Filter Banks and the Discrete Wavelet Transform*

Ramesh Gopinath

In many applications, one never has to deal directly with the scaling functions or wavelets. Only the coefficients $h(n)$ and $h_1(n)$ in the defining equations and and $c(k)$ and $d_j(k)$ in the expansions , , and need be considered, and they can be viewed as digital filters and digital signals respectively , . While it is possible to develop most of the results of wavelet theory using only filter banks, we feel that both the signal expansion point of view and the filter bank point of view are necessary for a real understanding of this new tool.

1 Analysis – From Fine Scale to Coarse Scale

In order to work directly with the wavelet transform coefficients, we will derive the relationship between the expansion coefficients at a lower scale level in terms of those at a higher scale. Starting with the basic recursion equation from

$$\phi(t) = \sum_n h(n)\ \sqrt{2}\,\phi(2t-n) \tag{1}$$

and assuming a unique solution exists, we scale and translate the time variable to give

$$\phi\left(2^j t-k\right) = \sum_n h(n)\ \sqrt{2}\,\phi\left(2\left(2^j t-k\right)-n\right) = \sum_n h(n)\ \sqrt{2}\,\phi\left(2^{j+1}t-2k-n\right) \tag{2}$$

which, after changing variables $m = 2k+n$, becomes

$$\phi\left(2^j t-k\right) = \sum_m h(m-2k)\ \sqrt{2}\,\phi\left(2^{j+1}t-m\right). \tag{3}$$

If we denote V_j as

$$V_j = \operatorname*{Span}_k\{2^{j/2}\,\phi\left(2^j t-k\right)\} \tag{4}$$

then

$$f(t) \in V_{j+1} \quad \Rightarrow \quad f(t) = \sum_k c_{j+1}(k)\ 2^{(j+1)/2}\,\phi\left(2^{j+1}t-k\right) \tag{5}$$

*Version 1.1: Nov 6, 2012 11:57 am -0600
†http://creativecommons.org/licenses/by/3.0/

is expressible at a scale of $j+1$ with scaling functions only and no wavelets. At one scale lower resolution, wavelets are necessary for the "detail" not available at a scale of j. We have

$$f(t) = \sum_k c_j(k)\ 2^{j/2}\,\phi\left(2^j t - k\right) + \sum_k d_j(k)\ 2^{j/2}\,\psi\left(2^j t - k\right) \tag{6}$$

where the $2^{j/2}$ terms maintain the unity norm of the basis functions at various scales. If $\phi_{j,k}(t)$ and $\psi_{j,k}(t)$ are orthonormal or a tight frame, the j level scaling coefficients are found by taking the inner product

$$c_j(k) = < f(t), \phi_{j,k}(t) > = \int f(t)\ 2^{j/2}\,\phi\left(2^j t - k\right)\ dt \tag{7}$$

which, by using (3) and interchanging the sum and integral, can be written as

$$c_j(k) = \sum_m h(m-2k) \int f(t)\ 2^{(j+1)/2}\,\phi\left(2^{j+1} t - m\right)\ dt \tag{8}$$

but the integral is the inner product with the scaling function at a scale of $j+1$ giving

$$c_j(k) = \sum_m h(m-2k)\ c_{j+1}(m). \tag{9}$$

The corresponding relationship for the wavelet coefficients is

$$d_j(k) = \sum_m h_1(m-2k)\ c_{j+1}(m). \tag{10}$$

1.1 Filtering and Down-Sampling or Decimating

In the discipline of digital signal processing, the "filtering" of a sequence of numbers (the input signal) is achieved by convolving the sequence with another set of numbers called the filter coefficients, taps, weights, or impulse response. This makes intuitive sense if you think of a moving average with the coefficients being the weights. For an input sequence $x(n)$ and filter coefficients $h(n)$, the output sequence $y(n)$ is given by

$$y(n) = \sum_{k=0}^{N-1} h(k)\ x(n-k) \tag{11}$$

There is a large literature on digital filters and how to design them , . If the number of filter coefficients N is finite, the filter is called a Finite Impulse Response (FIR) filter. If the number is infinite, it is called an Infinite Impulse (IIR) filter. The design problem is the choice of the $h(n)$ to obtain some desired effect, often to remove noise or separate signals , .

