Digital Signal Processing

Applications

Sanjit K Mitra

Contents

1 Applications of Digital Signal Processing **1**

1 Dual-Tone Multifrequency Signal Detection 1
2 Spectral Analysis of Sinusoidal Signals 5
3 Analysis of Speech Signals Using the STFT 11
4 Spectral Analysis of Random Signals 13
5 Musical Sound Processing 21
6 Digital Music Synthesis 35
7 Discrete-Time Analytic Signal Generation 37
8 Signal Compression 44
9 Transmultiplexers 51
10 Discrete Multitone Transmission of Digital Data 55
11 Oversampling A/D Converter 58
12 Oversampling D/A Converter 64
13 Sparse Antenna Array Design 69
14 Programs 73

Applications of Digital Signal Processing

As mentioned in Chapter 1 of Text, digital signal processing techniques are increasingly replacing conventional analog signal processing methods in many fields, such as speech analysis and processing, radar and sonar signal processing, biomedical signal analysis and processing, telecommunications, and geophysical signal processing. In this chapter, we include a few simple applications to provide a glimpse of the potential of DSP.

We first describe several applications of the discrete Fourier transform (DFT) introduced in Section 5.2. The first application considered is the detection of the frequencies of a pair of sinusoidal signals, called tones, employed in telephone signaling. Next, we discuss the use of the DFT in the determination of the spectral contents of a continuous-time signal. The effect of the DFT length and the windowing of the sequence are examined in detail here. In the following section, we discuss its application of the short-time Fourier transform (STFT) introduced in Section 5.11 of Text for the spectral analysis of nonstationary signals. We then consider the spectral analysis of random signals using both nonparametric and parametric methods. Application of digital filtering methods to musical sound processing is considered next, and a variety of practical digital filter structures useful for the generation of certain audio effects, such as artificial reverberation, flanging, phasing, filtering, and equalization, are introduced. Generation of discrete-time analytic signals by means of a discrete-time Hilbert transformer is then considered, and several methods of designing these circuits are outlined along with an application. The basic concepts of signal compression are reviewed next, along with a technique for image compression based on Haar wavelets. The theory and design of transmultiplexers are discussed in the following section. One method of digital data transmission employing digital signal processing methods is then introduced. The basic concepts behind the design of the oversampling A/D and D/A converters are reviewed in the following two sections. Finally, we review the sparse antenna array design for ultrasound scanners.

1 Dual-Tone Multifrequency Signal Detection

Dual-tone multifrequency (DTMF) signaling, increasingly being employed worldwide with push-button telephone sets, offers a high dialing speed over the dial-pulse signaling used in conventional rotary telephone sets. In recent years, DTMF signaling has also found applications requiring interactive control, such as in voice mail, electronic mail (e-mail), telephone banking, and ATM machines.

A DTMF signal consists of a sum of two tones, with frequencies taken from two mutually exclusive groups of preassigned frequencies. Each pair of such tones represents a unique number or a symbol. Decoding of a DTMF signal thus involves identifying the two tones in that signal and determining their

corresponding number or symbol. The frequencies allocated to the various digits and symbols of a push-button keypad are internationally accepted standards and are shown in Figure 1.35 of Text.[1] The four keys in the last column of the keypad, as shown in this figure, are not yet available on standard handsets and are reserved for future use. Since the signaling frequencies are all located in the frequency band used for speech transmission, this is an *in-band system*. Interfacing with the analog input and output devices is provided by *codec* (coder/decoder) chips or A/D and D/A converters.

Although a number of chips with analog circuitry are available for the generation and decoding of DTMF signals in a single channel, these functions can also be implemented digitally on DSP chips. Such a digital implementation surpasses analog equivalents in performance, since it provides better precision, stability, versatility, and reprogrammability to meet other tone standards and the scope for multichannel operation by time-sharing, leading to a lower chip count.

The digital implementation of a DTMF signal involves adding two finite-length digital sinusoidal sequences, with the latter simply generated by using look-up tables or by computing a polynomial expansion. The digital tone detection can be easily performed by computing the DFT of the DTMF signal and then measuring the energy present at the eight DTMF frequencies. The minimum duration of a DTMF signal is 40 ms. Thus, with a sampling rate of 8 kHz, there are at most $0.04 \times 8000 = 320$ samples available for decoding each DTMF digit. The actual number of samples used for the DFT computation is less than this number and is chosen so as to minimize the difference between the actual location of the sinusoid and the nearest integer value DFT index k.

