CHAPTER 26

Interpolation and Decimation

It is common to encounter in practice discrete-time systems involving sub-systems that operate at different sampling rates. In order to properly interface such sub-systems, it is necessary to be able to change the sampling rate of signals traversing the system, either by increasing the sampling rate or by decreasing it. The process of modifying the sampling rate of a signal is known as re-sampling or sampling rate conversion. If the sampling rate needs to be increased or decreased by an integer factor then the operations of interpolation or decimation can be used to accomplish the desired effect. If, on the other hand, the sampling rate needs to change by a rational factor then a combination of interpolation and decimation should be used. In this chapter we explain the steps that are involved in modifying the sampling rates of signals and discuss efficient structures for decimation and interpolation using polyphase realizations. In the next two chapters we apply these realizations to the discrete-time processing and filtering of signals by multirate systems.

26.1 DOWNSAMPLERS AND UPSAMPLERS

We discussed earlier in Secs. 9.7 and 14.2 the two procedures of signal downsampling and signal upsampling. These procedures will serve as building blocks for performing sampling rate conversion, as we explain in the next section. Here, and in preparation for our discussions in this chapter and the following chapters, we pause to highlight several useful properties of downsamplers and upsamplers. These properties will appear frequently in our treatment of multirate systems.

Downsampling

Recall that starting from a sequence $x(n)$, downsampling it by an integer factor M amounts to replacing $x(n)$ by the sequence

$$y(n) = x(nM), \quad n \geq 0 \tag{26.1}$$

In this construction, for every M samples of $x(n)$, a total of $M-1$ samples are discarded and one sample is retained. For example,

$$y(0) = x(0), \;\; y(1) = x(M), \;\; y(2) = x(2M), \;\; y(3) = x(3M) \tag{26.2}$$

Thus, observe that the samples of $x(n)$ at times $n = 1, 2, \ldots, M-1$ are discarded while the samples $x(0)$ and $x(M)$ are retained, and so forth. Observe further that the samples of $y(n)$ occur at a rate that is M times slower than the rate of $x(n)$. Specifically, the time lapse between samples $y(0)$ and $y(1)$, which amounts to one unit-time delay for the $y(\cdot)$ sequence, is equivalent to the time lapse between the M samples $\{x(0), x(1), \ldots, x(M)\}$, which amounts to a total of M unit-time delays for the $x(\cdot)$ sequence. We represent downsamplers by an arrow pointing downwards as shown in Fig. 26.1.

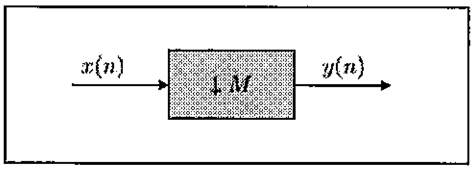

FIGURE 26.1 Block diagram representation of downsampling by an integer factor M.

Example 26.1 (Serial-to-parallel conversion)

Chains of downsamplers are very useful in performing serial-to-parallel conversion. Such chains appear frequently in the implementation of multirate systems. Let us refer to Fig. 26.2, which shows three branches of decimators, each by a factor of $M = 3$. In the diagram on the left, we show three parallel decimators, with input sequences $x(n)$, $x(n-1)$, and $x(n-2)$, respectively. Using definition (26.1), it is obvious that the corresponding output sequences will be $x(3n)$, $x(3n-1)$, and $x(3n-2)$; these results are obtained by simply replacing the variable n by $3n$. The diagram in the middle of the figure provides a more compact representation where two delay elements are added to generate the input sequences to the lower decimators.

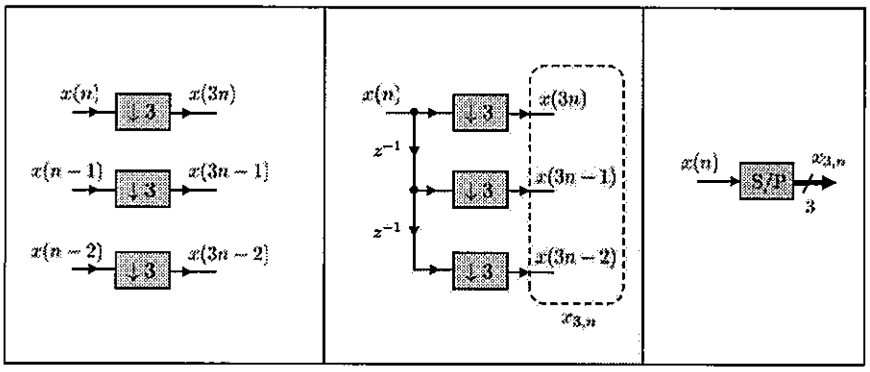

FIGURE 26.2 A chain of three downsamplers, each by a factor $M = 3$. The middle plot is the representation that we shall encounter often in our development. The diagram explains how the chain of downsamplers transforms a streaming sequence of samples into block vectors. In this example, the number of parallel branches is equal to the downsampling factor.

Let us illustrate the operation of the parallel branches by using letters to denote the samples of the sequence $x(n)$, say, as:

$$x(n) = \text{a b c d e f g h i j k l m n o p} \ldots \tag{26.3}$$

where the first sample a corresponds to time $n = 0$. Then, the samples of the delayed versions of $x(n)$ are given by

$$x(n-1) = \text{0 a b c d e f g h i j k l m n o p} \ldots \tag{26.4}$$

$$x(n-2) = \text{0 0 a b c d e f g h i j k l m n o p} \ldots \tag{26.5}$$

Accordingly, the samples at the outputs of the three decimators will be given by

$$x(3n) = \text{a d g j m p} \ldots \qquad (26.6)$$
$$x(3n-1) = \text{0 c f i l o} \ldots \qquad (26.7)$$
$$x(3n-2) = \text{0 b e h k n} \ldots \qquad (26.8)$$

If we collect the simultaneous samples at the output of the decimators into vectors of size 3 each, then the vectors that will appear in succession are given by

$$\left[\begin{array}{c|c|c|c|c|c} a & d & g & j & m & p \\ 0 & c & f & i & l & o \\ 0 & b & e & h & k & n \end{array}\right] \qquad (26.9)$$

To describe these vectors, we introduce the following block vector notation:

$$x_{3,n} = \begin{bmatrix} x(3n) \\ x(3n-1) \\ x(3n-2) \end{bmatrix} \qquad (26.10)$$

where the subscript 3 in $x_{3,n}$ is meant to indicate that this is a vector of size 3×1 with the number 3 also corresponding to the decimation factor. This vector captures the samples that appear simultaneously at the output of the three decimators. Now recall that the samples at the output of the decimators are generated at a rate that $M = 3$ times slower than the rate of the input sequence, $x(n)$. As such, a new vector $x_{3,n}$ is generated every $M = 3$ units of time relative to the original sequence, $x(n)$. The diagram on the far right in the figure provides a compact representation for the result: it shows that the chain of 3 decimators corresponds to a serial-to-parallel (S/P) converter. The converter receives the streaming samples of $x(n)$ and transforms them into block vectors of size $M = 3$ each. The thicker line at the output of the S/P converter is meant to indicate that it is a block vector, and the notation next to the line (short segment crossing the line and the number 3) represents the size of the block. More generally, M parallel branches of M−decimators with a chain of delays similar to the middle plot in Fig. 26.2 will result in block vectors of size M whose entries are given by

$$x_{M,n} \triangleq \begin{bmatrix} x(nM) \\ x(nM-1) \\ x(nM-2) \\ \vdots \\ x((n-1)M+1) \end{bmatrix} \qquad (26.11)$$

◇

Upsampling

Let us now examine the upsampling operation. Recall again that starting from a sequence $x(n)$, upsampling it by a factor L amounts to replacing the samples of $x(n)$ by the samples of the following sequence:

$$y(n) = \begin{cases} x(\frac{n}{L}), & \text{if } \frac{n}{L} \text{ is integer} \\ 0, & \text{otherwise} \end{cases} \qquad (26.12)$$

That is, upsampling amounts to inserting $L-1$ zeros between two successive samples of $x(n)$. We represent the upsampling operation by an upward arrow labeled by the factor L, as illustrated in Fig. 26.3. Observe that the samples of the original sequence, $x(n)$, are not lost or discarded during the upsampling operation. For this reason, as the next example illustrates, the upsampling operation is invertible: given the sequence $y(n)$, we can recover $x(n)$.

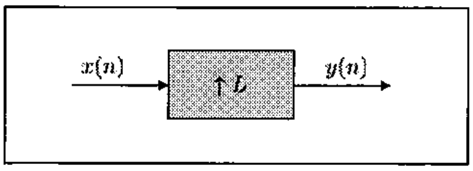

FIGURE 26.3 Block diagram representation of upsampling by a factor of L.

Example 26.2 (Upsampling can be reversed)

Assume a sequence $x(n)$ is upsampled by a factor L to generate a second sequence, $y(n)$. The upsampling operation inserts $L-1$ zeros between two successive samples of $x(n)$. Therefore, if the sequence $y(n)$ is now fed into a decimator by the same factor L, then this decimator ends up discarding the zero samples and we recover the original sequence $x(n)$. For this reason, the system formed by cascading an upsampler followed by a downsampler in Fig. 26.4, both of order L, behaves like an ideal link with transfer function equal to unity: the input to the cascade is $x(n)$ and its output will also be $x(n)$.

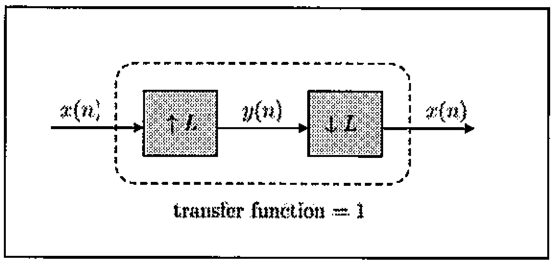

FIGURE 26.4 Upsampling by a factor L can be reversed by downsampling by the same factor.

◇

Example 26.3 (Parallel-to-serial conversion)

A chain of upsamplers can be used to perform parallel-to-serial conversion. Such chains also appear frequently in the implementation of multirate systems. Let us refer to Fig. 26.5, which shows three branches of upsamplers, each by a factor $L=3$. In the diagram on the left, we show three parallel upsamplers, with input sequences $x(3n)$, $x(3n-1)$, and $x(3n-2)$, respectively. The outputs of the samplers are combined through delay elements and we are claiming that the resulting output sequence is $x(n-2)$. This conclusion can be justified in a number of ways (e.g., it can be justified algebraically by carrying out calculations similar to what we do further ahead in Example 1.5). Here, we proceed instead from first principles as follows. Let us consider the same sequence $x(n)$ defined by (26.3) in Example 26.1. We already know from expressions (26.6)–(26.8) in that example that

$$\begin{aligned} x(3n) &= \text{a d g j m p} \ldots && (26.13)\\ x(3n-1) &= \text{0 c f i l o} \ldots && (26.14)\\ x(3n-2) &= \text{0 b e h k n} \ldots && (26.15) \end{aligned}$$

Upsampling these sequences by a factor $L = 3$ gives the following sequences, where two zeros are inserted between every pair of samples:

$$x_1(n) = \text{a00d00g00j00m00p}\ldots \tag{26.16}$$
$$x_2(n) = \text{000c00f00i00l00o}\ldots \tag{26.17}$$
$$x_3(n) = \text{000b00e00h00k00n}\ldots \tag{26.18}$$

Now we delay the first sequence by two units of time and delay the second sequence by one unit of time:

$$x_1(n-2) = \text{00a00d00g00j00m00p}\ldots \tag{26.19}$$
$$x_2(n-1) = \text{0000c00f00i00l00o}\ldots \tag{26.20}$$
$$x_3(n) = \text{000b00e00h00k00n}\ldots \tag{26.21}$$

If we add the samples of these three sequences pointwise we find that the samples at the output sequence are given by:

$$\text{00abcdefghijklmnop}\ldots = x(n-2) \tag{26.22}$$

as claimed. The diagram on the right side in the figure provides a compact representation of the result: it shows that the chain of 3 upsamplers corresponds to a parallel-to-serial (P/S) converter. The converter receives the block vectors $x_{3,n}$ and transforms them into streaming data with a delay of 2 units of time. More generally, L parallel branches of L—upsamplers with a chain of delays similar to the left plot in Fig. 26.5 will result in the output sequence, $x(n - M + 1)$, with a delay of $M - 1$ samples.

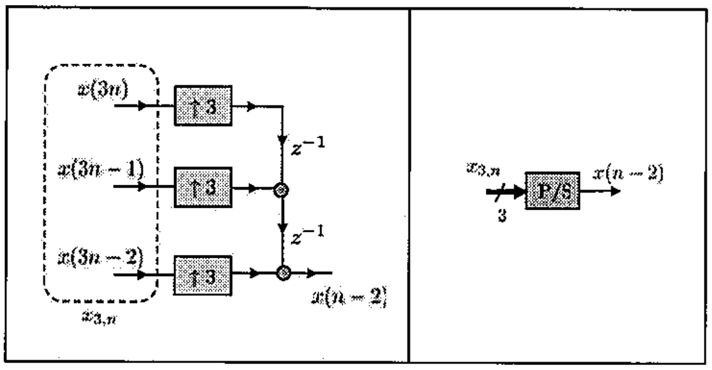

FIGURE 26.5 A chain of three upsamplers, each by a factor $L = 3$. The plot on the left is the representation that we shall encounter often in our development. The diagram explains how the chain of upsamplers transforms a sequence of block vectors into a streaming sequence. In this example, the number of parallel branches is equal to the upsampling factor. Observe that the output sequence has a delay of 2 samples.

◇

Example 26.4 (Perfect reconstruction)

The results of Examples 26.1 and 26.3 on S/P and P/S data conversion can be used to illustrate one fundamental concept in the study of multirate systems, namely, that of perfect reconstruction implementations. A multirate system is said to deliver perfect reconstruction (PR) if the transfer function between the input terminal and the output terminal is a pure delay. We shall examine PR systems in greater detail later in Secs. 27.5 and 28.6. Our objective here is to motivate one simple example. Thus, consider the implementation shown in Fig. 26.6, which simply combines the S/P and P/S structures from Figs. 26.2 and 26.5. The output signal will therefore be $x(n - M + 1)$ and the overall transfer function is seen to be $z^{-(M-1)}$.

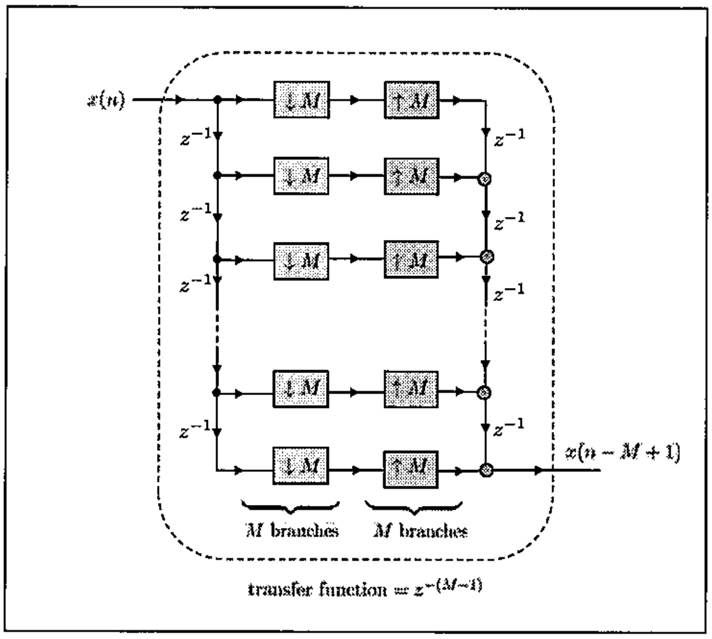

FIGURE 26.6 One example of a perfect reconstruction multirate system: the overall transfer function between the input terminal, $x(n)$, and the output terminal, $x(n-M+1)$, is equal to a pure delay and given by $z^{-(M-1)}$.

◇

Example 26.5 (Algebraic validation of PR property)

It is also possible to validate the perfect reconstruction property of multirate systems of the form shown in Fig. 26.6 algebraically, e.g., by using the z–transform representations of the various signals involved — see Prob. 26.23. Here we illustrate the calculations for the case involving two branches shown in Fig. 26.7. We denote the internal signals by $x_1(n)$, $u_1(n)$, and $u_2(n)$.

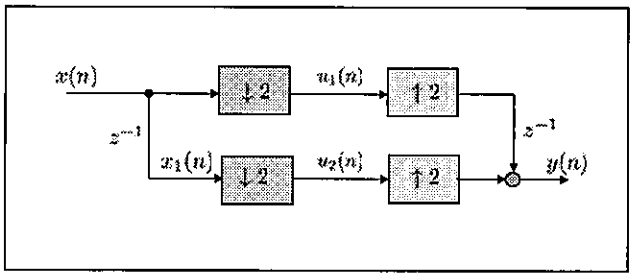

FIGURE 26.7 A perfect reconstruction filter bank consisting of two branches.

We know from the downsampling property (9.150) that

$$U_1(z) = \frac{1}{2}\left[X\left(z^{1/2}\right) + X\left(-z^{1/2}\right)\right] \tag{26.23}$$

$$U_2(z) = \frac{1}{2}\left[X_1\left(z^{1/2}\right) + X_1\left(-z^{1/2}\right)\right] \tag{26.24}$$

But since $X_1(z) = z^{-1}X(z)$, we get

$$U_2(z) = \frac{1}{2}\left[z^{-1/2}X\left(z^{1/2}\right) - z^{-1/2}X\left(-z^{1/2}\right)\right] \tag{26.25}$$

Subsequently, using the upsampling property (9.145),

$$\begin{aligned} Y(z) &= z^{-1}U_1(z^2) + U_2(z^2) \\ &= \frac{1}{2}\left[z^{-1}X(z) + z^{-1}X(-z)\right] + \frac{1}{2}\left[z^{-1}X(z) - z^{-1}X(-z)\right] \\ &= z^{-1}X(z) \end{aligned} \tag{26.26}$$

as claimed. That is, $y(n) = x(n-1)$.

$\diamond$

26.2 SAMPLING RATE CONVERSION

We now explain how the upsampling and downsampling operations can be used to achieve sampling rate conversions.

26.2.1 Increasing the Sampling Rate by an Integer Factor

We study first the case in which it is desired to increase the sampling rate (i.e., sampling frequency) of a signal by some integer factor (such as doubling or tripling it). We assume we are given samples $x(n)$ that have been generated by sampling a continuous-time signal, $x(t)$, at multiples of some sampling period, T_s, i.e.,

$$x(n) = x(t)|_{t=nT_s}, \quad n = 0, 1, 2\ldots \tag{26.27}$$

Our purpose is to replace the given samples $\{x(n)\}$ by another set of samples $\{x'(n)\}$ that would have been generated by sampling the original signal $x(t)$ at the finer sampling period:

$$T_s' = \frac{T_s}{L} \tag{26.28}$$

for some positive integer $L > 1$. In other words, the samples $x'(n)$ would correspond to the sampling operation:

$$x'(n) = x(t)|_{t=nT_s'}, \quad n = 0, 1, 2\ldots \tag{26.29}$$

The challenge is that we would like to generate the samples $\{x'(n)\}$ by working directly with the given samples $\{x(n)\}$ and without having to resort to the continuous-time signal $x(t)$ (which is generally unavailable). The process of transforming the sequence $x(n)$ to the sequence $x'(n)$ is called *interpolation* because it amounts to interpolating additional values between successive samples of $x(n)$.

In order to explain how this objective can be accomplished, it is useful to examine what happens in the frequency domain when we move from the sampling rate $\Omega_s = 2\pi/T_s$

radians/sec to the higher sampling rate $\Omega_s' = 2\pi/T_s'$ radians/sec. Thus, let $x_s(t)$ denote the sampled signal that results from sampling $x(t)$ at multiples of T_s:

$$x_s(t) = \sum_{n=-\infty}^{\infty} x(n)\delta(t - nT_s) \tag{26.30}$$

The amplitudes of the impulses within $x_s(t)$ are the samples $x(n)$. Likewise, let $x_s'(t)$ denote the sampled signal that results from sampling the same signal $x(t)$ at multiples of T_s':

$$x_s'(t) = \sum_{n=-\infty}^{\infty} x'(n)\delta(t - nT_s') \tag{26.31}$$

The amplitudes of the impulses within $x_s'(t)$ are now the desired samples $x'(n)$.

