CHAPTER

SUBBAND AND WAVELET-BASED CODING 12

CHAPTER OUTLINE

12.1 SUBBAND CODING BASICS

In many applications such as speech and audio analysis, synthesis, and compression, digital filter banks are often used. The filter bank system consists of two stages. The first stage, called the analysis stage, is in the form of filter bank decomposition, in which the signal is filtered into subbands along with a sampling rate decimation; the second stage interpolates the decimated subband signals to reconstruct the original signal. For the purpose of data compression, spectral information from each subband channel can be used to quantize the subband signal efficiently to achieve efficient coding.

Fig. 12.1 illustrates the basic framework for a four-channel filter bank analyzer and synthesizer. At the analysis stage, the input signal $x(n)$ at the original sampling rate f_s is divided via an analysis filter bank into four channels, $x_0(m)$, $x_1(m)$, $x_2(m)$, and $x_3(m)$, each at the decimated sampling rate f_s/M, where $M=4$. For the synthesizer, these four decimated signals are interpolated via a synthesis filter bank. The outputs from all four channels $[\bar{x}_0(n), \bar{x}_1(n), \bar{x}_2(n)$, and $\bar{x}_3(n)]$ of the synthesis filter bank are then combined to reconstruct the original signal $\bar{x}(n)$ at the original sampling rate f_s. Each channel essentially generates a bandpass signal. The decimated signal spectrum for channel 0 can be achieved via a standard downsampling process, while the decimated spectra of other channels can be obtained using the principle of undersampling of bandpass signals with the integer band discussed in Section 11.5, where the inherent frequency aliasing or image properties of decimation and interpolation are involved.

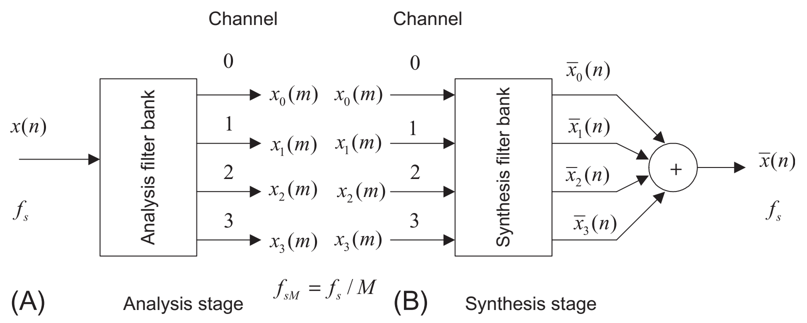


FIG. 12.1

Filter bank framework with an analyzer and a synthesizer.

The theoretical development will follow next. With a proper design of analysis and synthesis filter banks, we are able to achieve perfect reconstruction of the original signal.

Let us examine the spectral details of each band (subband). Fig. 12.2 depicts the spectral information of the analysis and synthesis stages, as shown in Fig. 12.2A and B. $H_0(z)$ and $G_0(z)$ are the analysis and synthesis filters of channel 0, respectively. At the analyzer (C–E), $x(n)$ is bandlimited by a lowpass filter $H_0(z)$ to get $w_0(n)$ and decimated by $M=4$ to obtain $x_0(m)$. At the synthesizer (F–H), $x_0(m)$ is upsampled by a factor of 4 to obtain $\overline{w}_0(n)$ and then goes through an anti-aliasing (synthesis) filter $G_0(z)$ to achieve the lowpass signal $\bar{x}_0(n)$.

Fig. 12.3 depicts the analysis and synthesis stages for channel 1 (see Fig. 12.3A and B). $H_1(z)$ and $G_1(z)$ are the bandpass analysis and synthesis filters, respectively. Similarly, at the analyzer (C–E), $x(n)$

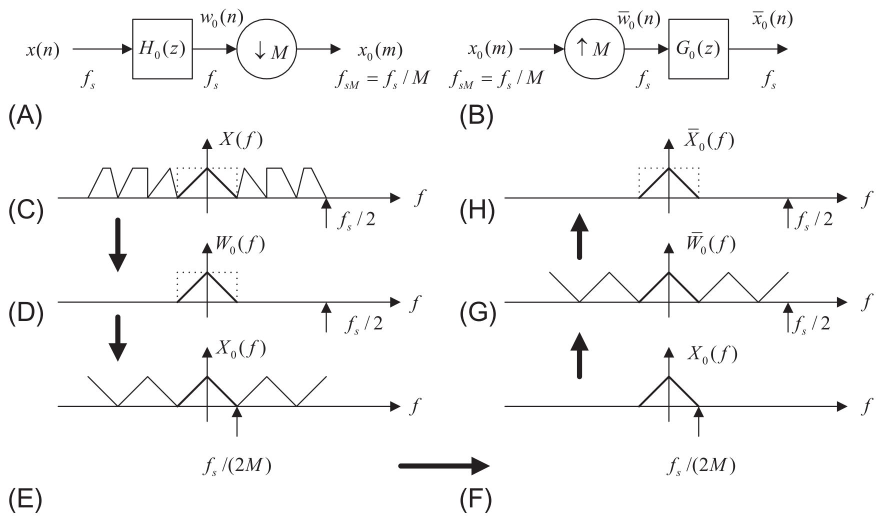


FIG. 12.2

Analysis and synthesis stages for channel 0.

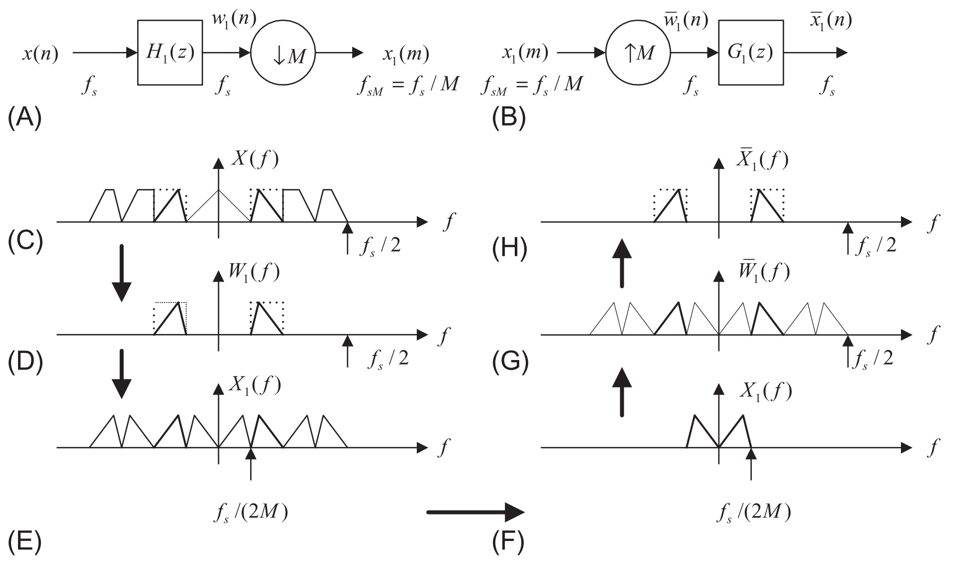


FIG. 12.3

Analysis and synthesis stages for channel 1.

is filtered by a bandpass filter $H_1(z)$ to get $w_1(n)$ and decimated by $M=4$ to obtain $x_1(m)$. Since the low-frequency edge of $W_1(z)$ is $f_c/B=1=$odd number, where $f_c=f_s/(2M)=B, f_c$ corresponds to the carrier frequency and B is the baseband bandwidth as described in Section 11.5, the reversed spectrum in the baseband occurs as shown in Fig. 12.3E. However, this is not a problem, since at the synthesizer as shown in Fig. 12.3 F and G, the spectral reversal occurs again so that $\overline{W}_1(z)$ will have the same spectral components as $W_1(z)$ at the analyzer. After $\overline{w}_1(n)$ goes through the anti-aliasing (synthesis) filter $G_1(z)$, we achieve the reconstructed bandpass signal $\overline{x}_1(n)$.

Fig. 12.4 describes the analysis and synthesis stages for channel 2. At the analyzer (C–E), $x(n)$ is filtered by a bandpass filter $H_2(z)$ to get $w_2(n)$ and decimated by $M=4$ to obtain $x_2(m)$. Similarly, considering the low-frequency edge of $W_2(z)$ as $f_c=2(f_s/(2M))=2B$, $f_c/B=2=$even, we obtain the non-reversed spectrum in the baseband as shown in Fig. 12.4F. At the synthesizer shown in Fig. 12.4G, the spectrum $\overline{W}_2(z)$ has the same spectral components as $W_2(z)$ at the analyzer. After $\overline{w}_2(n)$ is filtered by the synthesis bandpass filter, $G_2(z)$, we get the reconstructed bandpass signal $\overline{x}_2(n)$.

The process in channel 3 is similar to that in channel 1 with the spectral reversal effect and is illustrated in Fig. 12.5.

Now let us examine the theory. Without quantization of subband channels, perfect reconstruction of the filter banks (see Fig. 12.1) depends on the analysis and syntheses filter effects. To develop the perfect reconstruction required for the analysis and synthesis filters, consider a signal in a single channel flowing up to the synthesis filter in general as depicted in Fig. 12.6.

As shown in Fig. 12.6, $w(n)$ is the output signal from the analysis filter at the original sampling rate, that is,

$$W(z)=H(z)X(z). \tag{12.1}$$

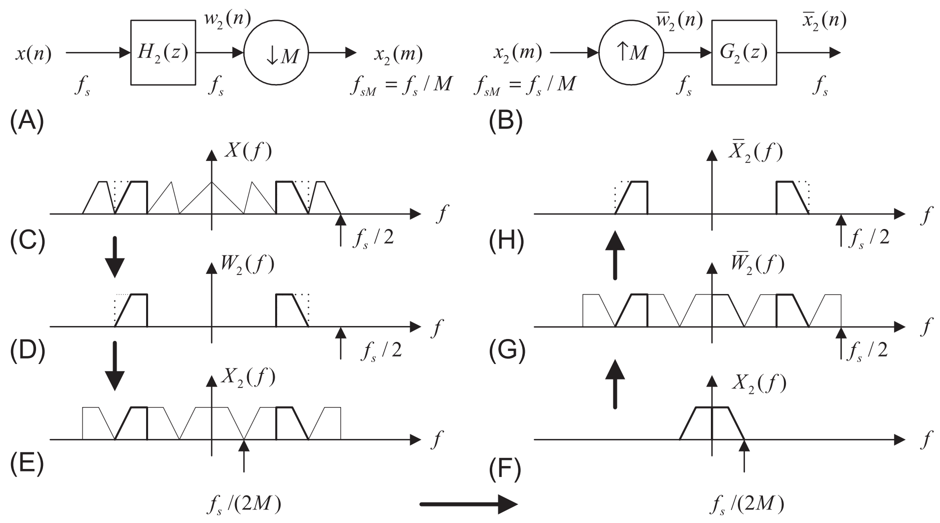


FIG. 12.4

Analysis and synthesis stages for channel 2.

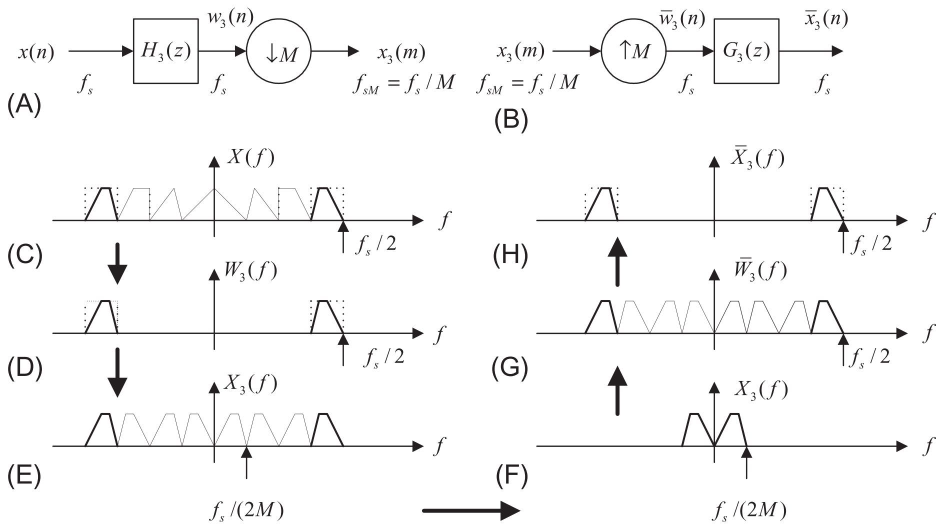


FIG. 12.5

Analysis and synthesis stages for channel 3.

$x_d(m)$ is the downsampled version of $w(n)$ while $\overline{w}(n)$ is the interpolated version of $w(n)$ prior to the synthesis filter and can be expressed as

$$\overline{w}(n) = \begin{cases} w(n) & n = 0, M, 2M, \dots \\ 0 & \text{otherwise} \end{cases}. \tag{12.2}$$

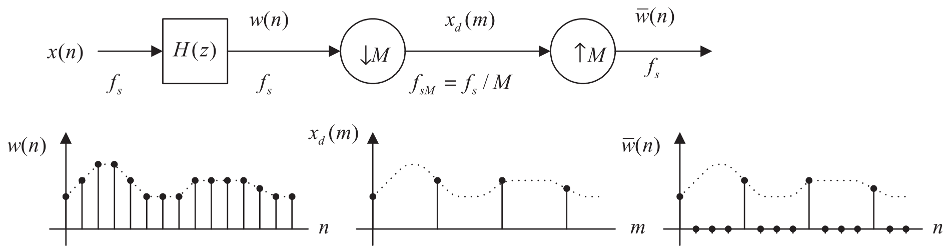


FIG. 12.6

Signal flow in one channel.

Using a delta function $\delta(n)$, that is, $\delta(n)=1$ for $n=0$ and $\delta(n)=0$ for $n \neq 0$, we can write $\bar{w}(n)$ as

$$\bar{w}(n) = \left[\sum_{k=0}^{\infty} \delta(n-kM)\right] w(n) = i(n)w(n). \tag{12.3}$$

where $i(n)$ is defined as

$$i(n) = \sum_{k=0}^{\infty} \delta(n-kM) = \delta(n) + \delta(n-M) + \delta(n-2M) + \cdots.$$

Clearly, $i(n)$ is a periodic function (impulse train with a period of M samples) as shown in Fig. 12.7.

We can determine the discrete Fourier transform of the impulse train with a period of M samples as

$$I(k) = \sum_{n=0}^{M-1} i(n) e^{-j\frac{2\pi kn}{M}} = \sum_{n=0}^{M-1} \delta(n) e^{-j\frac{2\pi kn}{M}} = 1. \tag{12.4}$$

Hence, using the inverse of discrete Fourier transform, $i(n)$ can be expressed as

$$i(n) = \frac{1}{M}\sum_{k=0}^{M-1} I(k) e^{j\frac{2\pi kn}{M}} = \frac{1}{M}\sum_{k=0}^{M-1} e^{j\frac{2\pi kn}{M}}. \tag{12.5}$$

Substituting Eq. (12.5) into Eq. (12.3) leads to

$$\bar{w}(n) = \frac{1}{M}\sum_{k=0}^{M-1} w(n) e^{j\frac{2\pi kn}{M}}. \tag{12.6}$$

Applying the z-transform to Eq. (12.6), we achieve the fundamental relationship between $W(z)$ and $\overline{W}(z)$:

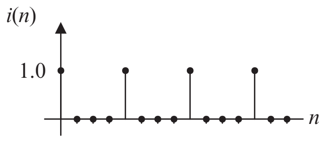


FIG. 12.7

Impulse train.

$$
\begin{aligned}
\overline{W}(z) &= \frac{1}{M}\sum_{k=0}^{M-1}\sum_{n=0}^{\infty} w(n)e^{j\frac{2\pi kn}{M}}z^{-n} = \frac{1}{M}\sum_{k=0}^{M-1}\sum_{n=0}^{\infty} w(n)\left(e^{-j\frac{2\pi k}{M}}z\right)^{-n} \\
&= \frac{1}{M}\sum_{k=0}^{M-1} W\left(e^{-j\frac{2\pi k}{M}}z\right) \\
&= \frac{1}{M}\left[W\left(e^{-j\frac{2\pi\times 0}{M}}z\right) + W\left(e^{-j\frac{2\pi\times 1}{M}}z\right) + \cdots + W\left(e^{-j\frac{2\pi\times(M-1)}{M}}z\right)\right].
\end{aligned}
\tag{12.7}
$$

Eq. (12.7) indicates that the signal spectrum $\overline{W}(z)$ before the synthesis filter is an average of the various modulated spectrum $W(z)$. Note that both $\overline{W}(z)$ and $W(z)$ are at the original sampling rate f_s. We will use this result for further development in the following section.

12.2 SUBBAND DECOMPOSITION AND TWO-CHANNEL PERFECT RECONSTRUCTION-QUADRATURE MIRROR FILTER BANK

To explore Eq. (12.7), let us begin with a two-band case as illustrated in Fig. 12.8.

Substituting $M = 2$ in Eq. (12.7), it follows that

$$
\overline{W}(z) = \frac{1}{2}\sum_{k=0}^{1} W\left(e^{-j\frac{2\pi k}{2}}z\right) = \frac{1}{2}[W(z) + W(-z)]. \tag{12.8}
$$

Applying for each band in Fig. 12.8 by substituting Eq. (12.1) into Eq. (12.8), we have

$$
Y_0(z) = \frac{1}{2}G_0(z)(H_0(z)X(z) + H_0(-z)X(-z)). \tag{12.9}
$$

$$
Y_1(z) = \frac{1}{2}G_1(z)(H_1(z)X(z) + H_1(-z)X(-z)). \tag{12.10}
$$

Since the synthesized signal $\overline{X}(z)$ is a sum of $Y_0(z)$ and $Y_1(z)$, it can be expressed as

$$
\begin{aligned}
\overline{X}(z) &= \frac{1}{2}(G_0(z)H_0(z) + G_1(z)H_1(z))X(z) \\
&\quad + \frac{1}{2}(G_0(z)H_0(-z) + G_1(z)H_1(-z))X(-z) \\
&= A(z)X(z) + S(z)X(-z).
\end{aligned}
\tag{12.11}
$$

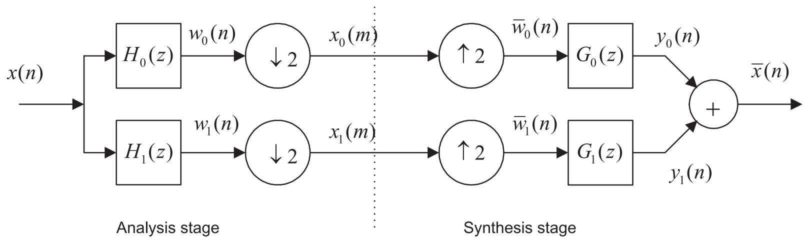


FIG. 12.8

Two-band filter bank system.

For perfect reconstruction, the recovered signal $\bar{x}(n)$ should be a scaled and delayed version of the original signal $x(n)$, that is, $\bar{x}(n) = cx(n-n_0)$. Hence, to achieve a perfect reconstruction, it is required that

$$S(z) = \frac{1}{2}(G_0(z)H_0(-z) + G_1(z)H_1(-z)) = 0 \tag{12.12}$$

$$A(z) = \frac{1}{2}(G_0(z)H_0(z) + G_1(z)H_1(z)) = cz^{-n_0}, \tag{12.13}$$

where c is the constant while n_0 is the delay introduced by the analysis and synthesis filters.