The DTMF decoder computes the DFT samples closest in frequency to the eight DTMF fundamental tones and their respective second harmonics. In addition, a practical DTMF decoder also computes the DFT samples closest in frequency to the second harmonics corresponding to each of the fundamental tone frequencies. This latter computation is employed to distinguish between human voices and the pure sinusoids generated by the DTMF signal. In general, the spectrum of a human voice contains components at all frequencies including the second harmonic frequencies. On the other hand, the DTMF signal generated by the handset has negligible second harmonics. The DFT computation scheme employed is a slightly modified version of Goertzel's algorithm, as described in Section 11.3.1 of Text, for the computation of the squared magnitudes of the DFT samples that are needed for the energy computation.

The DFT length N determines the frequency spacing between the locations of the DFT samples and the time it takes to compute the DFT sample. A large N makes the spacing smaller, providing higher resolution in the frequency domain, but increases the computation time. The frequency f_k in Hz corresponding to the DFT index (bin number) k is given by

$$f_k = \frac{kF_T}{N}, \qquad k = 0, 1, \ldots, N-1, \tag{1}$$

where F_T is the sampling frequency. If the input signal contains a sinusoid of frequency f_{in} different from that given above, its DFT will contain not only large-valued samples at values of k closest to Nf_{in}/F_T but also nonzero values at other values of k due to a phenomenon called leakage (see Example 11.16 of Text). To minimize the leakage, it is desirable to choose N appropriately so that the tone frequencies fall as close as possible to a DFT bin, thus providing a very strong DFT sample at this index value relative to all other values. For an 8-kHz sampling frequency, the best value of the DFT length N to detect the eight fundamental DTMF tones has been found to be 205 and that for detecting the eight second harmonics is 201.[2] Table 1 shows the DFT index values closest to each of the tone frequencies and their second

[1] International Telecommunication Union, *CCITT Red Book*, volume VI, Fascicle VI.1, October 1984.

[2] *Digital Signal Processing Applications Using the ADSP-2100 Family*, A. Mar, editor, Prentice Hall, Englewood Cliffs NJ, 1992.