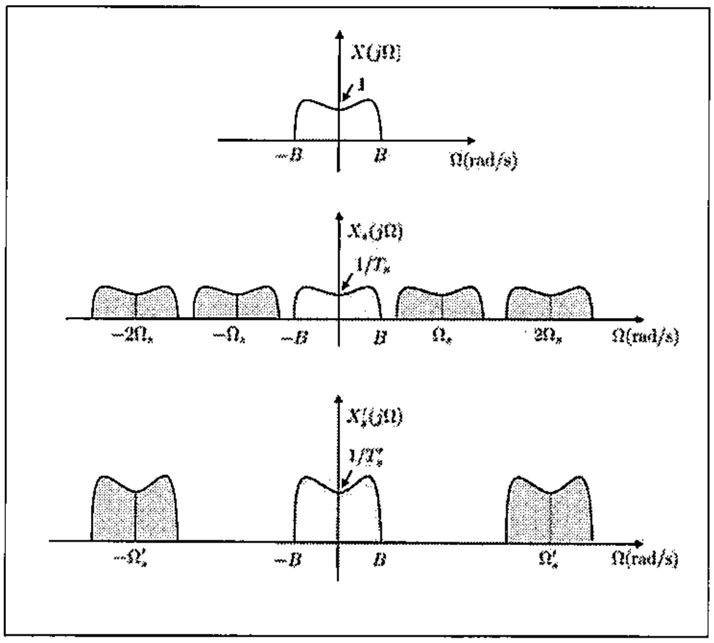

FIGURE 26.8 Sampling the signal $x(t)$, whose Fourier transform is shown in the top row, results in scaling the transform by $1/T_s$ and repeating it every Ω_s radians/second (middle plot). Likewise, sampling $x(t)$ at $\Omega_s' = 2\Omega_s$ radians/second results in scaling the transform by $1/T_s'$ and repeating it every Ω_s' radians/second (bottom plot).

Figure 26.8 illustrates the Fourier transforms of $x_s(t)$ and $x_s'(t)$ for the case $T_s' = T_s/2$ (or, equivalently, $\Omega_s' = 2\Omega_s$). The top row in the figure shows a generic Fourier transform $X(j\Omega)$ extending between $-B$ and B radians/second. The signal $x(t)$ is then sampled at the rate of Ω_s radians/second. The effect of the sampling operation in the frequency domain is that the transform $X(j\Omega)$ is scaled by $1/T_s$ and repeated every Ω_s radians/second to generate $X_s(j\Omega)$, as shown in the middle plot of the same figure. If we instead sample $x(t)$ at the higher rate $\Omega_s' = 2\Omega_s$, its Fourier transform is scaled by $1/T_s'$ and repeated every Ω_s' radians/second to generate $X_s'(t)$. The result is shown in the last row of Fig. 26.8. We observe that the repeated images in the Fourier transform of $x_s'(t)$ are further apart from each other than the images in the Fourier transform of $x_s(t)$; moreover, the images in $X_s'(j\Omega)$ are scaled by the larger factor $1/T_s'$.

It is instructive to examine what happens to the Fourier transforms of the sampled signals $x_s(t)$ and $x'_s(t)$ in the normalized domain by calling upon the results from Sec. 22.7. Recall that in the normalized domain, the frequency variable Ω (measured in radians/second) is replaced by the normalized variable ω (measured in radians/sample) through the transformation

$$\omega = \Omega \times (\text{sampling period}) \tag{26.32}$$

Under this mapping, the sampling frequency (whether it is Ω_s or Ω'_s) is always mapped to the same value $\omega = 2\pi$ radians/sample. Moreover, the Fourier transforms $X_s(j\Omega)$ and $X'_s(j\Omega)$ are replaced by the DTFTs of the sequences $x(n)$ and $x'(n)$, respectively, and these DTFTs are related to the Fourier transforms of the sampled signals as follows:

$$X(e^{j\omega}) = X_s\left(j\frac{\omega}{T_s}\right), \qquad X'(e^{j\omega}) = X'_s\left(j\frac{\omega}{T'_s}\right) \tag{26.33}$$

Figure 26.9 plots the DTFTs of the sequences $x(n)$ and $x'(n)$ for the same situation of Fig. 26.8. Observe how the DTFT of $x(n)$ now lies within the normalized range $\omega \in [-BT_s, BT_s]$ in the interval $[-\pi, \pi]$. Moreover, the DTFT of $x'(n)$ has its magnitude magnified by the factor $L = 2$ relative to the DTFT of $x(n)$ and its frequency support is compressed by the same factor so that its DTFT now lies within the smaller normalized range $\omega \in [-BT'_s, BT'_s]$ radians/sample.

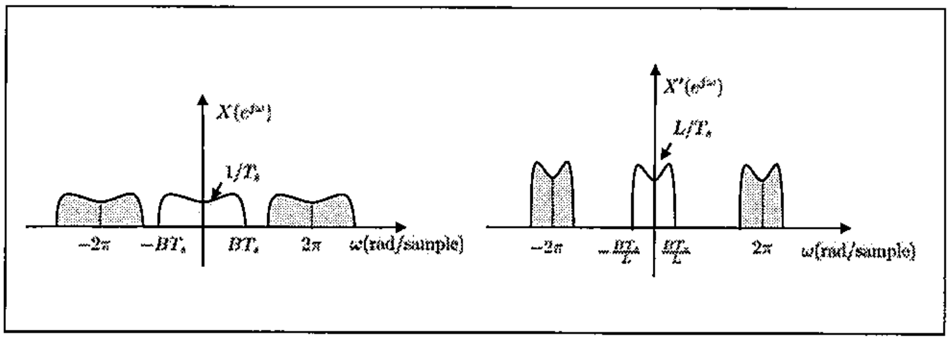

FIGURE 26.9 The bandwidth of the DTFT of $x'(n)$ is compressed by a factor of L and its amplitude is magnified by the same factor L relative to the DTFT of $x(n)$. The figure illustrates the case $L = 2$ for which $T'_s = T_s/2$.

Useful conclusion. *When the sampling rate of a signal $x(t)$ is increased by an integer factor L, the bandwidth range of $X'(e^{j\omega})$ will be reduced by L and its amplitude will be magnified by L, both in relation to $X(e^{j\omega})$.*

The discussion suggests that if we can develop a procedure that starts with $X(e^{j\omega})$ and produces $X'(e^{j\omega})$, then we should be able to arrive at the desired sequence $x'(n)$. This task can be accomplished with the aid of an upsampler — see Fig. 26.3. While studying upsampling in Sec. 14.2, we established the following useful result relating the DTFTs of a sequence and its upsampled version:

$$\boxed{Y(e^{j\omega}) = X\left(e^{j\omega L}\right)} \tag{26.34}$$

That is, the DTFT of the upsampled sequence, $y(n)$, is compressed in frequency by a factor of L relative to the DTFT of the original sequence, $x(n)$. In order to illustrate this effect, we consider the case $L = 2$ and refer to Fig. 26.10. The top row in the figure shows the Fourier transform, $X(j\Omega)$, of some signal $x(t)$; its frequency content is assumed to be bandlimited to the range $\Omega \in [-B, B]$ radians/second. The DTFT of the sequence $x(n)$ that results from sampling $x(t)$ at any rate $\Omega_s > 2B$ radians/second is displayed in the second row of the figure: the DTFT repeats itself every 2π radians/sample and is scaled by $1/T_s$. Moreover, the bandwidth of $x(n)$ in the normalized domain extends over the range $\omega \in [-BT_s, BT_s]$ radians/sample. In view of Nyquist condition that $\Omega_s > 2B$, and using $\Omega_s T_s = 2\pi$, we conclude that the normalized frequency BT_s satisfies $BT_s < \pi$ radians/sample in this example.

The DTFT of the upsampled sequence $y(n)$ appears in the third row of Fig. 26.10. Observe how, according to (26.34), the DTFT of $y(n)$ is compressed by a factor of $L = 2$ and, consequently, the images that were originally centered around $\pm 2\pi$ are now centered around $\pm 2\pi/L$. As a result, the DTFT of the upsampled sequence, $y(n)$, exhibits images within $[-\pi, \pi]$ relative to the DTFT of the original sequence, $x(n)$. The fourth row in Fig. 26.10 exhibits the frequency response of a low-pass filter, $H(e^{j\omega})$, with cutoff frequency π/L and gain L. The purpose of the filter is to remove the images that are present in $Y(e^{j\omega})$ around $\pm\pi/L$ and to generate the DTFT that is shown in the last row of the figure. Comparing with the DTFTs in Fig. 26.9, which tell us how the DTFTs of $x(n)$ and $x'(n)$ relate to each other, we conclude that the DTFT in the last row of Fig. 26.10 coincides with the DTFT of the desired sequence $x'(n)$.

Interpolation. *In summary, we conclude that interpolation by a factor of L can be accomplished by cascading an upsampler with a low-pass filter as illustrated in Fig. 26.11: the sequence $x(n)$ is upsampled by a factor of L followed by low-pass filtering with cutoff frequency $\frac{\pi}{L}$ radians/sample and gain L. The cascade combination of the upsampler and the low-pass filter is called an interpolator.*

Figure 26.12 illustrates schematically the operation of an interpolator for $L = 4$; it displays the upsampled sequence $y(n)$ and the interpolated output $x'(n)$. Observe that the samples of $x'(n)$ are obtained by operating *directly* on the available samples $x(n)$; the original continuous-time signal $x(t)$ is not needed. It is a remarkable feat that by working directly with $x(n)$, we are able to interpolate additional values between successive samples. This fact should come as no surprise because the sequence $x(n)$ has been generated by sampling $x(t)$ at a rate that satisfies Nyquist's criterion. As such, its samples contain sufficient information to represent the original signal $x(t)$ since, as we showed earlier in the reconstruction procedure (22.93), the value of $x(t)$, for any t, can be fully recovered from knowledge of the samples, $x(n)$.

26.2.2 Decreasing the Sampling Rate by an Integer Factor

Let us now study the reverse problem in which we are interested in replacing a sequence $x(n)$ by another sequence, $x''(n)$, whose samples correspond to sampling $x(t)$ at the larger sampling period:

$$T_s'' = MT_s \tag{26.35}$$

for some positive integer $M > 1$. That is, we would like to generate the sequence

$$x''(n) = x(t)\big|_{t=nT_s''}, \quad n = 0, 1, 2, \ldots \tag{26.36}$$

whose samples are separated further from each other than the samples of $x(n)$. Again, we would like to achieve this objective by working directly with the available samples $x(n)$, and without having to resort to the continuous-time signal, $x(t)$ (which is generally

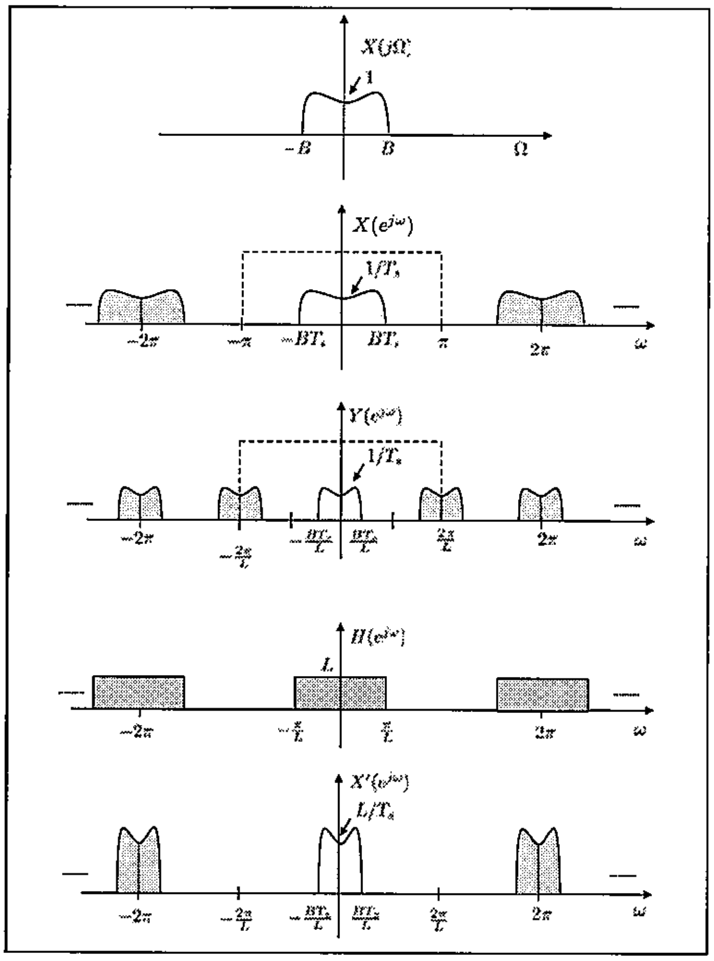

FIGURE 26.10 Illustration of the effect of upsamling on the DTFT of a sequence for the case $L = 2$. Observe how images are added within the range $[-\pi, \pi]$ in the DTFT of the upsampled sequence, $y(n)$, relative to the original sequence $x(n)$ (second and third rows). The fourth row shows the DTFT of an ideal low-pass filter, whose purpose is to extract the low-pass portion of $Y(e^{j\omega})$; this filter has gain L. The last row shows the resulting DTFT after low-pass filtering; it coincides with the DTFT of the desired sequence $x'(n)$.

unavailable). The operation of replacing $x(n)$ by $x''(n)$ is called *decimation* because we are decimating (eliminating) some sample values by increasing the sampling interval.

On first sight, it appears from (26.36) that we can obtain the desired samples $x''(n)$ directly from the samples $x(n)$ by simply discarding some of the samples since

$$x''(n) = x(nM) \tag{26.37}$$

However, as we now explain, this is generally not the manner by which the desired sequence $x''(n)$ is generated. The problem lies in the fact that $x''(n)$ amounts to sampling $x(t)$ at the lower sampling rate $\Omega''_s = 2\pi/T''_s$. As such, aliasing may occur if the value of Ω''_s is so low that it does not satisfy Nyquist's condition anymore, i.e., if it does not satisfy

$$\Omega''_s > 2B \tag{26.38}$$

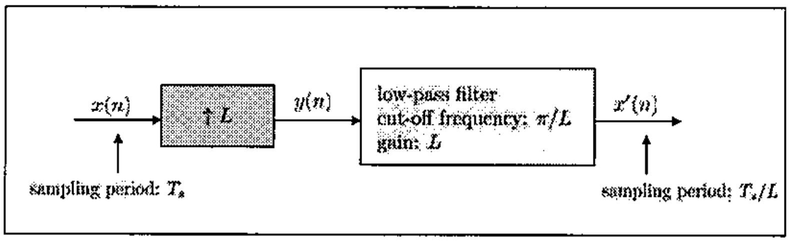

FIGURE 26.11 The cascade combination of an upsampler followed by a low-pass filter with cutoff frequency π/L and gain L results in upsampling the rate of $x(n)$ by a factor L.

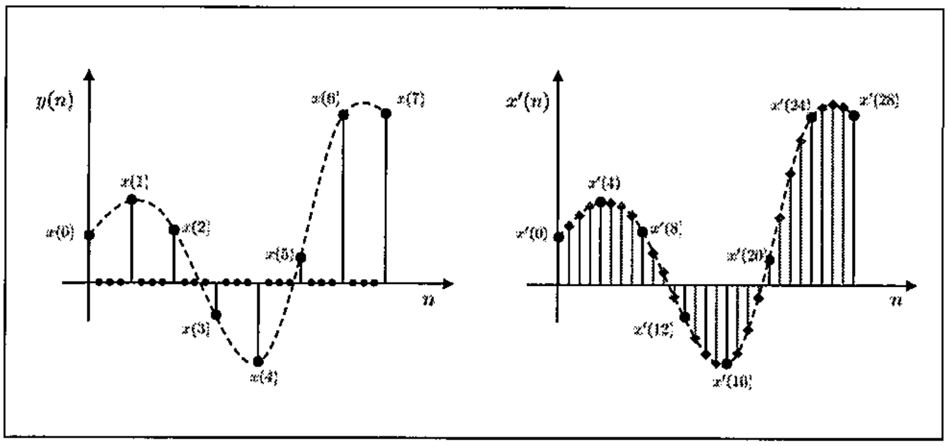

FIGURE 26.12 Example of an upsampled sequence $y(n)$ (left) and the interpolated version $x'(n)$ for $L = 4$ (right).

where B radians/second denotes the bandwidth of $x(t)$. If condition (26.38) is violated, then the samples $x''(n)$ defined by (26.36) would not correspond to a faithful representation of $x(t)$ because we would not be able to recover the signal $x(t)$ from them due to aliasing. In this case, relation (26.37) would not be valid. We illustrate this situation by examining what happens in the frequency domain when we move from the sampling rate $\Omega_s = 2\pi/T_s$ radians/second to the lower sampling rate $\Omega_s'' = 2\pi/T_s''$ radians/sec. Thus, let $x_s''(t)$ denote the sampled signal that results from sampling the signal $x(t)$ at multiples of T_s'':

$$x_s''(t) = \sum_{n=-\infty}^{\infty} x''(n)\delta(t - nT_s'') \tag{26.39}$$

where the amplitudes of the impulses within $x_s''(t)$ are the samples $x''(n)$.

Figure 26.13 illustrates the Fourier transforms of $x_s(t)$ and $x_s''(t)$ for the case $T_s'' = 2T_s$ (or, equivalently, $\Omega_s'' = \Omega_s/2$). The top row in the figure shows a generic Fourier transform extending over the range $\Omega \in [-B, B]$ radians/second. When the signal $x(t)$ is sampled at the rate of Ω_s radians/second, its Fourier transform is scaled by $1/T_s$ and repeated it every Ω_s radians/second to generate $X_s(j\Omega)$, as shown in the second row of the same figure. If we instead sample $x(t)$ at the lower rate $\Omega_s'' = \Omega_s/2$, then its Fourier transform is scaled by $1/T_s''$ and repeated every Ω_s'' radians/second to generate $X_s''(j\Omega)$, as shown in the third row of the figure.

Observe that the repeated images in the Fourier transform of $x_s''(t)$ are closer to each other than the images that appear in the Fourier transform of $x_s(t)$; moreover, the images

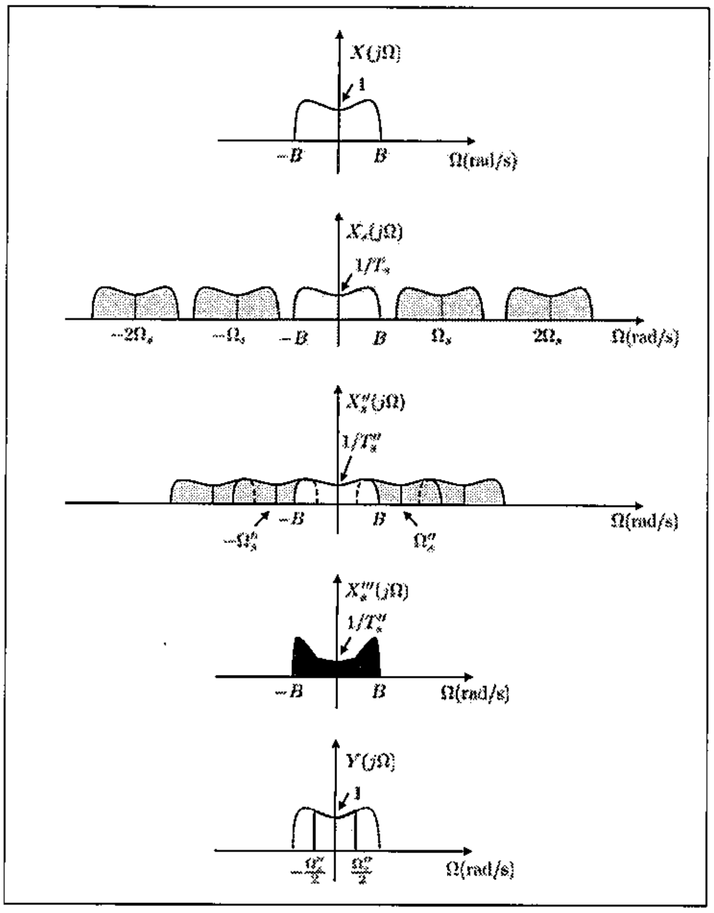

FIGURE 26.13 Sampling the signal $x(t)$, whose Fourier transform is shown in the top row, results in scaling its transform by $1/T_s$ and repeating it every Ω_s radians/second (second row). Likewise, sampling $x(t)$ at $\Omega''_s = \Omega_s/2$ radians/second results in scaling the transform by $1/T''_s$ and repeating it every Ω''_s radians/second (third row). Aliasing would not occur at the lower sampling rate if the original signal, $X(j\Omega)$, is first low-pass filtered in order to limit its bandwidth to the range $[-\Omega''_s/2, \Omega''_s/2]$. The bandlimited signal is denoted by $y(t)$ and its Fourier transform is displayed in the bottom plot.

in $X''_s(j\Omega)$ are scaled by the smaller factor $1/T''_s$. The fact that the images in $X''_s(j\Omega)$ get close to each other suggests that if Ω''_s is not large enough (specifically, if it does not satisfy (26.38)), then aliasing occurs. The third row in the figure illustrates this possibility. Clearly, if Ω''_s satisfies Nyquist's condition (26.38), then aliasing is avoided and construction (26.37) is sufficient; the construction maintains the samples of $x(n)$ that occur at multiples of M and discards all other samples. This operation can be accomplished with the aid of a downsampler, as shown in Fig. 26.14.