Forcing $S(z) = 0$ leads to the following relationship:

$$\frac{G_0(z)}{G_1(z)} = -\frac{H_1(-z)}{H_0(-z)}. \tag{12.14}$$

It follows that

$$G_0(z) = -H_1(-z) \tag{12.15}$$

$$G_1(z) = H_0(-z). \tag{12.16}$$

Substituting $G_0(z)$ and $G_1(z)$ into Eq. (12.13) gives

$$A(z) = \frac{1}{2}(H_0(-z)H_1(z) - H_0(z)H_1(-z)). \tag{12.17}$$

Assume $H_0(z)$ and $H_1(z)$ be N-tap FIR filters, where N is even, and let

$$H_1(z) = z^{-(N-1)}H_0\left(-z^{-1}\right) \tag{12.18}$$

and note that

$$H_1(-z) = -z^{-(N-1)}H_0\left(z^{-1}\right). \tag{12.19}$$

Substituting Eqs. (12.18) and (12.19) in Eq. (12.17), we can simplify Eq. (12.17) as

$$A(z) = \frac{1}{2}z^{-(N-1)}\left(H_0(z)H_0\left(z^{-1}\right) + H_0(-z)H_0\left(-z^{-1}\right)\right). \tag{12.20}$$

Finally, for perfect reconstruction, it requires that

$$H_0(z)H_0\left(z^{-1}\right) + H_0(-z)H_0\left(-z^{-1}\right) = R(z) + R(-z) = \text{constant}, \tag{12.21}$$

where

$$R(z) = H_0(z)H_0\left(z^{-1}\right) = a_{N-1}z^{N-1} + a_{N-2}z^{N-2} + \cdots + a_0 z^0 + \cdots + a_{N-1}z^{-(N-1)} \tag{12.22}$$

$$R(-z) = H_0(-z)H_0\left(-z^{-1}\right) = -a_{N-1}z^{N-1} + a_{N-2}z^{N-2} + \cdots + a_0 z^0 + \cdots - a_{N-1}z^{-(N-1)}. \tag{12.23}$$

It is important to note that the sum of $R(z)+R(-z)$ only consists of even order of powers of z, since the terms with odd powers of z cancel each other. Using algebraic simplification, we conclude that the coefficients of $R(z) = H(z)H(z^{-1})$ are essentially samples of the autocorrelation function given by

$$\rho(n) = \sum_{k=0}^{N-1} h_0(k)h_0(k+n) = \rho(-n) = h_0(n) \odot h_0(n), \tag{12.24}$$

where $\odot$ denotes the correlation operation. Hence, we require $\rho(n)=0$ for $n=$even and $n\neq 0$, that is,

$$\rho(2n)=\sum_{k=0}^{N-1} h_0(k)h_0(k+2n)=0. \tag{12.25}$$

For normalization, when $n=0$, we require

$$\sum_{k=0}^{N-1} |h_0(k)|^2 = 0.5. \tag{12.26}$$

We then obtain the filter design constraint as

$$\rho(2n)=\sum_{k=0}^{N-1} h_0(k)h_0(k+2n)=\delta(n). \tag{12.27}$$

For a two-band filter bank, $h_0(k)$ and $h_1(k)$ are designed as lowpass and highpass filters, respectively, which are essentially the quadrature mirror filters (QMF). Their expected frequency responses must satisfy Eq. (12.28) and are shown in Fig. 12.9.

$$\left|H_0\left(e^{j\Omega}\right)\right|^2+\left|H_1\left(e^{j\Omega}\right)\right|^2=1. \tag{12.28}$$

Eq. (12.28) implies that

$$R(z)+R(-z)=1. \tag{12.29}$$

To verify Eq. (12.29), we use

$$\left|H\left(e^{j\Omega}\right)\right|^2=H\left(e^{j\Omega}\right)H\left(e^{-j\Omega}\right)=H(z)H\left(z^{-1}\right)\Big|_{z=e^{j\Omega}}.$$

Eq. (12.28) becomes

$$H_0(z)H_0\left(z^{-1}\right)+H_1(z)H_1\left(z^{-1}\right)\Big|_{z=e^{j\Omega}}=1,$$

which is equivalent to

$$H_0(z)H_0\left(z^{-1}\right)+H_1(z)H_1\left(z^{-1}\right)=1.$$

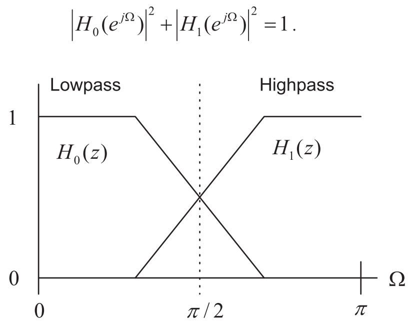


FIG. 12.9

Frequency responses for quadrature mirror filters.

From Eq. (12.18), we can verify that

$$H_1(z)H_1\left(z^{-1}\right) = H_0(-z)H_0\left(-z^{-1}\right).$$

Finally, we see that

$$H_0(z)H_0\left(z^{-1}\right) + H_1(z)H_1\left(z^{-1}\right) = H_0(z)H_0\left(z^{-1}\right) + H_0(-z)H_0\left(-z^{-1}\right) = R(z) + R(-z) = 1.$$

Once the lowpass analysis filter $H_0(z)$ is designed, the highpass filter can be obtained using the developed relationship in Eq. (12.18). The key equations are summarized as follows:

Filter design constraint equations for the lowpass filter $H_0(z)$:

$$\begin{aligned}&R(z) = H_0(z)H_0(z^{-1})\\&R(z) + R(-z) = 1\\&\rho(2n) = 0.5\delta(n).\end{aligned}$$

Equations for the other filters:

$$\begin{aligned}&H_1(z) = z^{-(N-1)}H_0(-z^{-1})\\&G_0(z) = -H_1(-z)\\&G_1(z) = H_0(-z).\end{aligned}$$

Design of the analysis and synthesis filters to satisfy the above conditions is very challenging. Smith and Barnwell (1984) were first to show that perfect reconstruction in a two-band filter bank is possible when the linear phase of the FIR filter requirement is relaxed. The Smith-Barnwell filters are called the conjugate quadrature filters (PR-CQF). 8- and 16-tap PR-CQF coefficients are listed in Table 12.1. As illustrated in Table 12.1, the filter coefficients are not symmetric; hence, the obtained analysis filter

Table 12.1 Smith-Barnwell PR-CQF Filters

8 Taps	16 Taps
0.0348975582178515	0.02193598203004352
−0.01098301946252854	0.001578616497663704
−0.06286453934951963	−0.06025449102875281
0.223907720892568	−0.0118906596205391
0.556856993531445	0.137537915636625
0.357976304997285	0.05745450056390939
−0.02390027056113145	−0.321670296165893
−0.07594096379188282	−0.528720271545339
	−0.295779674500919
	0.0002043110845170894
	0.02906699789467 96
	−0.03533486088708146
	−0.006821045322743358
	0.02606678468264118
	0.001033363491944126
	−0.01435930957477529

does not have a linear phase. The detailed design of Smith-Barnwell filters can be found in their research paper (Smith and Barnwell, 1984) and the design of other types of analysis and synthesis filters can be found in Akansu and Haddad (2001).

Now let us verify the filter constraint in the following example.

EXAMPLE 12.1

Use the 8-tap PR-CQF coefficients (Table 12.1) and MATLAB to verify the following conditions:

$$\rho(2n) = \sum_{k=0}^{N-1} h_0(k)h_0(k+2n) = 0.5\delta(n)$$

$$R(z) + R(-z) = 1,$$

and plot the magnitude frequency responses of the analysis and synthesis filters.

Solution:

Since $\rho(n) = \sum_{k=0}^{N-1} h_0(k)h_0(k+n)$, we perform the following:

$$\text{For } n = 0,\ \rho(0) = \sum_{k=0}^{8-1} h_0(k)h_0(k) = h_0^2(0) + h_0^2(1) + \cdots + h_0^2(7) = 0.5$$

$$\text{For } n = 1,\ \rho(1) = \sum_{k=0}^{8-1} h_0(k)h_0(k+1) = h_0(0)h_0(1) + h_0(1)h_0(2) + \cdots + h_0(6)h_0(7) = 0.3035$$

$$\text{For } n = -1,\ \rho(-1) = \sum_{k=0}^{8-1} h_0(k)h_0(k-1) = h_0(1)h_0(0) + h_0(2)h_0(1) + \cdots + h_0(7)h_0(6) = 0.3035$$

$$\text{For } n = 2,\ \rho(2) = \sum_{k=0}^{8-1} h_0(k)h_0(k+2) = h_0(0)h_0(2) + h_0(1)h_0(3) + \cdots + h_0(5)h_0(7) = 0.0$$

$$\text{For } n = -2,\ \rho(-2) = \sum_{k=0}^{8-1} h_0(k)h_0(k-2) = h_0(2)h_0(0) + h_0(3)h_0(1) + \cdots + h_0(7)h_0(5) = 0.0.$$

....

We can easily verify that $\rho(n) = 0$ for $n \neq 0$ and $n =$ even number.

Next, we use the MATLAB built-in function **xcorr()** to compute the autocorrelation coefficients. The results are listed as follows:

```
>>h0=[0.0348975582178515 -0.01098301946252854 -0.06286453934951963 …
   0.223907720892568 0.556856993531445 0.357976304997285 …
   -0.02390027056113145 -0.07594096379188282];
>>p=xcorr(h0,h0)
p = -0.0027 -0.0000 0.0175 0.0000 -0.0684 -0.0000 0.3035 0.5000
   0.3035 -0.0000 -0.0684 0.0000 0.0175 -0.0000 -0.0027
```

We can observe that there are 15 coefficients. The middle one is $\rho(0) = 0.5$ and. $\rho(\pm 2) = \rho(\pm 4) = \rho(\pm 6) = 0$ as well as $\rho(\pm 1) = 0.3035$, $\rho(\pm 3) = -0.0684$, $\rho(\pm 5) = 0.0175$, and $\rho(\pm 7) = -0.0027$.

Next, we write.

$$R(z) = -0.0027z^7 - 0.0000z^6 + 0.0175z^5 + 0.0000z^4 - 0.0684z^3 - 0.0000z^2 + 0.3035z^1 + 0.5000z^0 + 0.3035z^{-1} - 0.0000z^{-2} - 0.0684z^{-3} + 0.0000z^{-4} + 0.0175z^{-5} - 0.0000z^{-6} - 0.0027z^{-7}.$$

Substituting $z = -z$ in $R(z)$, it yields:

$$R(-z) = 0.0027z^7 - 0.0000z^6 - 0.0175z^5 + 0.0000z^4 + 0.0684z^3 - 0.0000z^2 - 0.3035z^1 + 0.5000z^0 - 0.3035z^{-1} - 0.0000z^{-2} + 0.0684z^{-3} + 0.0000z^{-4} - 0.0175z^{-5} - 0.0000z^{-6} + 0.0027z^{-7}.$$

Clearly, by adding the expressions $R(z)$ and $R(-z)$, we can verify that

$$R(z) + R(-z) = 1.$$

Using MATLAB, the PR-CQF frequency responses are plotted and shown in Fig. 12.10.

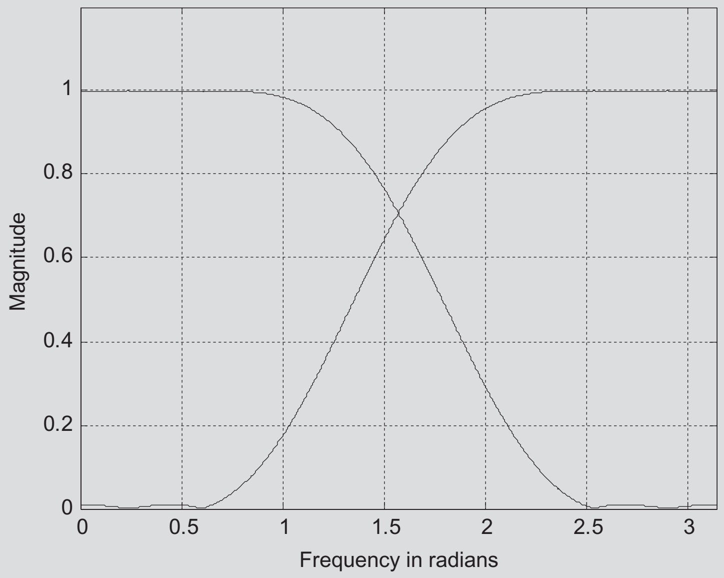


FIG. 12.10

Magnitude frequency responses of the analysis and synthesis filters in Example 12.1.

Fig. 12.11 shows the perfect reconstruction for performing the two-band system shown in Fig. 12.8 using two-band CQF filters. The MATLAB program is listed in Program 12.1, in which the quantization is deactivated. Since the obtained signal-to-noise ratio (SNR) = 135.5803 dB, a perfect reconstruction is achieved. Note that both $x_0(m)$ and $x_1(m)$ have half of the data samples, where $x_0(m)$ contains low-frequency components with more signal energy while $x_1(m)$ possesses high-frequency components with less signal energy.

MATLAB Program 12.1. Two-band subband system implementation.

```
% This program is for implementing the analysis and synthesis using two subbands.
close all; clear all;clc
%Smith-Barnwell PR-CQF 8-taps
h0=[0.0348975582178515-0.01098301946252854 -0.06286453934951963 ...
0.223907720892568 0.556856993531445 0.357976304997285 ...
-0.02390027056113145 -0.07594096379188282];
% Read data file 'orig.dat' with a sampling rate of 8 kHz
load orig.dat; % Load speech data
M=2; % Downsample factor
N=length(h0); PNones=ones(1,N); PNones(2:2:N)=-1;
h1=h0.*PNones; h1=h1(N:-1:1);
g0=-h1.*PNones; g1=h0.*PNones;
disp('check R(z)+R(-z)≥');
xcorr(h0,h0)
sum(h0.*h0)
w=0:pi/1000:pi;
fh0=freqz(h0,1,w); fh1=freqz(h1,1,w);
plot(w,abs(fh0),'k',w,abs(fh1),'k');grid; axis([0 pi 0 1.2]);
xlabel('Frequency in radians');ylabel('Magnitude')
figure
speech=orig;
%Analysis
sb_low=filter(h0,1,speech); sb_high=filter(h1,1,speech);
% Downsampling
sb_low=sb_low(1:M:length(sb_low)); sb_high=sb_high(1:M:length(sb_high));
% Quantization
sb_low=round((sb_low/2^15)*2^9)*2^(15-9); %Quantization with 10 bits
sb_high=round((sb_high/2^15)*2^5)*2^(15-5); % Quantization with 6 bits
% Syntheiss
low_sp=zeros(1,M*length(sb_low)); % Upsampling
low_sp(1:M:length(low_sp))=sb_low;
high_sp=zeros(1,M*length(sb_high)); high_sp(1:M:length(high_sp))=sb_high;
low_sp=filter(g0,1,low_sp); high_sp=filter(g1,1,high_sp);
rec_sig=2*(low_sp+high_sp);
% Signal alignment for SNR caculations
speech=[zeros(1,N-1) speech]; %Align the signal
subplot(4,1,1);plot(speech);grid,ylabel('x(n)');axis([0 20,000-20,000 20,000]);
subplot(4,1,2);plot(sb_low);grid,ylabel('0(m)'); axis([0 10,000-20,000 20,000]);
subplot(4,1,3);plot(sb_high);grid, ylabel('1(m)'); axis([0 10,000-2000 2000]);
subplot(4,1,4);plot(rec_sig);grid, ylabel('xbar(n)'),xlabel('Sample Number');
axis([0 20,000-20,000 20,000]);
NN=min(length(speech),length(rec_sig));
err=rec_sig(1:NN)-speech(1:NN);
SNR=sum(speech.*speech)/sum(err.*err);
disp('PR reconstruction SNR dB≥');
SNR=10*log10(SNR)
```

This two-band composition method can easily be extended to a multiband filter bank using a binary tree structure. Fig. 12.12 describes a four-band implementation. As shown in Fig. 12.12, the filter banks divide an input signal into two equal subbands, resulting in the low (L) and high (H) bands using PR-QMF. This two-band PR-QMF again splits L and H into half bands to produce quarter bands:

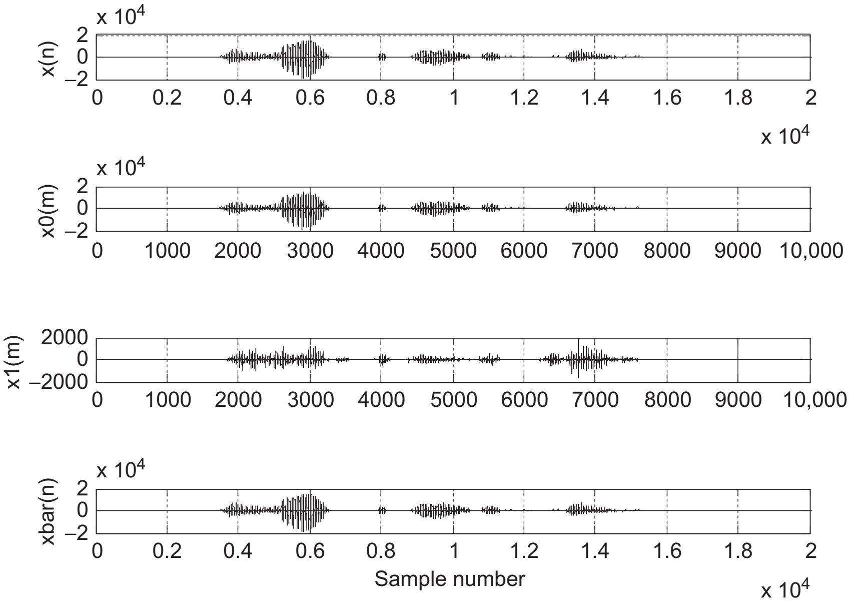
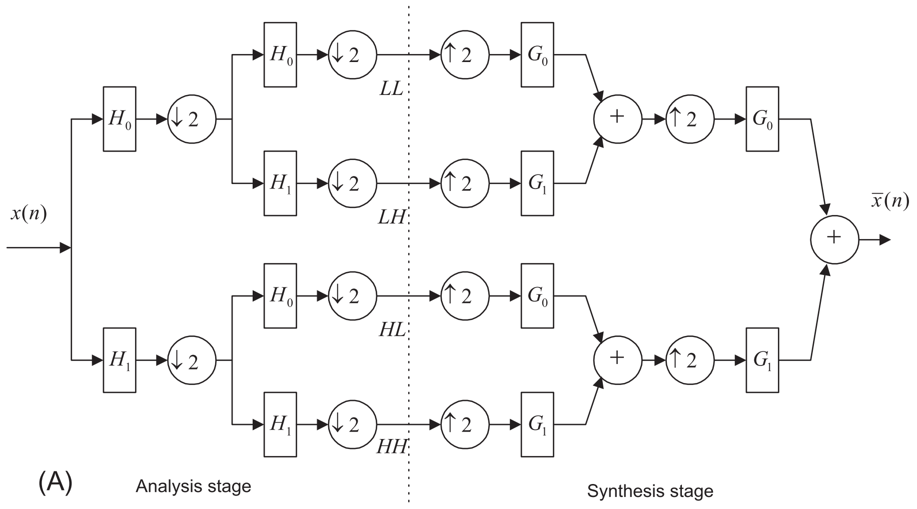


FIG. 12.11

Two-band analysis and synthesis.



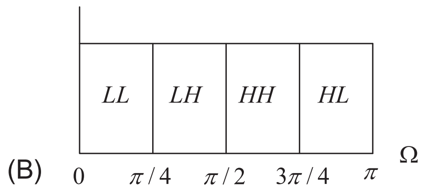


FIG. 12.12

Four-band implementation based on binary tree structure.

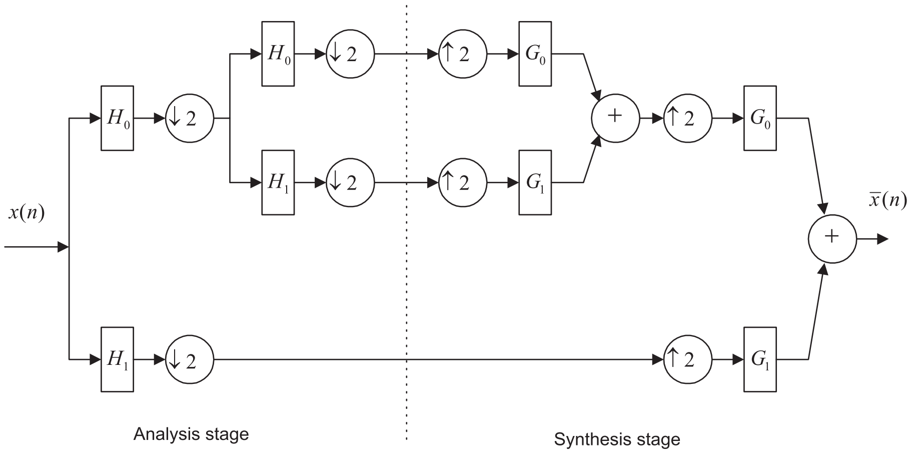


FIG. 12.13

Four-band implementation based on the dyadic tree structure.

LL, LH, HL, and HH. The four-band spectrum is labeled in Fig. 12.12B. Note that the HH band is actually centered in $[\pi/2, 3\pi/4]$ instead of $[3\pi/4, \pi]$.

In applications of signal coding, a dyadic subband tree structure is often used, as shown in Fig. 12.13, where the PR-QMF bank splits only the lower half of the spectrum into two equal bands at any level. Through continuation of splitting, we can achieve a coarser-and-coarser version of the original signal.

12.3 SUBBAND CODING OF SIGNALS

Subband analysis and synthesis can be successfully applied to signal coding, shown in Fig. 12.14 as an example for a two-band case. The analytical signals from each channel are filtered by the analysis filter, downsampled by a factor of 2, and quantized using quantizers Q_0 and Q_1 each with its assigned number of bits. The quantized codes are multiplexed for transmission or storage. At the synthesis stage, the

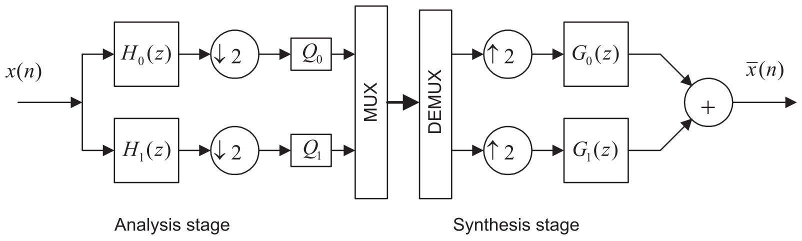


FIG. 12.14

Two-band filter bank system used for signal compression.