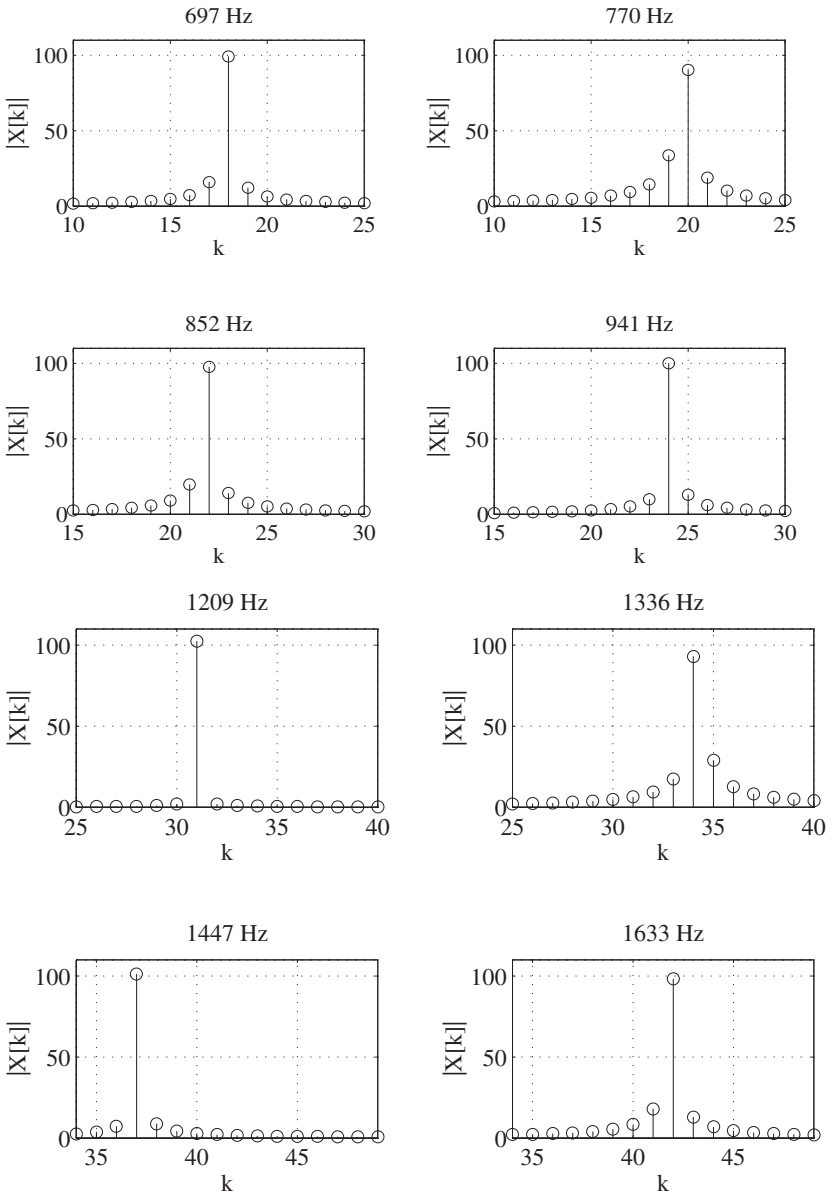


Figure 1: Selected DFT samples for each one of the DTMF tone signals for $N = 205$.

harmonics for these two values of N, respectively. Figure 1 shows 16 selected DFT samples computed using a 205-point DFT of a length-205 sinusoidal sequence for each of the fundamental tone frequencies.

Program A-1[3] can be used to demonstrate the DFT-based DTMF detection algorithm. The outputs generated by this program for the input symbol # are displayed in Figure 2.

[3]All MATLABprograms mentioned in this section are given in the Programs Section of the CD.

Table 1: DFT index values for DTMF tones for $N = 205$ and their second harmonics for $N = 201$.

Basic tone in Hz	**Exact k value**	**Nearest integer k value**	**Absolute error in k**
697	17.861	18	0.139
770	19.731	20	0.269
852	21.833	22	0.167
941	24.113	24	0.113
1209	30.981	31	0.019
1336	34.235	34	0.235
1477	37.848	38	0.152
1633	41.846	42	0.154
Second harmonic in Hz	**Exact k value**	**Nearest integer k value**	**Absolute error in k**
1394	35.024	35	0.024
1540	38.692	39	0.308
1704	42.813	43	0.187
1882	47.285	47	0.285
2418	60.752	61	0.248
2672	67.134	67	0.134
2954	74.219	74	0.219
3266	82.058	82	0.058

Adapted from *Digital Signal Processing Applications Using the ADSP-2100 Family*, A. Mar, editor, Prentice Hall, Englewood Cliffs NJ, 1992.

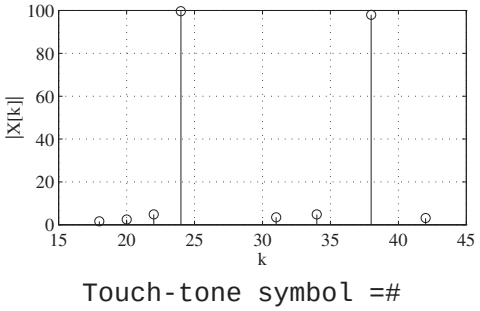


Figure 2: A typical output of Program A-1.

2 Spectral Analysis of Sinusoidal Signals

An important application of digital signal processing methods is in determining in the discrete-time domain the frequency contents of a continuous-time signal, more commonly known as *spectral analysis*. More specifically, it involves the determination of either the energy spectrum or the power spectrum of the signal. Applications of digital spectral analysis can be found in many fields and are widespread. The spectral analysis methods are based on the following observation. If the continuous-time signal $g_a(t)$ is reasonably band-limited, the spectral characteristics of its discrete-time equivalent $g[n]$ should provide a good estimate of the spectral properties of $g_a(t)$. However, in most cases, $g_a(t)$ is defined for $-\infty < t < \infty$, and as a result, $g[n]$ is of infinite extent and defined for $-\infty < n < \infty$. Since it is difficult to evaluate the spectral parameters of an infinite-length signal, a more practical approach is as follows. First, the continuous-time signal $g_a(t)$ is passed through an analog anti-aliasing filter before it is sampled to eliminate the effect of aliasing. The output of the filter is then sampled to generate a discrete-time sequence equivalent $g[n]$. It is assumed that the anti-aliasing filter has been designed appropriately, and hence, the effect of aliasing can be ignored. Moreover, it is further assumed that the A/D converter wordlength is large enough so that the A/D conversion noise can be neglected.