However, in general, downsampling can result in aliasing because the lower sampling rate, Ω''_s, maybe in violation of Nyquist's condition. When aliasing occurs, the transform $X''_s(j\Omega)$ over $\Omega \in [-B, B]$ will be a distorted version of the original transform, $X(j\Omega)$ — compare the top and fourth rows of Fig. 26.13; the fourth row shows the distorted transform, denoted by $X'''(j\Omega)$, over the same range $\Omega \in [-B, B]$. In this case, the samples $x''(n)$ would be representative of the signal $x'''(t)$ whose transform is $X'''(j\Omega)$ and not of $x(t)$. It is for this reason that decimation is instead performed by combining the downsampling step (26.37) with a pre-filtering step whose purpose is to avoid aliasing. The

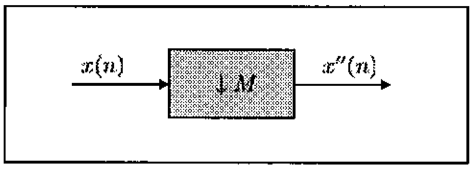

FIGURE 26.14 Block diagram representation of downsampling by a factor of M. When the lower sampling rate satisfies Nyquist's condition (26.38), this structure is sufficient to generate the desired samples $x''(n)$. More generally, as explained in the sequel, the structure used to perform decimation includes a pre-filtering step to avoid aliasing when Nyquist's condition (26.38) is not satisfied.

combination of the two operations of low-pass filtering and downsampling will result in an alternative downsampled sequence, $y''(n)$, such that $y''(n)$ is representative of a significant portion of $X(j\Omega)$ without distortion even when Nyquist's condition (26.38) is violated. To explain how this is accomplished, we note first that aliasing would not occur at the lower sampling rate if the original signal, $x(t)$, is first processed by a low-pass filter with gain equal to one and with cutoff frequency equal to $\Omega_s''/2$ in order to limit the bandwidth of the resulting signal, $y(t)$, to the range $\Omega \in [-\Omega_s''/2, \Omega_s''/2]$. Thus, let

$$Y(j\Omega) = X(j\Omega)H(j\Omega) \tag{26.40}$$

where $H(j\Omega)$ denotes the frequency response of the low-pass filter. The Fourier transforms $X(j\Omega)$ and $Y(j\Omega)$ are shown in the top and bottom rows of Fig. 26.13. Obviously, if (26.38) happens to be satisfied, then the signals $x(t)$ and $y(t)$ would coincide when $H(j\Omega)$ is an ideal or close-to-ideal low-pass filter. The filtering operation (26.40) helps ensure that the transforms $X(j\Omega)$ and $Y(j\Omega)$ essentially agree over the range of frequencies $\Omega \in [-\Omega_s''/2, \Omega_s''/2]$ (we say "essentially" to accommodate close-to-ideal low-pass filters).

Now let $y_s(t)$ denote the sampled signal that results from sampling the signal $y(t)$ at multiples of T_s:

$$y_s(t) = \sum_{n=-\infty}^{\infty} y(n)\delta(t - nT_s) \tag{26.41}$$

where

$$y(n) = y(t)|_{t=nT_s} \tag{26.42}$$

Then, the transformation from $x(t)$ to $y(t)$ can be accomplished directly in the discrete-time domain by working with $x(n)$ instead of $x(t)$ in order to generate the samples $y(n)$. This is achieved by low-pass filtering the sequence $x(n)$, as indicated in Fig. 26.15. The top two rows in the figure show the Fourier transforms of the sampled signals, $x_s(t)$ and $y_s(t)$, which are obtained from sampling $x(t)$ and $y(t)$, respectively, at the rate Ω_s radians/second. Both transforms are periodic with period Ω_s and scaled by $1/T_s$. The third and fourth rows of the figure show the DTFTs of the sequences $x(n)$ and $y(n)$. The dotted lines in the third row indicate the cutoff frequency of the discrete-time low-pass filter that is needed to transform $X(e^{j\omega})$ into $Y(e^{j\omega})$; the cutoff frequency is at $\omega_c = \pi/M$ radians/sample. This is because the cutoff frequency $\Omega_s''/2$ corresponds to the normalized frequency:

$$\omega_c = \frac{\Omega_s''}{2}T_s = \frac{2\pi}{2T_s''}T_s = \frac{\pi T_s}{T_s''} = \frac{\pi}{M} \tag{26.43}$$

Now the task of generating $y''(n)$ from $y(n)$ can be accomplished with the aid of a downsampler. Starting from the sequence $y(n)$, downsampling it by a factor M amounts

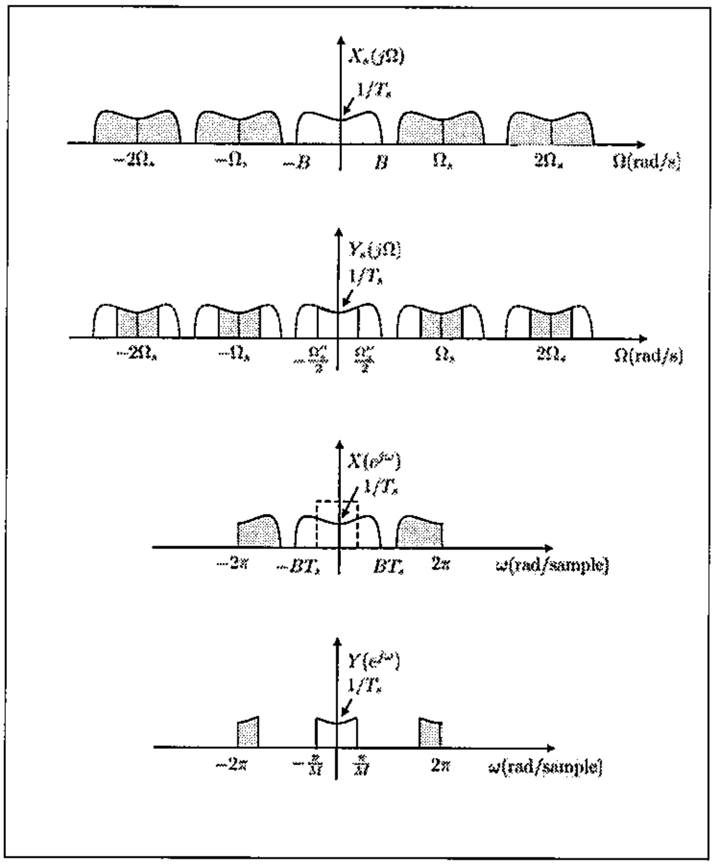

FIGURE 26.15 The top two rows show the Fourier transforms of the sampled signals, $x_s(t)$ and $y_s(t)$, which are obtained from sampling $x(t)$ and $y(t)$, respectively, at the rate Ω_s radians/second. Both transforms are periodic with period Ω_s and scaled by $1/T_s$. The third and fourth rows of the figure show the DTFTs of the sequences $x(n)$ and $y(n)$. The dotted lines in the third row indicate the cutoff frequency of the discrete-time low-pass filter that is needed to transform $X(e^{j\omega})$ into $Y(e^{j\omega})$; the cutoff frequency is at $\omega_c = \pi/M$ radians/sample.

to replacing its samples by the sequence

$$y''(n) = y(nM) \tag{26.44}$$

This operation amounts to retaining only samples of $y(n)$ that occur at multiples of M and discarding all other samples. The sequence $y''(n)$ obtained via (26.44) would coincide with the desired sequence $x''(n)$ from (26.36) when the sequences $x(n)$ and $y(n)$ agree with each other (which happens in the absence of aliasing, i.e., when Nyquist's condition (26.38) is satisfied). On the other hand, when aliasing occurs, there is distortion in going from $x(n)$ to $y(n)$ due to the low-pass filtering operation. Nevertheless, the samples $y''(n)$ uniquely determine $y(t)$, whose Fourier transform agrees with the transform of $x(t)$ over the range $\Omega \in [-\Omega_s''/2, \Omega_s''/2]$. We shall adopt the sequence $y''(n)$ as the desired downsampled sequence $x''(n)$ from (26.36).

It is instructive to examine the DTFT of the sequence $y''(n)$ more closely. Recall that we established earlier in Sec. 14.2 the following result, which relates the DTFTs of $y(n)$

and its downsampled version $y''(n)$:

$$\boxed{Y''(e^{j\omega}) = \frac{1}{M}\sum_{k=0}^{M-1} Y\left(e^{\frac{j(\omega-2\pi k)}{M}}\right)} \tag{26.45}$$

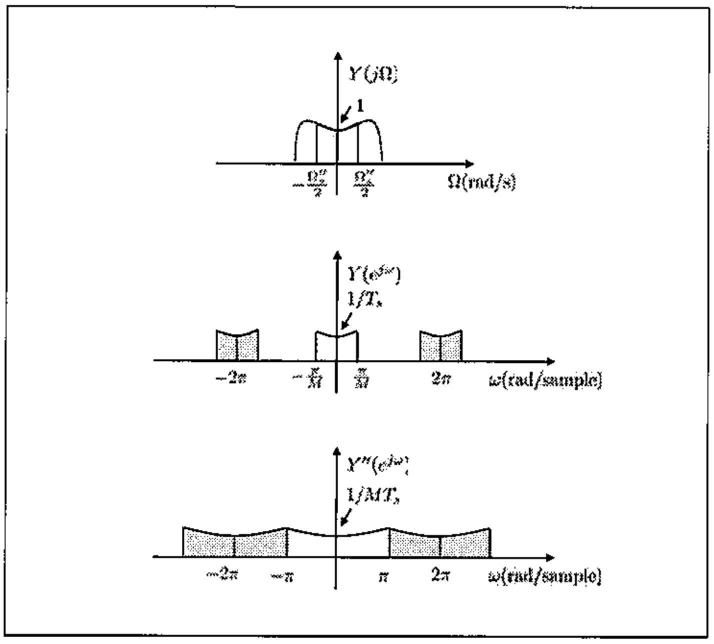

FIGURE 26.16 The bandwidth of the DTFT of $y''(n)$ is expanded by a factor of $M = 2$ and its amplitude is scaled down by the same factor M relative to the DTFT of $y(n)$.

Useful conclusion. *When the sampling rate of a signal $y(t)$ is decreased by an integer factor M, the bandwidth range of $Y''(e^{j\omega})$ will be expanded by M and its amplitude will be reduced by M, both in relation to $Y(e^{j\omega})$.*

Figure 26.16 illustrates the DTFTs of the sequences $y(n)$ and $y''(n)$ for the same situation of Fig. 26.15. Observe how the DTFT of $y''(n)$ is (scaled and) expanded and appears within the larger normalized range $\omega \in [-\pi, \pi]$. In particular, for this example, since the sampling rate is decreased by an integer factor of $M = 2$, we find that the bandwidth of $Y''(e^{j\omega})$ is increased by a factor of $M = 2$ compared to the bandwidth of $Y(e^{j\omega})$. More generally, when the sampling rate is decreased by an integer factor of M, then the bandwidth of $Y''(e^{j\omega})$ will be increased by the same factor M and its amplitude will be scaled down by M as well.

Decimation. *In summary, decimation by a factor of M can be accomplished by cascading a low-pass filter with gain one and cutoff frequency π/M with a downsampler, as illustrated in Fig. 26.17: the sequence $x(n)$ is first low-pass filtered with cutoff frequency π/M followed by downsampling by a factor M. The cascade combination of the low-pass filter and the downsampler is called a decimator.*

We shall denote the output of the decimator by $x''(n)$ instead of $y''(n)$ from now on.

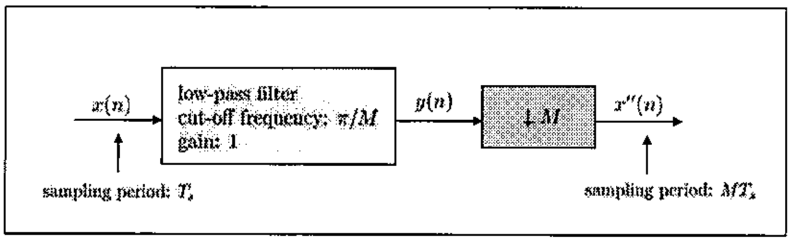

FIGURE 26.17 The cascade combination of a low-pass filter with cutoff frequency π/M and gain one followed by a downsampler results in downsampling the rate of $x(n)$ by a factor M. Moving forward, we will denote the output of the decimator by $x''(n)$ instead of $y''(n)$.

26.2.3 Modifying the Sampling Rate by a Rational Factor

Now that we know how to increase or decrease the sampling rate of a sequence by an integer factor, we can cascade decimators and interpolators to achieve sampling rate conversions by rational factors. For example, the series cascade of an interpolator that increases the sampling rate by a factor of L with a decimator that reduces the sampling rate by a factor of M results in a system that modifies the sampling rate by the rational factor L/M. This structure is shown in the top part of Fig. 26.18 where the output sequence is denoted by $x_r(n)$. The interpolator and the decimator must be used in the order indicated in the figure, with the interpolator coming first. This is because performing decimation first generally leads to loss of information due to the pre-filtering operation that is intended to prevent aliasing; this pre-filtering step limits the bandwidth of the signal to $[-\pi/M, \pi/M]$ radians/sample and discards the portion of the spectrum that lies outside this range. On the other hand, performing interpolation first already limits the bandwidth of the resulting signal to π/L radians/sample, which can be useful for the subsequent decimation operation. The two low-pass filters can be combined into a single low-pass filter with gain L and cutoff frequency:

$$\omega_c = \min\left\{\frac{\pi}{L}, \frac{\pi}{M}\right\} \quad \text{(radians/sample)} \tag{26.46}$$

The resulting structure is shown in the bottom part of the same figure.

Example 26.6 (Sampling rate conversion)

A continuous-time signal $x(t)$ has been sampled at the rate of 6 KHz. The result is a sequence $x(n)$. We would like to generate the sequence $x_r(n)$ that would have resulted from sampling the original signal $x(t)$ at the rate of 8 KHz. In this case, we want to modify the sampling rate by the factor

$$\frac{8}{6} = \frac{4}{3} \tag{26.47}$$

If we select the integers $L = 4$ and $M = 3$, then the sequence $x_r(n)$ can be generated by using the structure of Fig. 26.18 where the low-pass filter should have gain 4 and cutoff frequency $\pi/4$ radians/sample.

◇

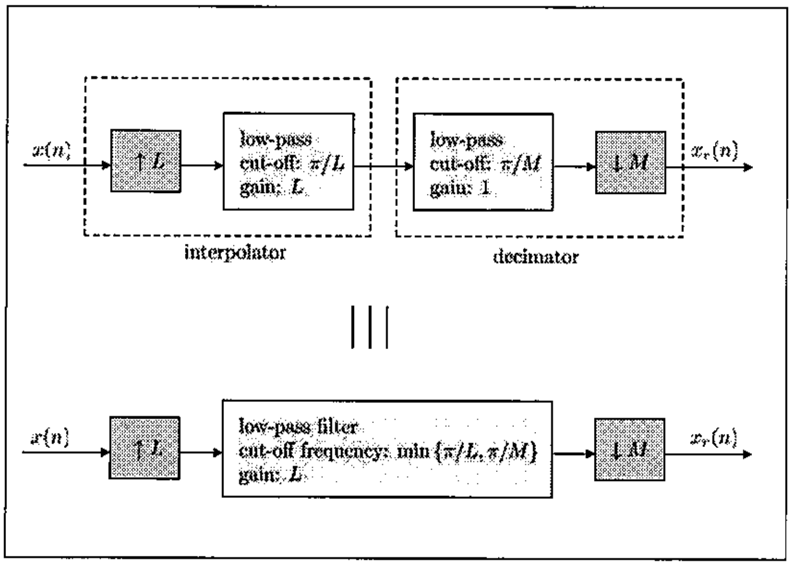

FIGURE 26.18 The series cascade of an interpolator and a decimator allows modifying the sampling rate of a sequence by a rational factor L/M.

Example 26.7 (Audio signals)

Audio signals are usually bandlimited at $B = 20$KHz. These signals are sampled at $F_s = 48$ KHz in professional audio systems and at $F_s = 44.1$ KHz in compact disc (CD) storage systems. To transfer data stored in one system to the other we need to match their sampling rates. For example, in order to convert the sampling rate of audio recorded on CDs to match the sampling rate of audio recorded by professional systems we need to modify the sampling rate by a factor of

$$\frac{48}{44.1} = \frac{480}{441} = \frac{160}{147} \tag{26.48}$$

If we select the integers $L = 160$ and $M = 147$, then the sampling rate conversion can be generated by using the structure of Fig. 26.18 where the low-pass filter should have gain 160 and cutoff frequency $\pi/160$ radians/sample.

◇

Example 26.8 (Multisage re-sampling)

Let us continue with the same example dealing with audio signals, which requires using an interpolator with $L = 160$ and a decimator with $M = 147$. Now observe that the integers L and M can be factored as

$$L = 2 \times 2 \times 2 \times 2 \times 2 \times 5 \tag{26.49}$$
$$M = 3 \times 7 \times 7 \tag{26.50}$$

This decomposition suggests that the interpolator $L = 160$ and the decimator $M = 147$ can be implemented in multiple stages by using a sequence of interpolators and decimators of lower orders. In order to accomplish an interpolation factor of $L = 160$ we can interpolate in multiple stages as follows by using, for example, the smallest factors first (so that the largest factor is used at the highest sampling rate):

1. First stage uses $L = 2$.
2. Second stage uses $L = 2$.

3. Third stage uses $L = 2$.
4. Fourth stage uses $L = 2$.
5. Fifth stage uses $L = 2$.
6. Sixth stage uses $L = 5$.

Likewise, in order to accomplish a decimation factor of $M = 147$ we can decimate in multiple stages as follows by using, for example, the largest factors first (so that the largest factor is again used at the highest sampling rate):

1. First stage uses $M = 7$.
2. Second stage uses $M = 7$.
3. Third stage uses $M = 3$.

The resulting structure is shown in Fig. 26.19.

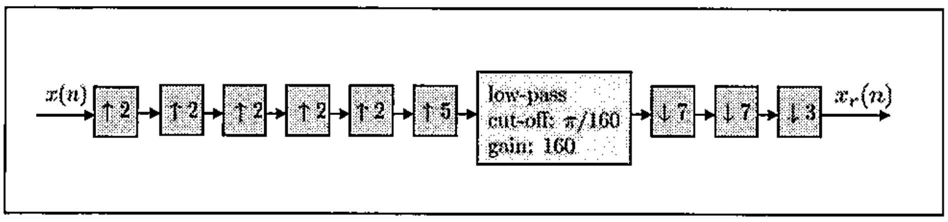

FIGURE 26.19 A cascade of upsamplers and downsamplers for Example 26.8.

◇

26.3 POLYPHASE REALIZATIONS

The interpolation and decimation structures reproduced in Fig. 26.20 are highly inefficient. This is because the low-pass filters are either operating on unnecessary samples or generating unnecessary samples. For example, the upsampler adds $L - 1$ zeros between every two successive samples of $x(n)$. This means that for every collection of L samples, the low-pass filter in the interpolator implementation operates on $L - 1$ zero samples, which is a wasteful operation. Likewise, the downsampler discards $M - 1$ samples from every M samples generated by the low-pass filter in the decimator implementation. These observations motivate us to seek more efficient structures. In the process of deriving the alternative structures in this section, we will introduce two important concepts that will play a prominent role in the study of multirate systems. The first concept is that of *polyphase realizations* and the second concept is related to a set of *equivalence relations*.

26.3.1 Equivalence Relations

To begin with, recall from the discussion in Sec. 9.7 that downsampling a sequence $x(n)$ by a factor of M amounts to retaining only one sample of $x(n)$ for every M samples. Specifically, the following samples are retained:

$$x(0), x(M), x(2M), \ldots, x(nM), \ldots \tag{26.51}$$

and all other samples are discarded. It follows that downsampling the *sum* of two sequences by a factor of M can be equivalently accomplished by downsampling each sequence separately by the same factor M and then adding the downsampled sequences. This equivalence

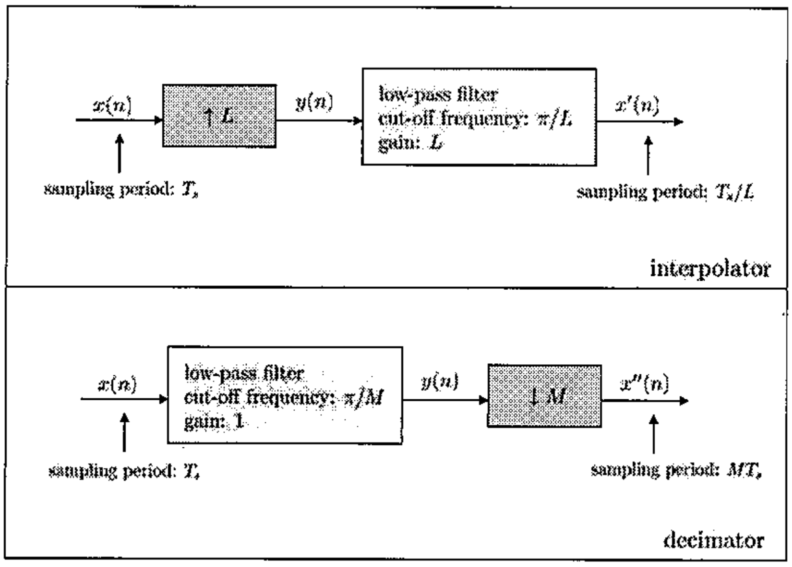

FIGURE 26.20 The interpolator (left) and decimator (right) structures involve the use of low-pass filters either following an upsampler or preceding a downsampler.

is illustrated in the first row of Fig. 26.21. Likewise, downsampling a scaled multiple of a sequence is equivalent to scaling the downsampled version of the sequence, as illustrated in the second row of Fig. 26.21.