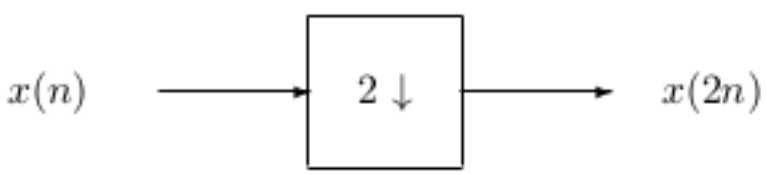

Figure 1: The Down Sampler of Decimator

In multirate digital filters, there is an assumed relation between the integer index n in the signal $x(n)$ and time. Often the sequence of numbers are simply evenly spaced samples of a function of time. Two basic operations in multirate filters are the down-sampler and the up-sampler. The down-sampler (sometimes simply called a sampler or a decimator) takes a signal $x(n)$ as an input and produces an output of $y(n) = x(2n)$. This is symbolically shown in . In some cases, the down-sampling is by a factor other than two and in some cases, the output is the odd index terms $y(n) = x(2n+1)$, but this will be explicitly stated if it is important.

In down-sampling, there is clearly the possibility of losing information since half of the data is discarded. The effect in the frequency domain (Fourier transform) is called aliasing which states that the result of this loss of information is a mixing up of frequency components , . Only if the original signal is band-limited (half of the Fourier coefficients are zero) is there no loss of information caused by down-sampling.

We talk about digital filtering and down-sampling because that is exactly what and (10) do. These equations show that the scaling and wavelet coefficients at different levels of scale can be obtained by convolving the expansion coefficients at scale j by the time-reversed recursion coefficients $h(-n)$ and $h_1(-n)$ then down-sampling or decimating (taking every other term, the even terms) to give the expansion coefficients at the next level of $j-1$. In other words, the scale-j coefficients are "filtered" by two FIR digital filters with coefficients $h(-n)$ and $h_1(-n)$ after which down-sampling gives the next coarser scaling and wavelet coefficients. These structures implement Mallat's algorithm , and have been developed in the engineering literature on filter banks, quadrature mirror filters (QMF), conjugate filters, and perfect reconstruction filter banks , , , , , , and are expanded somewhat in Chapter of this book. Mallat, Daubechies, and others showed the relation of wavelet coefficient calculation and filter banks. The implementation of equations and (10) is illustrated in where the down-pointing arrows denote a decimation or down-sampling by two and the other boxes denote FIR filtering or a convolution by $h(-n)$ or $h_1(-n)$. To ease notation, we use both $h(n)$ and $h_0(n)$ to denote the scaling function coefficients for the dilation equation .