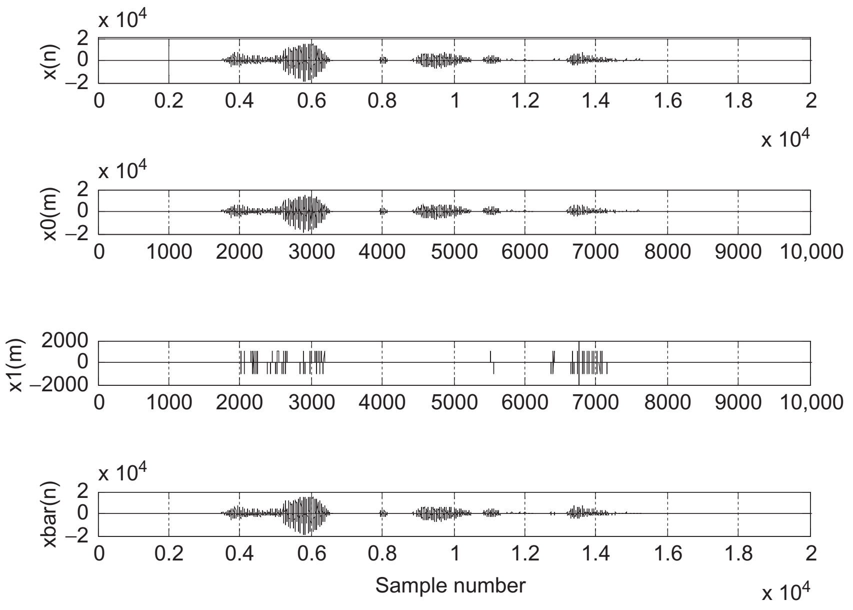


FIG. 12.15

Two-subband compression for speech data.

received or recovered quantized signals are demultiplexed, upsampled by a factor of 2, and processed by the synthesis filters. Then the output signals from all the channels are added to reconstruct the original signal. Since the signal from each analytical channel is quantized, the resultant scheme is a lossy compression one. The coding quality can be measured using the SNR.

Fig. 12.15 shows the speech coding results using the subband coding system (two-band) with the following specifications:

Sampling rate is 8 ksps (kilosamples per second)

Sample size = 16 bits/sample

Original data rate = 8 kHz × 16 bits = 128 kbps (kilobits per second)

We assign 10 bits for Q_0 (since the low-band signal contains more energy) and 6 bits for Q_1. We obtain a new data rate as (10 + 6) bits × 8 ksps/2 = 64 kbps. The implementation is shown in MATLAB Program 12.1 with the activated quantizers. Note that $x_0(m)$ and $x_1(m)$ shown in Fig. 12.15 are the quantized versions using Q_0 and Q_1. The measured SNR is 24.51 dB.

Fig. 12.16 shows the results using a four-band system. We designate both Q_0 and Q_1 as 11 bits, Q_3 as 10 bits and Q_2 as 0 bits (discarded). Note that the HL band contains the highest frequency components with the lowest signal energy level (see Fig. 12.12B). Hence, we discard HL band information to increase the coding efficiency. Therefore, we obtain the data rate as (11 + 11 + 10 + 0) bits × 8 ksps/4 = 64 kbps. The measured SNR is 27.06 dB. Four-band system offers possibility of signal quality improvement over the two-band system. Plots for the original speech, reconstructed speech, and quantized signal version for each subband are displayed in Fig. 12.17.

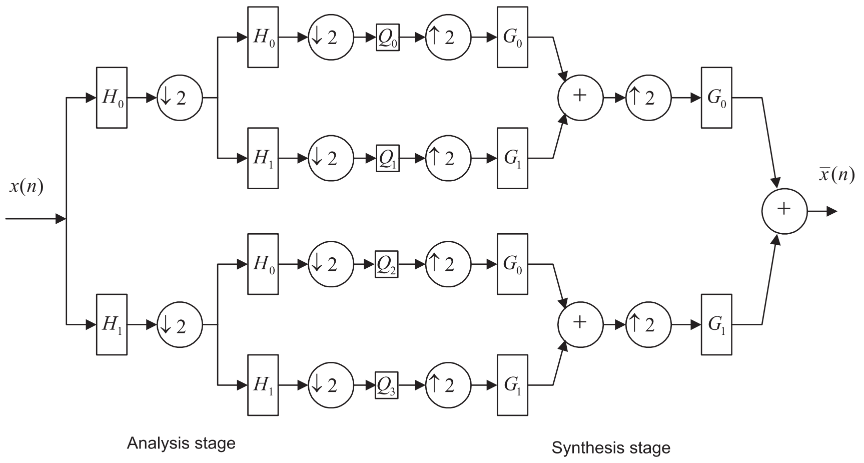


FIG. 12.16

Four-subband compression for speech data.

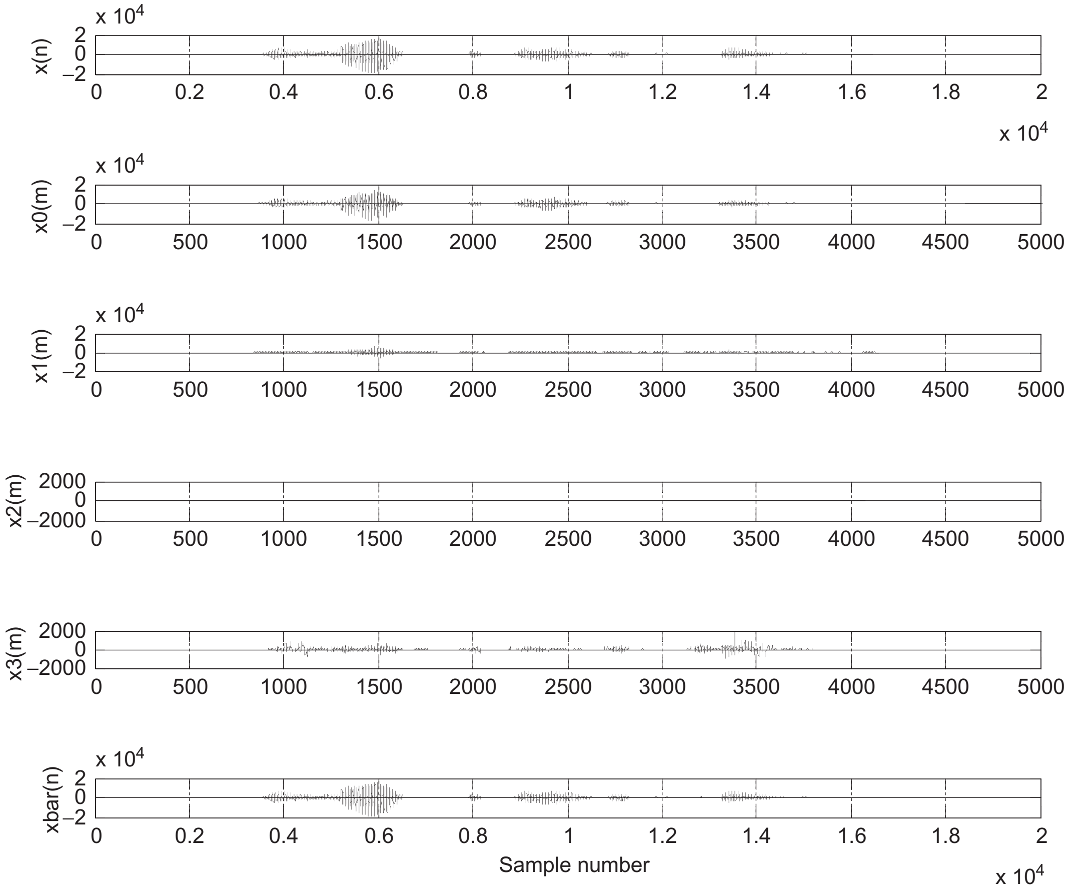


FIG. 12.17

Four-subband compression for 16-bit speech data and SNR = 27.5 dB.

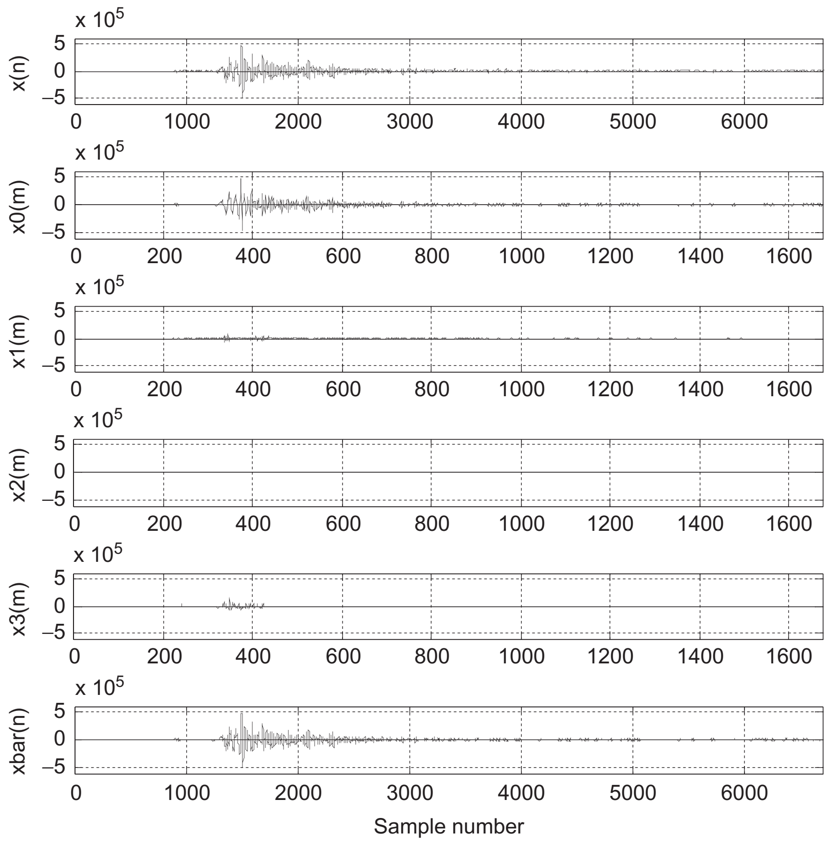


FIG. 12.18

Four-subband compression for 32-bit seismic data and SNR = 36 dB.

Fig. 12.18 shows the results using a four-band system (Fig. 12.16) for encoding seismic data. The seismic signal (provided by the USGS Albuquerque Seismological Laboratory) has a sampling rate of 15 Hz with 6700 data samples, and each sample is encoded in 32 bits. For a four-band system, the bit allocations for all bands are listed below: Q_0 (LL) = 22 bits, Q_1 (LH) = 22 bits, Q_3 (HL) = 0 (discarded), and Q_4 (HH) = 20 bits. We achieve a compression ratio of 2:1 with SNR = 36.00 dB.

12.4 WAVELET BASICS AND FAMILIES OF WAVELETS

Wavelet transform has become a powerful tool for signal processing. It offers time-frequency analysis to decompose the signal in terms of a family of wavelets or a set of basic functions, which have a fixed shape but can be shifted and dilated in time. The wavelet transform can present a signal with a good time resolution or a good frequency resolution. There are two types of wavelet transforms: the continuous wavelet transform (CWT) and the discrete wavelet transform (DWT). Specifically, the DWT provides an efficient tool for signal coding. It operates on discrete samples of the signal and has a relation

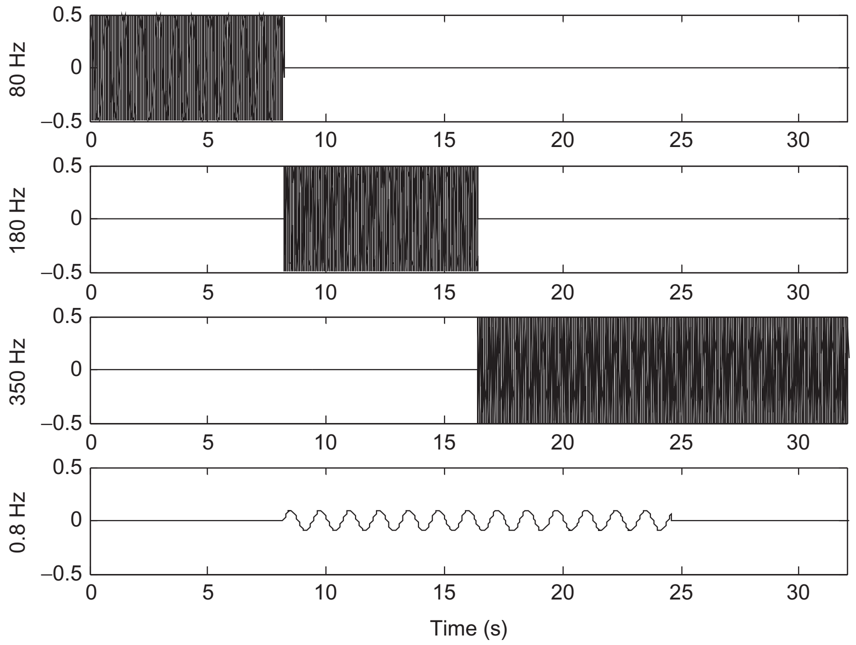


FIG. 12.19

Individual signal components.

with the dyadic subband coding as described in Section 12.2. The DWT resembles other discrete transform, such as the DFT or the DCT. In this section, without going too in-depth with mathematics, we review basics of CWT, which will lay out the foundation. Next, we emphasize the DWT for applications of signal coding.

Let us examine a signal sampled at 1 kHz with 1024×32 (32678) samples given by

$$
\begin{aligned}
x(t) = {} & 0.5\cos(2\pi \times 80t)[u(t) - u(t-8)] + \sin(2\pi \times 180t)[u(t-8) - u(t-16)] \\
& + \sin(2\pi \times 250t)[u(t-16) - u(t-32)] + 0.1\sin(2\pi \times 0.8t)[u(t-8) - u(t-24)].
\end{aligned}
\tag{12.30}
$$

The signal contains four sinusoids: 80 Hz for $0 \le t < 8$ s, 180 Hz for $8 \le t < 16$ s, 350 Hz for $16 \le t \le 32$ s, and finally 0.8 Hz for $8 \le t \le 24$ s. All the signals are plotted separately in Fig. 12.19 while Fig. 12.20 shows the combined signal and its DFT spectrum.

Based on the traditional spectral analysis shown in Fig. 12.20, we can identify the frequency components of 80, 180, and 350 Hz. However, the 0.8-Hz component and transient behaviors such as the start and stop time instants of the sinusoids (discontinuity) cannot be observed from the spectrum. Fig. 12.21 depicts the wavelet transform of the same signal. The horizontal axis is time in seconds while the vertical axis is index j, which is inversely proportional to the scale factor ($a = 2^{-j}$). As will be discussed, the larger the scale factor (the smaller index j), the smaller the frequency value. The amplitudes of the wavelet transform are displayed according to the intensity. The brighter the intensity, the larger the amplitude. The areas with brighter intensities indicate the strongest resonances between the signal and wavelets of various frequency scales and time shifts. In Fig. 12.21, four different frequency

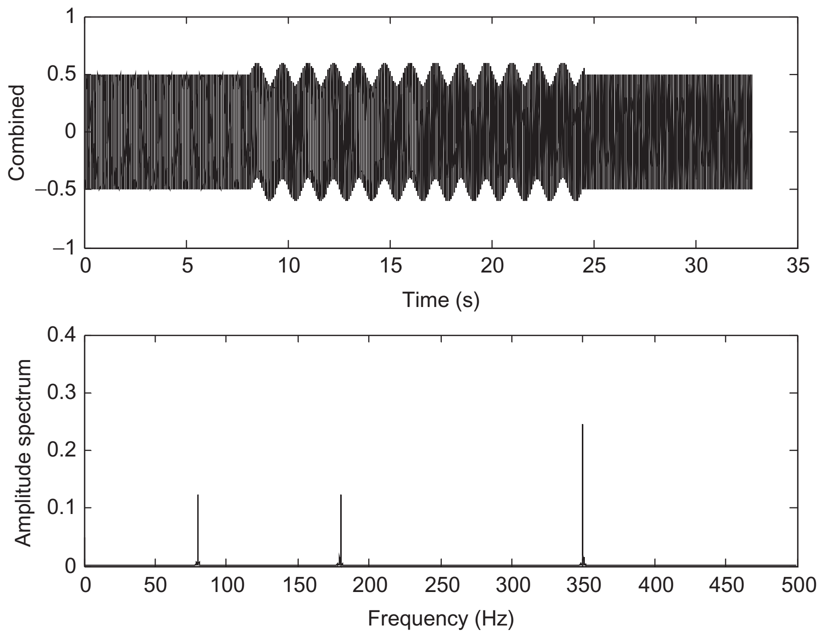


FIG. 12.20

Combined signal and its spectrum.

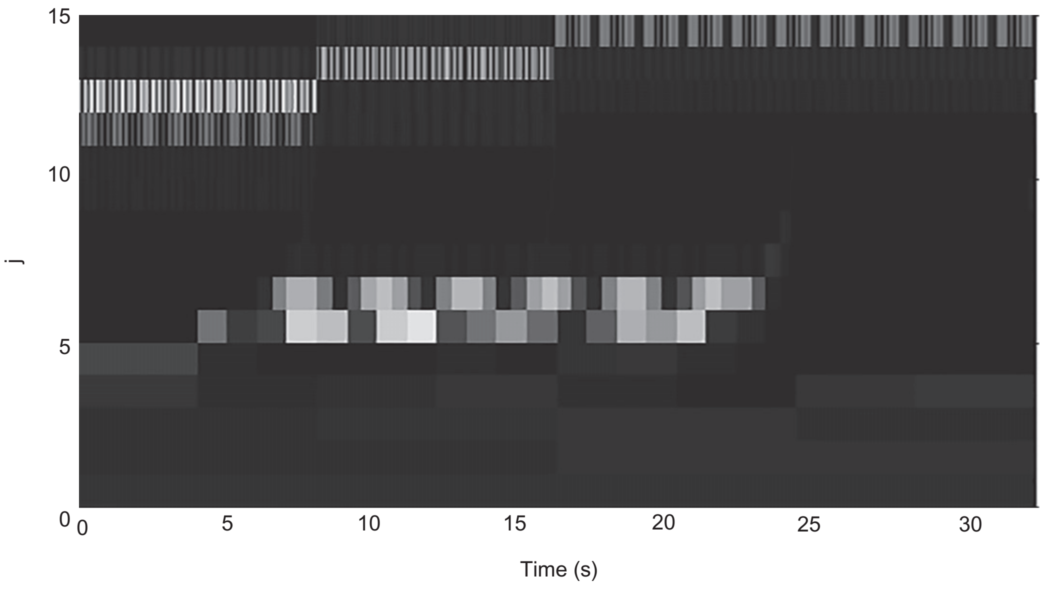


FIG. 12.21

Wavelet transform amplitudes.

components and the discontinuities of the sinusoids are displayed as well. We can further observe the fact that the finer the frequency resolution, the coarser the time resolution. For example, we can clearly identify the start and stop times for 80-, 180-, and 350-Hz frequency components, but frequency resolution is coarse, since index j has larger frequency spacing. However, for the 0.8-Hz sinusoid, we have fine frequency resolution (small frequency spacing so we can see the 0.8-Hz sinusoid) and coarse time resolution as evidence in which the start and stop times are blurred.

The CWT is defined as

$$W(a,b) = \int_{-\infty}^{\infty} f(t)\psi_{ab}(t)dt, \tag{12.31}$$

where $W(a,b)$ is the wavelet transform and $\psi_{ab}(t)$ is called the mother wavelet, which is defined as:

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right). \tag{12.32}$$

The wavelet function consists of two important parameters: scaling a and translation b. A scaled version of a function $\psi(t)$ with a scale factor of a is defined as $\psi(t/a)$. Consider a basic function $\psi(t) = \sin(\omega t)$ when $a = 1$. When $a > 1$, $\psi(t) = \sin(\omega t/a)$ is the scaled function with a frequency less than ω rad/s. When $a < 1$, $\psi(t) = \sin(\omega t/a)$ has a frequency larger than ω. Fig. 12.22 shows the scaled wavelet functions.

A translated version of a function $\psi(t)$ with a shifted time constant b is defined as $\psi(t-b)$. Fig. 12.23 shows several translated versions of the wavelet. A scaled and translated function $\psi(t)$ is given by $\psi((t-b)/a)$. This means that $\psi((t-b)/a)$ changes frequency and time shift. Several combined scaling and translated wavelets are displayed in Fig. 12.24.

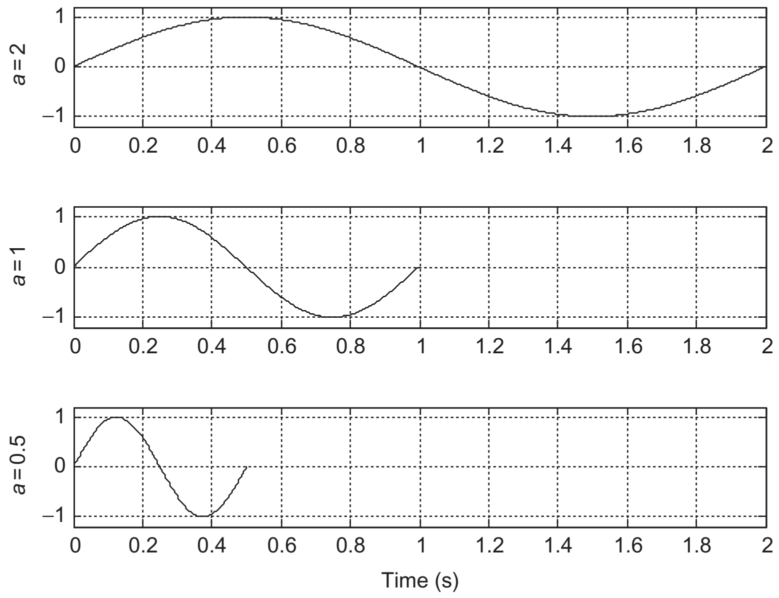


FIG. 12.22

Scaled wavelet functions.