In a similar manner, recall that upsampling a sequence $x(n)$ by a factor of L amounts to inserting $L-1$ zeros between two successive samples of $x(n)$. It then follows that upsampling a sequence by a factor of L and transmitting the upsampled sequence simultaneously over two separate branches is equivalent to first transmitting the original sequence over these branches and upsampling the bifurcated signals separately. This equivalence is illustrated in the third row of Fig. 26.21. Likewise, upsampling a scaled multiple of a sequence is equivalent to scaling the upsampled version of the sequence, as illustrated in the last row of Fig. 26.21.

Initial Structure for Decimation

Now let us refer to the structure shown in the top part of Fig. 26.22, which consists of some FIR filter $H(z)$ followed by a downsampler of factor M. When the filter $H(z)$ happens to be low-pass with cut-off frequency π/M, then this structure serves as a decimator. More generally, we will examine the structure for arbitrary FIR filters, $H(z)$, say, of the form:

$$H(z) = h(0) + h(1)z^{-1} + h(2)z^{-2} + \ldots + h(N-1)z^{-(N-1)} \qquad (26.52)$$

where N denotes the duration of the impulse response sequence, $h(n)$. The bottom left-part of Fig. 26.22 replaces $H(z)$ by its direct realization (cf. Sec. 23.1).

Since the downsampler maintains only one of every M samples generated by the FIR filter, it is obviously wasteful to generate the samples that are going to be discarded. To get an indication about how wasteful the direct implementation can be, we observe that each sample $y(n)$ at the output of $H(z)$ requires the evaluation of $N-1$ additions and N multiplications. Since M such samples need to be generated by the filter before a single sample, $x''(n)$, is generated by the downsampler, we find that the evaluation of each output sample, $x''(n)$, in this manner would require a total of MN multiplications and $M(N-1)$

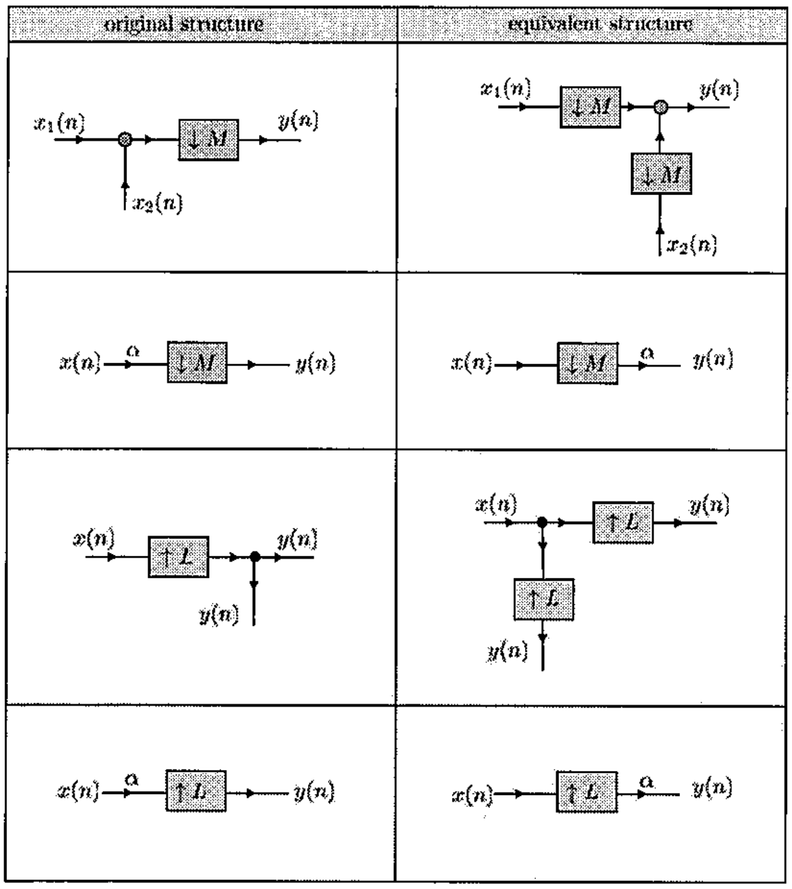

FIGURE 26.21 Downsamping the sum of two sequences by a factor of M is equivalent to adding the downsampled versions of both sequences by the same factor (first row). Likewise, downsampling a scaled multiple of a sequence is equivalent to scaling the downsampled version of the sequence (second row). On the other hand, upsampling before or after the bifurcation results in the same sequences $y(n)$ (third row), while upsampling a scaled multiple of a sequence is equivalent to scaling the upsampled version of the sequence (last row).

additions. A more efficient implementation can be obtained by exploiting the equivalences of Fig. 26.21.

We start by noting that the downsampler in the bottom left-part of Fig. 26.22 appears after a summation node. We can therefore move it backwards and successively through the various summation nodes and use the equivalences in the first two rows of Fig. 26.21 to arrive at the equivalent structure shown in the bottom right-part of Fig. 26.22. In this equivalent structure, all multiplications by the coefficients $\{h(n)\}$ now appear after the downsamplers. It then follows that the evaluation of each output sample, $x''(n)$, now involves N multiplications and $N-1$ additions. In this way, we are able to reduce the total number of multiplications and additions by a factor of M for each sample $x''(n)$.

Initial Structure for Interpolation

Likewise, let us consider the structure shown in the top part of Fig. 26.23 with an FIR filter $H(z)$ following an upsampler. Again, when the filter $H(z)$ happens to be low-pass with cut-off frequency π/L, then this structure serves as an interpolator. More generally, we will examine the structure for arbitrary FIR filters, $H(z)$, with transfer functions expressed

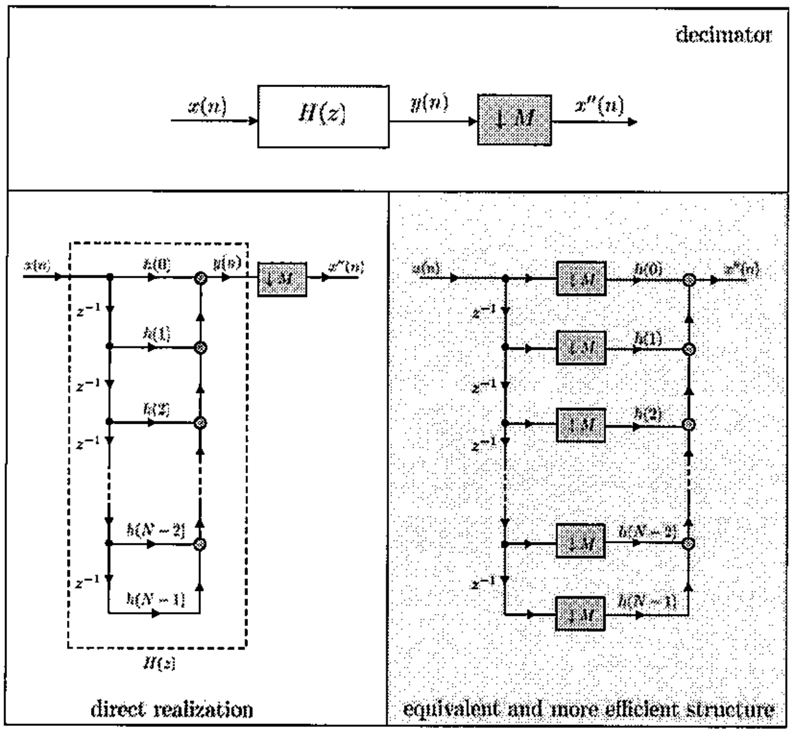

FIGURE 26.22 An FIR filter, $H(z)$, is followed by a downsampler. The top part is a block diagram realization and the bottom left-part replaces $H(z)$ by its direct realization. The bottom right-part is a more efficient implementation that reduces the amount of computations by a factor M.

in the form (26.52). The bottom left-part of Fig. 26.23 replaces $H(z)$ by its transposed realization (cf. Sec. 23.3); observe that in this case we are using the transposed realization for the FIR filter as opposed to its direct realization, as was the case with downsampling in Fig. 26.22. The reason for this choice is that the transposed structure will allow us to exploit the equivalences of Fig. 26.21 more directly.

Since the upsampler inserts $L-1$ zeros between successive samples of $x(n)$, most of the samples that are being fed into $H(z)$ are zero samples. This is again a wasteful operation. Indeed, in this implementation, the generation of each sample, $x'(n)$, by $H(z)$ would generally require the evaluation of N multiplications and $N-1$ additions. Therefore, the generation of a succession of L successive samples of $x'(n)$ would require a total of LN multiplications and $L(N-1)$ additions. However, only one of every L samples fed into $H(z)$ (and multiplied by its coefficients $\{h(n)\}$) is nonzero and, therefore, most of the multiplications are being wasted during the generation of the L successive samples of $x'(n)$. A more efficient implementation can be obtained by resorting to the equivalences of Fig. 26.21.

Thus, observe that the upsampler in the bottom left-part of Fig. 26.23 appears before a bifurcation node. We can therefore move it forward and successively through the bifurcation nodes and apply the equivalences in the last two rows of Fig. 26.21 to arrive at the equivalent structure shown in the bottom right-part of Fig. 26.23. In this equivalent structure, multiplications by the coefficients $\{h(n)\}$ occur before the upsamplers. It then follows that the evaluation of L successive output samples, $x'(n)$, now involves N multiplications and $N-1$ additions and we are therefore able to reduce the total number of multiplications and additions by a factor of L.

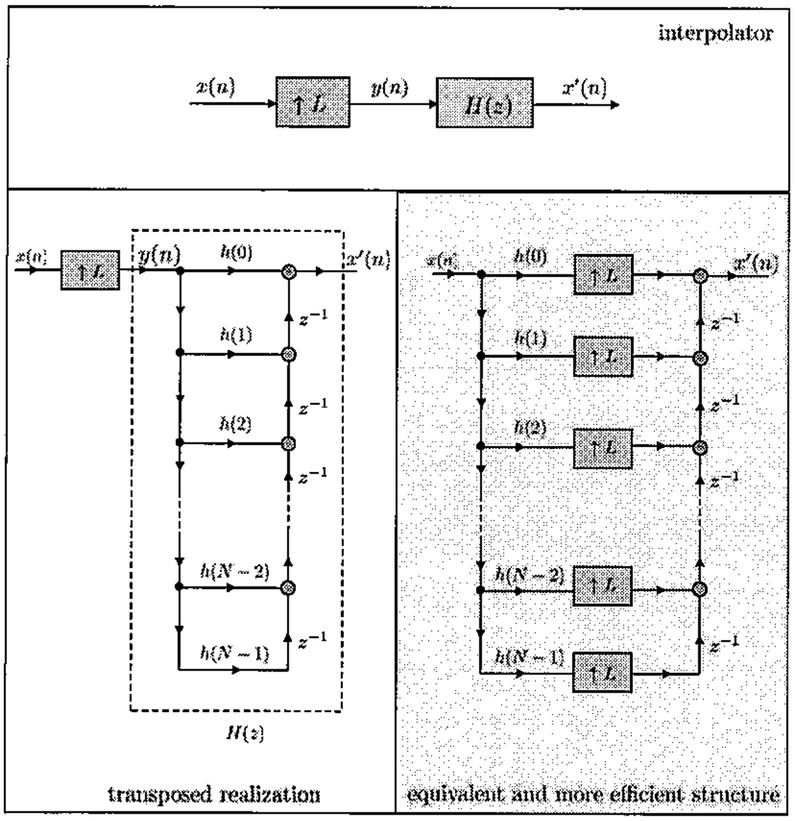

FIGURE 26.23 An FIR filter, $H(z)$, follows an upsampler (top) and a transposed realization is used for $H(z)$ in the bottom left-part of the figure. A more efficient realization is shown in the bottom right-part of the figure, which reduces the amount of computations by a factor L.

26.3.2 Polyphase Decomposition

The efficient filter structures derived in Figs. 26.22 and 26.23 were motivated by starting from the direct and transposed realizations of the FIR filter $H(z)$. We shall explain in the next section that these efficient structures are actually special cases of more general *polyphase* realizations for the filter $H(z)$. In the meantime, and in preparation for that discussion, we need to explain what is meant by the polyphase decomposition of an FIR filter.

Recall that filtering a signal $x(n)$ through a filter $H(z)$ amounts to convolving the signal with the impulse response sequence of the filter. Now, we described earlier in Sec. 19.3 two block convolution methods that were based on the idea of segmenting one of the sequences into smaller segments and transforming the original convolution problem into the computation of several smaller convolutions. Polyphase realizations of FIR filters are based on a similar idea: they partition the system $H(z)$ into smaller sub-systems (called polyphase components) and filter the signal $x(n)$ through these smaller sub-systems prior to combining the results.

To introduce the idea, consider the example of an FIR filter of duration $N = 6$, say,

$$H(z) = h(0) + h(1)z^{-1} + h(2)z^{-2} + h(3)z^{-3} + h(4)z^{-4} + h(5)z^{-5} \tag{26.53}$$

We group together the terms involving the even-indexed samples of $h(n)$, and group separately the terms involving the odd-indexed samples of $h(n)$, and write

$$\begin{aligned} H(z) &= \left[h(0)+h(2)z^{-2}+h(4)z^{-4}\right] \;+\; \left[h(1)z^{-1}+h(3)z^{-3}+h(5)z^{-5}\right] \\ &= \left[h(0)+h(2)z^{-2}+h(4)z^{-4}\right] \;+\; z^{-1}\left[h(1)+h(3)z^{-2}+h(5)z^{-4}\right] \end{aligned} \tag{26.54}$$

If we now introduce the second-order polynomials in z^{-1}:

$$E_o(z) \;\stackrel{\Delta}{=}\; h(0)+h(2)z^{-1}+h(4)z^{-2} \tag{26.55}$$

$$E_1(z) \;\stackrel{\Delta}{=}\; h(1)+h(3)z^{-1}+h(5)z^{-2} \tag{26.56}$$

then we can express $H(z)$ in terms of these components as follows:

$$H(z) \;=\; E_o(z^2) \;+\; z^{-1}E_1(z^2) \tag{26.57}$$

The factors $\{E_o(z), E_1(z)\}$ are called the *second-order* polyphase components of $H(z)$. The decomposition (26.57) indicates that $H(z)$ can be implemented as the parallel connection of two subsystems: $E_o(z^2)$ and $z^{-1}E_1(z^2)$ — see the two plots in the middle row of Fig. 26.24, where the rightmost plot uses a transposed implementation with the delays appearing on the right-hand side (cf. Sec. 23.3). Since $E_o(z^2)$ and $E_1(z^2)$ are FIR filters, any of the realizations discussed in Chapter 23 can be used to implement them.

We could have pursued an alternative decomposition of $H(z)$ in terms of *third-order* polyphase components as follows:

$$H(z) \;=\; E_o(z^3) \;+\; z^{-1}E_1(z^3) \;+\; z^{-2}E_2(z^3) \tag{26.58}$$

where now

$$E_o(z) \;=\; h(0)+h(3)z^{-1} \tag{26.59}$$

$$E_1(z) \;=\; h(1)+h(4)z^{-1} \tag{26.60}$$

$$E_2(z) \;=\; h(2)+h(5)z^{-1} \tag{26.61}$$

In this case, the polyphase components are formed as follows. We cycle through the samples of $h(n)$: (a) the first sample, $h(0)$, is assigned to $E_o(z)$; (b) the second sample, $h(1)$, is assigned to $E_1(z)$ and (c) the third sample, $h(2)$, is assigned to $E_2(z)$. Then, we repeat with $\{h(3), h(4), h(5)\}$ assigned to $\{E_o(z), E_1(z), E_2(z)\}$, respectively. Clearly, the third-order polyphase component, $E_o(z)$, in (26.59) is different from the second-order polyphase component, $E_o(z)$, in (26.55). Likewise, for the other polyphase components. The polyphase decomposition (26.58) again indicates that the same $H(z)$ can be implemented as the parallel connection of three subsystems: $E_o(z^3)$, $z^{-1}E_1(z^3)$, and $z^{-2}E_2(z^3)$ — see the bottom plots in Fig. 26.24.

For the sake of comparison, let us write down the polyphase components of order 6 (which coincides with the length $N = 6$ of the example we started with in (26.53)). In this case we write

$$H(z) \;=\; \sum_{k=0}^{5} z^{-k}E_k(z^6) \tag{26.62}$$

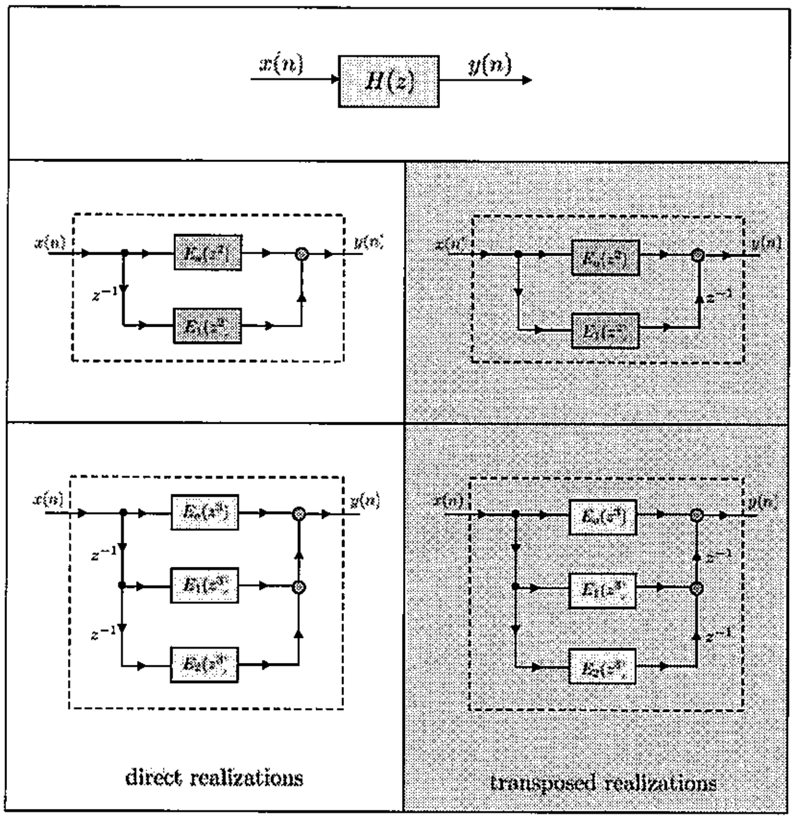

FIGURE 26.24 The left part of the figure shows direct realizations of an FIR filter, $H(z)$, as the parallel cascade of second-order (middle) and third-order (bottom) polyphase structures. The right part of the figure shows transposed realizations of the same filter as the parallel cascade of second-order (middle) and third-order (bottom) polyphase structures. In the transposed realizations, the delay elements appear on the right-hand side of the implementations.

where

$$E_o(z) = h(0) \tag{26.63a}$$
$$E_1(z) = h(1) \tag{26.63b}$$
$$E_2(z) = h(2) \tag{26.63c}$$
$$E_3(z) = h(3) \tag{26.63d}$$
$$E_4(z) = h(4) \tag{26.63e}$$
$$E_5(z) = h(5) \tag{26.63f}$$

It is seen that the polyphase components now trivialize to the samples of the impulse response sequence. The resulting parallel structure coincides with the direct realization we encountered earlier in Fig. 26.22 for the filter $H(z)$.

More generally, an $L-$th order polyphase decomposition of an FIR filter $H(z)$ takes the form:

$$\boxed{H(z) \triangleq \sum_{k=0}^{L-1} z^{-k} E_k(z^L)} \tag{26.64}$$

where

$$E_k(z) \triangleq \sum_{n=0}^{\lfloor N/L \rfloor - 1} h(nL+k)z^{-n} \tag{26.65}$$

and the notation $\lfloor x \rfloor$ denotes the integer part of x.

26.3.3 Polyphase Structures for Decimation and Interpolation

The polyphase decomposition of FIR filters can be used to motivate more general structures for decimation and interpolation than those encountered earlier in Figs. 26.22 and 26.23. To do so, we establish two additional equivalence results, also known as the Noble identities. These identities enable us to exchange filters with downsamplers and upsamplers. When the identities are combined with the earlier equivalence results from Fig. 26.21, we will end up with a set of useful tools that allows us to move upsamplers and downsamplers within multirate systems to more desirable locations.