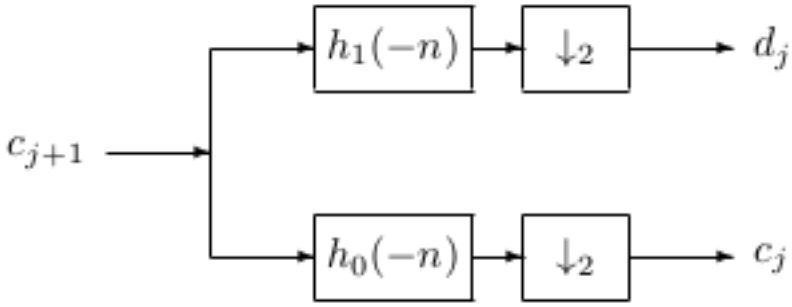

Figure 2: Two-Band Analysis Bank

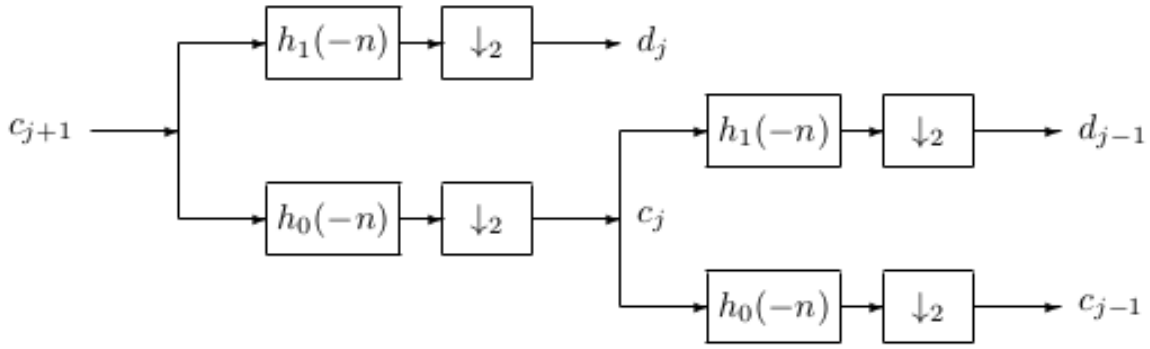

Figure 3: Two-Stage Two-Band Analysis Tree

As we will see in Chapter , the FIR filter implemented by $h(-n)$ is a lowpass filter, and the one implemented by $h_1(-n)$ is a highpass filter. Note the average number of data points out of this system is the same as the number in. The number is doubled by having two filters; then it is halved by the decimation back to the original number. This means there is the possibility that no information has been lost and it will be possible to completely recover the original signal. As we shall see, that is indeed the case. The aliasing occurring in the upper bank can be "undone" or cancelled by using the signal from the lower bank. This is the idea behind perfect reconstruction in filter bank theory , .

This splitting, filtering, and decimation can be repeated on the scaling coefficients to give the two-scale structure in . Repeating this on the scaling coefficients is called *iterating the filter bank*. Iterating the filter bank again gives us the three-scale structure in .

The frequency response of a digital filter is the discrete-time Fourier transform of its impulse response (coefficients) $h(n)$. That is given by

$$H(\omega) = \sum_{n=-\infty}^{\infty} h(n)\, e^{i\omega n}. \tag{12}$$

The magnitude of this complex-valued function gives the ratio of the output to the input of the filter for a sampled sinusoid at a frequency of ω in radians per seconds. The angle of $H(\omega)$ is the phase shift between the output and input.

The first stage of two banks divides the spectrum of $c_{j+1}(k)$ into a lowpass and highpass band, resulting in the scaling coefficients and wavelet coefficients at lower scale $c_j(k)$ and $d_j(k)$. The second stage then divides that lowpass band into another lower lowpass band and a bandpass band. The first stage divides the spectrum into two equal parts. The second stage divides the lower half into quarters and so on. This results in a logarithmic set of bandwidths as illustrated in . These are called "constant-Q" filters in filter bank language because the ratio of the band width to the center frequency of the band is constant. It is also interesting to note that a musical scale defines octaves in a similar way and that the ear responds to frequencies in a similar logarithmic fashion.

For any practical signal that is bandlimited, there will be an upper scale $j = J$, above which the wavelet coefficients, $d_j(k)$, are negligibly small . By starting with a high resolution description of a signal in terms of the scaling coefficients c_J, the analysis tree calculates the DWT

down to as low a resolution, $j = j_0$, as desired by having $J - j_0$ stages. So, for $f(t) \in V_J$, using we have

$$\begin{aligned} f(t) &= \sum_k c_J(k)\ \phi_{J,k}(t) \\ &= \sum_k c_{J-1}(k)\ \phi_{J-1,k}(t) + \sum_k d_{J-1}(k)\ \psi_{J-1,k}(t) \\ f(t) &= \sum_k c_{J-2}(k)\ \phi_{J-2,k}(t) + \sum_k \sum_{j=J-2}^{J-1} d_j(k)\ \psi_{j,k}(t) \\ f(t) &= \sum_k c_{j_0}(k)\ \phi_{j_0,k}(t) + \sum_k \sum_{j=j_0}^{J-1} d_j(k)\ \psi_{j,k}(t) \end{aligned} \tag{13}$$

which is a finite scale version of . We will discuss the choice of j_0 and J further in Chapter .