This and the following two sections provide a review of some spectral analysis methods. In this section, we consider the Fourier analysis of a stationary signal composed of sinusoidal components. In Section 3, we discuss the Fourier analysis of nonstationary signals with time-varying parameters. Section 4 considers the spectral analysis of random signals.[4]

For the spectral analysis of sinusoidal signals, we assume that the parameters characterizing the sinusoidal components, such as amplitudes, frequencies, and phase, do not change with time. For such a signal $g[n]$, the Fourier analysis can be carried out by computing its Fourier transform $G(e^{j\omega})$:

$$G(e^{j\omega}) = \sum_{n=-\infty}^{\infty} g[n]e^{-j\omega n}. \tag{2}$$

In practice, the infinite-length sequence $g[n]$ is first windowed by multiplying it with a length-N window $w[n]$ to make it into a finite-length sequence $\gamma[n] = g[n] \cdot w[n]$ of length N. The spectral characteristics of the windowed finite-length sequence $\gamma[n]$ obtained from its Fourier transform $\Gamma(e^{j\omega})$ then is assumed to provide a reasonable estimate of the Fourier transform $G(e^{j\omega})$ of the discrete-time signal $g[n]$. The Fourier transform $\Gamma(e^{j\omega})$ of the windowed finite-length segment $\gamma[n]$ is next evaluated at a set of $R(R \geq N)$ discrete angular frequencies equally spaced in the range $0 \leq \omega < 2\pi$ by computing its R-point discrete Fourier transform (DFT) $\Gamma[k]$. To provide sufficient resolution, the DFT length R is chosen to be greater than the window N by zero-padding the windowed sequence with $R-N$ zero-valued samples. The DFT is usually computed using an FFT algorithm.

We examine the above approach in more detail to understand its limitations so that we can properly make use of the results obtained. In particular, we analyze here the effects of windowing and the evaluation of the frequency samples of the Fourier transform via the DFT.

Before we can interpret the spectral content of $\Gamma(e^{j\omega})$, that is, $G(e^{j\omega})$, from $\Gamma[k]$, we need to reexamine the relations between these transforms and their corresponding frequencies. Now, the relation between the R-point DFT $\Gamma[k]$ of $\gamma[n]$ and its Fourier transform $\Gamma(e^{j\omega})$ is given by

$$\Gamma[k] = \Gamma(e^{j\omega})\big|_{\omega=2\pi k/R}, \qquad 0 \leq k \leq R-1. \tag{3}$$

[4]For a detailed exposition of spectral analysis and a concise review of the history of this area, see R. Kumaresan, "Spectral analysis", In S.K. Mitra and J.F. Kaiser, editors, *Handbook for Digital Signal Processing*, chapter 16, pages 1143–1242. Wiley-Interscience, New York NY, 1993.

The normalized discrete-time angular frequency ω_k corresponding to the DFT bin number k (DFT frequency) is given by

$$\omega_k = \frac{2\pi k}{R}. \tag{4}$$

Likewise, the continuous-time angular frequency Ω_k corresponding to the DFT bin number k (DFT frequency) is given by

$$\Omega_k = \frac{2\pi k}{RT}. \tag{5}$$

To interpret the results of the DFT-based spectral analysis correctly, we first consider the frequency-domain analysis of a sinusoidal sequence. Now an infinite-length sinusoidal sequence $g[n]$ of normalized angular frequency ω_o is given by

$$g[n] = \cos(\omega_o n + \phi). \tag{6}$$

By expressing the above sequence as

$$g[n] = \tfrac{1}{2}\left(e^{j(\omega_o n+\phi)} + e^{-j(\omega_o n+\phi)}\right) \tag{7}$$

and making use of Table 3.3 of Text, we arrive at the expression for its Fourier transform as

$$G(e^{j\omega}) = \pi \sum_{\ell=-\infty}^{\infty} \left(e^{j\phi}\delta(\omega - \omega_o + 2\pi\ell) + e^{-j\phi}\delta(\omega + \omega_o + 2\pi\ell)\right). \tag{8}$$

Thus, the Fourier transform is a periodic function of ω with a period 2π containing two impulses in each period. In the frequency range, $-\pi \le \omega < \pi$, there is an impulse at $\omega = \omega_o$ of complex amplitude $\pi e^{j\phi}$ and an impulse at $\omega = -\omega_o$ of complex amplitude $\pi e^{-j\phi}$.