Two Additional Equivalence Relations: Noble Identities

Thus, consider the equivalences indicated in Fig. 26.25. The top row considers the cascade of an upsampler and a generic polyphase component of the form $E(z^L)$. The equivalence result tells us that the order of the upsampler and the polyphase component can be reversed with $E(z^L)$ replaced by $E(z)$. To establish the validity of this result, we call upon the $z-$transformation property (9.145) to note that, for the block on the left-hand side in the top row we have:

$$\begin{aligned} W_1(z) &= X(z^L) & (26.66)\\ Y(z) &= W_1(z)E(z^L) = X(z^L)E(z^L) & (26.67)\end{aligned}$$

while for the block on the right-hand side in the top row we have

$$\begin{aligned} W_2(z) &= X(z)E(z) & (26.68)\\ Y(z) &= W_2(z^L) = X(z^L)E(z^L) & (26.69)\end{aligned}$$

We conclude that both blocks have the same input-output relation. This calculation establishes the equivalence result in the top row of Fig. 26.25.

Interpretation for interpolation relation. Let $e(n)$ denote the impulse response sequence of the filter $E(z)$. Then, it is straightforward to conclude that the impulse response sequence of $E(z^L)$ is obtained from $e(n)$ by upsampling it by a factor of L, i.e., by inserting $L-1$ zeros between successive samples of $e(n)$. For instance, consider the example

$$E(z) = e(0) + e(1)z^{-1} + e(2)z^{-2} \tag{26.70}$$

with impulse response sequence $\{e(0), e(1), e(2)\}$ then

$$E(z^3) = e(0) + e(1)z^{-3} + e(2)z^{-6} \tag{26.71}$$

with impulse response sequence $\{e(0), 0, 0, e(1), 0, 0, e(2)\}$. Now refer to the first row of Fig. 26.25. In the block on the right, we are convolving $x(n)$ with $e(n)$ and then upsampling the result. In the block on the left, we are first upsampling both $x(n)$ and $e(n)$ and

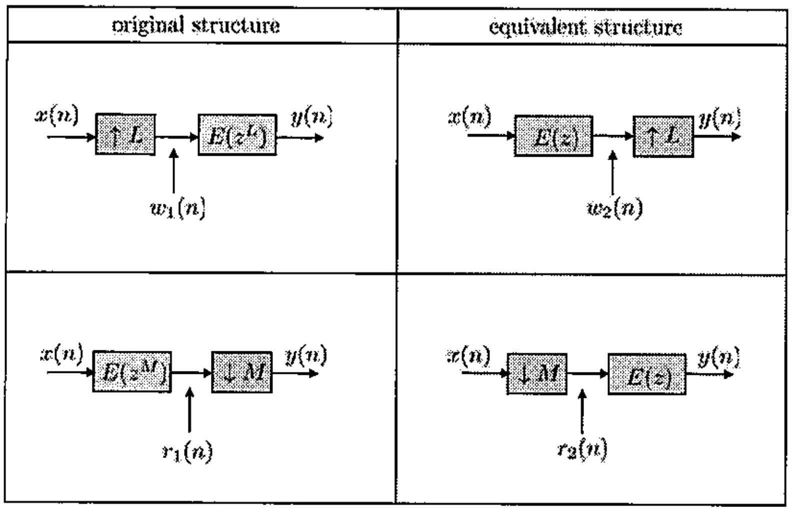

FIGURE 26.25 Noble identities showing the cascade equivalences for interpolation (top) and decimation (bottom) structures.

then convolving them. Thus, we find that *upsampling the convolution of two sequences is equivalent to convolving the upsampled versions of the sequences.*

◇

Let us now consider the equivalence result pertaining to decimation. The bottom row in Fig. 26.25 considers the cascade of a generic polyphase component of the form $E(z^M)$ and a downsampler. The equivalence result tells us that the order of the polyphase component and the downsampler can be switched with $E(z^M)$ replaced by $E(z)$. To establish the validity of this result, we call upon the z−transformation property (9.156) to note that, for the block on the right-hand side in the bottom row we have

$$R_2(z) = \frac{1}{M}\sum_{k=0}^{M-1} X\left(W_M^k z^{1/M}\right), \quad W_M = e^{-j2\pi/M} \tag{26.72}$$

$$Y(z) = R_2(z)E(z) = E(z)\cdot\frac{1}{M}\sum_{k=0}^{M-1} X\left(W_M^k z^{1/M}\right) \tag{26.73}$$

while for the block on the left-hand side in the bottom row we have

$$R_1(z) = X(z)E(z^M) \tag{26.74}$$

$$\begin{aligned} Y(z) &= \frac{1}{M}\sum_{k=0}^{M-1} R_1\left(W_M^k z^{1/M}\right) \\ &= \frac{1}{M}\sum_{k=0}^{M-1} X(z)E(z^M)\Big|_{z=W_M^k z^{1/M}} \\ &= E(z)\cdot\frac{1}{M}\sum_{k=0}^{M-1} X\left(W_M^k z^{1/M}\right) \end{aligned} \tag{26.75}$$

We conclude that both blocks have the same input-output relation. This calculation establishes the equivalence result in the bottom row of the figure.

Interpretation for decimation relation. The impulse response sequence of the polyphase component $E(z^M)$, denoted by $e_u(n)$, is again the upsampled version of the impulse response sequence of $E(z)$, denoted by $e(n)$. In the block on the left-hand side of the bottom row in Fig. 26.25, we are convolving $x(n)$ with $e_u(n)$ and then downsampling the result. In the block on the right-hand side, we are replacing both $x(n)$ and $e_u(n)$ by their downsampled versions and convolving them. Thus, we find that *downsampling the convolution of a sequence with a polyphase component is equivalent to convolving the downsampled versions of the sequences.*

◇

Decimation

Now consider the decimation structure shown in the top part of Fig. 26.26 where the FIR filter $H(z)$ is followed by a downsampler of factor M. On the left-hand side in the bottom part of the figure, the filter $H(z)$ is replaced by a polyphase realization of order M. Again, as remarked earlier, since the downsampler keeps only one of every M samples generated by $H(z)$, it is obviously wasteful to generate the samples that are going to be discarded by the downsampler. A more efficient implementation can be obtained by appealing to the equivalences of Figs. 26.21 and 26.25. Doing so results in the polyphase structure shown on the right in the bottom row of Fig. 26.26.

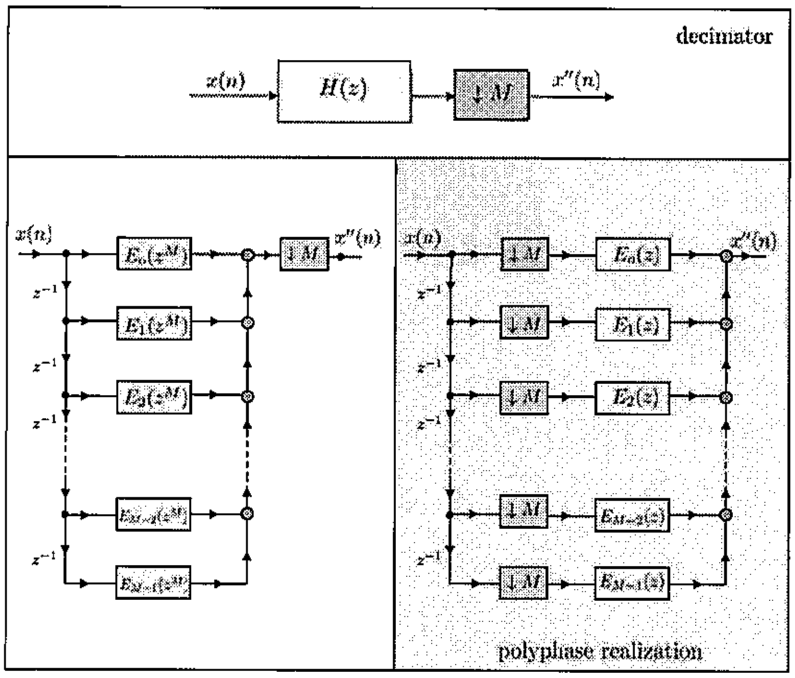

FIGURE 26.26 Applying the equivalence results of Figs. 26.21 and 26.25 to the realization on the left results in the efficient polyphase realization shown on the right for decimation by a factor M.

Interpolation

Likewise, consider the interpolator structure shown in the top part of Fig. 26.27 where an upsampler with factor L is followed by the FIR filter $H(z)$. On the left-hand side in the bottom part of the figure, the filter $H(z)$ is replaced by a transposed polyphase realization of order L.

Again, as remarked earlier, since the upsampler inserts $L-1$ zeros between successive samples of $x(n)$, most multiplications that are performed within the filters $E_k(z)$ are wasted. A more efficient implementation can be obtained by appealing to the equivalences of Figs. 26.21 and 26.25. Doing so results in the polyphase structure shown on the right in the bottom row of Fig. 26.27, which we refer to as a type-I implementation. There is an equivalent implementation, known as type-II, which simply amounts to renaming the filters $E_k(z)$ as explained next.

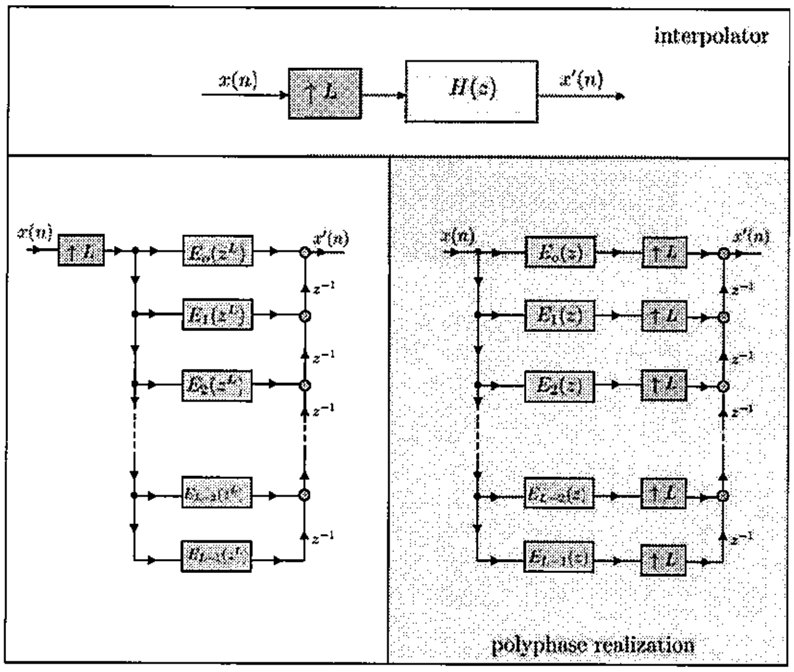

FIGURE 26.27 Applying the equivalence results of Figs. 26.21 and 26.25 to the realization on the left results in the efficient type-I polyphase realization shown on the right for interpolation by a factor L.

Assume we rename the polyphase components as follows:

$$\boxed{E_k(z) \;=\; R_{L-1-k}(z), \quad k = 0, 1, \ldots, L-1} \tag{26.76}$$

so that the relation between the filters $E_k(z)$ and $R_k(z)$ is

$$\left\{\begin{array}{lcl} E_o(z) & = & R_{L-1}(z) \\ E_1(z) & = & R_{L-2}(z) \\ & \vdots & \\ E_{L-1}(z) & = & R_o(z) \end{array}\right. \tag{26.77}$$

Note in particular that while $E_k(z)$ runs from $E_o(z)$ to $E_{L-1}(z)$, the indices of the filters $R_k(z)$ run from $R_{L-1}(z)$ down to $R_o(z)$. With this new notation, we can rewrite the $L-$th order polyphase composition (26.64) of $H(z)$ in the equivalent form:

$$\boxed{H(z) \triangleq \sum_{k=0}^{L-1} z^{-(L-1-k)} R_k(z^L)} \tag{26.78}$$

where

$$R_k(z) = E_{L-1-k}(z) \tag{26.79}$$

$$E_k(z) = \sum_{n=0}^{\lfloor N/L \rfloor -1} h(nL+k)z^{-n} \tag{26.80}$$

Using the polyphase components $R_k(z)$, it is straightforward to redraw the type-I implementation of Fig. 26.27 in the form shown in Fig. 26.28. Note that the signal $x'(n)$ is now generated at the bottom right corner.

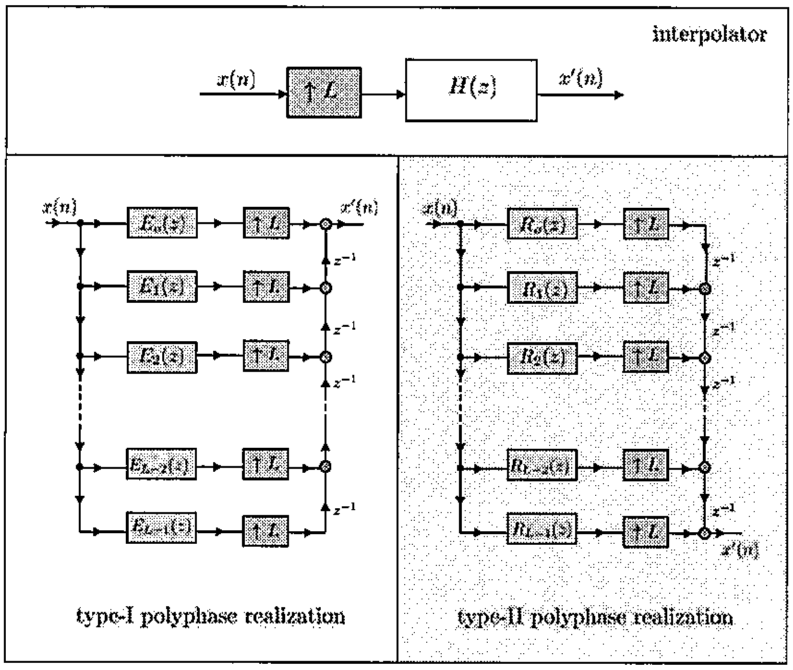

FIGURE 26.28 Type-I polyphase realization of an interpolator of order L (left) and type-II realization of the same interpolation in terms of the polyphase components $R_k(z)$ as opposed to the $E_k(z)$.

26.4 NYQUIST FILTERS

We still need to explain how to select the low-pass filter, $H(z)$, and/or its polyphase components, which are needed in the decimator and interpolator implementations. We shall discuss later in Chapter 29 several techniques for designing FIR filters. In this section, we motivate one useful form for $H(z)$ in the interpolator structure; the form belongs to the class of Nyquist filters, which are defined below. We shall encounter this class of filters again in Chapter 27 when we study complimentary frequency responses.

Thus, refer to the top part of Fig. 26.23, which shows the generic structure of an interpolator, consisting of an upsampler followed by a low-pass FIR filter. The role of the upsampler is to generate the sequence $y(n)$ by inserting $L-1$ zeros between successive samples of $x(n)$. As such, the samples of $y(n)$ at values of n that correspond to multiples of L agree with those of $x(n)$, i.e.,

$$y(nL) = x(n), \quad n = 0, 1, 2, \ldots \tag{26.81}$$

and the samples of $y(n)$ are zero at all other time instants n. The low-pass filter, $H(z)$, operates on the zero and non-zero samples of $y(n)$ and generates the interpolated sequence, $x'(n)$. It would be useful if the filter $H(z)$ had the property that it keeps the samples $y(nL)$ untouched, and only interpolates to fill in the zero samples of $y(n)$.

The class of Nyquist filters that we introduce next satisfies this property and, therefore, it would be desirable to implement $H(z)$ as a Nyquist filter. Although these filters leave the samples $y(nL)$ untouched, they nevertheless introduce some delay. This is because, as anticipated in Sec. 16.1, discrete-time filters have group delays and, therefore, it is natural to expect that the samples of the output sequence $x'(n)$, including any untouched samples, will have some delay in relation to the samples of the input sequence, $y(n)$. This situation is illustrated in Fig. 26.29.

26.4.1 Sample Preservation Property

The impulse response sequence of a causal FIR Nyquist filter should therefore satisfy the following conditions:

$$\boxed{h(r) = \alpha \quad \text{and} \quad h(r+\ell L) = 0} \tag{26.82}$$

for some constant α and where $r \geq 0$ and ℓ are integers. That is, $h(n)$ assumes the value α at $n = r$ and is zero at all time instants n that are multiples of L away from r (the integer ℓ can assume positive or negative values). The samples of $h(n)$ at all other values of n are not restricted (they may assume zero or nonzero values). In view of property (26.82), we find from the convolution sum expression

$$x'(n) \;=\; \sum_{k=0}^{n} y(k)h(n-k) \tag{26.83}$$

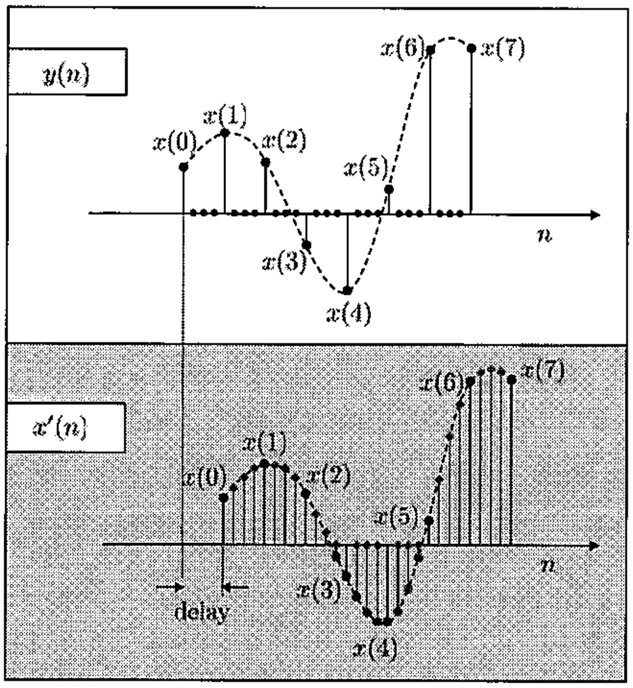

FIGURE 26.29 Nyquist filters have the property that they leave the nonzero samples of $y(n)$ untouched in $x'(n)$ apart from some delay.

that the output samples $x'(n)$ will be given by:

$$x'(r) = h(r)\,y(0) = \alpha\,x(0) \tag{26.84a}$$

$$x'(r+L) = h(r)\,y(L) = \alpha\,x(1) \tag{26.84b}$$

$$x'(r+2L) = h(r)\,y(2L) = \alpha\,x(2) \tag{26.84c}$$

$$x'(r+3L) = h(r)\,y(3L) = \alpha\,x(3) \tag{26.84d}$$

$$\vdots \qquad \vdots$$

since the samples of $y(k)$ are zero except when k is a multiple of L in which case $y(nL) = x(n)$. We conclude from the above relations that, as desired, the samples of $x(n)$ are left unchanged, except for scaling by α. Specifically, it holds that

$$\boxed{x'(r+nL) = \alpha\,y(nL) = \alpha\,x(n), \quad n \geq 0} \tag{26.85}$$

Although Nyquist filters can be IIR as well, it is sufficient for our purposes to focus on FIR choices. In this case, we observe from (26.82) that many of the coefficients of the filter will be zero, namely, all coefficients at time indices $r + nL$ with the exception of $h(r)$. As such, the implementation of Nyquist filters requires a reduced number of multiplications when compared with the implementation of regular FIR filters.

26.4.2 Polyphase Characterization

The polyphase structure of Nyquist filters has useful properties. To see this, we introduce the $L-$th order polyphase decomposition of a Nyquist filter, $H(z)$, say, as

$$H(z) = \sum_{k=0}^{L-1} z^{-k} E_k(z^L) \tag{26.86}$$

In general, the value of r used in the definition (26.82) can assume values that may be smaller than, equal to, or even larger than L. Thus, let

$$\boxed{r' \triangleq r \bmod L} \tag{26.87}$$

so that r' lies in the range

$$0 \leq r' \leq L-1 \tag{26.88}$$

Then, it is straightforward to verify from property (26.82), and from the definition (26.64) of polyphase components, that the polyphase component of $H(z)$ of index r' is given by:

$$\boxed{E_{r'}(z^L) \;=\; \alpha\, z^{-(r-r')}} \tag{26.89}$$

Example 26.9 (Polyphase representation)

Let us illustrate property (26.89) by means of an example. Consider a Nyquist impulse response sequence $h(n)$ with samples as indicated below:

$n:$	0	1	2	3	4	5	6	7	8	9	10	11	12
$h(n):$	0	$h(1)$	0	$h(3)$	$\boxed{\alpha}$	$h(5)$	0	$h(7)$	0	$h(9)$	0	$h(11)$	0

This situation corresponds to the choices $r = 4$ and $L = 2$ so that $r' = 0$. The second-order polyphase decomposition of $H(z)$ is given by

$$\left\{\begin{array}{lcl} E_o(z^2) & = & \alpha z^{-4} \\ E_1(z^2) & = & h(1) + h(3)z^{-2} + h(5)z^{-4} + h(7)z^{-6} + h(9)z^{-8} + h(11)z^{-10} \\ H(z) & = & E_o(z^2) + z^{-1}E_1(z^2) \end{array}\right. \tag{26.90}$$

These results agree with (26.89). Consider now the alternative example

$n:$	0	1	2	3	4	5	6	7	8	9	10	11	12
$h(n):$	$h(0)$	$h(1)$	0	$h(3)$	$h(4)$	$\boxed{\alpha}$	$h(6)$	$h(7)$	0	$h(9)$	$h(10)$	0	$h(12)$

(26.91)

This situation corresponds to the choices $r = 5$ and $L = 3$ so that $r' = 2$. The third-order polyphase decomposition of $H(z)$ is given by

$$\left\{\begin{array}{lcl} E_o(z^3) & = & h(0) + h(3)z^{-3} + h(6)z^{-6} + h(9)z^{-9} + h(12)z^{-12} \\ E_1(z^3) & = & h(1) + h(4)z^{-3} + h(7)z^{-6} + h(10)z^{-9} \\ E_2(z^3) & = & \alpha z^{-3} \\ H(z) & = & E_o(z^3) + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \end{array}\right. \tag{26.92}$$

These results again agree with (26.89).