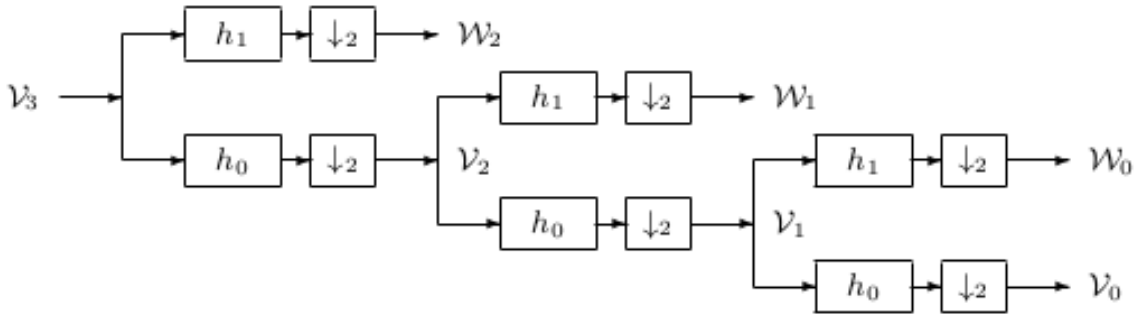

Figure 4: Three-Stage Two-Band Analysis Tree

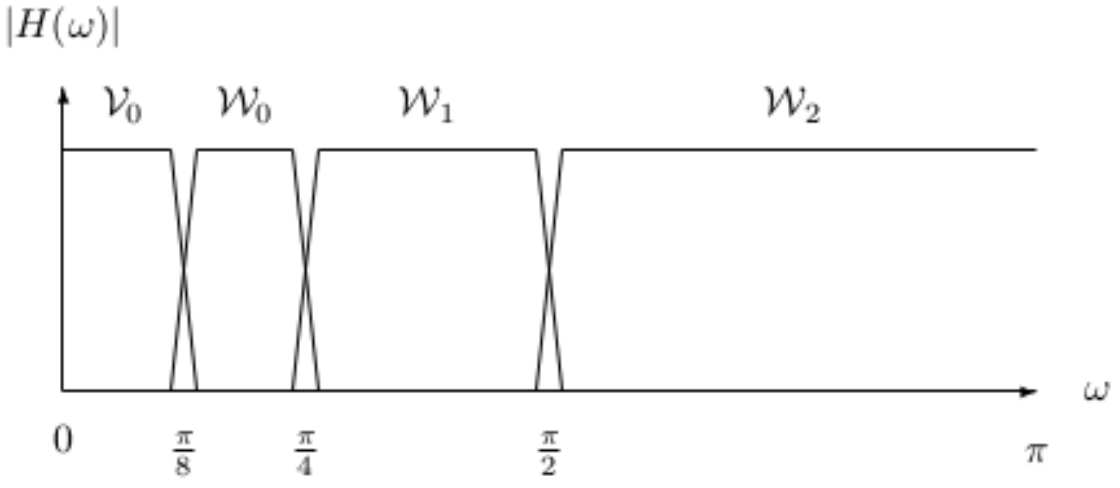

Figure 5: Frequency Bands for the Analysis Tree

2 Synthesis – From Coarse Scale to Fine Scale

As one would expect, a reconstruction of the original fine scale coefficients of the signal can be made from a combination of the scaling function and wavelet coefficients at a coarse resolution. This is derived by considering a signal in the $j+1$ scaling function space $f(t) \in V_{j+1}$. This function can be written in terms of the scaling function as

$$f(t) = \sum_k c_{j+1}(k)\ 2^{(j+1)/2}\,\phi\left(2^{j+1}t - k\right) \tag{14}$$

or in terms of the next scale (which also requires wavelets) as

$$f(t) = \sum_k c_j(k)\ 2^{j/2}\,\phi\left(2^j t - k\right) + \sum_k d_j(k)\ 2^{j/2}\,\psi\left(2^j t - k\right). \tag{15}$$

Substituting (1) and into (15) gives

$$f(t) = \sum_k c_j(k) \sum_n h(n)\ 2^{(j+1)/2}\,\phi\left(2^{j+1}t - 2k - n\right) + \sum_k d_j(k) \sum_n h_1(n)\ 2^{(j+1)/2}\,\phi\left(2^{j+1}t - 2k - n\right). \tag{16}$$