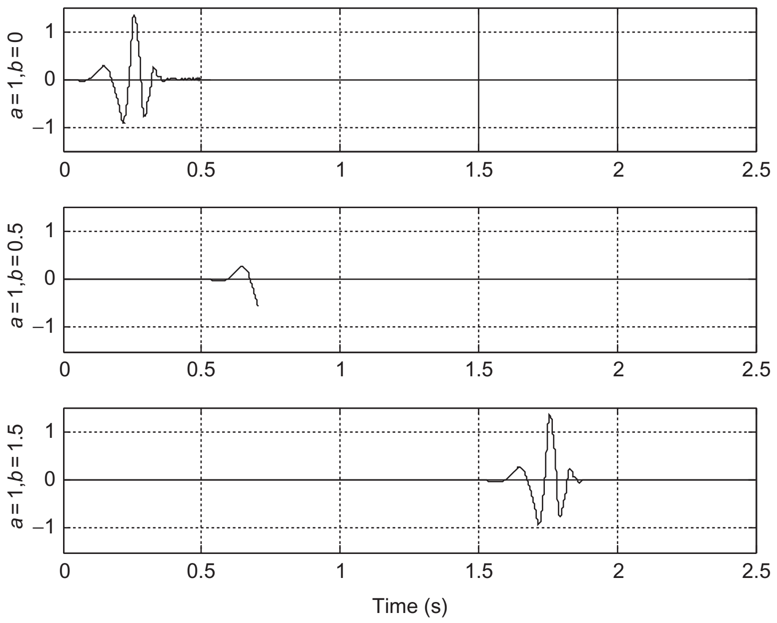


FIG. 12.23

Translated wavelet functions.

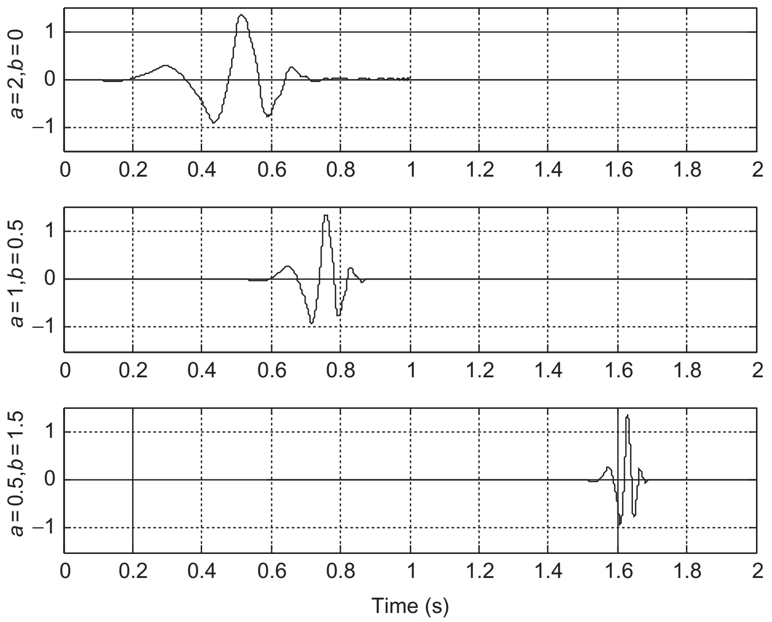


FIG. 12.24

Scaled and translated wavelet functions.

Besides these two properties, a wavelet function must satisfy admissibility and regularity conditions (vanishing moment up to certain order). Admissibility requires that the wavelet (mother wavelet) have a bandpass-limited spectrum and a zero average in the time domain, which means that wavelets must be oscillatory. Regularity requires that wavelets have some smoothness and concentration in both time and frequency domains. This topic is beyond the scope of this book and the details can be found in Akansu and Haddad (2001). There exists a pair of wavelet functions: the father wavelet (also called the scaling function) and mother wavelet. Fig. 12.25 shows a simplest pair of wavelets: the Haar father wavelet and mother wavelet.

To devise an efficient wavelet transform algorithm, we let the scale factor be a power of two, that is,

$$a = 2^{-j}. \tag{12.33}$$

Note that the larger the index j, the smaller the scale factor $a = 2^{-j}$. The time shift becomes

$$b = k2^{-j} = ka. \tag{12.34}$$

Substituting Eqs. (12.33), (12.34) into the base function gives

$$\psi\left(\frac{t-b}{a}\right) = \psi\left(\frac{t-ka}{a}\right) = \psi\left(a^{-1}t - k\right) = \psi\left(2^{j}t - k\right). \tag{12.35}$$

We can define a mother wavelet at scale j and translation k as

$$\psi_{jk}(t) = 2^{j/2}\psi\left(2^{j}t - k\right). \tag{12.36}$$

Similarly, a father wavelet (scaling function) at scale j and translation k is defined as

$$\phi_{jk}(t) = 2^{j/2}\phi\left(2^{j}t - k\right). \tag{12.37}$$

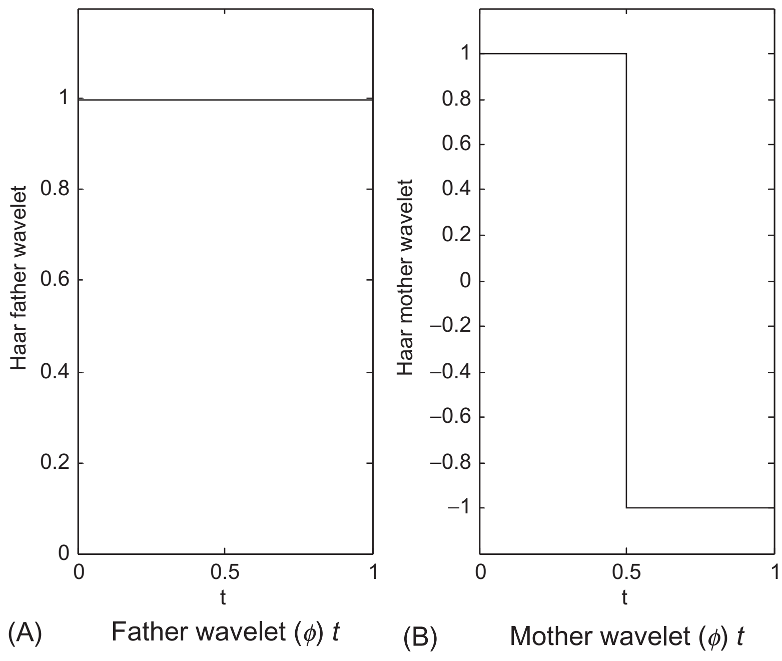


FIG. 12.25

Haar father and mother wavelets. (A) Father wavelet $\phi(t)$. (B) Mother wavelet $\psi(t)$.

EXAMPLE 12.2

Sketch the Haar father wavelet families for four different scales: $j=0, 1, 2, 3$ for a period of 1 s.

Solution:

Based on Eq. (12.37), we can determine the wavelet at each required scale as follows:
For $j=0$, $\phi_{0k}(t)=\phi(t-k)$, only $k=0$ is required to cover a second-s duration
For $j=1$, $\phi_{1k}(t)=\sqrt{2}\phi(2t-k)$, $k=0$ and $k=1$ are required
For $j=2$, $\phi_{2k}(t)=2\phi(4t-k)$, we need $k=0, 1, 2, 3$
For $j=3$, $\phi_{3k}(t)=2\sqrt{2}\phi(8t-k)$, we need $k=0, 1, 2, \cdots, 7$.
Using Fig. 12.25A, we obtain the plots as shown in Fig. 12.26.

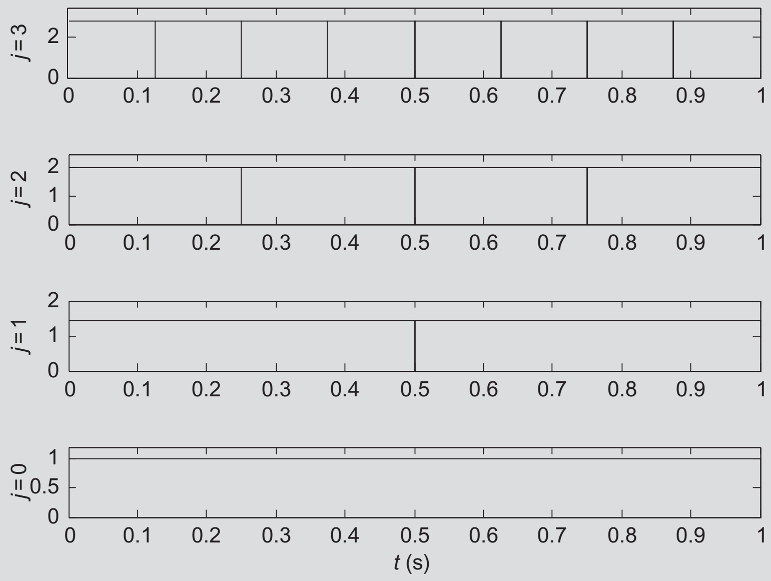


FIG. 12.26

Haar father wavelets at different scales and translations.

EXAMPLE 12.3

Sketch the Haar mother wavelet families for four different scales: $j=0, 1, 2, 3$ for a period of 1 s.

Solution:

Based on Eq. (12.36), we have.
For $j=0$, $\psi_{0k}=\psi(t-k)$, $k=0$ and $k=1$ are required
For $j=1$, $\psi_{1k}=\sqrt{2}\psi(2t-k)$, $k=0$ and $k=1$ are required
For $j=2$, $\psi_{2k}=2\psi(4t-k)$, we need $k=0, 1, 2, 3$
For $j=3$, $\psi_{3k}(t)=2\sqrt{2}\psi(8t-k)$, we need $k=0, 1, 2, \cdots, 7$.
Using Fig. 12.24B, we obtain the plots shown in Fig. 12.27.

Continued

EXAMPLE 12.3—CONT'D

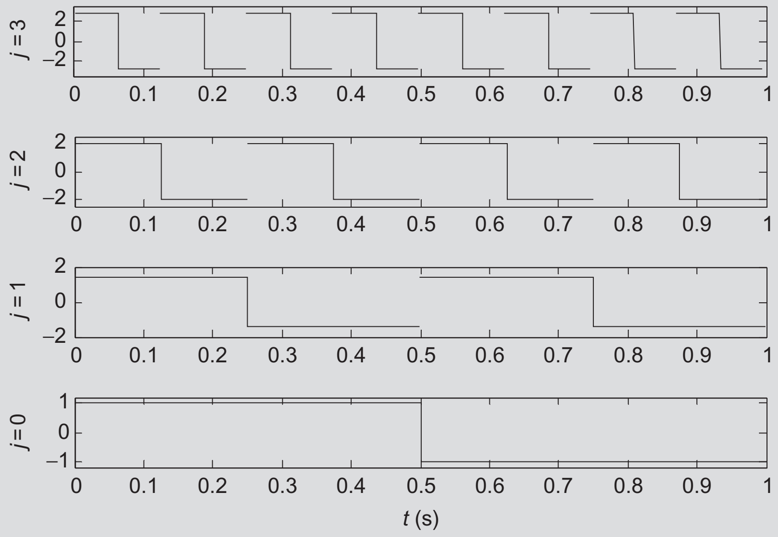


FIG. 12.27

Haar mother wavelet at different scales and translations.

A signal can be expanded by father wavelets (scaling function and its translations) at level j. More accuracy can be achieved by using larger j. The expanded function is approximated by

$$f(t) \approx f_j(t) = \sum_{k=-\infty}^{\infty} c_j(k) 2^{j/2} \phi\left(2^j t - k\right) = \sum_{k=-\infty}^{\infty} c_j(k) \phi_{jk}(t), \tag{12.38}$$

where the wavelet coefficients $c_j(k)$ can be determined by an inner product:

$$c_j(k) = f(t)\phi_{jk}(t) = \int f(t) 2^{j/2} \phi\left(2^j t - k\right) dt. \tag{12.39}$$

EXAMPLE 12.4

Approximate the following function using the Haar scaling function at level $j = 1$.

$$f(t) = \begin{cases} 2 & 0 \le t < 0.5 \\ 1 & 0.5 \le t \le 1 \end{cases}.$$

Solution:

Substituting $j = 1$ in Eq. (12.38), it leads to

$$f(t) \approx f_1(t) = \sum_{k=-\infty}^{\infty} c_1(k)\phi_{1k}(t) = \sum_{k=-\infty}^{\infty} c_1(k) 2^{1/2} \phi(2t - k).$$

We only need $k=0$ and $k=1$ to cover the range: $0 \le t \le 1$, that is,

$$f(t) = c_1(0)2^{1/2}\phi(2t) + c_1(1)2^{1/2}\phi(2t-1).$$

Note that

$$\phi(2t) = \begin{cases} 1 & \text{for } 0 \le t \le 0.5 \\ 0 & \text{elsewhere} \end{cases} \text{ and } \phi(2t-1) = \begin{cases} 1 & \text{for } 0.5 \le t \le 1 \\ 0 & \text{elsewhere} \end{cases}.$$

Applying Eq. (12.39) yields

$$c_1(0) = \int_0^{1/2} f(t)2^{1/2}\phi(2t)dt = \int_0^{1/2} 2 \times 2^{1/2} \times 1dt = 2^{1/2}.$$

Similarly,

$$c_1(1) = \int_{1/2}^{1} f(t)2^{1/2}\phi(2t-1)dt = \int_{1/2}^{1} 1 \times 2^{1/2} \times 1dt = 0.5 \times 2^{1/2}.$$

Then substituting the coefficients $c_1(0)$ and $c_1(1)$ leads to

$$f_1(t) = 2^{1/2} \times 2^{1/2}\phi(2t) + 0.5 \times 2^{1/2}2^{1/2}\phi(2t-1) = 2\phi(2t) + \phi(2t-1) = f(t).$$

Eq. (12.39) can also be approximated numerically:

$$c_j(k) \approx \sum_{m=0}^{M-1} f(t_m)\phi_{jk}(t_m)\Delta t = \sum_{m=0}^{M-1} f(t_m)2^{j/2}\phi\left(2^j t_m - k\right)\Delta t,$$

where $t_m = m\Delta t$ is the time instant, Δt denotes the time step, and M is the number of intervals.

In this example, if we chose $\Delta t = 0.2$, then $M = 5$ and $t_m = m\Delta t$. The numerical calculations are listed:

$$c_1(0) \approx \sum_{m=0}^{4} f(t_m)2^{1/2}\phi(2t_m)\Delta t = 2^{1/2} \times [f(0) \times \phi(0) + f(0.2) \times \phi(0.4)$$

$$+ f(0.4) \times \phi(0.8) + f(0.6) \times \phi(1.2) + f(0.8) \times \phi(1.6)]\Delta t$$

$$= 2^{1/2}(2 \times 1 + 2 \times 1 + 2 \times 1 + 1 \times 0 + 1 \times 0) \times 0.2 = 1.2 \times 2^{1/2}$$

$$c_1(1) \approx \sum_{m=0}^{4} f(t_m)2^{1/2}\phi(2t_m - 1)\Delta t = 2^{1/2} \times [f(0) \times \phi(-1) + f(0.2) \times \phi(-0.6)$$

$$+ f(0.4) \times \phi(-0.2) + f(0.6) \times \phi(0.2) + f(0.8) \times \phi(0.6)]\Delta t$$

$$= 2^{1/2} \times (2 \times 0 + 2 \times 0 + 2 \times 0 + 1 \times 1 + 1 \times 1 \times 0.2 = 0.4 \times 2^{1/2}.$$

Finally, we have

$$f_1(t) = 1.2 \times 2^{1/2} \times 2^{1/2}\phi(2t) + 0.4 \times 2^{1/2}2^{1/2}\phi(2t-1) = 2.4\phi(2t) + 0.8\phi(2t-1) \approx f(t).$$

It is clear that there is a numerical error. The error can be reduced when a smaller time interval Δt is adopted.

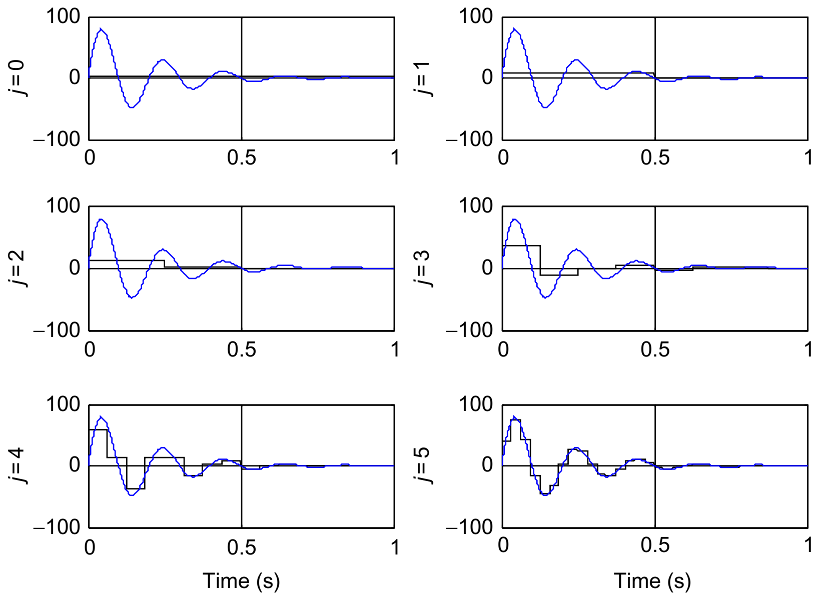


FIG. 12.28

Signal expanded by Haar father wavelets.

Fig. 12.28 demonstrates the approximation of a sinusoidal delaying function using the scaling functions (Haar father wavelets) at different scales, that is, $j=0$, 1, 2, 4, 5.

Now, let us examine the function approximation at resolution $j=1$:

$$f_1(t) \approx \sum_{k=-\infty}^{\infty} c_1(k)\phi_{1k}(2t) = \sum_{k=-\infty}^{\infty} c_1(k)\sqrt{2}\phi(2t-k)$$
$$= c_1(0)\phi_{10}(2t) + c_1(1)\phi_{11}(2t) = c_1(0)\sqrt{2}\phi(2t) + c_1(1)\sqrt{2}\phi(2t-1).$$

We also look at another possibility at a coarser scale with both the scaling functions (father wavelets) and mother wavelets, that is, $j=0$:

$$f_1(t) \approx \sum_{k=-\infty}^{\infty} c_0(k)\phi_{0k}(t) + \sum_{k=-\infty}^{\infty} d_0(k)\psi_{0k}(t)$$
$$= c_0(0)\phi_{00}(t) + d_0(0)\psi_{00}(t) = c_0(0)\phi(t) + d_0(0)\psi(t).$$

Furthermore, we see that

$$\begin{aligned} f_1(t) &\approx c_0(0)\phi(t) + d_0(0)\psi(t) \\ &= c_0(0)(\phi(2t) + \phi(2t-1)) + d_0(0)(\phi(2t) - \phi(2t-1)) \\ &= \frac{1}{\sqrt{2}}(c_0(0) + d_0(0))\phi_{00}(2t) + \frac{1}{\sqrt{2}}(c_0(0) - d_0(0))\phi_{01}(2t-1) \\ &= c_1(0)\phi_{10}(2t) + c_1(1)\phi_{10}(2t). \end{aligned}$$

We observe that

$$c_1(0) = \frac{1}{\sqrt{2}}(c_0(0) + d_0(0))$$
$$c_1(1) = \frac{1}{\sqrt{2}}(c_0(0) - d_0(0)).$$

This means that

$$S_1 = S_0 \cup W_0.$$

where S_1 contains functions in terms of basis scaling functions at $\phi_{1k}(t)$, and the function can also be expanded using the scaling functions $\phi_{0k}(t)$ and wavelet functions $\psi_{0k}(t)$ at a coarser level $j-1$. In general, the following statement is true:

$$\begin{aligned} S_j &= S_{j-1} \cup W_{j-1} = \left[S_{j-2} \cup W_{j-2}\right] \cup W_{j-1} \\ &= \left\{\left[S_{j-3} \cup W_{j-3}\right] \cup W_{j-2}\right\} \cup W_{j-1} \\ \cdots &= S_0 \cup W_0 \cup W_1 \cup \cdots \cup W_{j-1}. \end{aligned}$$

Hence, the approximation of $f_j(t)$ can be expressed as

$$\begin{aligned} f(t) \approx f_j(t) &= \sum_{k=-\infty}^{\infty} c_j(k)\phi_{jk}(t) \\ &= \sum_{k=-\infty}^{\infty} c_{j-1}(k)\phi_{(j-1)k}(t) + \sum_{k=-\infty}^{\infty} d_{j-1}(k)\psi_{(j-1)k}(t) \\ &= \sum_{k=-\infty}^{\infty} c_{(j-1)}(k)2^{(j-1)/2}\phi\left(2^{(j-1)}t - k\right) + \sum_{k=-\infty}^{\infty} d_{(j-1)}(k)2^{(j-1)/2}\psi\left(2^{(j-1)}t - k\right). \end{aligned}$$

Repeating the expansion of the first sum leads to

$$\begin{aligned} f(t) \approx f_J(t) &= \sum_{k=-\infty}^{\infty} c_0(k)\phi_{0k}(t) + \sum_{j=0}^{J-1}\sum_{k=-\infty}^{\infty} d_j(k)\psi_{jk}(t) \\ &= \sum_{k=-\infty}^{\infty} c_0(k)\phi(t-k) + \sum_{j=0}^{J-1}\sum_{k=-\infty}^{\infty} d_j(k)2^{j/2}\psi\left(2^j t - k\right), \end{aligned} \tag{12.40}$$

where the mother wavelet coefficients $d_j(k)$ can also be determined by the inner product:

$$d_j(k) = \langle f(t)\psi_{jk}(t)\rangle = \int f(t)2^{j/2}\psi\left(2^j t - k\right)dt. \tag{12.41}$$

Fig. 12.29 demonstrates the function approximation (Fig. 12.28) with the base scaling function at resolution $j=0$, and mother wavelets at scales $j=0, 1, 2, 3, 4$. The combined approximation ($J=5$) using Eq. (12.40) is shown in Fig. 12.30.

