illustrative figure is given in chapter 4, section 4.2.4.)

The set of \( N \) wavelet packet coefficients which contain the least entropy are selected to represent the signal. That is, we want the signal information to be concentrated within as few coefficients as possible. To find these coefficients the WP array, such as the one we saw in figure 3.37, is inspected from the bottom upwards. At each scale, each pair of partitioned coefficient sets (the ‘children’) are compared with those from which they were derived (their ‘parent’). If the combined children’s coefficients have a smaller entropy than those of their parent then they are selected. The entropy of this block, together with the entropy assigned to its pair is less than its parent above hence it is selected. The entropy of this block, plus the entropy assigned to its pair have less entropy than their parent above and hence is selected. The combined entropy of these children is assigned to their parent in order that entropy comparisons can be made further up the tree.

![Wavelet packet coefficient selection](image)

Figure 3.40. Wavelet packet coefficient selection.
Figure 3.41. Wavelet packet decomposition of a simple signal. (a) Signal (top) with wavelet packet decomposition (below). The coefficient values are plotted as heights. The scale indices, $m = 1$ to 6, are given down the left-hand side of the plot. Trace WP contains the best selection of wavelet packets and trace WT contains the wavelet transform decomposition for comparison. (b) The time–frequency tiling associated with each wavelet packet decomposition in (a). Larger coefficient values are plotted darker.
retained. If not, the parent’s coefficients are retained. When the children are selected their entropy value is assigned to their parent in order that subsequent entropy comparisons can be made further up the tree. This is shown schematically in figure 3.40. Once the whole WP array has been inspected in this way, we get an optimal tiling of the time–frequency plane (with respect to the localization of coefficient energies). This tiling provides the best basis for the signal decomposition.

Figure 3.41 (continued). (c) The time–frequency tiling associated with the best wavelet packet decomposition (left) and wavelet decomposition (right). (d) The 16 largest coefficients from (c): wavelet packet decomposition (left) and wavelet decomposition (right). (e) The reconstruction of the signal using only the 16 largest coefficients given in (d): wavelet packet (left) and wavelet (right).

Copyright © 2002 IOP Publishing Ltd.
Figure 3.41 illustrates the wavelet packet method on a simple discrete signal composed of a sampled sinusoid plus a spike. The signal is 64 data points in length. A Haar wavelet is used to decompose the signal. Figure 3.41(a) shows the wavelet packet coefficients below the original signal for each stage in the WP algorithm. The coefficients are displayed as histograms. The bottom two traces contain the coefficients corresponding to the best wavelet packet basis and the ‘traditional’ discrete wavelet basis respectively. The WP tiling of the coefficient energies in the time–frequency plane for each scale is given in figure 3.41(b). The larger coefficient

Figure 3.42. Wavelet packet decomposition of a simple signal using a Daubechies D20 wavelet. (a) The wavelet packet decomposition of the signal shown at the top of figure 3.31(a). (b) The time–frequency tiling associated with the best wavelet packet decomposition (left) and wavelet decomposition (right).
energies are shaded darker in the plot. In figure 3.41(c) the optimal WP tiling is compared with the wavelet transform tiling of the time–frequency plane. The plots in figure 3.41(d) outline the 16 largest coefficients in both time–frequency planes of figure 3.41(c). These are used to reconstruct the signals shown in figure 3.41(e). The 16 largest wavelet packet coefficients contain 98.6% of the signal energy, whereas the 16 wavelet transform coefficients contain 96.5% of the signal energy. The wavelet packet reconstruction using the selected coefficients is visibly smoother than the reconstruction using the traditional wavelet transform coefficients. Figure 3.42 contains the same signal as figure 3.41. However, this time the WP decomposition is performed using a Daubechies D20 wavelet (refer back to figure 3.15). Using this wavelet results in a different tiling of the time–frequency plane for the WP method (compare the left-hand plots of figures 3.42(b) and 3.41(c)). Again, the 16 largest coefficients are used in the signal reconstruction. The oscillatory parts of both reconstructions shown in figure 3.42(d) are visibly smoother than their Haar counterparts in the previous figure (figure 3.41(e)). We expect this as the D20 is more smoothly oscillatory than the Haar. Note, however, comparing figures 3.42(c) and 3.41(d), we see that the signal spike leads to a single high frequency tile for both Haar decompositions but respectively five and four high frequency tiles for the D20 wavelet. The more compact support of the Haar wavelet has allowed for a better localization of the signal spike, but it does makes it less able than the D20 to represent the smooth oscillations in the signal. The energies of the reconstructed signals for the D20 decompositions using only the largest 16 coefficients are 99.8% (WP) and 99.7% (WT), indicating the data compression possibilities of the techniques.

Figure 3.42 (continued). The 16 largest coefficients from figure 3.31(c): wavelet packet decomposition (left) and wavelet decomposition (right). (d) The reconstruction of the signal using only the 16 largest coefficients given in figure 3.31(d): wavelet packet (left) and wavelet (right).
3.10  Endnotes

3.10.1  Chapter keywords and phrases

(You may find it helpful to jot down your understanding of each of them.)