Because all of these functions are orthonormal, multiplying (14) and (16) by $\phi\left(2^{j+1}t - k'\right)$ and integrating evaluates the coefficient as

$$c_{j+1}(k) = \sum_m c_j(m)\ h(k-2m) + \sum_m d_j(m)\ h_1(k-2m). \tag{17}$$

2.1 Filtering and Up-Sampling or Stretching

For synthesis in the filter bank we have a sequence of first up-sampling or stretching, then filtering. This means that the input to the filter has zeros inserted between each of the original terms. In other words,

$$y(2n) = x(n) \quad \text{and} \quad y(2n+1) = 0 \tag{18}$$

where the input signal is stretched to twice its original length and zeros are inserted. Clearly this up-sampling or stretching could be done with factors other than two, and the two equation above could have the $x(n)$ and 0 reversed. It is also clear that up-sampling does not lose any information. If you first up-sample then down-sample, you are back where you started. However, if you first down-sample then up-sample, you are not generally back where you started.

Our reason for discussing filtering and up-sampling here is that is exactly what the synthesis operation (17) does. This equation is evaluated by up-sampling the j scale coefficient sequence $c_j(k)$, which means double its length by inserting zeros between each term, then convolving it with the scaling coefficients $h(n)$. The same is done to the j level wavelet coefficient sequence and the results are added to give the $j+1$ level scaling function coefficients. This structure is illustrated in where $g_0(n) = h(n)$ and $g_1(n) = h_1(n)$. This combining process can be continued to any level by combining the appropriate scale wavelet coefficients. The resulting two-scale tree is shown in .

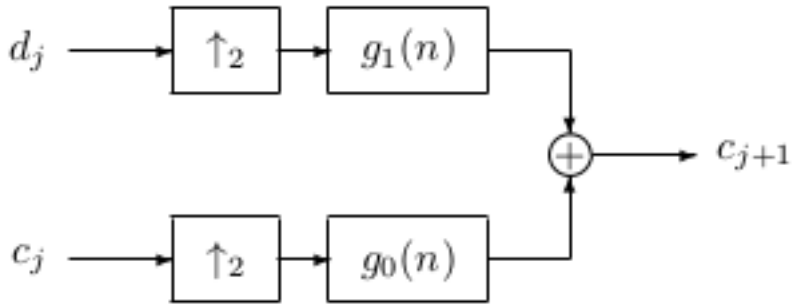

Figure 6: Two-Band Synthesis Bank

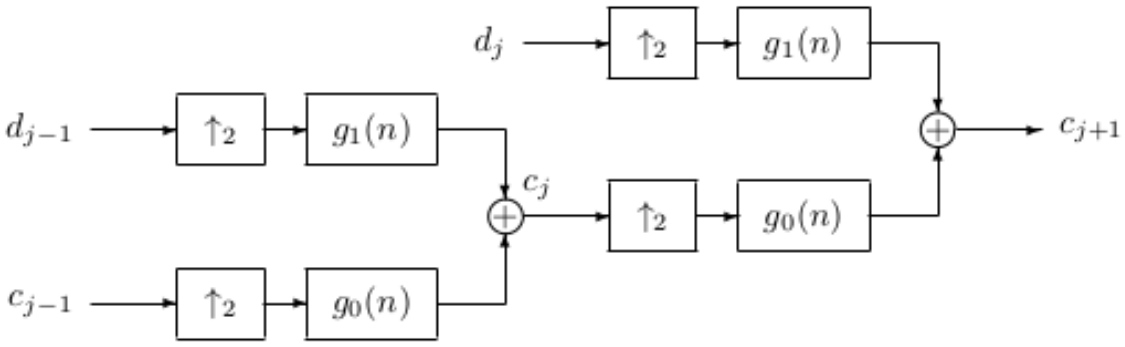

Figure 7: Two-Stage Two-Band Synthesis Tree

3 Input Coefficients

One might wonder how the input set of scaling coefficients c_{j+1} are obtained from the signal to use in the systems of Figures and . For high enough scale, the scaling functions act as "delta functions" with the inner product to calculate the high scale coefficients as simply a sampling of $f(t)$, . If the samples of $f(t)$ are above the Nyquist rate, they are good approximations to the scaling coefficients at that scale, meaning no wavelet coefficients are necessary at that scale. This approximation is particularly good if moments of the scaling function are zero or small. These ideas are further explained in and Chapter .