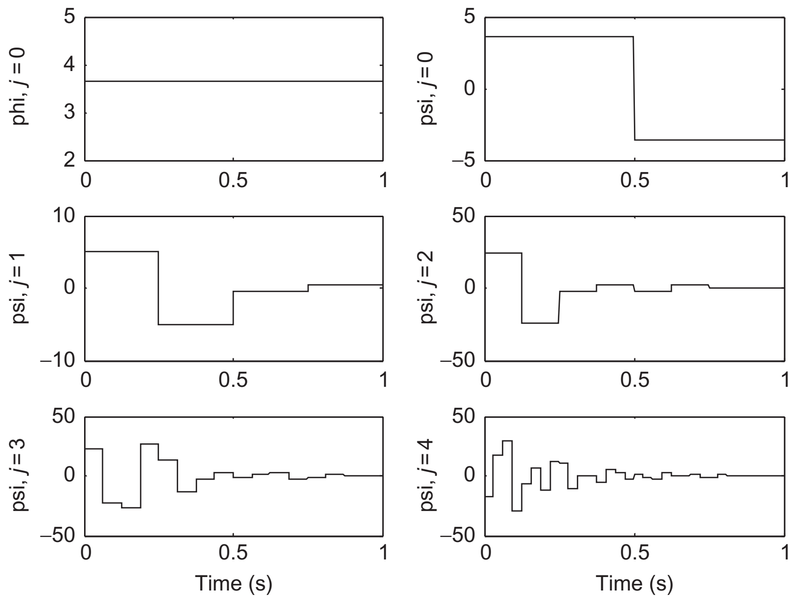


FIG. 12.29

Approximations using Haar scaling functions and mother wavelets.

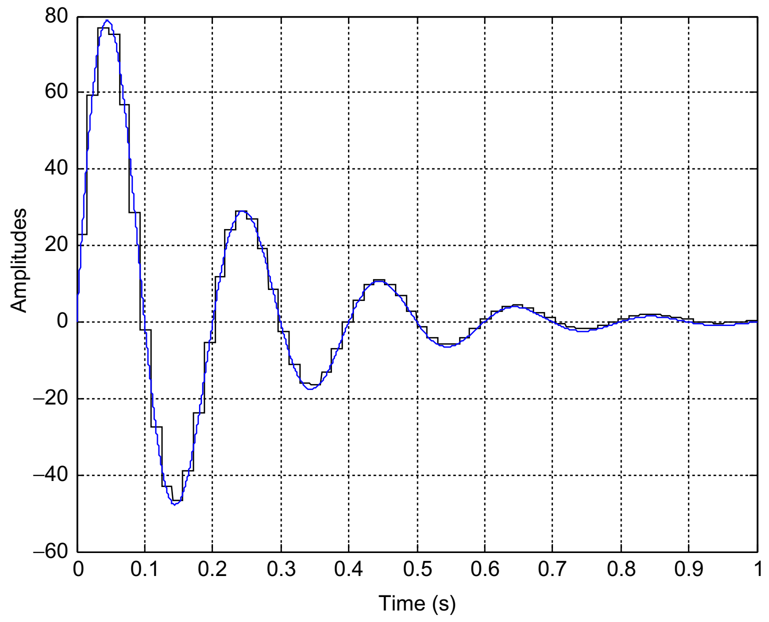


FIG. 12.30

Signal coded using the wavelets at resolution $J = 5$.

12.5 MULTIRESOLUTION EQUATIONS

There are two very important equations for multiresolution analysis. Each scaling function can be constructed by a linear combination of translations with the doubled frequency of a base scaling function $\phi(2t)$, that is,

$$\phi(t) = \sum_{k=-\infty}^{\infty} \sqrt{2} h_0(k) \phi(2t-k), \tag{12.42}$$

where $h_0(k)$ is a set of the scaling function coefficients (wavelet filter coefficients). The mother wavelet function can also be built by a sum of translations with the double frequency of the base scaling function $\phi(2t)$, that is,

$$\psi(t) = \sum_{k=-\infty}^{\infty} \sqrt{2} h_1(k) \phi(2t-k), \tag{12.43}$$

where $h_1(k)$ is another set of wavelet filter coefficients. Let us verify these two relationships via Example 12.5.

EXAMPLE 12.5

Determine $h_0(k)$ for the Haar father wavelet.

Solution:

From Eq. (12.42), we can express

$$\phi(t) = \sqrt{2} h_0(0) \phi(2t) + \sqrt{2} h_0(1) \phi(2t-1).$$

Then we deduce that

$$h_0(0) = h_0(1) = 1/\sqrt{2}.$$

Fig. 12.31 shows that the Haar father wavelet is a sum of two scaling functions at scale $j=1$.

EXAMPLE 12.6

Determine $h_1(k)$ for the Haar mother wavelet.

Solution:

From Eq. (12.43), we can write

$$\psi(t) = \sqrt{2} h_1(0) \phi(2t) + \sqrt{2} h_1(1) \phi(2t-1).$$

Hence, we deduce that

$$h_1(0) = 1/\sqrt{2} \text{ and } h_1(1) = -1/\sqrt{2}.$$

Fig. 12.32 shows that the Haar mother wavelet is a difference of two scaling functions at scale $j=1$.

Note that the relation between $H_0(z)$ and $H_1(z)$ exists and is given by

$$h_1(k) = (-1)^k h_0(N-1-k). \tag{12.44}$$

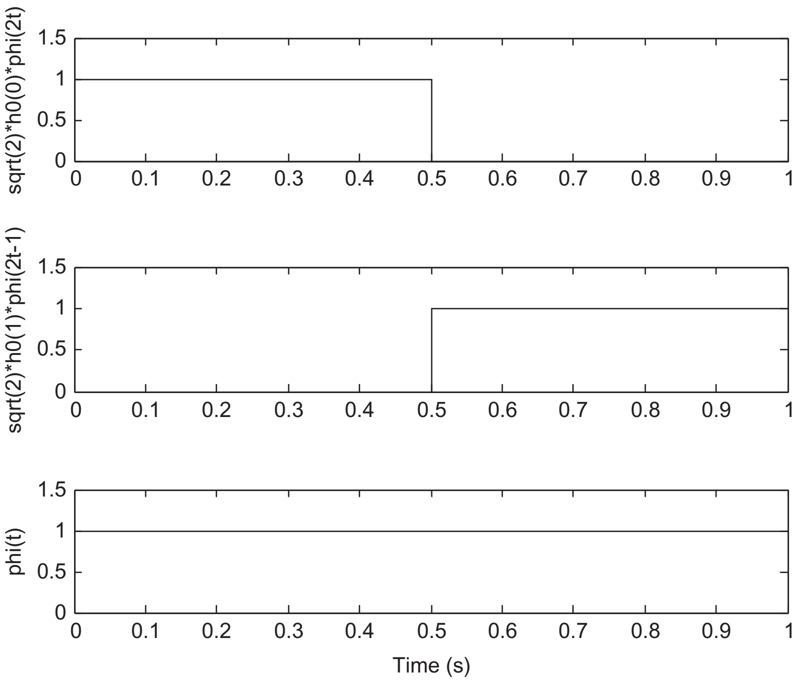


FIG. 12.31

Haar wavelets in Example 12.5.

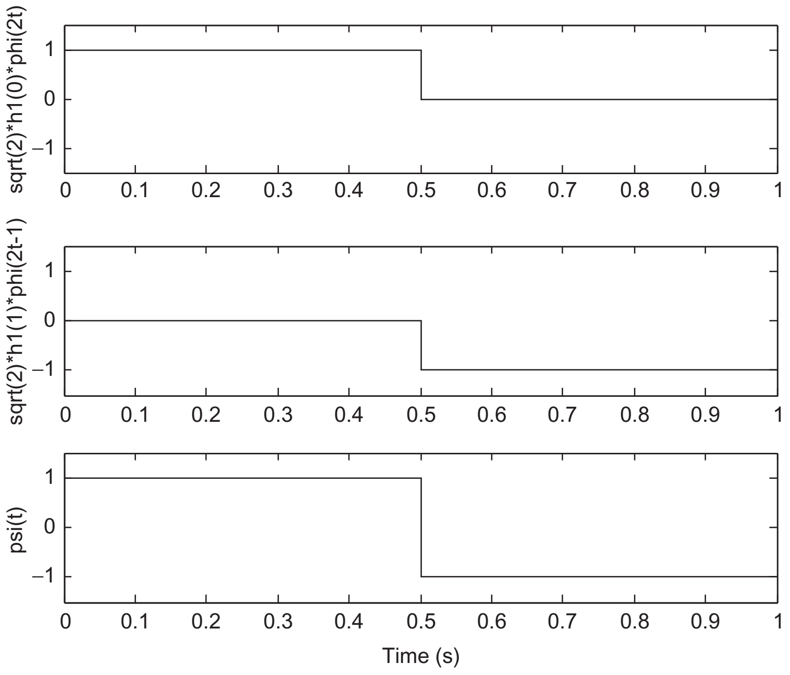


FIG. 12.32

Haar wavelets in Example 12.6.

We can verify Eq. (12.44) for the Haar wavelet:

$$h_1(k) = (-1)^k h_0(1-k).$$

Then

$$h_1(0) = (-1)^0 h_0(1-0) = h_0(1) = 1/\sqrt{2}$$
$$h_1(1) = (-1)^1 h_0(1-1) = -h_0(0) = -1/\sqrt{2}.$$

This means that once we obtain the coefficients of $h_0(k)$, the coefficients $h_1(k)$ can be determined via Eq. (12.44). We do not aim to obtain wavelet filter coefficients here. The topic is beyond the scope of this book and the details are given in Akansu and Haddad (2001). Instead, some typical filter coefficients for Haar and Doubechies are presented in Table 12.2.

We can apply the Daubechies-4 filter coefficients to examine multiresolution Eqs. (12.42), (12.43). From Table 12.2, we have

$$h_0(0) = 0.4830, h_0(1) = 0.8365, h_0(2) = 0.2241, \text{ and } h_0(3) = -0.1294.$$

We then expand Eq. (12.42) as

$$\phi(t) = \sqrt{2}h_0(0)\phi(2t) + \sqrt{2}h_0(1)\phi(2t-1) + \sqrt{2}h_0(2)\phi(2t-2) + \sqrt{2}h_0(3)\phi(2t-3).$$

Fig. 12.33 shows each component at resolution $j=1$ and the constructed scaling function $\phi(t)$. The original scaling function $\phi(t)$ is also included as shown in the last plot for comparison.

With the given coefficients $h_0(k)$ and applying Eq. (12.44), we can obtain the wavelet coefficients $h_1(k)$ as

$$h_1(0) = -0.1294, h_1(1) = -0.2241, h_1(2) = 0.8365, \text{ and } h_1(3) = -0.4830.$$

Expanding Eq. (12.43) leads to

$$\psi(t) = \sqrt{2}h_1(0)\phi(2t) + \sqrt{2}h_1(1)\phi(2t-1) + \sqrt{2}h_1(2)\phi(2t-2) + \sqrt{2}h_1(3)\phi(2t-3).$$

Similarly, Fig. 12.34 displays each component at resolution $j=1$ and the constructed mother wavelet function $\psi(t)$. The last plot displays the original mother wavelet function $\psi(t)$ for comparison.

Table 12.2 Typical Wavelet Filter Coefficients $h_0(k)$

Haar	Daubechies 4	Daubechies 6	Daubechies 8
0.707106781186548	0.482962913144534	0.332670552950083	0.230377813308896
0.707106781186548	0.836516303737808	0.806891509311093	0.714846570552915
	0.224143868042013	0.459877502118492	0.630880767929859
	−0.129409522551260	−0.135011020010255	−0.027983769416859
		−0.085441273882027	−0.187034811719093
		0.035226291885710	0.030841381835561
			0.032883011666885
			−0.010597401785069

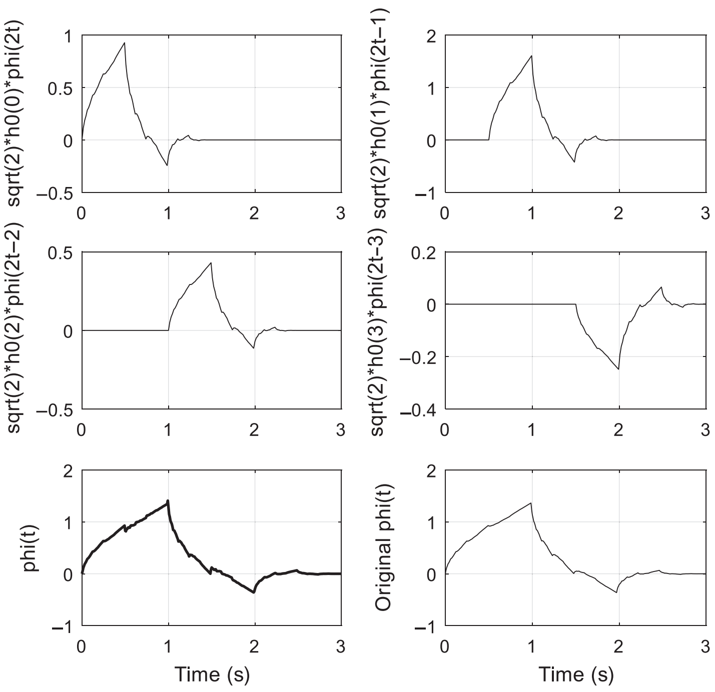


FIG. 12.33

Constructed four-tap Daubechies father wavelet.

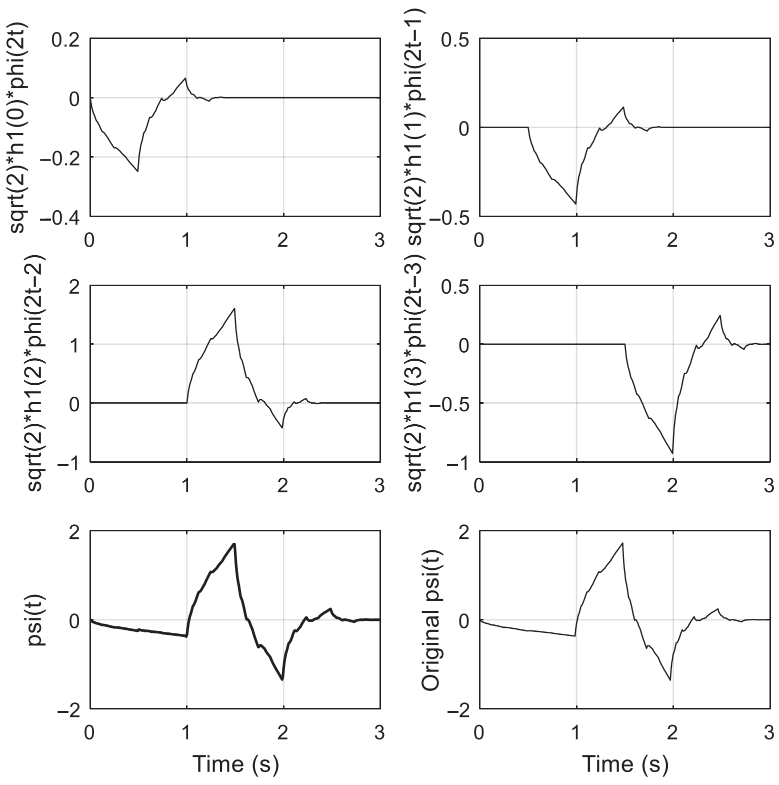


FIG. 12.34

Constructed four-tap Daubechies mother wavelet.

12.6 DISCRETE WAVELET TRANSFORM

Now let us examine the discrete wavelet transform (DWT). We begin with coding a signal using a wavelet expansion as shown in Eq. (12.45):

$$f(t) \approx f_{j+1}(t) = \sum_{k=-\infty}^{\infty} c_j(k) 2^{j/2} \phi(2^j t - k) + \sum_{k=-\infty}^{\infty} d_j(k) 2^{j/2} \psi(2^j t - k). \tag{12.45}$$

By applying and continuing to apply Eq. (12.45), $f(t)$ can be coded at any level we wish. Furthermore, by recursively applying Eq. (12.45) until $j=0$, we can obtain the signal expansion using all the mother wavelets as well as a one scaling function at scale $j=0$, that is,

$$f(t) \approx f_J(t) = \sum_{k=-\infty}^{\infty} c_0(k) \phi(t-k) + \sum_{j=0}^{J-1} \sum_{k=-\infty}^{\infty} d_j(k) 2^{j/2} \psi(2^j t - k). \tag{12.46}$$

All $c_j(k)$ and all $d_j(k)$ are called the wavelet coefficients. They are essentially weights for the scaling function(s) and wavelet functions (mother wavelets). The DWT computes these wavelet coefficients. On the other hand, given the wavelet coefficients, we are able to reconstruct the original signal by applying the inverse discrete wavelet transform (IDWT).

Based on the wavelet theory without proof (see Appendix F), we can perform the DWT using the analysis equations as follows:

$$c_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h_0(m-2k) \tag{12.47}$$

$$d_j(k) = \sum_{m=-\infty}^{\infty} c_{j+1}(m) h_1(m-2k), \tag{12.48}$$

where $h_0(k)$ are the lowpass wavelet filter coefficients listed in Table 12.2, while $h_1(k)$, the highpass filter coefficients, can be determined by

$$h_1(k) = (-1)^k h_0(N-1-k). \tag{12.49}$$

These lowpass and highpass filters are called the quadrature mirror filters (QMF). As an example, the frequency responses of the four-tap Daubechies wavelet filters are plotted in Fig. 12.35.

Next, we need to determine the filter inputs $c_{j+1}(k)$ in Eqs. (12.47), (12.48). In practice, since j is a large number, the function $\phi(2^j t - k)$ appears to be close to an impulse-like function, that is, $\phi(2^j t - k) \approx 2^{-j}\delta(t - k2^{-j})$. For example, the Haar scaling function can be expressed as $\phi(t) = u(t) - u(t-1)$, where $u(t)$ is the step function. We can easily get $\phi(2^5 t - k) = u(2^5 t - k) - u(2^5 t - 1 - k) = u(t - k2^{-5}) - u(t - (k+1)2^{-5})$ for $j=5$, which is a narrow pulse with a unit height and a width 2^{-5} located at $t = k2^{-5}$. The area of the pulse is therefore equal to 2^{-5}. When j approaches to a bigger positive integer, $\phi(2^j t - k) \approx 2^{-j}\delta(t - k2^{-j})$. Therefore, $f(t)$ approximated by the scaling function at level j is rewritten as

$$\begin{aligned} f(t) \approx f_j(t) &= \sum_{k=-\infty}^{\infty} c_j(k) 2^{j/2} \phi(2^j t - k) \\ &= \cdots + c_j(0) 2^{j/2} \phi(2^j t) + c_j(1) 2^{j/2} \phi(2^j t - 1) + c_j(2) 2^{j/2} \phi(2^j t - 2) + \cdots \\ &\approx \cdots + c_j(0) 2^{-j/2} \delta(t) + c_j(1) 2^{-j/2} \delta(t - 1 \times 2^{-j}) + c_j(2) 2^{-j/2} \delta(t - 2 \times 2^{-j}) + \cdots. \end{aligned} \tag{12.50}$$

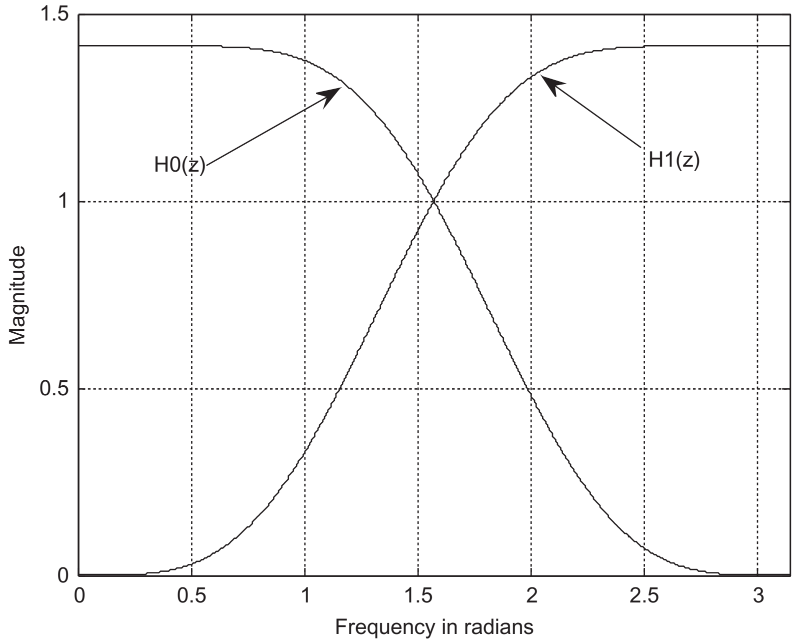


FIG. 12.35

Frequency responses for 4-tap Daubechies filters.