◇

Returning to property (26.89), note that in the special case when $r' = 0$, we get

$$E_o(z^L) = \alpha z^{-r} \tag{26.93}$$

The condition $r' = 0$ occurs when r is a multiple of L, say, $r = \ell_o L$ for some integer ℓ_o. In this case, condition (26.82) becomes

$$h(\ell_o L) = \alpha \quad \text{and} \quad h(\ell L) = 0 \ \text{ for all } \ell \neq \ell_o \tag{26.94}$$

and this result translates into a useful frequency normalization property that is satisfied by the Nyquist filter, namely, shifted versions of its frequency response add up to a pure delay. Specifically, let W_L denote the $L-$th root of unity, i.e.,

$$\boxed{W_L \triangleq e^{-j\frac{2\pi}{L}}} \tag{26.95}$$

Then, when $r' = 0$, the transfer function of a Nyquist filter satisfies the property:

$$\boxed{\sum_{k=0}^{L-1} H\left(zW_L^k\right) \ = \ L\alpha z^{-r}} \tag{26.96}$$

Proof: Let us establish the result for $L = 3$ for convenience of presentation. The same argument applies to other values of L. Thus, let $H(z)$ be the transfer function of an FIR Nyquist filter and introduce its third-order polyphase decomposition:

$$H(z) = \alpha z^{-r} + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \tag{26.97}$$

where we used the already known fact from (26.93) that $E_o(z^3) = \alpha z^{-r}$. Note also that since, by assumption, $r' = 0$ then r is a multiple of $L = 3$. It follows that

$$\begin{aligned} H(zW_3) &= \alpha z^{-r}W_3^{-r} + z^{-1}W_3^{-1}E_1(z^3W_3^3) + z^{-2}W_3^{-2}E_2(z^3W_3^3) \\ &= \alpha z^{-r} + z^{-1}W_3^{-1}E_1(z^3) + z^{-2}W_3^{-2}E_2(z^3), \quad \text{since } W_3^3 = 1 \end{aligned} \tag{26.98}$$

and

$$\begin{aligned} H(zW_3^2) &= \alpha z^{-r}W_3^{-2r} + z^{-1}W_3^{-2}E_1(z^3W_3^6) + z^{-2}W_3^{-4}E_2(z^3W_3^6) \\ &= \alpha z^{-r} + z^{-1}W_3^{-2}E_1(z^3) + z^{-2}W_3^{-4}E_2(z^3) \end{aligned} \tag{26.99}$$

Now note that

$$1 + W_3^{-1} + W_3^{-2} \ = \ 0 \tag{26.100}$$
$$1 + W_3^{-2} + W_3^{-4} \ = \ 0 \tag{26.101}$$

and, hence,

$$H(z) + H(zW_3) + H(zW_3^2) \ = \ 3\alpha z^{-r} \tag{26.102}$$

as claimed.

◇

Replacing z by $e^{j\omega}$, relation (26.95) translates into the following property in the frequency domain when $r' = 0$:

$$\boxed{\sum_{k=0}^{L-1} H\left(e^{j\left(\omega - \frac{2\pi k}{L}\right)}\right) = L\alpha e^{-j\omega r}} \tag{26.103}$$

This result indicates that if the frequency response of a Nyquist filter is shifted by multiples of $2\pi/L$ radians/sample, and all responses are added together, then the result is a flat response of magnitude $L\alpha$ and phase $-\omega r$. Figure 26.30 illustrates the frequency normalization property by showing images of $H(e^{j\omega})$ when $r = 0$.

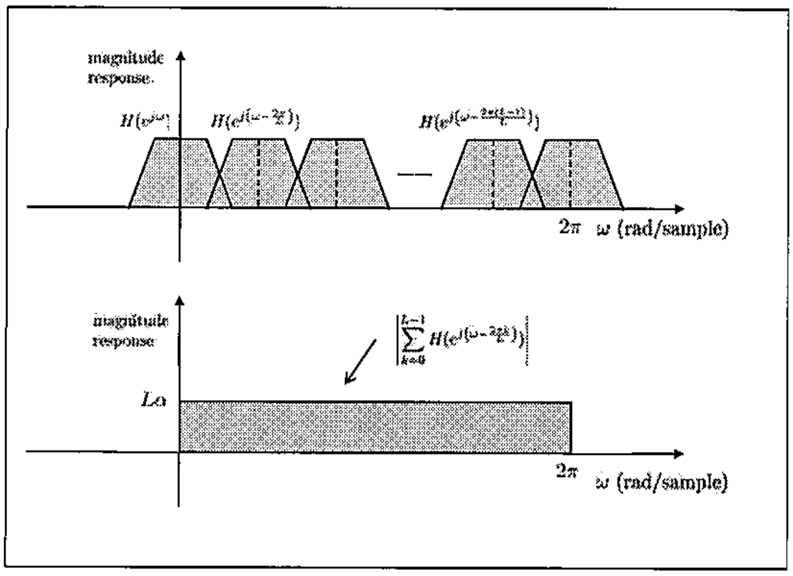
magnitude response
$H(e^{j\omega})$
$H(e^{j(\omega-\frac{2\pi}{L})})$
$H(e^{j(\omega-\frac{2\pi(L-1)}{L})})$
2π
ω (rad/sample)
magnitude response
$\left|\sum_{k=0}^{L-1} H(e^{j(\omega-\frac{2\pi k}{L})})\right|$
$L\alpha$
2π
ω (rad/sample)

FIGURE 26.30 Shifting the frequency response of the Nyquist filter $H(z)$ (in the case $r = 0$) to the locations $2\pi k/L$ radians/sample, and adding up all responses, results in a flat response of value $L\alpha$ over the range $\omega \in [0, 2\pi]$.

26.4.3 Design Procedure

We now present one method to design linear-phase causal FIR Nyquist filters; we shall discuss this method and other methods in greater detail later in Chapter 29 when we consider techniques for designing FIR filters. The method we discuss below is patterned on the window design method from future Sec. 29.2.

Our objective is to design a causal low-pass filter, $H(z)$, whose impulse response sequence is real-valued and symmetric with duration N. We assume, for reasons to become clear soon, that N has the form:

$$\boxed{N = 2r + 1} \tag{26.104}$$

for some positive integer r. That is, N is odd and its value is defined in terms of r. Since N is odd and the impulse response sequence is symmetric, we see that we are seeking a type-I FIR filter (cf. Table 16.1).

We start from the infinitely-long and non-causal impulse response sequence:

$$h_d(n) \;=\; \frac{\sin\omega_c(n-n_o)}{\pi(n-n_o)} \tag{26.105}$$

with some delay equal to n_o and for some angular frequency ω_c radians/sample. If we recall from Table 13.1 the following DTFT pair:

$$x(n) = \frac{\sin\omega_c n}{\pi n} \quad\longleftrightarrow\quad X(e^{j\omega}) = \begin{cases} 1, & |\omega| \le \omega_c \\ 0, & \text{otherwise} \end{cases} \tag{26.106}$$

then it becomes clear that the frequency response corresponding to the impulse response sequence $h_d(n)$ has low-pass frequency characteristics since its DTFT is given by

$$H_d(e^{j\omega}) \;=\; \begin{cases} e^{-j\omega n_o}, & |\omega| \le \omega_c \\ 0, & \text{otherwise} \end{cases} \tag{26.107}$$

with cutoff frequency at ω_c radians/sample. In other words, the magnitude response is equal to one in the range $\omega \in [-\omega_c, \omega_c]$ and is zero outside this range. Moreover, the phase response is linear over the same range $\omega \in [-\omega_c, \omega_c]$.

Observe that $h_d(n)$ is not a causal sequence and, therefore, is physically unrealizable. Moreover, $h_d(n)$ has infinite duration. One way to approximate $h_d(n)$ by a causal FIR filter is to truncate $h_d(n)$ to a finite duration sequence by keeping some of its samples and ignoring the rest. Therefore, we set

$$h(n) \;=\; h_d(n), \quad 0 \le n \le N-1 \tag{26.108}$$

where N is odd. In order to continue to ensure linear phase characteristics, we impose the symmetry condition

$$h(n) \;=\; h(N-1-n), \quad 0 \le n \le N-1 \tag{26.109}$$

This can be guaranteed if we select n_o to satisfy:

$$n_o = \frac{N-1}{2} \;=\; r \tag{26.110}$$

We see now that condition (26.104), which was imposed on N, serves to ensure that the resulting value for n_o is an integer (and equal to r). Then,

$$h(n) \;=\; \frac{\sin\omega_c(n-r)}{\pi(n-r)}, \quad 0 \le n \le N-1 \tag{26.111}$$

We further select the cut-off frequency ω_c to be

$$\omega_c = \frac{\pi}{L} \quad \text{(radians/sample)} \tag{26.112}$$

so that the FIR filter that we start from is the following:

$$\boxed{h(n) \;=\; \frac{\sin\left(\frac{\pi(n-r)}{L}\right)}{\pi(n-r)}, \;, \quad 0 \le n \le N-1} \tag{26.113}$$

Observe that this filter satisfies the properties

$$h(n) = \begin{cases} 1/L, & n = r \\ 0, & n = r + \ell L, \text{ for all } \ell \neq 0 \end{cases} \tag{26.114}$$

and, therefore, $h(n)$ satisfies the Nyquist property (26.82) with $\alpha = 1/L$. Note further that if we multiply $h(n)$ by any of the windows discussed in Sec. 29.2, say,

$$\boxed{h_w(n) = h(n)w(n), \quad 0 \leq n \leq N-1} \tag{26.115}$$

we arrive at other causal FIR filters, $h_w(n)$, that also satisfy the Nyquist property (26.82) albeit with $\alpha = w(0)/L$. Figure 26.31 illustrates the result of this design procedure using $\omega_c = \pi/4$, $L = 4$, $N = 51$, $r = 25$, and a rectangular window. Observe how $h(n)$ is symmetric around $n = 25$ (which is the value of r). Observe also that all samples of $h(n)$ that are at multiples of four steps from $n = 25$ to the left and to the right are equal to zero.

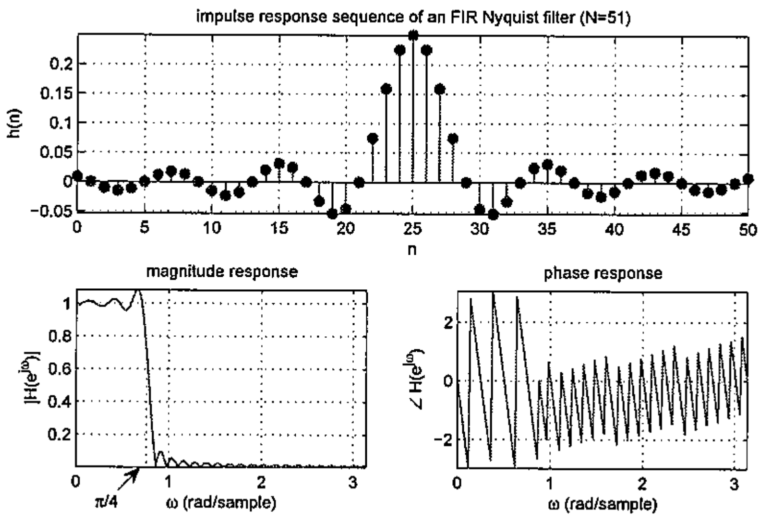

FIGURE 26.31 Impulse response sequence of the Nqyuist filter (26.113) of duration $N = 51$ samples for $L = 4$(top). The resulting magnitude and phase responses over the interval $\omega \in [0, \pi]$ are shown in the bottom plots. The cutoff frequency of the low-pass filter is around $\omega_c = \pi/4$ radians/sample.

Example 26.10 (Nyquist filter for $L = 2$)

Let us design a second Nyquist filter of duration $N = 21$ samples for an interpolator with $L = 2$. This case corresponds to assuming $r = 10$ (since we want N to satisfy $N = 2r + 1$). From (26.113), the low-pass filter is

$$h(n) = \frac{\sin\left(\frac{\pi(n-10)}{2}\right)}{\pi(n-10)}, \quad 0 \leq n \leq 20 \tag{26.116}$$

We now select the Hamming window from Table 29.1 and set the filter impulse response sequence to

$$h_w(n) = \frac{\sin\left(\frac{\pi(n-10)}{2}\right)}{\pi(n-10)} \cdot \left[0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right)\right], \quad 0 \le n \le 20 \tag{26.117}$$

Figure 26.32 plots the resulting impulse response sequence of the Nyquist filter. Observe how the sample at $n = 10$ is nonzero and equal to $h_w(10) = 0.5$, while the samples of $h_w(n)$ are zero at $n = 0, 2, 4, 6, 8, 12, 14, 16, 18, 20$.

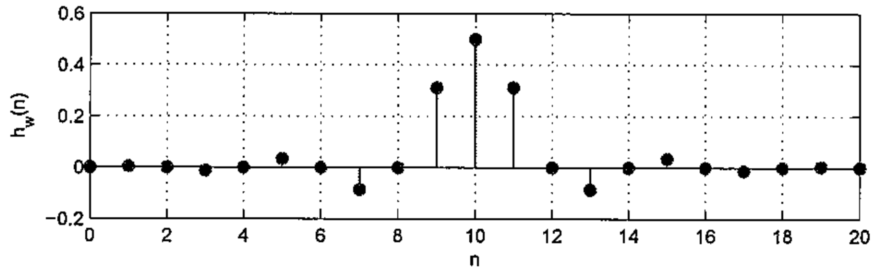

FIGURE 26.32 A Nyquist filter of duration $N = 21$ for the case $L = 2$.

Observe further than since in this case $r = 10$, we have

$$\begin{aligned} r' &= r \bmod L \\ &= 10 \bmod 2 \\ &= 0 \end{aligned} \tag{26.118}$$

so that from (26.89),

$$E_o(z^2) = h(r)z^{-r} = 0.5z^{-10} \tag{26.119}$$

and, according to (26.103), the frequency response in this case satisfies

$$H_w(e^{j\omega}) + H_w(e^{j(\omega-\pi)}) = e^{-j10\omega} \tag{26.120}$$

Figure 26.33 plots the magnitude responses of $H_w(e^{j\omega})$ and its shifted version, $H_w(e^{j(\omega-\pi)})$, over the interval $\omega = [0, \pi]$.

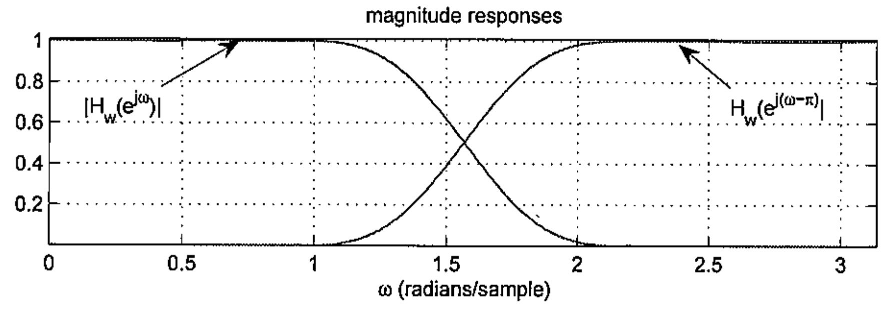

FIGURE 26.33 Magnitude plots of $H_w(e^{j\omega})$ and $H_w(e^{j(\omega-\pi)})$ for the case $L = 2$ in Example 26.10.

◇

26.4.4 Half-Band Filters

We continue to denote the impulse response sequence of the Nyquist filter by $h(n)$ whether it is windowed or not (i.e., whether we choose it as $h(n)$ or $h_w(n)$). Now, when $L = 2$ and $h(n)$ is real-valued, the Nyquist filters are also called half-band filters (the reason for the name is explained further ahead). For example, the filter designed in Example 26.10 is a half-band filter with $\alpha = 1/2$. For $L = 2$, property (26.82) becomes

$$\boxed{h(r) = \alpha \quad \text{and} \quad h(r + 2\ell) = 0} \tag{26.121}$$

for nonzero integers ℓ, so that

$$\boxed{x'(r + 2n) = \alpha\, x(n)} \tag{26.122}$$

In terms of the second-order polyphase decomposition of $H(z)$, we have

$$H(z) = E_o(z^2) \ + \ z^{-1}E_1(z^2) \tag{26.123}$$

where

$$\boxed{\begin{cases} E_o(z^2) = \alpha\, z^{-r}, & \text{when } r \text{ is even} \\ E_o(z^2) = \alpha\, z^{-(r-1)}, & \text{when } r \text{ is odd} \end{cases}} \tag{26.124}$$

Moreover, the frequency normalization property (26.96) becomes

$$\boxed{H(z) \ + \ H(-z) \ = \ 2\alpha z^{-r}} \tag{26.125}$$

If we write $-z \ = \ e^{-j\pi}z$ and replace z by $e^{j\omega}$ we get that the frequency response of a half-band filter satisfies:

$$\boxed{H(e^{j\omega}) \ + \ H(e^{j(\omega-\pi)}) \ = \ 2\alpha e^{-j\omega r}} \tag{26.126}$$

From (26.125) we also get directly

$$H(e^{j\omega}) \ + \ H(-e^{j\omega}) \ = \ 2\alpha e^{-j\omega r} \tag{26.127}$$

However, from the properties of the DTFT (see Sec. 14.1 and Prob. 14.19), we know that when the impulse response sequence, $h(n)$, is real-valued, the frequency response satisfies the symmetry property

$$H(-e^{j\omega}) \ = \ H(e^{j(\pi-\omega)}) \tag{26.128}$$

so that we also have

$$\boxed{H(e^{j\omega}) \ + \ H(e^{j(\pi-\omega)}) \ = \ 2\alpha e^{-j\omega r}} \tag{26.129}$$

If we introduce the change of variables $\omega \ = \ \frac{\pi}{2} - \theta$, then this equality implies that the quantities

$$H\left(e^{j\left(\frac{\pi}{2}-\theta\right)}\right) \quad \text{and} \quad H\left(e^{j\left(\frac{\pi}{2}+\theta\right)}\right) \tag{26.130}$$

add up to a complex number whose magnitude is 2α for all θ. This situation was illustrated earlier in Fig. 26.33. We say that the filter $H(e^{j\omega})$ exhibits symmetry about the half-band frequency, $\omega = \pi/2$, and hence, the name, half-band filter.

26.5 APPLICATION: SUBBAND CODING

One important application of the decimation and interpolation concepts of this chapter arises in the context of speech and audio coding and image compression. It is well-known, for example, that the frequency content of speech is limited to about 4KHz. However, some frequency bands are perceptually more relevant than other bands. Subband coding is a quantization procedure that is based on the idea of assigning more bits to encode the information in the relevant bands and fewer bits to encode the information in the remaining bands.

More generically, consider a signal whose frequency content extends to B KHz (e.g., $B = 4$KHz). We divide the range $[0, B]$ into octave subbands as follows. First, the entire interval is divided in two equal parts. Subsequently, the first half $[0, \frac{B}{2}]$ is subdivided into two equal parts and finally the range $[0, \frac{B}{4}]$ is subdivided into two equal parts as well. In this way, we end up with four frequency intervals as illustrated in Fig. 26.34 for both cases of un-normalized frequencies (measured in Hz) and normalized frequencies ω (measured in radians/sample). The normalized frequency range assumes that the sampling frequency is at $F_s = 2B$ KHz so that $\omega = \pi$ radians/sample corresponds to the frequency B KHz. The mapping between the normalized and un-normalized values of frequencies is given by

$$\omega = 2\pi\left(\frac{f}{F_s}\right) \qquad (26.131)$$

where F_s denotes the sampling frequency and f denotes the un-normalized frequency (say, both measured in Hz).