An alternative approach is to "prefilter" the signal samples to make them a better approximation to the expansion coefficients. This is discussed in .

This set of analysis and synthesis operations is known as Mallat's algorithm , . The analysis filter bank efficiently calculates the DWT using banks of digital filters and down-samplers, and the synthesis filter bank

calculates the inverse DWT to reconstruct the signal from the transform. Although presented here as a method of calculating the DWT, the filter bank description also gives insight into the transform itself and suggests modifications and generalizations that would be difficult to see directly from the wavelet expansion point of view. Filter banks will be used more extensively in the remainder of this book. A more general development of filter banks is presented in .

Although a pure wavelet expansion is possible as indicated in and , properties of the wavelet are best developed and understood through the scaling function. This is certainly true if the scaling function has compact support because then the wavelet is composed of a finite sum of scaling functions given in .

In a practical situation where the wavelet expansion or transform is being used as a computational tool in signal processing or numerical analysis, the expansion can be made finite. If the basis functions have finite support, only a finite number of additions over k are necessary. If the scaling function is included as indicated in or (6), the lower limit on the summation over j is finite. If the signal is essentially bandlimited, there is a scale above which there is little or no energy and the upper limit can be made finite. That is described in Chapter .

4 Lattices and Lifting

An alternative to using the basic two-band tree-structured filter bank is a lattice-structured filter bank. Because of the relationship between the scaling filter $h(n)$ and the wavelet filter $h_1(h)$ given in , some of the calculation can be done together with a significant savings in arithmetic. This is developed in Chapter .

Still another approach to the calculation of discrete wavelet transforms and to the calculations of the scaling functions and wavelets themselves is called "lifting." Although it is related to several other schemes , , , , this idea was first explained by Wim Sweldens as a time-domain construction based on interpolation . Lifting does not use Fourier methods and can be applied to more general problems (e.g., nonuniform sampling) than the approach in this chapter. It was first applied to the biorthogonal system and then extended to orthogonal systems . The application of lifting to biorthogonal is introduced in later in this book. Implementations based on lifting also achieve the same improvement in arithmetic efficiency as the lattice structure do.

5 Different Points of View

5.1 Multiresolution versus Time-Frequency Analysis

The development of wavelet decomposition and the DWT has thus far been in terms of multiresolution where the higher scale wavelet components are considered the "detail" on a lower scale signal or image. This is indeed a powerful point of view and an accurate model for many signals and images, but there are other cases where the components of a composite signal at different scales and/or time are independent or, at least, not details of each other. If you think of a musical score as a wavelet decomposition, the higher frequency notes are not details on a lower frequency note; they are independent notes. This second point of view is more one of the time-frequency or time-scale analysis methods , , , , and may be better developed with wavelet packets (see), M-band wavelets (see), or a redundant representation (see), but would still be implemented by some sort of filter bank.

5.2 Periodic versus Nonperiodic Discrete Wavelet Transforms

Unlike the Fourier series, the DWT can be formulated as a periodic or a nonperiodic transform. Up until now, we have considered a nonperiodic series expansion over $-\infty < t < \infty$ with the calculations made by the filter banks being an on-going string of coefficients at each of the scales. If the input to the filter bank has a certain rate, the output at the next lower scale will be two sequences, one of scaling function coefficients $c_{j-1,k-1}$ and one of wavelet coefficients $d_{j-1,k-1}$, each, after down-sampling, being at half the rate of the input. At the next lower scale, the same process is done on the scaling coefficients to give a total output of three strings, one at half rate and two at quarter rate. In other words, the calculation of the wavelet

transform coefficients is a multirate filter bank producing sequences of coefficients at different rates but with the average number at any stage being the same. This approach can be applied to any signal, finite or infinite in length, periodic or nonperiodic. Note that while the average output rate is the same as the average input rate, the number of output coefficients is greater than the number of input coefficients because the length of the output of convolution is greater than the length of the input.