On the other hand, if we sample $f(t)$ using the same sample interval $T_s = 2^{-j}$ (time resolution), the discrete-time function can be expressed as

$$f(n) = f(nT_s) = \cdots + f(0T_s)T_s\delta(t - T_s) + f(T_s)T_s\delta(t - T_s) + f(2T_s)T_s\delta(n - 2T_s) + \cdots. \tag{12.51}$$

Hence, comparing Eq. (12.50) with the discrete-time version in Eq. (12.51), it follows that

$$c_j(k)2^{-j/2} = f(k)T_s. \tag{12.52}$$

Substituting $T_s = 2^{-j}$ in Eq. (12.52) leads to

$$c_j(k) = 2^{-j/2} f(k). \tag{12.53}$$

With the obtained sequence $c_j(k)$ using sample values $f(k)$, we can perform the DWT using Eqs. (12.47), (12.48). Furthermore, Eqs. (12.47) and (12.48) can be implemented using a dyadic tree structure similar to the subband coding case. Fig. 12.36 depicts the case for $j = 2$.

Note that the reversed sequences $h_0(-k)$ and $h_1(-k)$ are used in the analysis stage. Similarly, the IDWT (synthesis equation) can be developed (see Appendix F) and expressed as

$$c_{j+1}(k) = \sum_{m=-\infty}^{\infty} c_j(m)h_0(k - 2m) + \sum_{m=-\infty}^{\infty} d_j(m)h_1(k - 2m). \tag{12.54}$$

Finally, the signal amplitude can be rescaled by

$$f(k) = 2^{j/2} c_j(k). \tag{12.55}$$

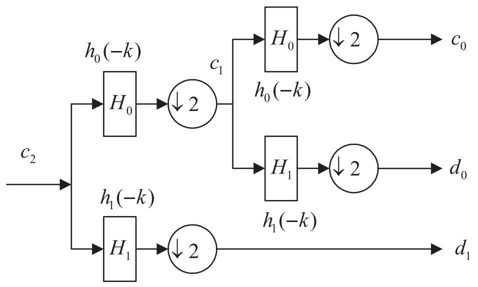


FIG. 12.36

Analysis using the dyadic subband coding structure.

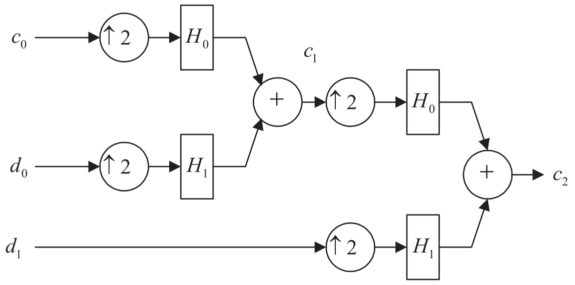


FIG. 12.37

Synthesis using the dyadic subband coding structure.

An implementation for $j=2$ using the dyadic subband coding structure is illustrated in Fig. 12.37.
Now, let us study the DWT and IDWT in the following examples.

EXAMPLE 12.7

Given the sample values as [4 2 −1 0], use the Haar wavelets to determine the wavelet coefficients.

Solution:

Form the filter inputs: $c_2(k) = 2^{-2/2} \times [4 \ \ 2 \ \ -1 \ \ 0] = \left[2 \ \ 1 \ \ -\frac{1}{2} \ \ 0\right]$.
The acquired Haar wavelet filter coefficients are listed as

$$h_0(k) = \left[\frac{1}{\sqrt{2}} \ \ \frac{1}{\sqrt{2}}\right] \text{ and } h_1(k) = \left[\frac{1}{\sqrt{2}} \ \ -\frac{1}{\sqrt{2}}\right].$$

The function is expanded by the scaling functions as

$$f(t) \approx f_2(t) = \sum_{k=-\infty}^{\infty} c_j(k) 2^{j/2} \phi(2^j t - k)$$
$$= 4 \times \phi(4t) + 2 \times \phi(4t-1) - 1 \times \phi(4t-2) + 0 \times \phi(4t-3).$$

Continued

EXAMPLE 12.7—CONT'D

We will verify this expression later. Applying the wavelet analysis equations, we have

$$c_1(k)=\sum_{m=-\infty}^{\infty} c_2(m)h_0(m-2k)$$
$$d_1(k)=\sum_{m=-\infty}^{\infty} c_2(m)h_1(m-2k).$$

Specifically,

$$c_1(0)=\sum_{m=-\infty}^{\infty} c_2(m)h_0(m)=c_2(0)h_0(0)+c_2(1)h_0(1)=2\times\frac{1}{\sqrt{2}}+1\times\frac{1}{\sqrt{2}}=\frac{3\sqrt{2}}{2}$$
$$c_1(1)=\sum_{m=-\infty}^{\infty} c_2(m)h_0(m-2)=c_2(2)h_0(0)+c_2(3)h_0(1)=\left(-\frac{1}{2}\right)\times\frac{1}{\sqrt{2}}+0\times\frac{1}{\sqrt{2}}=-\frac{1}{2\sqrt{2}}$$
$$d_1(0)=\sum_{m=-\infty}^{\infty} c_2(m)h_1(m)=c_2(0)h_1(0)+c_2(1)h_1(1)=2\times\frac{1}{\sqrt{2}}+1\times\left(-\frac{1}{\sqrt{2}}\right)=\frac{1}{\sqrt{2}}$$
$$d_1(1)=\sum_{m=-\infty}^{\infty} c_2(m)h_1(m-2)=c_2(2)h_1(0)+c_2(3)h_1(1)=\left(-\frac{1}{2}\right)\times\frac{1}{\sqrt{2}}+0\times\left(-\frac{1}{\sqrt{2}}\right)=-\frac{1}{2\sqrt{2}}.$$

Using the subband coding method in Fig. 12.36 yields.

```
>> x0=rconv([1 1]/sqrt(2),[2 1 -0.5 0])
x0 =2.1213  0.3536  -0.3536  1.4142
>> c1=x0(1:2:4)
c1 = 2.1213  -0.3536
>> x1=rconv([1 -1]/sqrt(2),[2 1 -0.5 0])
x1 = 0.7071  1.0607  -0.3536  -1.4142
>> d1=x1(1:2:4)
d1 =0.7071  -0.3536
```

where MATLAB function **rconv()** for filter operations with the reversed filter coefficients is listed in Section 12.8. Repeating for the next level, we have

$$c_0(k)=\sum_{m=-\infty}^{\infty} c_1(m)h_0(m-2k)$$
$$d_0(k)=\sum_{m=-\infty}^{\infty} c_1(m)h_1(m-2k).$$

Thus,

$$c_0(0)=\sum_{m=-\infty}^{\infty} c_1(m)h_0(m)=c_1(0)h_0(0)+c_1(1)h_0(1)=\frac{3\sqrt{2}}{2}\times\frac{1}{\sqrt{2}}+\left(-\frac{1}{2\sqrt{2}}\right)\times\frac{1}{\sqrt{2}}=\frac{5}{4}$$

$$d_0(0) = \sum_{m=-\infty}^{\infty} c_1(m)h_1(m) = c_1(0)h_1(0) + c_1(1)h_1(1) = \frac{3\sqrt{2}}{2} \times \frac{1}{\sqrt{2}} + \left(-\frac{1}{2\sqrt{2}}\right) \times \left(-\frac{1}{\sqrt{2}}\right) = \frac{7}{4}.$$

MATLAB verifications are shown below:

```
>> xx0=rconv([1 1]/sqrt(2),c1)
xx0 =1.2500 1.2500
>> c0=xx0(1:2:2)
c0 =1.2500
>> xx1=rconv([1 -1]/sqrt(2),c1)
xx1 =1.7500 -1.7500
>> d0=xx1(1:2:2)
d0 =1.7500
```

Finally, we pack the wavelet coefficients $w_2(k)$ at $j = 2$ together as

$$w_2(k) = [c_0(0)d_0(0)d_1(0)d_1(1)] = \left[\frac{5}{4}\frac{7}{4}\frac{1}{\sqrt{2}} - \frac{1}{2\sqrt{2}}\right].$$

Then the function can be expanded using one scaling function and three mother wavelet functions.

$$f(t) \approx f_2(t) = \sum_{k=-\infty}^{\infty} c_0(k)\phi(t-k) + \sum_{j=0}^{1}\sum_{k=-\infty}^{\infty} d_j(k)2^{j/2}\psi\left(2^j t - k\right)$$
$$= \frac{5}{4}\phi(t) + \frac{7}{4}\psi(t) + \psi(2t) - \frac{1}{2}\psi(2t-1).$$

Fig. 12.38 shows the plots for each function and the combined function to verify that $f(t)$ does have amplitudes of 4, 2, -1, and 0.

We can use the MATLAB function **dwt()** provided in Section 12.8 to compute the DWT coefficients.

```
function w = dwt(h0,c,kLevel)
% h0 = wavelet filter coefficients (lowpass filter)
% c = input vector
% kLevel = level
% w= wavelet coefficients
```

The results are verified as follows:

```
>> w=dwt([1/sqrt(2) 1/sqrt(2)][4 2 -1 0]/2,2)'
w = 1.2500 1.7500 0.7071 -0.3536
```

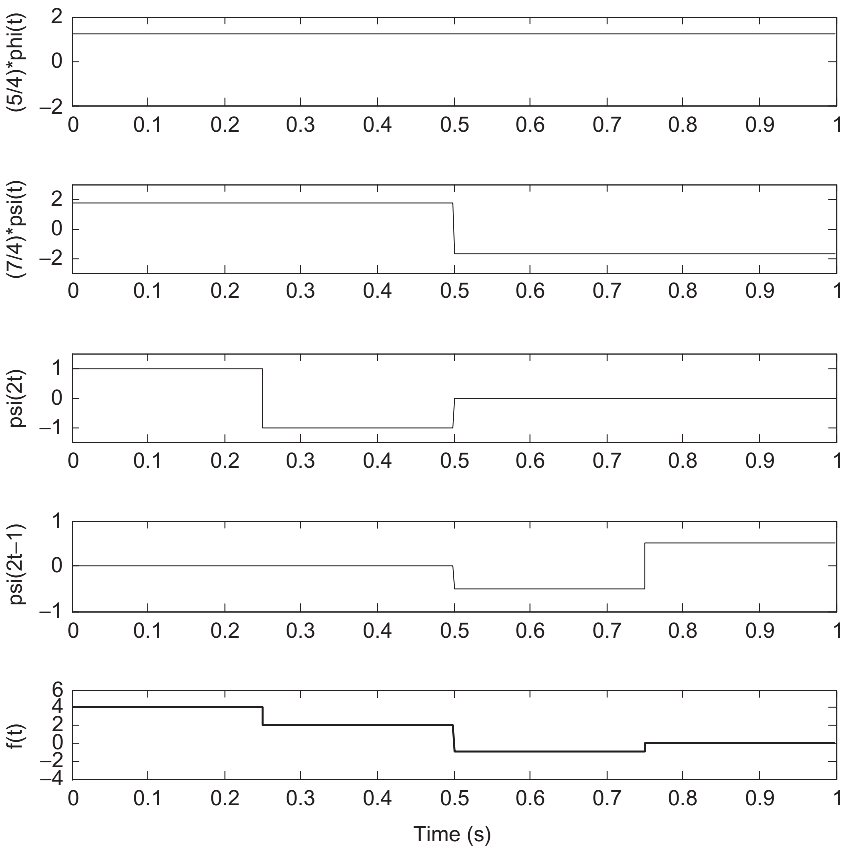
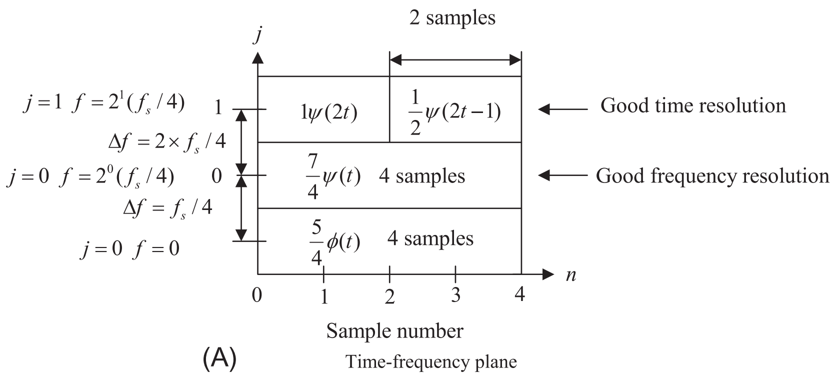


FIG. 12.38

Signal reconstructed using the Haar wavelets in Example 12.7.

From Example 12.7, we can create a time-frequency plot of the DWT amplitudes in two dimensions as shown in Fig. 12.39. Assuming that the sampling frequency is f_s, we have the smallest frequency resolution as $f_s/N = f_s/4$, where $N = 4$. When j ($j = 0$) is small, we achieve a small frequency resolution $\Delta f = f_s/4$ and each wavelet presents four samples. In this case, we have a good frequency resolution but a poor time resolution. Similarly, when j ($j = 1$) is a large value, the frequency resolution becomes $\Delta f = 2f_s/4$ and each wavelet presents two samples (more details in time domain). Hence, we achieve a good time resolution but a poor frequency resolution. Note that the DWT cannot achieve good resolutions in both frequency and time at the same time. The time-frequency plot of the DWT amplitudes in terms of their intensity is shown in Fig. 12.39B, and the time and frequency plane for the DWT is shown in Fig. 12.40.



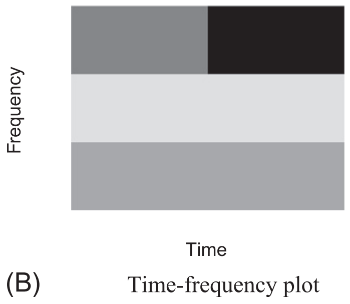


FIG. 12.39

Time-frequency plot of the DWT amplitudes. (A) Time-frequency plane, (B) time-frequency plot.

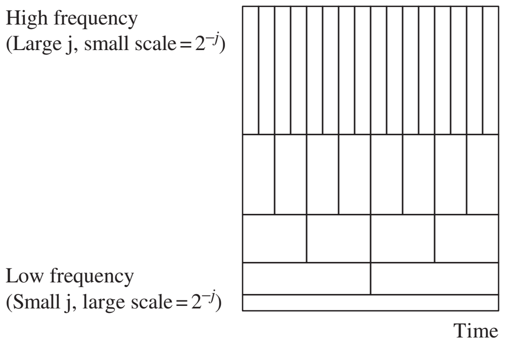

High frequency
(Large j, small scale $=2^{-j}$)
Low frequency
(Small j, large scale $=2^{-j}$)
Time

FIG. 12.40

Time-frequency plane.

EXAMPLE 12.8

Given the wavelet coefficients obtained using the Haar wavelet filters

$$[c_0(0)d_0(0)d_1(0)d_1(1)] = \left[\frac{5}{4}\,\frac{7}{4}\,\frac{1}{\sqrt{2}}\ -\frac{1}{2\sqrt{2}}\right],$$

perform the IDWT.

Solution:

From Eq. (12.54), we get

$$c_1(k) = \sum_{m=-\infty}^{\infty} c_0(m)h_0(k-2m) + \sum_{m=-\infty}^{\infty} d_0(m)h_1(k-2m).$$

Then we recover coefficients $c_1(k)$ as

$$\begin{aligned} c_1(0) &= \sum_{m=-\infty}^{\infty} c_0(m)h_0(-2m) + \sum_{m=-\infty}^{\infty} d_0(m)h_1(-2m) \\ &= c_0(0)h_0(0) + d_0(0)h_1(0) = \frac{5}{4}\times\frac{1}{\sqrt{2}} + \frac{7}{4}\times\frac{1}{\sqrt{2}} = \frac{3\sqrt{2}}{2} \\ c_1(1) &= \sum_{m=-\infty}^{\infty} c_0(m)h_0(1-2m) + \sum_{m=-\infty}^{\infty} d_0(m)h_1(1-2m) \\ &= c_0(0)h_0(1) + d_0(0)h_1(1) = \frac{5}{4}\times\frac{1}{\sqrt{2}} + \frac{7}{4}\times\left(-\frac{1}{\sqrt{2}}\right) = -\frac{1}{2\sqrt{2}}. \end{aligned}$$

MATLAB verification is given as.

```
>> c1=fconv([1 1]/sqrt(2),[5/4 0])+fconv([1 -1]/sqrt(2),[7/4 0])
  c1 = 2.1213 -0.3536
```

where the MATLAB function **fconv()** for filter operations with the forward filter coefficients is listed in Section 12.8. Again, from Eq. (12.54), we obtain

$$c_2(k) = \sum_{m=-\infty}^{\infty} c_1(m)h_0(k-2m) + \sum_{m=-\infty}^{\infty} d_1(m)h_1(k-2m).$$

Substituting the achieved wavelet coefficients $c_2(k)$, we yield

$$\begin{aligned} c_2(0) &= \sum_{m=-\infty}^{\infty} c_1(m)h_0(-2m) + \sum_{m=-\infty}^{\infty} d_1(m)h_1(-2m) \\ &= c_1(0)h_0(0) + d_1(0)h_1(0) = \frac{3\sqrt{2}}{2}\times\left(\frac{1}{\sqrt{2}}\right) + \frac{1}{\sqrt{2}}\times\frac{1}{\sqrt{2}} = 2 \\ c_2(1) &= \sum_{m=-\infty}^{\infty} c_1(m)h_0(1-2m) + \sum_{m=-\infty}^{\infty} d_1(m)h_1(1-2m) \\ &= c_1(0)h_0(1) + d_1(0)h_1(1) = \frac{3\sqrt{2}}{2}\times\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}\left(-\frac{1}{\sqrt{2}}\right) = 1 \end{aligned}$$

$$c_2(2) = \sum_{m=-\infty}^{\infty} c_1(m)h_0(2-2m) + \sum_{m=-\infty}^{\infty} d_1(m)h_1(2-2m)$$

$$= c_1(1)h_0(0) + d_1(1)h_1(0) = \left(-\frac{1}{2\sqrt{2}}\right) \times \frac{1}{\sqrt{2}} + \left(-\frac{1}{2\sqrt{2}}\right) \times \frac{1}{\sqrt{2}} = -\frac{1}{2}$$

$$c_2(3) = \sum_{m=-\infty}^{\infty} c_1(m)h_0(3-2m) + \sum_{m=-\infty}^{\infty} d_1(m)h_1(3-2m)$$

$$= c_1(1)h_0(1) + d_1(1)h_1(1) = \left(-\frac{1}{2\sqrt{2}}\right) \times \frac{1}{\sqrt{2}} + \left(-\frac{1}{2\sqrt{2}}\right) \times \left(-\frac{1}{\sqrt{2}}\right) = 0.$$

We can verify the results using the MATLAB program as follows:

```
>> c2=fconv([1 1]/sqrt(2),[3*sqrt(2)/2 0 -1/(2*sqrt(2)) 0])+fconv([1 -1]/sqrt(2),[1/
sqrt(2) 0 -1/(2*sqrt(2)) 0])
c2 = 2.0000 1.0000 -0.5000   0
```

Scaling the wavelet coefficients, we finally recover the original sample values as

$$f(k) = 2^{2/2}[2\ 1\ -0.5\ 0] = [4\ 2\ -1\ 0].$$

Similarly, we can use the MATLAB function **idwt()** provided in Section 12.8 to perform the IDWT.
idwt.m

```
function c = idwt(h0,w,kLevel)
% h0 = wavelet filter coefficients (lowpass filter)
% w= wavelet coefficients
% kLevel = level
% c = input vector
```

Appling the MATLAB function **idwt()** leads to.

```
f=2*idwt([1/sqrt(2) 1/sqrt(2)][5/4 7/4 1/sqrt(2) -1/(2*sqrt(2))],2)'
f = 4.0000 2.0000 -1.0000 0.0000
```

Since $2^{j/2}$ scales signal amplitudes down in the analysis stage while scales them up back in the synthesis stage, we can omit $2^{j/2}$ by using $c(k) = f(k)$ directly in practice.

EXAMPLE 12.9

Given the sample values [4 2 − 1 0], use the provided MATLAB DWT (dwt.m) and IDWT (idwt.m) and specified wavelet filter to perform the DWT and IWDT without using the scale factor $2^{j/2}$.

(a) Haar wavelet filter

(b) four-tap Daubechies wavelet filter.