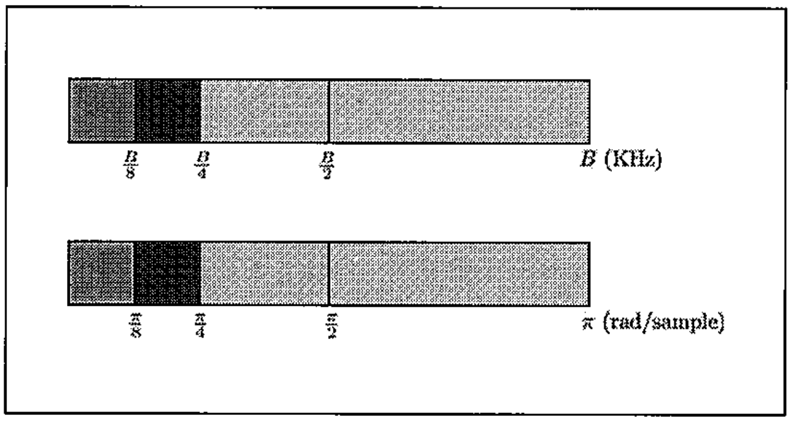

FIGURE 26.34 The range of frequencies $[0, B]$ KHz is subdivided into four octaves, designated by the four different colors. The first two regions have width $\pi/8$ each, the third region has width $\pi/4$, and the fourth region has width $\pi/2$ radians/sample. The lower plot shows the normalized frequencies assuming a sampling frequency of $F_s = 2B$ KHz. Relation (26.131) can be used to determine the normalized frequencies for other sampling rates.

Analysis Filter Bank Based on Ideal Filters

Now, let $x(n)$ denote the original signal, obtained at F_s samples per second. Let further $H_\ell(e^{j\omega})$ and $H_h(e^{j\omega})$ denote an ideal low-pass filter and an ideal high-pass filter, respectively, whose frequency responses are defined as follows:

$$H_\ell(e^{j\omega}) = \begin{cases} 1, & 0 \le \omega \le \frac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \quad \text{(ideal low-pass filter)} \tag{26.132}$$

$$H_h(e^{j\omega}) = \begin{cases} 0, & 0 \le \omega \le \frac{\pi}{2} \\ 1, & \text{otherwise} \end{cases} \quad \text{(ideal high-pass filter)} \tag{26.133}$$

Filtering $x(n)$ through $H_\ell(e^{j\omega})$ would extract the low-pass frequency components of the input sequence, while filtering it through $H_h(e^{j\omega})$ would extract its high frequency components. The sequences at the output of both of these filters will each have a bandwidth that is equal to $\pi/2$ radians/sampleand, therefore, each of the sequences can be decimated by a factor of 2 without the occurrence of aliasing. We let $x_\ell(n)$ and $x_h(n)$ denote the resulting sequences at the output of the decimators — see the first stage on the left side of Fig. 26.35. Figure 26.36 illustrates the frequency content of a sequence $x(n)$ and its decimated versions, $x_\ell(n)$ and $x_h(n)$.

Subband coding benefits from decompositions of this type where the input sequence is decomposed into various subband components, such that the bandwidth of each of these components is smaller than the bandwidth of the original signal. In this way, the subband components can be processed at lower rates and the processed components are then combined together to reconstruct the desired output sequence at the same rate as the original input sequence.

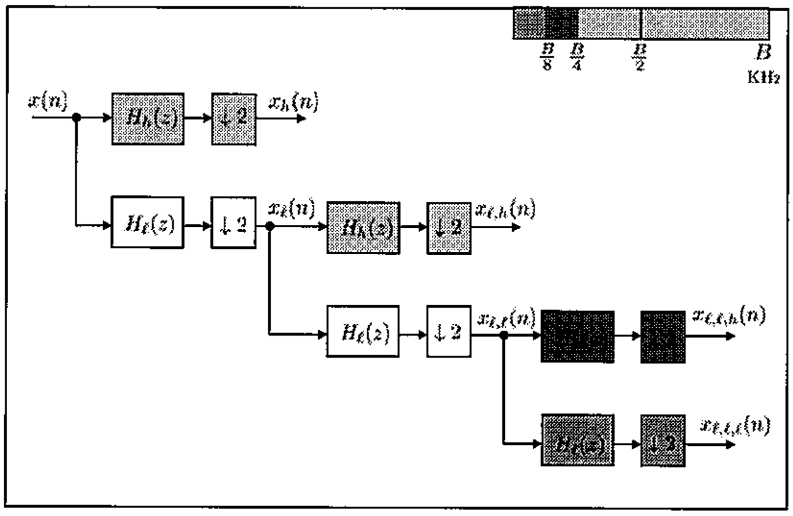

FIGURE 26.35 Decomposition of the input sequence $x(n)$ into four subband components through the use of three stages of low-pass/high-pass filters. The color codes indicate the range of frequencies that correspond to each of the component signals $\{x_h(n), x_{\ell,h}(n), x_{\ell,\ell,h}, x_{\ell,\ell,\ell}(n)\}$.

We can repeat the decomposition process by processing $x_\ell(n)$ through the same filters $H_\ell(e^{\omega})$ and $H_h(e^{\omega})$. This process splits the frequency content of $x_\ell(n)$ into two low-pass and high-pass components and decimates the corresponding output sequences. This step results in the sequences $x_{\ell,\ell}(n)$ and $x_{\ell,h}(n)$ shown in Fig. 26.36. We can apply the process one more time to the sequence $x_{\ell,\ell}(n)$ to obtain $x_{\ell,\ell,\ell}(n)$ and $x_{\ell,\ell,h}(n)$. In this way, we

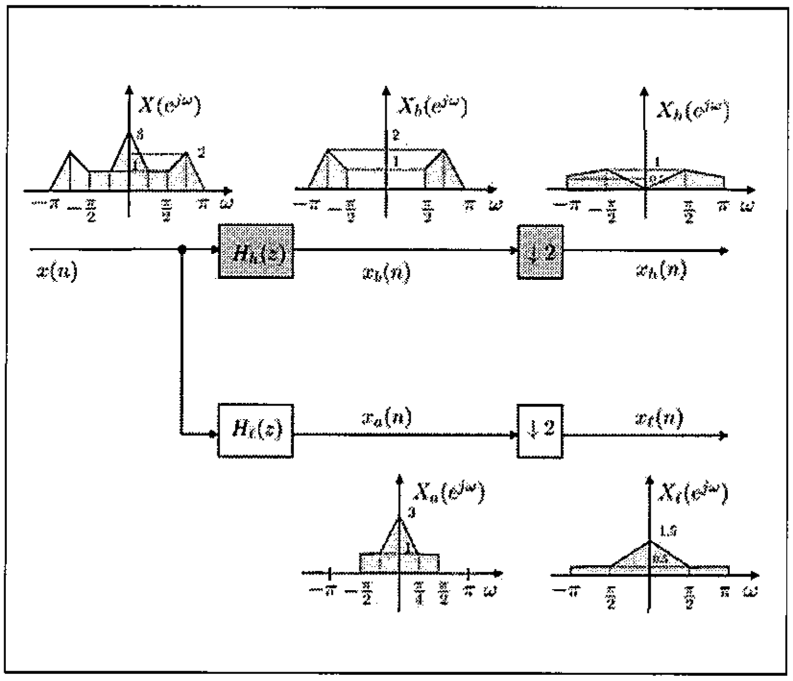

FIGURE 26.36 Illustration of the frequency content of the sequences involved in the first stage of the subband decomposition shown in Fig. 26.35.

end up with an *analysis* structure that consists of three successive stages. The structure generates the four sequences $\{x_{\ell,\ell,\ell}(n), x_{\ell,\ell,h}, x_{\ell,h}(n), x_h(n)\}$, which represent the content of the original speech signal within the octave subbands shown earlier in Fig. 26.34. The color codes used in Fig. 26.36 indicate the range of frequencies that correspond to each of the component signals $\{x_h(n), x_{\ell,h}(n), x_{\ell,\ell,h}, x_{\ell,\ell,\ell}(n)$.

For speech signals, the sequence $x_{\ell,\ell,\ell}(n)$ contains the most relevant spectral content. As such, these samples can be quantized by the encoder using a higher number of bits before being stored or transmitted over a channel. On the other hand, the samples of the sequences $\{x_{\ell,\ell,h}, x_{\ell,h}(n), x_h(n)\}$ are quantized by their respective encoders by fewer bits in accordance with the relevance of their spectral content. For example, one may use 16, 8, 4, and 2 bits, respectively, to quantize the samples of $\{x_{\ell,\ell,\ell}(n), x_{\ell,\ell,h}, x_{\ell,h}(n), x_h(n)\}$.

Synthesis Filter Bank Based on Ideal Filters

We now consider the reverse problem, which deals with recovering the original sequence, $x(n)$, from the samples of its subband components $\{x_{\ell,\ell,\ell}(n), x_{\ell,\ell,h}, x_{\ell,h}(n), x_h(n)\}$. It turns out that a delayed version of $x(n)$ can be recovered by reversing the operations performed by the analysis filter bank. One way to achieve this task is as follows.

Let $G_\ell(e^{j\omega})$ and $G_h(e^{j\omega})$ denote ideal low-pass and high-pass filters that are defined in a manner similar to $H_\ell(e^{j\omega})$ and $H_h(e^{j\omega})$, namely,

$$G_\ell(e^{j\omega}) = \begin{cases} 1, & 0 \le \omega \le \frac{\pi}{2} \\ 0, & \text{otherwise} \end{cases} \quad \text{(ideal low-pass filter)} \qquad (26.134)$$

$$G_h(e^{j\omega}) = \begin{cases} 0, & 0 \le \omega \le \frac{\pi}{2} \\ 1, & \text{otherwise} \end{cases} \quad \text{(ideal high-pass filter)} \qquad (26.135)$$

Then the operations shown in Fig. 26.37 upsample the subband sequences (which halves their frequency ranges) and extracts the necessary portions of their frequency content to reconstruct a delayed version of the original sequence $x(n)$. The structure is motivated as follows.

We explain in the sequel that reconstruction filters $\{G_l(z), G_h(z)\}$ can be determined such that the 2−branch structure of future Fig. 26.39 maps an arbitrary input signal $x(n)$ into a delayed version of it, namely, the resulting output signal will satisfy $y(n) = x(n-d)$, for some delay d — see, for example, expression (26.153) for which $d = 1$. The reason for the delay is the group delay that will exist between the input and output terminals of the combined analysis-synthesis filter bank in Fig. 26.39.

Now, if we examine the two lowest branches in Fig. 26.37, and recall that the signals $\{x_{\ell,\ell,h}(n), x_{\ell,\ell,\ell}(n)\}$ are generated from the analysis of signal $x_{\ell,\ell}(n)$ in Fig. 26.35, we conclude that the output of the synthesis operation resulting from these lowest branches in Fig. 26.37 will be $x_{\ell,\ell}(n-d)$. For this reason, the signal $x_{\ell,h}(n)$ is delayed by d units of time so that $\{x_{\ell,h}(n-d), x_{\ell,\ell}(n-d)\}$ now feed into the second synthesis stage in Fig. 26.37 to generate $x_\ell(n-2d)$. Finally, the signals $\{x_h(n-2d), x_\ell(n-2d)\}$ feed into the last synthesis stage to generate $x(n-3d)$, which is a delayed version of the original input sequence. The factors of 2 on the various branches in the *synthesis* filter bank are meant to undo the scaling by $1/2$ that was introduced during the decomposition operation by the decimators in Fig. 26.35; these factors can be incorporated into the definition of the transfer functions $G_\ell(z)$ and $G_h(z)$ (we actually take this alternate route below when we derive the quadrature mirror filter bank implementation corresponding to Fig. 26.39. In that case, the multiplications by 2 in Fig. 26.37 can be removed).

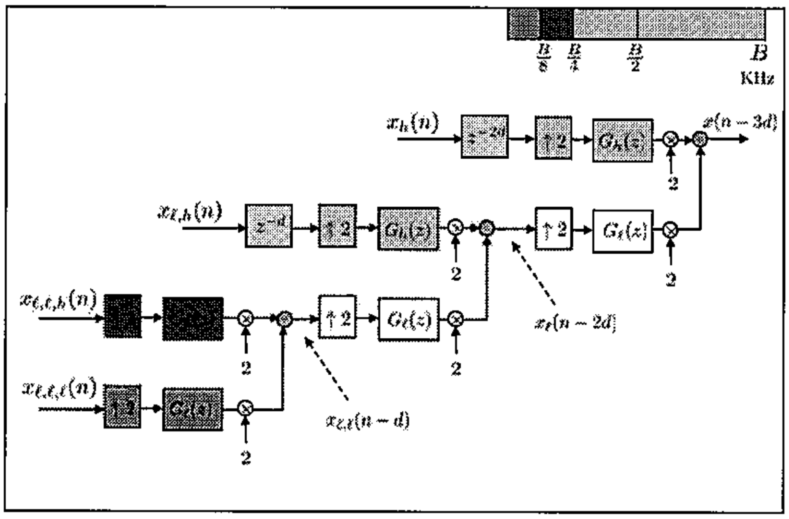

FIGURE 26.37 Synthesis of (a delayed version of) the input sequence from its subband components. The color codes indicate the range of frequencies that correspond to each of the component signals $\{x_h(n), x_{\ell,h}(n), x_{\ell,\ell,h}, x_{\ell,\ell,\ell}(n)\}$.

Using Quadrature Mirror Filters

The exposition so far assumed ideal properties for the filters $\{H_\ell(z), H_h(z), G_\ell(z), G_h(z)\}$. However, such ideal filters are un-realizable because they exhibit sharp transitions between their passband and stopband regions. In practice, the actual filters that are employed in the analysis and synthesis filter banks exhibit transition regions — we shall study the design of low-pass and high-pass filters in great detail later in Chapters 29–31. For our discussions in this section, it is sufficient to recognize that any low-pass and high-pass filters that will result from these design procedures will have transition regions. For example, the frequency response of a low-pass filter will not transition immediately from high gain to zero gain at the cut-off frequency; instead, there will be some fast but gradual transition between both gain levels. This behavior is illustrated schematically for both low-pass and high-pass filters in Fig. 26.38 — compare the ideal responses in the first row with the non-ideal responses in the second row. Note that, in each case, the transition region of the filter usually starts prior to the cut-off frequency and ends beyond it.

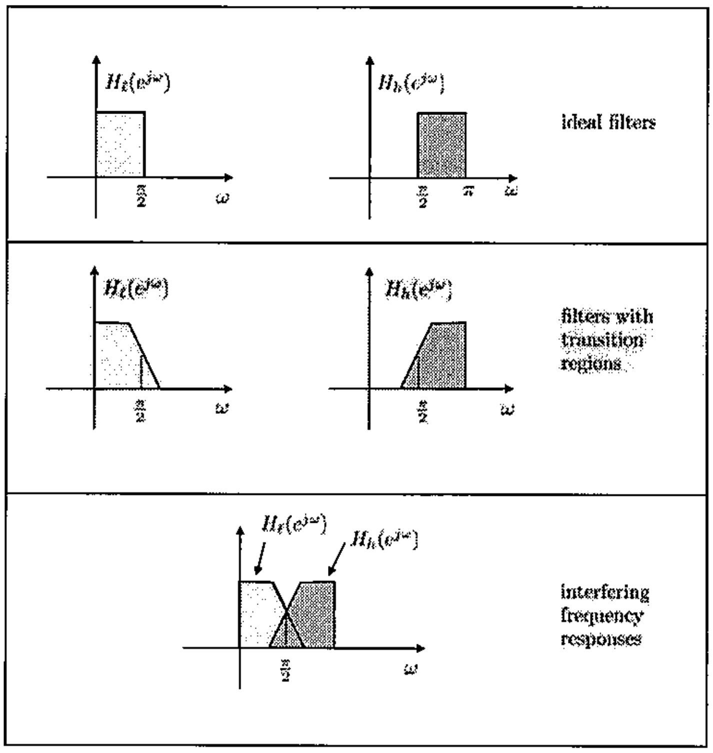

FIGURE 26.38 The top row shows the frequency characteristics of the ideal low-pass and high-pass filters over $\omega \in [0, \pi]$ radians/sample. The middle row shows the frequency characteristics of low-pass and high-pass filters with transition regions. For example, the frequency response of the low-pass filter starts dropping before the cut-off frequency $\omega_c = \pi/2$ and its transition region extends beyond ω_c.

Due to the non-ideal transition regions, it is not possible to separate the frequency contents of the various subband signals into disjoint regions anymore; some overlap between adjacent subbands will occur. The main question of interest is whether it is still possible to recover the original sequence, or a delayed version of it, when non-ideal filters are utilized. The answer is in the affirmative. We shall study in some detail perfect reconstruction filter banks later in Sec. 27.5. Here, we introduce the idea in the context of subband coding.

We illustrate the main idea by focusing on one analysis and synthesis stage. Thus, refer to Fig. 26.39, which shows one analysis stage with two branches and the corresponding synthesis stage. The input and output signals are denoted by $x(n)$ and $y(n)$, respectively. The signals at the output of the decimators are denoted by $x_\ell(n)$ and $x_h(n)$. We would like to determine conditions on the analysis and synthesis filters, $\{H_\ell(z), H_h(z)\}$

and $\{G_\ell(z), G_h(z)\}$, in order to ensure that the transfer function from $x(n)$ to $y(n)$ acts as a pure delay, say, as z^{-d} for some integer d.

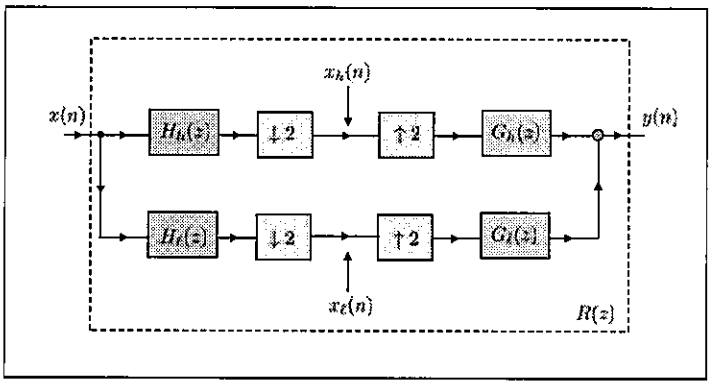

FIGURE 26.39 A 2−branch filter bank mapping an arbitrary signal $x(n)$ into another signal $y(n)$. The analysis and synthesis filters $\{H_\ell(z), H_h(z), G_\ell(z), G_h(z)\}$ can be designed to ensure that the mapping from $x(n)$ to $y(n)$ is a pure delay.

Using the z−transform representation, we have

$$\begin{aligned} X_\ell(z) &= \frac{1}{2}H_\ell(z^{1/2})X(z^{1/2}) + \frac{1}{2}H_\ell(-z^{1/2})X(-z^{1/2}) && (26.136)\\ X_h(z) &= \frac{1}{2}H_h(z^{1/2})X(z^{1/2}) + \frac{1}{2}H_h(-z^{1/2})X(-z^{1/2}) && (26.137)\\ Y(z) &= X_\ell(z^2)G_\ell(z) + X_h(z^2)G_h(z) && (26.138) \end{aligned}$$

so that

$$\begin{aligned} Y(z) &= \frac{1}{2}\left[H_\ell(z)G_\ell(z) + H_h(z)G_h(z)\right]X(z) + \\ & \frac{1}{2}\left[H_\ell(-z)G_\ell(z) + H_h(-z)G_h(z)\right]X(-z) && (26.139) \end{aligned}$$

The expression for $Y(z)$ contains a contribution from $X(z)$ and a contribution from $X(-z)$. The term involving $X(-z)$ is due to the aliasing effect that arises from the downsampling stage. The aliasing effect can be eliminated if the analysis and synthesis filters satisfy the following condition:

$$\boxed{H_\ell(-z)G_\ell(z) + H_h(-z)G_h(z) = 0} \quad \text{(alias-free condition)} \qquad (26.140)$$

We shall pursue this condition in more detail later in Sec. 27.5. Here, it suffices to say that one possible choice for the synthesis filters is

$$G_h(z) = -2H_\ell(-z), \qquad G_\ell(z) = 2H_h(-z) \qquad (26.141)$$

where the scaling by 2 is incorporated for convenience.

Now recall that $H_\ell(z)$ is a low-pass filter. Its high-pass counterpart, $H_h(z)$, can be chosen as

$$\boxed{H_h(z) = H_\ell(-z)} \qquad (26.142)$$

When the analysis filters $\{H_\ell(z), H_h(z)\}$ satisfy this property, and $\{G_\ell(z), G_h(z)\}$ are selected according to (26.141), then the alias-free filter-bank of Fig. 26.39 becomes known as a quadrature mirror filter (QMF) bank. The reason for this designation is the following. Assume the low-pass filter $H_\ell(z)$ has a real-valued impulse response sequence. Then, its magnitude frequency response, $|H_\ell(e^{j\omega})|$, is symmetric about the vertical axis, i.e., $|H_\ell(e^{j\omega})| = |H_\ell(e^{-j\omega})|$. It follows from (26.142) that:

$$\begin{aligned} |H_h(e^{j\omega})| &= |H_\ell(-e^{j\omega})| \\ &= |H_\ell(e^{j(\omega-\pi)})| \\ &= |H_\ell(e^{j(-\omega+\pi)})| \\ &= |H_\ell(e^{j(\pi-\omega)})| \end{aligned} \tag{26.143}$$

That is,

$$\boxed{|H_h(e^{j\omega})| = \left|H_\ell\left(e^{j(\pi-\omega)}\right)\right|} \tag{26.144}$$

This result indicates that the magnitude responses of the analysis filters, $H_\ell(z)$ and $H_h(z)$, are symmetric about the quadrature frequency, $\omega = \pi/2$, as illustrated in Fig. 26.40 and, hence, the name Quadrature Mirror Filter Bank.