- **discrete wavelet transform**
- **wavelet/detail coefficients**
- **wavelet frames**
- **tight frame**
- **orthonormal basis**
- **dyadic grid**
- **scaling function**
- **approximation coefficients**
- **inverse discrete wavelet transform**
- **discrete approximation**
- **continuous approximation**
- **signal detail**
- **multiresolution**
- **scaling equation**
- **scaling coefficients**
- **compact support**
- **fast wavelet transform**
- **decomposition algorithm**
- **reconstruction algorithm**
- **wraparound**
- **scale index**
- **sequential indexing**
- **level indexing**
- **hard thresholding**
- **soft thresholding**
- **scale thresholding**
- **Daubechies wavelets**
- **upsampled**
- **Symmlets**
- **Coiflets**
- **translation invariance**
- **redundant/stationary/translation**
- **invariant/non-decimated/maximal**
- **overlap/discrete wavelet transform**
- **biorthogonal wavelets**
- **wavelet packet transforms**
- **Shannon entropy measure**

3.10.2  Further resources

discrete orthonormal wavelets: the Haar wavelet and the Daubechies D4 wavelet. There are many other wavelets we have not considered, most notably the Shannon, Meyer, and Battle-Lemarié wavelets. More information on these can be found in, for example, Chui (1997), Mallat (1998), Daubechies (1992) and Blatter (1998). More examples of the multiresolution analysis of simple discrete signals can be found in the book by Walker (1999). Books providing a more mathematical account of discrete wavelet transforms are those by Hernandez and Weiss (1996), Benedetto and Frazier (1994), Chui (1992a,b), Walter (1994) and Percival and Walden (2000). There is also a lot of useful information on the web. The appendix contains a list of useful websites from which to begin a search.

The book by Starck et al. (1998) contains a lot of good introductory material on two-dimensional discrete wavelet transforms, covering many of their practical applications including remote sensing, image compression, multiscale vision models, object detection and multiscale clustering. A nice illustrative paper on the application of two-dimensional wavelet transforms is that by Jiang et al (1999), who investigate the three-dimensional surface of orthopaedic joint prostheses. We considered only square two-dimensional transforms in this chapter, where the horizontal and vertical scalings were kept the same. If they are allowed to vary independently, we get rectangular transforms. Also possible are two-dimensional transforms which are not simply tensor products but wavelets constructed ‘intrinsically’ for higher dimensions (Jawerth and Sweldens, 1994). There are many applications of wavelet packets cited in the rest of this book. Quinquis (1998) provides a nice introduction to wavelet packets, while the paper by Hess-Nielsen and Wickerhauser (1996) and Wickerhauser’s (1994) book provide a more in-depth account.

We mentioned briefly the redundant wavelet transform, a variant of the discrete wavelet transform which produces \( N \) coefficients at each level and is translation invariant. This has been found useful in statistical applications: see for example Coifman and Donoho (1995), Lang et al (1996) and Nason and Silverman (1995). A concise summary of the various wavelet thresholding methods developed over recent years is to be found in the paper by Abramovich et al (2000) together with a comprehensive list of references on the subject. In addition, the book by Ogden (1997) provides a more detailed overview together with numerous examples. The universal threshold method is detailed in the paper by Donoho and Johnstone (1994) together with another global method, the \textit{minimax thresholding method}. Donoho and Johnstone (1995) also developed a scheme which uses the wavelet coefficients at each scale to choose a scale-dependent threshold. This method is known as the \textit{SURE} or \textit{SureShrink} method after ‘Stein’s Unbiased Risk Estimate’ on which it is based. They suggest a \textit{HYBRID} method to be used in practice for decompositions where, at some scales, the wavelet representation is sparse, i.e. a large number of the coefficients are zero (or very near zero). The \textit{HYBRID} method uses the \textit{SURE} method, but defaults to the universal threshold for scales where the data representation is sparse due to the noise overwhelming the little signal contributed from the nonzero coefficients. A number of other methods have been proposed: including those based on cross validation, e.g. Nason (1996), Jansen and Bultheel (1999); Bayesian approaches, e.g. Chipman et al (1997), Vidakovich (1998), Abramovich et al (1998); and the Lorentz curve (Katul and Vidakovich, 1998).

Often we want to compress the signal into as few detail coefficients as possible without losing too much information for data compression applications such as speech and audio transmission. We can do this by finding a wavelet which decomposes the signal into a few large amplitude detail coefficients, which we retain, and many coefficients close to zero, which we discard. How many coefficients we set to zero affects the subsequent quality of the reconstruction. In practice for larger data sets and more suitable wavelets we can get good compression ratios without the loss of significant detail, where the term ‘significant’ is determined by the user. Here we have not considered compression schemes, including quantization and encoding, but rather the reader is referred in the first instance to the simplified explanation of Walker (1999) and the more detailed treatments in Chui (1997) and Mallat (1998).

Note that all examples in this chapter have used wraparound to deal with the signal edges, i.e. the part of the wavelet spilling over the end of the signal is placed back at the start. This is the simplest and one of the most common treatments of edge effects for a finite length signal, and it results in exactly the same number of decomposition coefficients as original signal components. However, it is not always the best for the application. Other methods were shown in chapter 2, figure 2.35. In addition, there are families of boundary wavelets which are intrinsically defined on finite length signals (see for example the discussions in Jawerth and Sweldens (1994) and Chui (1997)). Take particular care when using off-the-shelf wavelet software packages as they may employ other boundary methods as the default setting, for example zero padding, which results in slightly more than $N$ coefficients resulting from the full decomposition. In addition, we have employed one version of the scaling coefficient reordering $b_k$ (equation (3.25)). Again this is very much software dependent and you will find alternative ordering employed by various software packages.