Solution:

(a) From Table 12.2, the Haar wavelet filter coefficients are

$$h_0 = \left(\frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}}\right).$$

Applying the MATLAB function **dwt()** and **idwt()**, we have.

```
>> w=dwt([1/sqrt(2) 1/sqrt(2)][4 2 -1 0],2)'
w = 2.5000 3.5000 1.4142 -0.7071
>> f=idwt([1/sqrt(2) 1/sqrt(2)],w,2)'
w = 4.0000 2.0000 -1.0000   0
```

(b) From Table 12.2, the 4-tap Duabechies wavelet filter coefficients are.

```
h0=[0.482962913144534 0.836516303737808 0.224143868042013 -0.129409522551260];
```

MATLAB program verification is demonstrated below:

```
>> w=dwt([0.482962913144534 0.836516303737808 0.224143868042013 -
0.129409522551260][4 2 -1 0],2)'
w = 2.5000 2.2811 -1.8024 2.5095
>> f=idwt([0.482962913144534 0.836516303737808 0.224143868042013 -
0.129409522551260],w,2)'
f = 4.0000 2.0000 -1.0000   0
```

12.7 WAVELET TRANSFORM CODING OF SIGNALS

We can apply the DWT and IWDT for data compression and decompression. The compression and decompression involves two stages, that is, the analysis stage and the synthesis stage. At analysis stage, the wavelet coefficients are quantized based on their significance. Usually, we assign more bits to the coefficient in a coarser scale, since the corresponding subband has larger signal energy and low-frequency components. We assign a small number of bits to the coefficient, which resides in a finer scale, since the corresponding subband has lower signal energy and high-frequency components. The quantized coefficients can be efficiently transmitted. The DWT coefficients are laid out in a format described in Fig. 12.41. The coarse coefficients are placed toward the left side. For example, in Example 12.7, we organized the DWT coefficient vector as

$$w_2(k) = [c_0(0)d_0(0)d_1(0)d_1(1)] = \left[\frac{5}{4}\ \frac{7}{4}\ \frac{1}{\sqrt{2}}\ -\frac{1}{2\sqrt{2}}\right].$$

$$w_2(k) = [c_0(0)\ d_0(0)\ d_1(0)\ d_1(1)] = \left[\frac{5}{4}\ \frac{7}{4}\ \frac{1}{\sqrt{2}}\ -\frac{1}{2\sqrt{2}}\right]$$

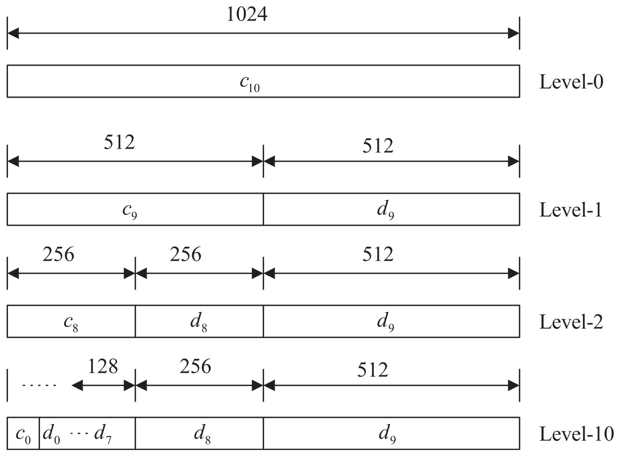


FIG. 12.41

DWT coefficient layout.

Let us look at the following simulation examples.

EXAMPLE 12.10

Given a 40-Hz sinusoidal signal plus random noise sampled at 8000 Hz with 1024 samples,

$$x(n) = 100\cos(2\pi \times 40nT) + 10 \times randn,$$

where $T = 1/8000$ s and *randn* is a random noise generator with a unit power and Gaussian distribution. Use a 16-bit code for each wavelet coefficient and write a MATLAB program to perform data compressions for each of the following ratios: 2:1, 4:1, 8:1, and 16:1. Plot the reconstructed waveforms.

Solution:

We use the 8-tap Daubechies filter as listed in Table 12.2. We achieve the data compression by dropping the high subband coefficients for each level consecutively and coding each wavelet coefficient in the lower subband using 16 bits. For example, we achieve the 2:1 compression ratio by omitting 512 high-frequency coefficients at the first level, 4:1 by omitting 512 high-frequency coefficients at first level and 256 high-frequency coefficients at the second level, and so on. The recovered signals are plotted in Fig. 12.42. SNR =21 dB is achieved for the 2:1 compression ratio. As we can see, when more and more higher frequency coefficients are dropped, the reconstructed signal contains less and less details. The recovered signal with the compression of 16:1 presents the least details but shows the smoothest signal. On the other hand, omitting the high-frequency wavelet coefficients can be very useful for a signal denoising application, in which the high-frequency noise contaminating the clean signal is removed. A complete MATLAB program is given in Program 12.2.

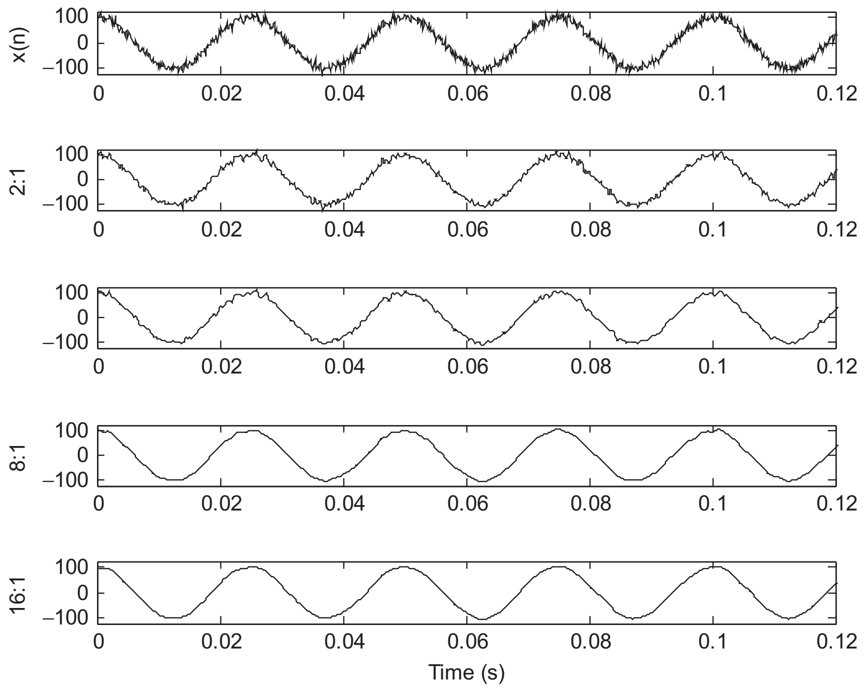


FIG. 12.42

Reconstructed signal at various compression ratios.

Program 12.2. Wavelet data compression.

```
close all; clear all;clc
t=0:1:1023;t=t/8000;
x=100*cos(40*2*pi*t)+10*randn(1,1024);
h0=[0.230377813308896  0.714846570552915  0.630880767929859 ...
  -0.027983769416859  -0.187034811719092  0.030841381835561 ....
  0.032883011666885  -0.010597401785069];
N=1024; nofseg=1
rec_sig=[]; rec_sig2t1=[]; rec_sig4t1=[]; rec_sig8t1=[]; rec_sig16t1=[];
for i=1:nofseg
  sp.=x((i-1)*1024+1:i*1024);
  w=dwt(h0,sp.,10);
% Quantization
  wmax=round(max(abs(w)));
  wcode=round(2^15*w/wmax); % 16-bit code for storage
  w=wcode*wmax/2^15; % Recovered wavelet coefficients
  w(513:1024)=zeros(1512); % 2:1 compression ratio
  sig_rec2t1=idwt(h0,w,10);
  rec_sig2t1=[rec_sig2t1 sig_rec2t1'];
  w(257:1024)=0; % 4:1 compression ratio
  sig_rec4t1=idwt(h0,w,10);
  rec_sig4t1=[rec_sig4t1 sig_rec4t1'];
  w(129:1024)=0; % 8:1 compression ratio
```

```
    sig_rec8t1=idwt(h0,w,10);
    rec_sig8t1=[rec_sig8t1 sig_rec8t1'];
    w(65:1024)=0; % 16:1 compression ratio
    sig_rec16t1=idwt(h0,w,10);
    rec_sig16t1=[rec_sig16t1 sig_rec16t1'];
  end
  subplot(5,1,1),plot(t,x,'k'); axis([0 0.12-120,120]);ylabel('x(n)');
  subplot(5,1,2),plot(t,rec_sig2t1,'k'); axis([0 0.12-120,120]);ylabel('2:1');
  subplot(5,1,3),plot(t,rec_sig4t1,'k'); axis([0 0.12-120,120]);ylabel(4:1);
  subplot(5,1,4),plot(t,rec_sig8t1,'k'); axis([0 0.12-120,120]);ylabel('8:1');
  subplot(5,1,5),plot(t,rec_sig16t1,'k'); axis([0 0.12-120,120]);ylabel('16:1');
  xlabel('Time (Sec.)')
  NN=min(length(x),length(rec_sig2t1)); axis([0 0.12-120,120]);
  err=rec_sig2t1(1:NN)-x(1:NN);
  SNR=sum(x.*x)/sum(err.*err);
  disp('PR reconstruction SNR dB≥');
  SNR=10*log10(SNR)
```

Fig. 12.43 shows the wavelet compression for 16-bit speech data sampled at 8 kHz. The original speech data is divided into speech segments, each with 1024 samples. After applying the DWT to each segment, the coefficients, which correspond to high frequency components indexed from 513 to 1024, are discarded in order to achieve the coding efficiency. The reconstructed speech data has a compression ratio 2:1 with SNR = 22 dB. The MATLAB program is given in Program 12.3.

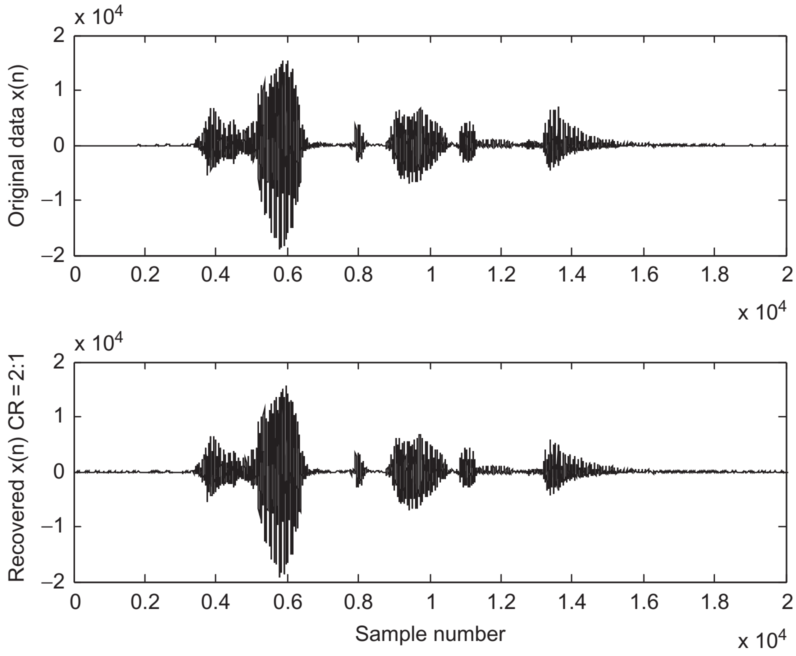


FIG. 12.43

Reconstructed speech signal with compression ratio of 2 and SNR = 22 dB.

Program 12.3. Wavelet data compression for speech segments.

```
close all; clear all;clc
load orig.dat; %Load speech data
h0=[0.230377813308896  0.714846570552915  0.630880767929859 ...
  -0.027983769416859  -0.187034811719092  0.030841381835561 ....
  0.032883011666885  -0.010597401785069];
N=length(orig);
nofseg=ceil(N/1024);
speech=zeros(1,nofseg*1024);
speech(1:N)=orig(1:N);% Making the speech length to be multiple of 1024 samples
rec_sig=[];
for i=1:nofseg
  sp.=speech((i-1)*1024+1:i*1024);
  w=dwt(h0,sp.,10);
% Quantization
  w=(round(2^15*w/2^15))*2^(15-15);
  w(513:1024)=zeros(1512); % Omitting the high frequency coefficients
  sp_rec=idwt(h0,w,10);
  rec_sig=[rec_sig sp_rec'];
end
subplot(2,1,1),plot([0:length(speech)-1],speech,'k');axis([0 20,000-20,000
20,000]);
ylabel('Original data x(n)');
subplot(2,1,2),plot([0:length(rec_sig)-1],rec_sig,'k');axis([0 20,000-20,000
20,000]);
xlabel('Sample number');ylabel('Recovered x(n) CR=2:1');
NN=min(length(speech),length(rec_sig));
err=rec_sig(1:NN)-speech(1:NN);
SNR=sum(speech.*speech)/sum(err.*err);
disp('PR reconstruction SNR dB≥');
SNR=10*log10(SNR)
```

Fig. 12.44 displays the wavelet compression for 16-bit ECG data using Program 12.3. The reconstructed ECG data has a compression ratio of 2:1 with SNR = 33.8 dB.

Fig. 12.45 illustrates an application of signal denoising using the DWT with coefficient threshold. During the analysis stage, the obtained DWT coefficient (quantization is not necessary) is set to zero if its value is less than the predefined threshold as depicted in Fig. 12.45. This simple technique is called the hard threshold. Usually, the small wavelet coefficients are related to the high-frequency components in signals. Therefore, setting high-frequency components to zero is the same as lowpass filtering.

An example is shown in Fig. 12.46. The first plot depicts a 40-Hz noisy sinusoidal signal (sine wave plus noise with SNR = 18 dB) and the clean signal with a sampling rate of 8000 Hz. The second plot shows that after zero threshold operations, 67% of coefficients are set to zero and the recovered signal

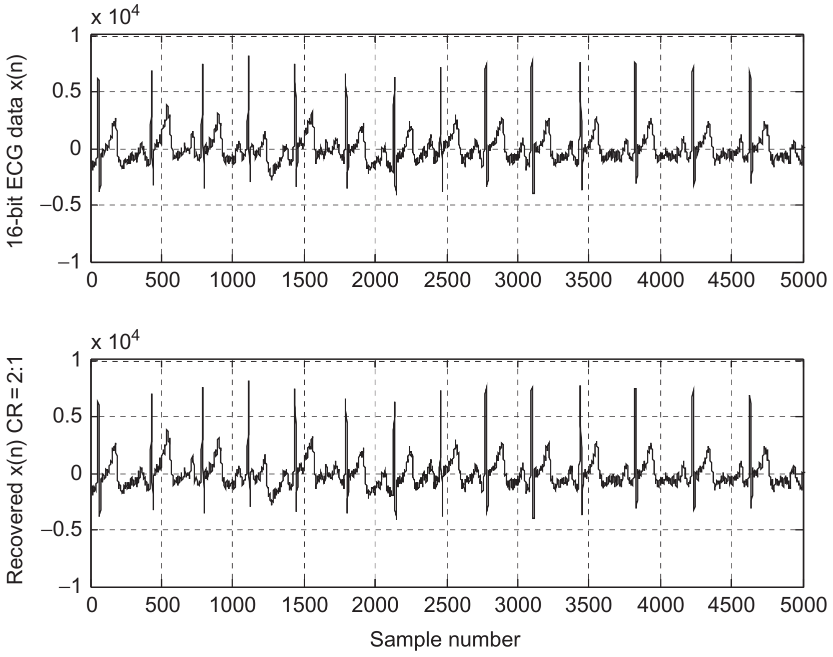


FIG. 12.44

Reconstructed ECG signal with the compression ratio of 2 and SNR = 33.8 dB.

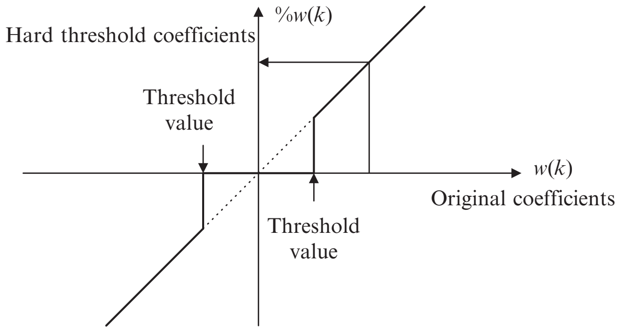


FIG. 12.45

Hard threshold for the DWT coefficients.

has a SNR = 19 dB. Similarly, the third and fourth plots illustrate that 93% and 97% of coefficients are set to zero after threshold operations and the recovered signals have the SNR = 23 and 28 dB, respectively. As an evidence that the signal is smoothed, that is, the high frequency noise is attenuated, the wavelet denoise technique is equivalent to lowpass filtering.

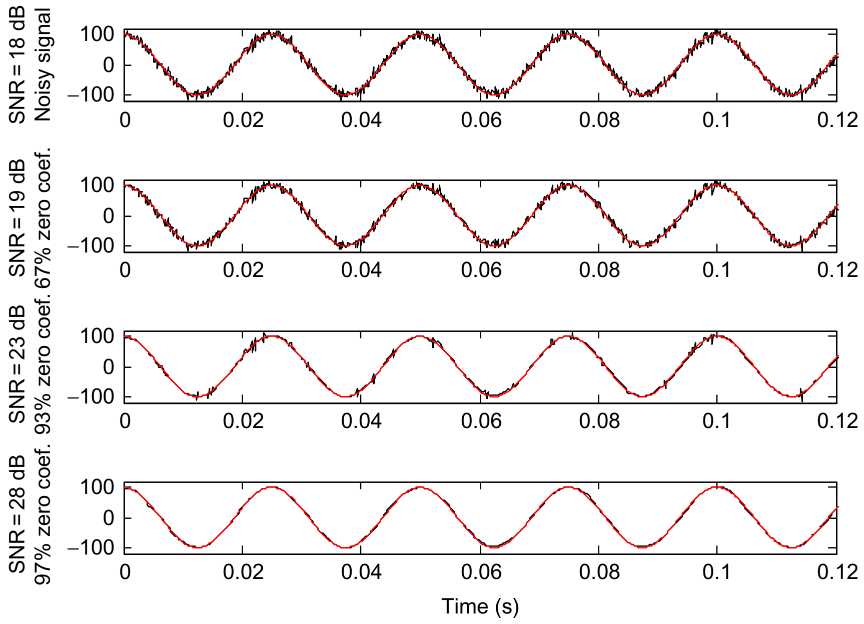


FIG. 12.46

Signal denoising using wavelet transform coding.

12.8 MATLAB PROGRAMS

In this section, four key MATLAB programs are included. **rconv()** and **fconv()** perform circular convolutions with the reversed filter coefficients and the forward filter coefficients, respectively. **dwt()** and **idwt()** are the programs to compute the DWT coefficients and IDWT coefficients. The resolution level can be specified.

Program 12.4. Circular convolution with the reversed filter coefficients (rconv.m).

```
function [y]=rconv(h,c)
% Circular convolution using the reversed filter coefficients h(-k)
% h=filter coefficients
% c=input vector
% y=output vector
N=length(c); M=length(h);
xx=zeros(1,M+N-1);
xx(1:N)=c;
xx(N+1:N+M-1)=c(1:M-1); % Use periodized input
for n=1:N;
  y(n)=0;
  for m=1:M
    y(n)=y(n)+h(m)*xx(n+m-1);
  end
end
```

Program 12.5. Circular convolution with the forward filter coefficients (fconv.m).

```
function [y]=fconv(h,c)
% Circular convolution using the forward filter coefficients h(k)
% h=filter coefficients
% c=input vector
% y=output vector
N=length(c); M=length(h);
x(1:N+M-1)=zeros(1,N+M-1);
  for j=1:N
  x(j:M+(j-1))=x(j:M+(j-1))+c(j)*h;
  end
  for i=N+M-1:-1:N+1
  x(i-N)=x(i-N)+x(i); % Circular convolution
  end
  y=x(1:N);
```

Program 12.6. DWT coefficients (dwt.m).

```
function w=dwt(h0,c,kLevel)
% w=dwt(h,c,k)
% Computes wavelet transform coefficients for a vector c using the
% orthonormal wavelets defined by the coefficients h
% h=wavelet coefficients
% c=input vector
% kLelvel=level
% w=wavelet coefficients
n=length(c); m=length(h0);
h1=h0(m:-1:1); h1(2:2:m)=-h1(2:2:m);
h0=h0(:)'; h1=h1(:)';
c=c(:); w=c;
x=zeros(n+m-2,1);
% Perform decomposition through k levels
% at each step, x=periodized version of x-coefficients
for j=1:kLevel
  x(1:n)=w(1:n);
  for i=1:m-2
  x(n+i)=x(i);
  end
  for i=1:n/2
  w(i)=h0 * x(1+2*(i-1):m+2*(i-1));
  w(n/2+i)=h1* x(1+2*(i-1):m+2*(i-1));
  end
  n=n/2;
end
```

Program 12.7. IDWT coefficients (idwt.m).

```
function c=idwt(h0,w,kLevel)
% c=idwt(h0,w,kLevel)
% Computes the inverse fast wavelet transform from data W using the
% orthonormal wavelets defined by the coefficients.
% h0=wavelet filter coefficients
% w=wavelet coefficients
% kLevel=level
% c=IDWT coefficients
n=length(w); m=length(h0);
h1=h0(m:-1:1); h1(2:2:m)=-h1(2:2:m);
h0=h0(:); h1=h1(:);
w=w(:); c=w;
x=zeros(n+m-2,1);
% Perform the reconstruction through k levels
% x=periodized version of x-coefficients
n=n/2^kLevel;
for i=1:kLevel
  x(1:2*n+m-2)=zeros(2*n+m-2,1);
  for j=1:n
  x(1+2*(j-1):m+2*(j-1))=x(1+2*(j-1):m+2*(j-1))+c(j)*h0+w(n+j)*h1;
  end
  for i=2*n+m-2:-1:2*n+1
  x(i-2*n)=x(i-2*n)+x(i);
  end
  c(1:2*n)=x(1:2*n);
  n=2 * n;
end
```

12.9 SUMMARY

1. A signal can be decomposed using a filter bank system. The filter bank contains two stages: the analysis stage and the synthesis stage. The analysis stage applies analysis filters to decompose the signal into multichannels. The signal from each channel is downsampled and coded. At the synthesis stage, the recovered signal from each channel is upsampled and processed using its synthesis filter. Then the outputs from all the synthesis filters are combined to produce the recovered signal.
2. Perfect reconstruction conditions for two-band case are derived to design the analysis and synthesis filters. The conditions consist of a half-band filter requirement and normalization. Once the lowpass analysis filter coefficients are obtained, the coefficients for other filters can be achieved from the derived relationships.
3. In a binary tree structure, a filter bank divides an input signal into two equal subbands, resulting in the low and high bands. Each band again splits into low and high bands to produce quarter bands. The process continues in this form.
4. The dyadic structure implementation of the filter bank first splits the input signal into low and high bands and then continues into split the low band only each time.