To analyze $g[n]$ in the spectral domain using the DFT, we employ a finite-length version of the sequence given by

$$\gamma[n] = \cos(\omega_o n + \phi), \qquad 0 \le n \le N-1. \tag{9}$$

The computation of the DFT of a finite-length sinusoid has been considered in Example 11.16 of Text. In this example, using Program 11_10, we computed the DFT of a length-32 sinusoid of frequency 10 Hz sampled at 64 Hz, as shown in Figure 11.32(a) of Text. As can be seen from this figure, there are only two nonzero DFT samples, one at bin $k = 5$ and the other at bin $k = 27$. From Eq. (5), bin $k = 5$ corresponds to frequency 10 Hz, while bin $k = 27$ corresponds to frequency 54 Hz, or equivalently, -10 Hz. Thus, the DFT has correctly identified the frequency of the sinusoid.

Next, using the same program, we computed the 32-point DFT of a length-32 sinusoid of frequency 11 Hz sampled at 64 Hz, as shown in Figure 11.32(b) of Text. This figure shows two strong peaks at bin locations $k = 5$ and $k = 6$, with nonzero DFT samples at other bin locations in the positive half of the frequency range. Note that the bin locations 5 and 6 correspond to frequencies 10 Hz and 12 Hz, respectively, according to Eq. (5). Thus the frequency of the sinusoid being analyzed is exactly halfway between these two bin locations.

The phenomenon of the spread of energy from a single frequency to many DFT frequency locations as demonstrated by Figure 11.32(b) of Text is called *leakage*. To understand the cause of this effect, we recall that the DFT $\Gamma[k]$ of a length-N sequence $\gamma[n]$ is given by the samples of its discrete-time Fourier transform (Fourier transform) $\Gamma(e^{j\omega})$ evaluated at $\omega = 2\pi k/N$, $k = 0, 1, \ldots, N-1$. Figure 3 shows the Fourier transform of the length-32 sinusoidal sequence of frequency 11 Hz sampled at 64 Hz. It can be seen that the DFT samples shown in Figure 11.32(b) of Text are indeed obtained by the frequency samples of the plot of Figure 3.

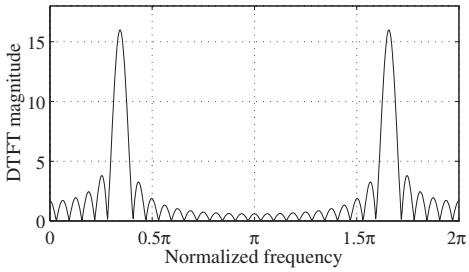


Figure 3: Fourier transform of a sinusoidal sequence windowed by a rectangular window.

To understand the shape of the Fourier transform shown in Figure 3, we observe that the sequence of Eq. (9) is a windowed version of the infinite-length sequence $g[n]$ of Eq. (6) obtained using a rectangular window $w[n]$:

$$w[n] = \begin{cases} 1, & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Hence, the Fourier transform $\Gamma(e^{j\omega})$ of $\gamma[n]$ is given by the frequency-domain convolution of the Fourier transform $G(e^{j\omega})$ of $g[n]$ with the Fourier transform $\Psi_R(e^{j\omega})$ of the rectangular window $w[n]$:

$$\Gamma(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} G(e^{j\varphi})\Psi_R(e^{j(\omega-\varphi)})\,d\varphi, \tag{11}$$

where

$$\Psi_R(e^{j\omega}) = e^{-j\omega(N-1)/2}\frac{\sin(\omega N/2)}{\sin(\omega/2)}. \tag{12}$$

Substituting $G(e^{j\omega})$ from Eq. (8) into Eq. (11), we arrive at

$$\Gamma(e^{j\omega}) = \tfrac{1}{2}e^{j\phi}\Psi_R(e^{j(\omega-\omega_o)}) + \frac{1}{2}e^{-j\phi}\Psi_R(e^{j(\omega+\omega_o)}). \tag{13}$$