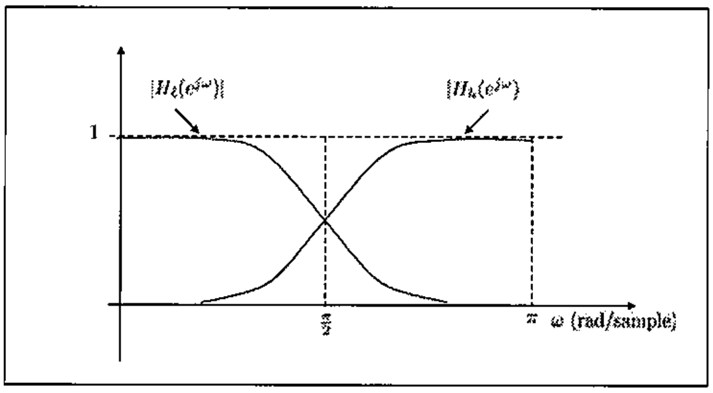

FIGURE 26.40 The magnitude responses of the analysis filters of a QMF are symmetric about $\omega = \pi/2$ rad/sample.

In view of (26.142), the alias-free condition (26.140) for the QMF filter bank can be met by selecting the synthesis filters as:

$$\boxed{G_\ell(z) = 2H_h(-z) = 2H_\ell(z), \qquad G_h(z) = -2H_\ell(-z)} \tag{26.145}$$

Thus, note that all filters in the QMF implementation are defined in terms of the single prototype filter, $H_\ell(z)$. Using (26.139), we find that the transfer function from $x(n)$ to $y(n)$ is given by

$$\boxed{R(z) = H_\ell^2(z) - H_\ell^2(-z)} \tag{26.146}$$

which still does not necessarily evaluate to the desired pure delay.

Haar Filters

Let us now select the low-pass filter $H_\ell(z)$ as the simple averaging (or Haar) filter

$$H_\ell(z) = \frac{1}{2}\left(1 + z^{-1}\right) \tag{26.147}$$

This means that $x_\ell(n)$ is obtained by averaging the samples $x(n)$ and $x(n-1)$:

$$x_\ell(n) = \frac{1}{2}(x(n) + x(n-1)) \tag{26.148}$$

Then, according to (26.142), the corresponding high-pass filter is the difference filter

$$H_h(z) = \frac{1}{2}\left[1 - z^{-1}\right] \tag{26.149}$$

This means that $x_h(n)$ is obtained by averaging the difference between two successive samples:

$$x_\ell(n) = \frac{1}{2}(x(n) - x(n-1)) \tag{26.150}$$

Using (26.145), the synthesis filters are then selected as

$$\begin{aligned} G_\ell(z) &= 1 + z^{-1} && (26.151) \\ G_h(z) &= -1 + z^{-1} && (26.152) \end{aligned}$$

and, from (26.146), the overall transfer function from $x(n)$ to $y(n)$ becomes a pure delay — see Fig. 26.41:

$$R(z) = z^{-1} \tag{26.153}$$

In this way, the input-output mapping of the filter-bank behaves like a pure delay even though the analysis and synthesis filters are not ideal (low-pass or high-pass) filters.

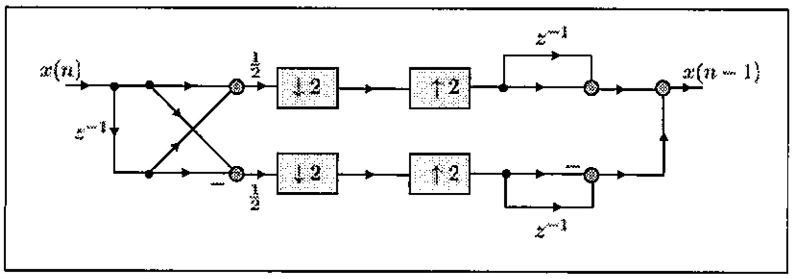

FIGURE 26.41 QMF filter bank that results from selecting $H_\ell(z) = \frac{1}{2}(1 + z^{-1})$.

Practice Questions:

(a) Assume B = 4.8KHz and F_s = 9.6KHz. What are the frequency limits of the octave subbands in Fig. 26.34?

(b) Consider a speech signal of duration 2 seconds. If each sample is quantized to 16 bits, how much memory is needed to store the quantized speech signal? Assume now the frequency range $[0, 4.8]$KHz is subdivided into four octaves and $\{16, 8, 4, 2\}$ bits are used to quantize the samples that fall into the first, second, third, and fourth octaves. Assume further that each octave contains an equal number of samples. How much memory is needed to store the quantized speech signal?

(c) Consider the same speech signal of duration 2 seconds. Assume initially that each sample is quantized to 12 bits. How much memory is needed to store the speech signal? Assume the frequency range $[0, 4.8]$KHz is subdivided into four octaves and $\{16, 8, 4, 2\}$ bits are used to quantize the samples that fall into the first, second, third, and fourth octaves. Assume further that a fraction p of the speech samples lie within the first octave (quantized with 16 bits). Give a lower bound on p such that the amount of memory that is needed to store the speech signal using subband coding would be less than what is needed using 12 bit quantization.

(d) Use expression (26.45) to show that the DTFTs of the sequences $x_\ell(n)$ and $x_h(n)$ shown in Fig. 26.35 when ideal low-pass and high-pass filters are used are given by

$$X_\ell(e^{j\omega}) = \frac{1}{2}X\left(e^{\frac{j\omega}{2}}\right), \quad X_h(e^{j\omega}) = \frac{1}{2}X\left(e^{\frac{j(\omega-2\pi)}{2}}\right), \quad \omega \in [0, \pi] \qquad (26.154)$$

Justify the form of the frequency responses shown in Fig. 26.36.

(e) Consider the choices (with noncausal analysis filters):

$$H_\ell(z) = \frac{1}{2}(1+z), \quad H_h(z) = -\frac{1}{2}(1-z), \quad G_\ell(z) = 1 + z^{-1}, \quad G_h(z) = -1 + z^{-1} \qquad (26.155)$$

Verify that these choices satisfy the alias-free condition (26.140) and that the overall transfer function of the filter-bank of Fig. 26.39 is $R(z) = 1$.

◇

26.6 REFERENCES AND COMMENTARIES

Multirate Digital Signal Processing by R. E. Crochiere and L. R. Rabiner, Prentice-Hall, 1983. ISBN
Multirate Systems and Filter Banks by P. P. Vaidyanathan, Prentice-Hall, 1992.
RONALD E. CROCHIERE, SENIOR MEMBER, IEEE, AND LAWRENCE R. RABINER, Interpolation and Decimation of Digital Signals-Tutorial Review PROCEEDINGS OF THE IEEE, VOL. 69, NO. 3, pp. 300–331, MARCH 1981

Nyquist filters
Noble identities
subband coding
Haar filter
QMF filter
polyphase decomposition

26.7 PROBLEMS

P 26.1 Draw a block diagram representation as in Fig. 26.11 to increase the sampling rate of a sequence $x(n)$ by a factor of 3. What is the gain and cutoff frequency of the low-pass filter $H(z)$ in radians/sample. If the sequence $x(n)$ is obtained by sampling $x(t)$ at 12KHz, at what sampling rate is $H(z)$ operating? What is its cutoff frequency in Hz?

P 26.2 Draw a block diagram representation as in Fig. 26.17 to decrease the sampling rate of a sequence $x(n)$ by a factor of 3. What is the gain and cutoff frequency of the low-pass filter $H(z)$ in radians/sample. If the sequence $x(n)$ is obtained by sampling $x(t)$ at 12KHz, at what sampling rate is $H(z)$ operating? What is its cutoff frequency in Hz?

P 26.3 Refer to Fig. 26.11 with an upsampling factor of $L = 3$. Assume $X(e^{j\omega})$ is as shown in Fig. 26.42 over the range $[-\pi, \pi]$. Draw the DTFTs of the sequences $y(n)$ and $x'(n)$.

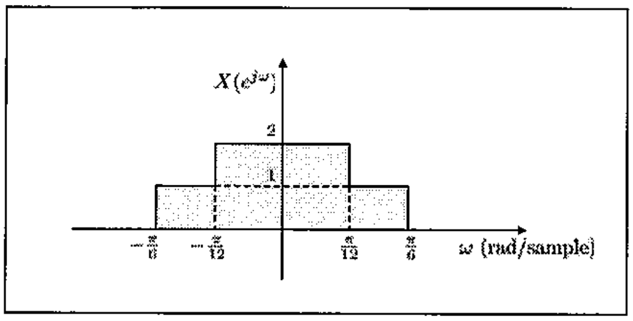

FIGURE 26.42 DTFT of the sequence $x(n)$ for Prob. 26.3.

P 26.4 Refer to Fig. 26.11 with an upsampling factor of $L = 3$. Assume $X(e^{j\omega})$ is as shown in Fig. 26.43 over the range $[-\pi, \pi]$. Draw the DTFTs of the sequences $y(n)$ and $x'(n)$.

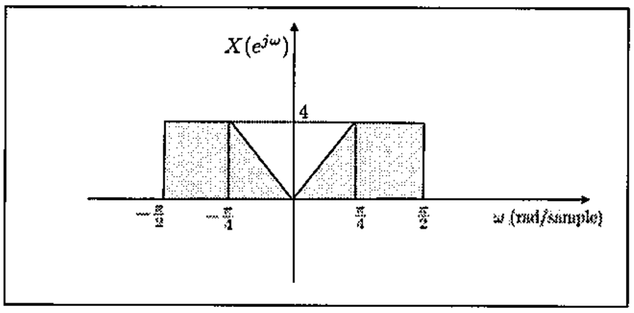

FIGURE 26.43 DTFT of the sequence $x(n)$ for Prob. 26.4.

P 26.5 Refer to Fig. 26.17 with a downsampling factor of $M = 3$. Assume $X(e^{j\omega})$ is as shown in Fig. 26.44 over the range $[-\pi, \pi]$. Draw the DTFTs of the sequences $y(n)$ and $x''(n)$.

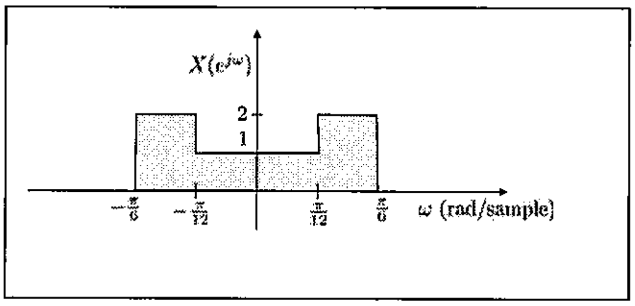

FIGURE 26.44 DTFT of the sequence $x(n)$ for Prob. 26.5.

P 26.6 Refer to Fig. 26.17 with a downsampling factor of $M = 3$. Assume $X(e^{j\omega})$ is as shown in Fig. 26.45 over the range $[-\pi, \pi]$. Draw the DTFTs of the sequences $y(n)$ and $x''(n)$.

P 26.7 Draw a block diagram representation as in Fig. 26.18 to convert the sampling rate of a sequence from 10KHz to 15KHz. At what sampling rate is $H(z)$ operating in Hz?

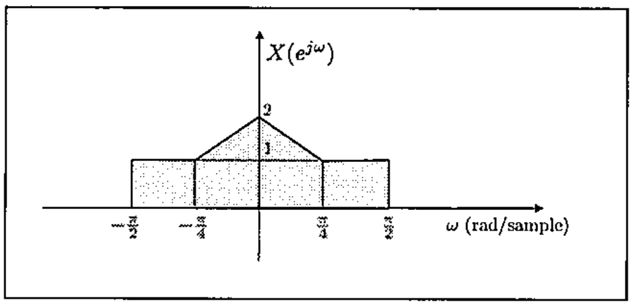

FIGURE 26.45 DTFT of the sequence $x(n)$ for Prob. 26.6.

P 26.8 Draw a block diagram representation as in Fig. 26.18 to convert the sampling rate of a sequence from 10KHz to 6KHz. At what sampling rate is $H(z)$ operating in Hz?

P 26.9 Draw a multistage rate converter as in Example 26.8 for $L = 24$ and $M = 42$.

P 26.10 Draw a multistage rate converter as in Example 26.8 for $L = 150$ and $M = 180$.

P 26.11 A signal $x(n)$ is first downsampled by a factor M and the result is downsampled by a factor M' to generate $y(n)$. Show that the succession of two downsamplers can be replaced by a single downsampler whose factor is the product MM'.

P 26.12 A signal $x(n)$ is first upsampled by a factor L and the result is upsampled by a factor L' to generate $y(n)$. Show that the succession of two upsamplers can be replaced by a single upsampler whose factor is the product LL'.

P 26.13 Explain why upsampling by a factor L is an invertible transformation. How can you recover the original sequence from its upsampled version?

P 26.14 Explain why downsampling by a factor M is a non-invertible transformation?

P 26.15 A signal $x(n)$ is first downsampled by a factor M and the result is upsampled by a factor L to generate $y_1(n)$. In an alternative implementation, the location of the downsampler and upsampler are reversed: the same signal $x(n)$ is first upsampled by a factor L and the result is downsampled by a factor M to generate $y_2(n)$. Show that the output sequences will coincide (i.e., the upsampler and downsampler can be interchanged) if, and only if, the factors M and L are coprime, namely, their only common factor is the number one.

P 26.16 Consider the multirate system shown in Fig. 26.46 and the DTFT shown in Fig. 26.45. Draw the DTFTs of the sequences $y(n), z(n), s(n)$, and $r(n)$.

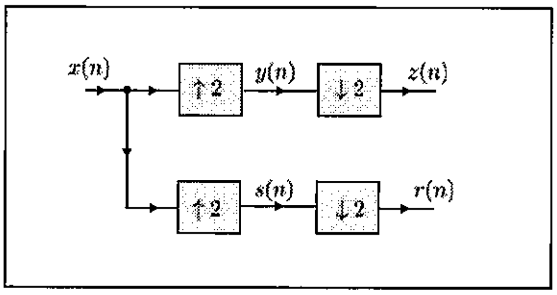

FIGURE 26.46 Multirate system for Prob. 26.16.

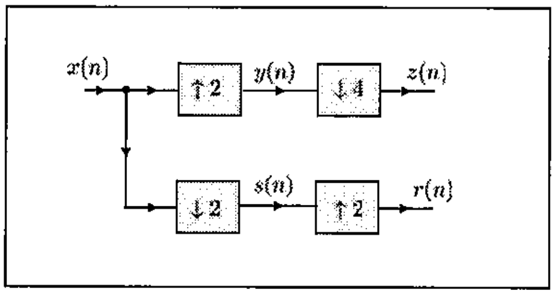

FIGURE 26.47 Multirate system for Prob. 26.17.

P 26.17 Consider the multirate system shown in Fig. 26.47 and the DTFT shown in Fig. 26.45. Draw the DTFTs of the sequences $y(n), z(n), s(n)$, and $r(n)$.

P 26.18 Determine the second and third-order polyphase components of the transfer function

$$H(z) = 1 + \frac{1}{2}z^{-1} + \frac{1}{3}z^{-2} + \frac{1}{4}z^{-3} + \frac{1}{5}z^{-5}$$

Draw block diagram representations for $H(z)$ in terms of its polyphase components.

P 26.19 Draw an efficient polyphase realization for the structure shown in Fig. 26.48 assuming

$$H(z) = 1 - \frac{1}{2}z^{-1} + \frac{1}{3}z^{-2} - \frac{1}{4}z^{-3} + \frac{1}{5}z^{-5}$$

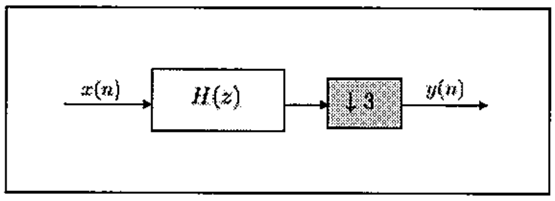

FIGURE 26.48 Decimator system for Prob. 26.19.

P 26.20 Draw efficient type-I and type-II polyphase realizations for the structure shown in Fig. 26.49 assuming

$$H(z) = 1 - \frac{1}{2}z^{-1} + \frac{1}{3}z^{-2} - \frac{1}{4}z^{-3} + \frac{1}{5}z^{-5}$$

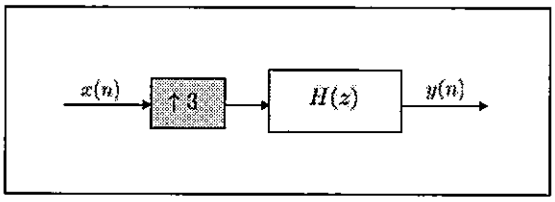

FIGURE 26.49 Interpolator system for Prob. 26.20.

P 26.21 Let $E_o(z)$ denote the zero-th polyphase component of order L of an FIR filter, $H(z)$. Show that the cascade shown on the left in Fig. 26.50 is equivalent to the LTI system shown on the right in the same figure.

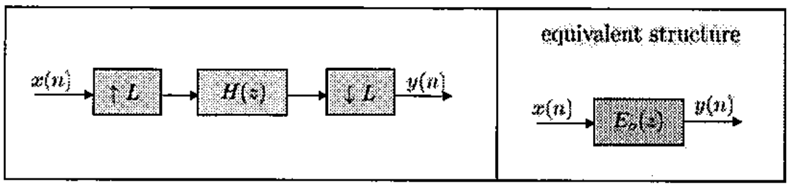

FIGURE 26.50 The cascade on the left is equivalent to the LTI system on the right.

P 26.22 The DTFT of a real-valued sequence $x(n)$ is shown in the top part of Fig. 26.51 over the interval $[0, \pi]$. Plot the DTFTs of the sequences $\{x_k(\cdot)\}$ appearing in the bottom part of the same figure for $k = 1, 2, \ldots, 6$.

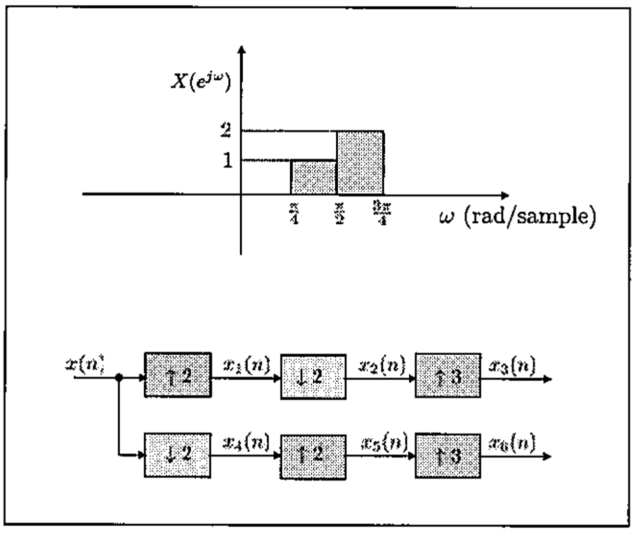

FIGURE 26.51 Multirate structure for Prob. 26.22.

P 26.23 Consider the M−branch filter bank shown in Fig. 26.52. Establish algebraically that the transfer function from $x(n)$ to $y(n)$ is a pure delay of $M - 1$ samples, i.e., $Y(z) = z^{-(M-1)}X(z)$.

P 26.24 Explain why the term $X(-z)$ in (26.139) arises from aliasing.

P 26.25 Explain why when a filter $H_\ell(z)$ is low-pass, then the filter $H_h(z) = H_\ell(-z)$ has hight-pass characteristics.

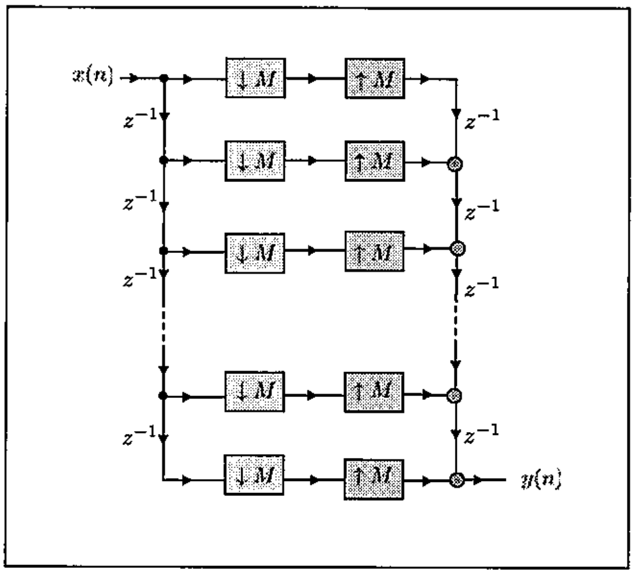

FIGURE 26.52 An M−branch filter bank for Prob. 26.23.