An alternative formulation that can be applied to finite duration signals or periodic signals (much as the Fourier series) is to make all of the filter bank filters cyclic or periodic convolution which is defined by

$$y(n) = \sum_{\ell=0}^{N-1} h(\ell)\, x(n-\ell), \tag{19}$$

for $n, \ell = 0, 1, \cdots, N-1$ and all indices and arguments are evaluated modulo N. For a length N input at scale $j = J$, we have after one stage two length $N/2$ sequences, after two stages, one length $N/2$ and two length $N/4$ sequences, and so on. If $N = 2^J$, this can be repeated J times with the last stage being length one; one scaling function coefficient and one wavelet coefficient. An example of how the periodic DWT of a length 8 can be seen .

$c_j(\mathrm{k})$	$d_j(\mathrm{k})$	$d_{\mathrm{j+1}}(\mathrm{k})$	$d_{\mathrm{j+1}}(\mathrm{k}+1)$	$d_{\mathrm{j+2}}(\mathrm{k})$	$d_{\mathrm{j+2}}(\mathrm{k}+1)$	$d_{\mathrm{j+2}}(\mathrm{k}+2)$	$d_{\mathrm{j+2}}(\mathrm{k}+3)$

Figure 8: The length-8 DWT vector

The details of this periodic approach are developed in Chapter showing the aliasing that takes place in this system because of the cyclic convolution (19). This formulation is particularly clean because there are the same number of terms in the transform as in the signal. It can be represented by a square matrix with a simple inverse that has interesting structure. It can be efficiently calculated by an FFT although that is not needed for most applications.

For most of the theoretical developments or for conceptual purposes, there is little difference in these two formulations. However, for actual calculations and in applications, you should make sure you know which one you want or which one your software package calculates. As for the Fourier case, you can use the periodic form to calculate the nonperiodic transform by padding the signal with zeros but that wastes some of the efficiency that the periodic formulation was set up to provide.

5.3 The Discrete Wavelet Transform versus the Discrete-Time Wavelet Transform

Two more points of view concern looking at the signal processing methods in this book as based on an expansion of a signal or on multirate digital filtering. One can look at Mallat's algorithm either as a way of calculating expansion coefficients at various scales or as a filter bank for processing discrete-time signals. The first is analogous to use of the Fourier series (FS) where a continuous function is transformed into a discrete sequence of coefficients. The second is analogous to the discrete Fourier transform (DFT) where a discrete function is transformed into a discrete function. Indeed, the DFT (through the FFT) is often used to calculate the Fourier series coefficients, but care must be taken to avoid or minimize aliasing. The difference in these views comes partly from the background of the various researchers (i.e., whether they are "wavelet people" or "filter bank people"). However, there are subtle differences between using the series expansion of the signal (using the discrete wavelet transform (DWT)) and using a multirate digital filter bank on samples

of the signal (using the discrete-time wavelet transform (DTWT)). Generally, using both views gives more insight into a problem than either achieves alone. The series expansion is the main approach of this book but filter banks and the DTWT are also developed in Chapters and .

5.4 Numerical Complexity of the Discrete Wavelet Transform

Analysis of the number of mathematical operations (floating-point multiplications and additions) shows that calculating the DTWT of a length-N sequence of numbers using Mallat's algorithm with filter banks requires $O(N)$ operations. In other words, the number of operations is linear with the length of the signal. What is more, the constant of linearity is relatively small. This is in contrast to the FFT algorithm for calculating the DFT where the complexity is $O(N\,log(N))$ or calculating a DFT directly requires $O\left(N^2\right)$ operations. It is often said that the FFT algorithm is based on a "divide and conquer" scheme, but that is misleading. The process is better described as a "organize and share" scheme. The efficiency (in fact, optimal efficiency) is based on organizing the calculations so that redundant operations can be shared. The cascaded filtering (convolution) and down-sampling of Mallat's algorithm do the same thing.

One should not make too much of this difference between the complexity of the FFT and DTWT. It comes from the DTWT having a logarithmic division of frequency bands and the FFT having a uniform division. This logarithmic scale is appropriate for many signals but if a uniform division is used for the wavelet system such as is done for wavelet packets (see) or the redundant DWT (see Chapter), the complexity of the wavelet system becomes $O(N\,log(N))$.

If you are interested in more details of the discrete wavelet transform and the discrete-time wavelet transform, relations between them, methods of calculating them, further properties of them, or examples, see and Chapter.