As indicated by Eq. (13), the Fourier transform $\Gamma(e^{j\omega})$ of the windowed sequence $\gamma[n]$ is a sum of the frequency shifted and amplitude scaled Fourier transform $\Psi_R(e^{j\omega})$ of the window $w[n]$, with the amount of frequency shifts given by $\pm\omega_o$. Now, for the length-32 sinusoid of frequency 11 Hz sampled at 64 Hz, the normalized frequency of the sinusoid is $11/64 = 0.172$. Hence, its Fourier transform is obtained by frequency shifting the Fourier transform $\Psi_R(e^{j\omega})$ of a length-32 rectangular window to the right and to the left by the amount $0.172 \times 2\pi = 0.344\pi$, adding both shifted versions, and then amplitude scaling by a factor 1/2. In the normalized angular frequency range 0 to 2π, which is one period of the Fourier transform, there are two peaks, one at 0.344π and the other at $2\pi(1-0.172) = 1.656\pi$, as verified by Figure 3. A 32-point DFT of this Fourier transform is precisely the DFT shown in Figure 11.32(b) of Text. The two peaks of the DFT at bin locations $k = 5$ and $k = 6$ are frequency samples of the main lobe located on both sides of the peak at the normalized frequency 0.172. Likewise, the two peaks of the DFT at bin locations $k = 26$ and $k = 27$ are frequency samples of the main lobe located on both sides of the peak at the normalized frequency 0.828. All other DFT samples are given by the samples of the sidelobes of the Fourier transform of the window causing the leakage of the frequency components at $\pm\omega_o$ to other bin locations, with the amount of leakage determined by the relative amplitude of the main lobe and the sidelobes. Since the relative sidelobe level $A_{s\ell}$, defined by the ratio in dB of the amplitude of the main

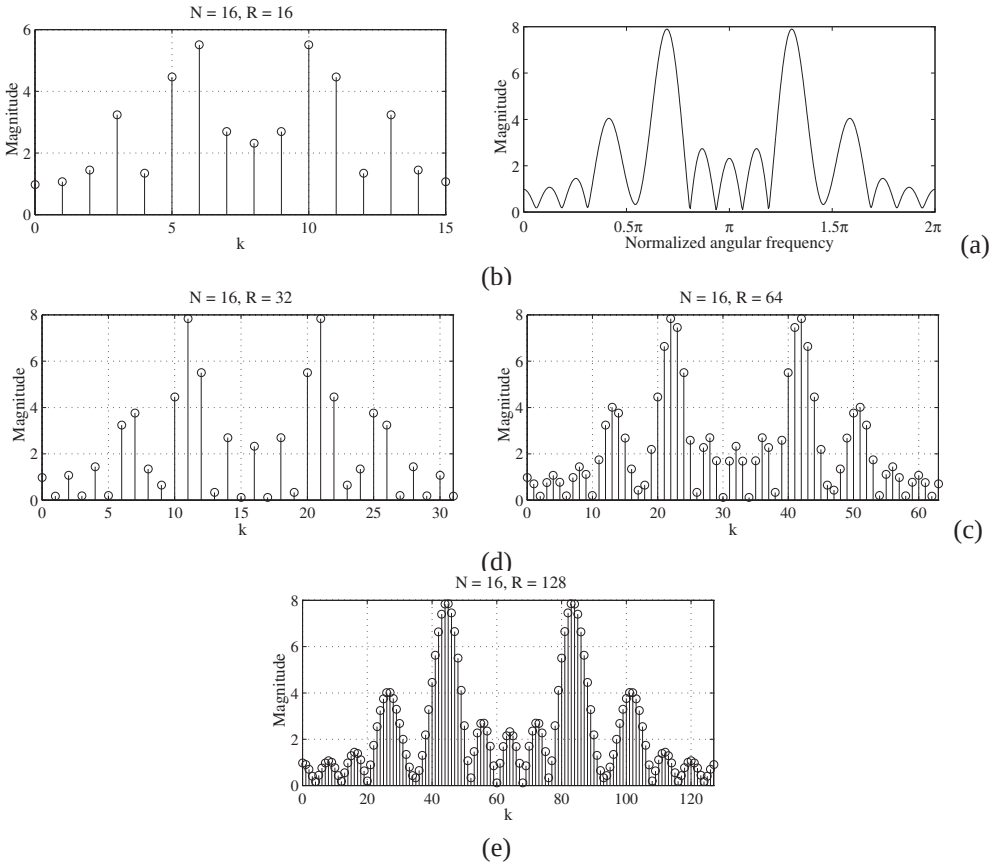


Figure 4: (a)–(e) DFT-based spectral analysis of a sum of two finite-length sinusoidal sequences of normalized frequencies 0.22 and 0.34, respectively, of length 16 each for various values of DFT lengths.