5. By quantizing each subband channel using the assigned number bits based on the signal significance (more bits are assigned to code the sample for a channel with large signal energy while less bits are assigned to code the samples with small signal energy), the subband-coding method demonstrates efficiency for data compression.
6. The wavelet transform can identify the frequencies in a signal and the times when the signal elements occur and end.
7. The wavelet transform provides either good frequency resolution or good time resolution, but not both.
8. The wavelet has two important properties: scaling and translation. The scaling process is related to changes of wavelet frequency (oscillation) while the translation is related to the time localization.
9. A family of wavelets contains a father wavelet and a mother wavelet, and their scaling and translation versions. The father wavelet and its scaling and translation are called the scaling functions while the mother wavelet and its scaling and translation are called the wavelet function.
   Each scaling function and wavelet function can be presented using the scaling functions in the next finer scale.
10. A signal can be approximated from a sum of weighted scaling functions and wavelet functions. The weights are essentially the DWT coefficients. A signal can also be coded at any desired level using the smaller-scale wavelets.
11. Implementation of DWT and IDWT consists of the analysis and synthesis stages, which are similar to the subband-coding scheme. The implementation uses the dyadic structure but with each analysis filter coefficients in a reversed format.
12. The DWT and IDWT are very effective for data compression or signal denoising by eliminating smaller DWT coefficients, which correspond to higher frequency components.

12.10 PROBLEMS

12.1 Given each of the following downsampling systems (Fig. 12.47A and B) and input spectrum $W(f)$, sketch the downsampled spectrum $X(f)$.

12.2 Given each of the following upsampling systems (Fig. 12.48A and B) and input spectrum $X(f)$, sketch the upsampled spectrum $\overline{W}(f)$. Note that the sampling rate for input $x(m)$ $f_{sM}=f_s/4$ and the output-sampling rate $\overline{w}(n)$ isf_s.

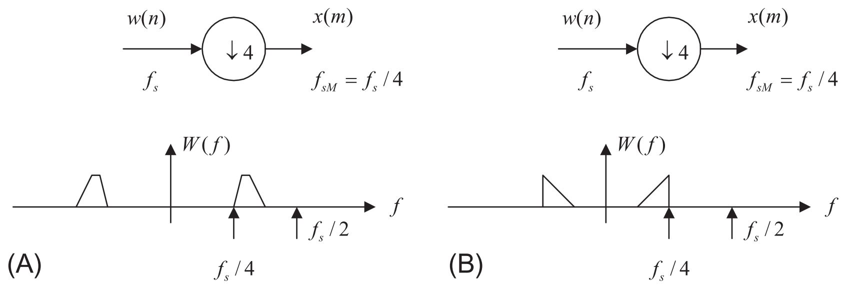


FIG. 12.47

Downsampling systems in Problem 12.1.

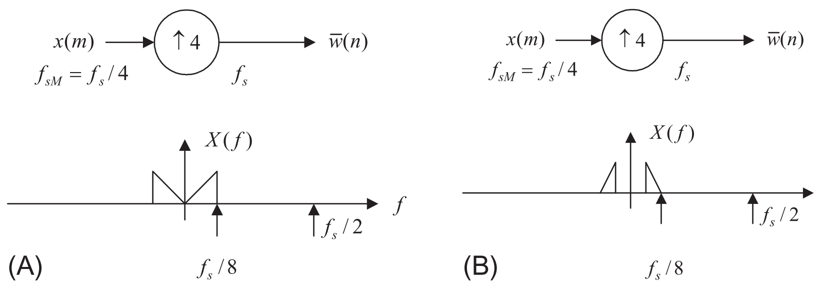


FIG. 12.48

Upsampling systems in Problem 12.2.

12.3 Given each of the following down- and upsampling systems (Fig. 12.49A and B) and the input spectrum $W(f)$, sketch output spectrum $\overline{Y}(f)$ and express $\overline{Y}(f)$ in terms of $W(f)$.

12.4 Given each of the following down- and upsampling systems (Fig. 12.50A and B) and the input spectrum $W(f)$, sketch output spectrum $\overline{Y}(f)$ and express $\overline{Y}(f)$ in terms of $W(f)$.

12.5 Given $H_0(z) = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}z^{-1}$, determine $H_1(z)$, $G_0(z)$, and $G_1(z)$.

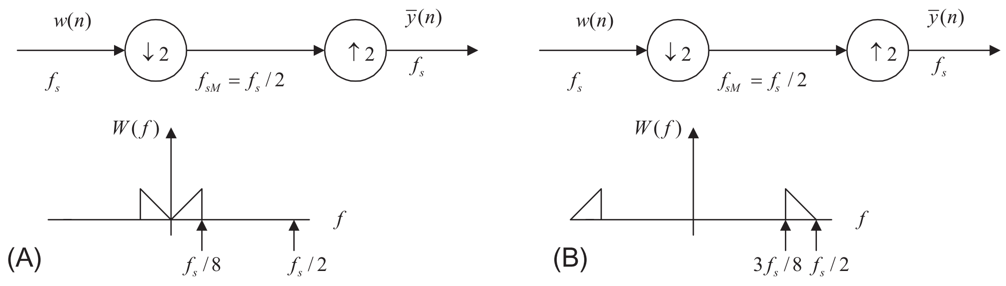


FIG. 12.49

Down- and upsampling systems in Problem 12.3.

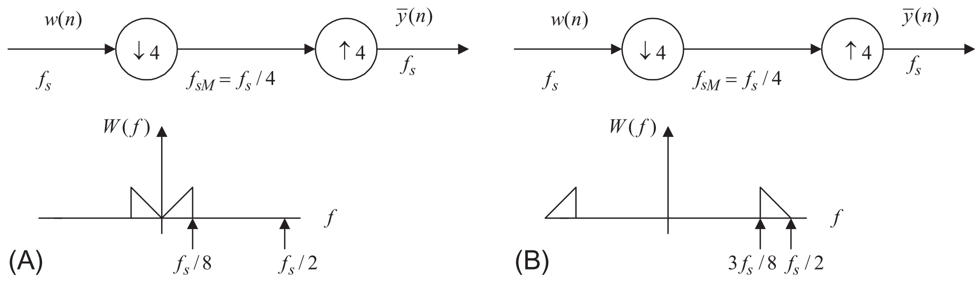


FIG. 12.50

Down- and upsampling systems in Problem 12.4.

12.6 Given $H_0(z) = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} z^{-1}$, verify the following conditions:

$$\rho(2n) = \sum_{k=0}^{N-1} h_0(k) h_0(k+2n) = \delta(n)$$
$$R(z) + R(-z) = 2,$$

and plot the magnitude frequency responses of analysis and synthesis filters.

12.7 Given

$$H_0(z) = 0.483 + 0.837z^{-1} + 0.224z^{-2} - 0.129z^{-3},$$

determine $H_1(z)$, $G_0(z)$, and $G_1(z)$.

12.8 Given

$$H_0(z) = 0.483 + 0.837z^{-1} + 0.224z^{-2} - 0.129z^{-3},$$

verify the following conditions:

$$\rho(2n) = \sum_{k=0}^{N-1} h_0(k) h_0(k+2n) = \delta(n)$$
$$R(z) + R(-z) = 2.$$

12.9 Draw a four-band dyadic tree structure of a subband system including the analyzer and the synthesizer.

12.10 Draw an eight-band dyadic tree structure of a subband system including the analyzer and the synthesizer.

12.11 Given a function in Fig. 12.51

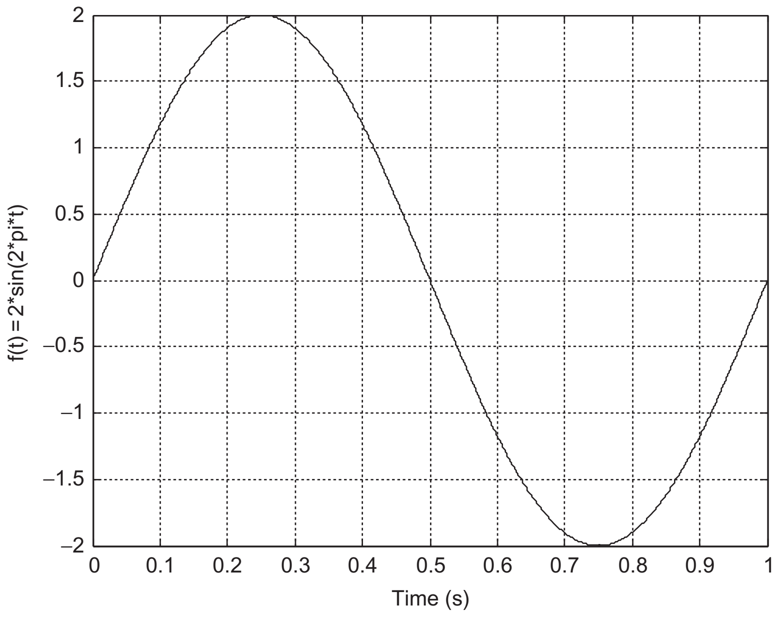


FIG. 12.51

A sine function in Problem 12.11.

Sketch

(a) $f(4t)$
(b) $f(t-2)$
(c) $f(2t-3)$
(d) $f(t/2)$
(e) $f(t/4-0.5)$.

12.12 Given a father wavelet (base scaling function) in base scale plotted in Fig. 12.52A, determine a and b for each of the wavelets plotted in Fig. 12.52B and C.

12.13 Given the signal as depicted in Fig. 12.53,

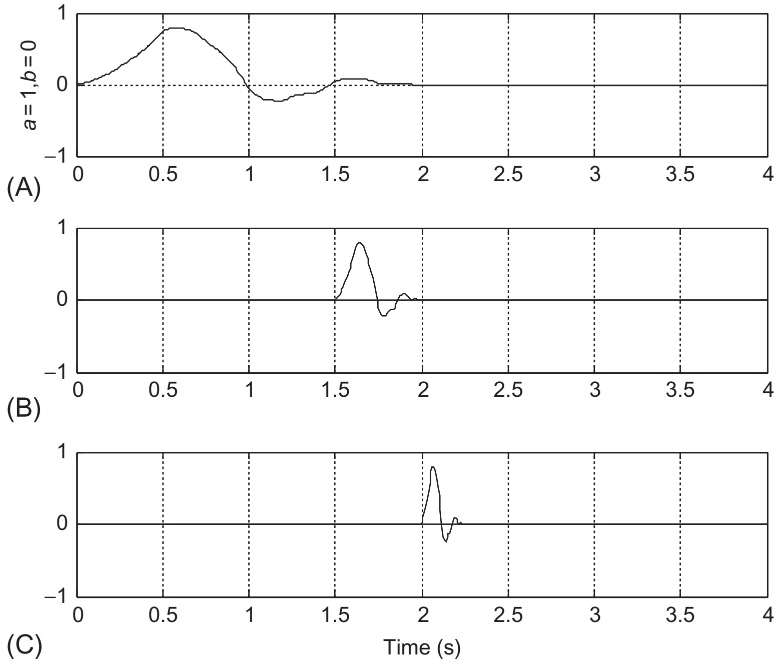


FIG. 12.52

Wavelets in Problem 12.12.

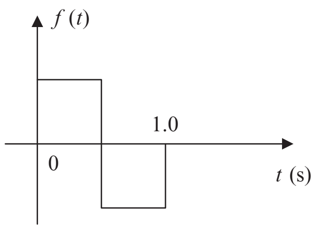


FIG. 12.53

The function in Problem 12.13.

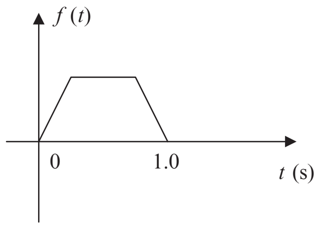


FIG. 12.54

A trapezoidal function in Problem 12.14.

Sketch

(a) $f(4t)$
(b) $f(t-2)$
(c) $f(2t-3)$
(d) $f(t/2)$
(e) $f(t/4-0.5)$.

12.14 Given the signal as shown in Fig. 12.54

Sketch

(a) $f(4t)$
(b) $f(t-2)$
(c) $f(2t-3)$
(d) $f(t/2)$
(e) $f(t/4-1)$.

12.15 Sketch the Haar father wavelet families for three different scales: $j=0, 1, 2$ for a period of 2 s.

12.16 Sketch the Haar mother wavelet families for three different scales: $j=0, 1, 2$ for a period of 2 s.

12.17 Use the Haar wavelet family to expand the signal depicted in Fig. 12.55.
(a) Use only scaling functions $\phi(2t-k)$.
(b) Use scaling functions and wavelets: $\phi(t)$ and $\psi(t-k)$.

12.18 Use the Haar wavelet family to expand the signal depicted in Fig. 12.56.
(a) Use only scaling functions $\phi(4t-k)$.
(b) Use scaling functions and wavelets: $\phi(2t-k)$ and $\psi(2t-k)$.
(c) Using the scaling function and wavelets: $\phi(t)$, $\psi(2t-k)$, and $\psi(t-k)$.

12.19 Use the Haar wavelet family to expand the signal

$$x(t) = \sin(2\pi t)\text{for}0 \le t \le 1.$$

(a) Use only scaling functions $\phi(2t-k)$.
(b) Use scaling functions and wavelets: $\phi(t)$ and $\psi(t-k)$.

12.20 Use the Haar wavelet family to expand the signal

$$x(t) = e^{-5t}\text{for}0 \le t \le 1.$$

(a) Use only scaling functions $\phi(2t-k)$.
(b) Use scaling functions and wavelets: $\phi(t)$ and $\psi(t-k)$.

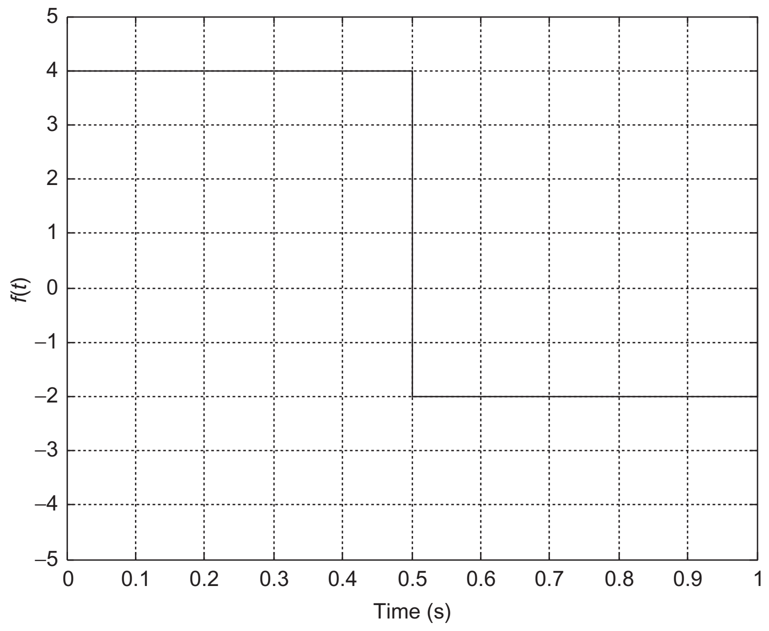


FIG. 12.55

A gate function in Problem 12.17.

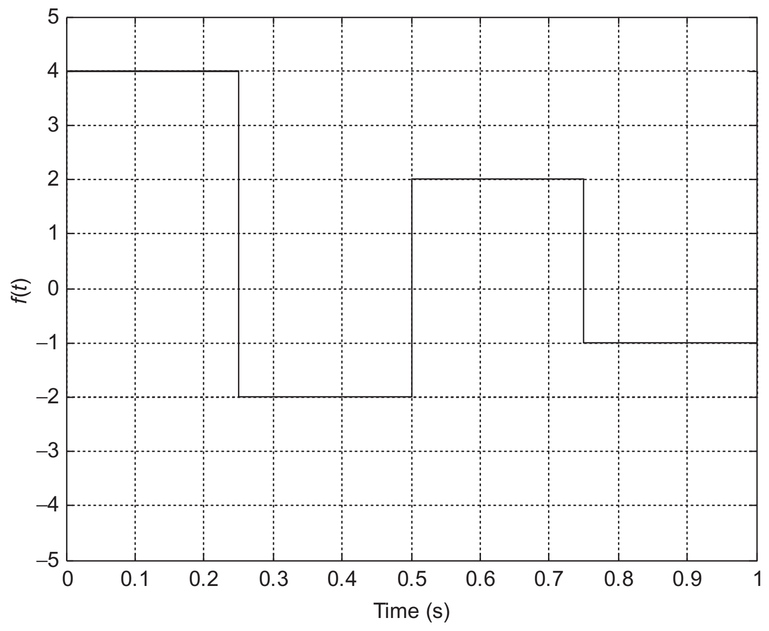


FIG. 12.56

A piecewise function in Problem 12.18.

12.21 Verify the following equations using the Haar wavelet families:

(a) $\phi(2t) = \sum_{k=-\infty}^{\infty} \sqrt{2} h_0(k)\phi(4t-k)$

(b) $\psi(2t) = \sum_{k=-\infty}^{\infty} \sqrt{2} h_1(k)\phi(4t-k)$.

12.22 Given the four-tap Daubechies wavelet coefficients

$$h_0(k) = [0.483\ 0.837\ 0.224\ -0.129].$$

Determine $h_1(k)$ and plot magnitude frequency responses for both $h_0(k)$ and $h_1(k)$.

12.23 Given the sample values [8 −2 4 1], use the Haar wavelet to determine the level-2 wavelet coefficients.

12.24 Given the sample values [8 −2 4 3 0 −1 −2 0], use the Haar wavelet to determine level-3 wavelet coefficients.

12.25 Given the level-2 wavelet coefficients [4 2 −1 2], use the Haar wavelet to determine the sampled signal vector $f(k)$.

12.26 Given the level-3 wavelet coefficients [4 2 −1 2 0 0 0 0], use the Haar wavelet to determine the sampled signal vector $f(k)$.

12.27 Given the level-1 wavelet coefficients [4 2 −1 2], use the Haar wavelet to determine the sampled signal vector $f(k)$.

12.28 The four-level DWT coefficients are packed below:
$W = [100\ 20\ 16\ –5\ –3\ 4\ 2\ –6\ 4\ 6\ 1\ 2\ –3\ 0\ 2\ –1]$.
List the wavelet coefficients to achieve each of the following compression ratios:

(a) 2:1

(b) 4:1

(c) 8:1

(d) 16:1.

MATLAB Problems

Use MATLAB to solve Problems 12.29–12.31.

12.29 Use the 16-tap PR-CQF coefficients and MATLAB to verify the following conditions:

$$\rho(2n) = \sum_{k=0}^{N-1} h_0(k)h_0(k+2n) = \delta(n);$$

$$R(z) + R(-z) = 2.$$

Plot the frequency responses for $h_0(k)$ and $h_1(k)$.

12.30 Use MATLAB functions provided in Section 12.8 [**dwt(), idwt()**] to verify Problems 12.23–12.27.

12.31 Given a 20-Hz sinusoidal signal plus random noise sampled at 8000 Hz with 1024 samples,

$$x(n) = 100\cos(2\pi \times 20nT) + 50 \times randn,$$

where $T = 1/8000$ s and *randn* is a random noise generator with a unit power and Gaussian distribution.

(a) Use a 16-bit code for each wavelet coefficient and write a MATLAB program to perform data compressions with the following ratios: 2:1, 4:1, 8:1, 16:1, and 32:1;

(b) Measure the SNR in dB for each case.

(c) Plot the reconstructed waveform for each case.

MATLAB Projects

12.32 Data compression using subband coding:

Given 16-bit speech data ("speech.dat") and using the four-band subband-coding method, write a MATLAB program to compress speech signal with the following specifications:

(a) 16 bits for each of subband coefficients, coding LL, LH, HL, HH subbands, and measure the SNR in dB;

(b) 16 bits for each of subband coefficients, coding LL band, discarding LH, HL, and HH subbands;

(c) 16 bits for each of subband coefficients, coding LL, LH band, discarding HL, and HH subbands;

(d) 16 bits for each of subband coefficients, coding LL, LH, HL band, discarding HH subband;

(e) Measure SNR in dB for (a), (b), (c), and (d);

(f) Determine the achieved compression ratios for (a), (b), (c), (d);

(g) Repeat (a) to (f) for seismic data ("seismic.dat") in which each sample is encoded using 32 bits instead of 16 bits.

12.33 Wavelet-based data compression:

Given 16-bit speech data ("speech.dat") and using the wavelet-coding method with 16-tap Daubechies wavelet filters, write a MATLAB program to compress the speech signal with the following specification:

(a) 16 bits for each of wavelet coefficients, compression 2:1;

(b) 16 bits for each of wavelet coefficients, compression 4:1;

(c) 16 bits for each of wavelet coefficients, compression 8:1;

(d) 16 bits for each of wavelet coefficients, compression 16:1;

(e) 16 bits for each of wavelet coefficients, compression 32:1;

(f) Measure SNR in dB for (a), (b), (c), (d), and (f);

(g) Repeat (a) to (f) for seismic data ("seismic.dat") in which each sample is encoded using 32 bits instead of 16 bits.