lobe to that of the largest sidelobe, of the rectangular window is very high, there is a considerable amount of leakage to the bin locations adjacent to the bins showing the peaks in Figure 11.32(b) of Text.

The above problem gets more complicated if the signal being analyzed has more than one sinusoid, as is typically the case. We illustrate the DFT-based spectral analysis approach by means of Examples 1 through 3. Through these examples, we examine the effects of the length R of the DFT, the type of window being used, and its length N on the results of spectral analysis.

EXAMPLE 1 Effect of the DFT Length on Spectral Analysis

The signal to be analyzed in the spectral domain is given by

$$x[n] = \tfrac{1}{2}\sin(2\pi f_1 n) + \sin(2\pi f_2 n), \qquad 0 \le n \le N-1. \tag{14}$$

Let the normalized frequencies of the two length-16 sinusoidal sequences be $f_1 = 0.22$ and $f_2 = 0.34$. We compute the DFT of their sum $x[n]$ for various values of the DFT length R. To this end, we use Program A-2 whose input data are the length `N` of the signal, length `R` of the DFT, and the two frequencies f_1 and f_2. The program generates the two sinusoidal sequences, forms their sum, then computes the DFT of the sum and plots the DFT samples. In this example, we fix $N = 16$ and vary the DFT length R from 16 to 128. Note that when `R > N`, the M-file `fft(x,R)` automatically zero-pads the sequence `x` with `R-N` zero-valued samples.

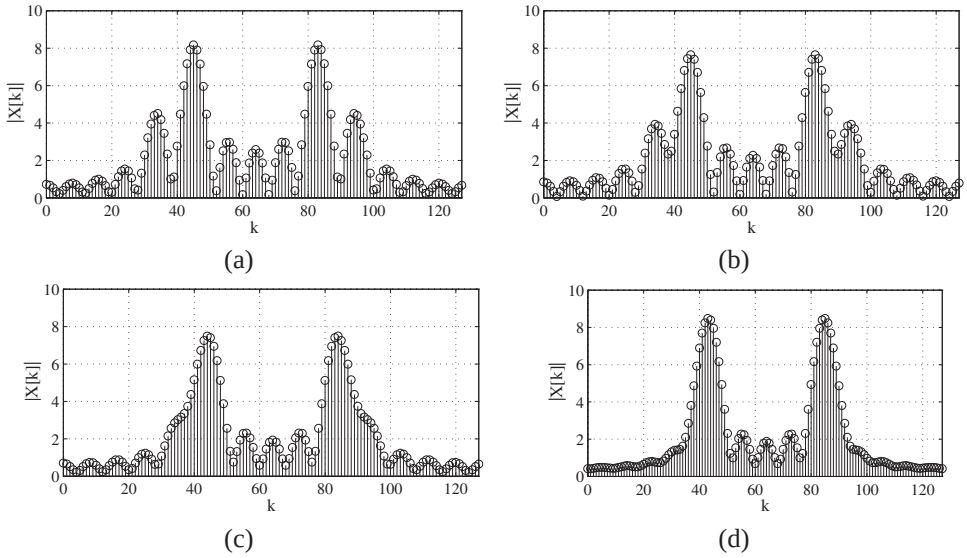


Figure 5: Illustration of the frequency resolution property: (a) $f_1 = 0.28$, $f_2 = 0.34$; (b) $f_1 = 0.29$, $f_2 = 0.34$; (c) $f_1 = 0.3$, $f_2 = 0.34$; and (d) $f_1 = 0.31$, $f_2 = 0.34$.

Figure 4(a) shows the magnitude $|X[k]|$ of the DFT samples of the signal $x[n]$ of Eq. (14) for $R = 16$. From the plot of the magnitude $|X(e^{j\omega})|$ of the Fourier transform given in Figure 4(b), it is evident that the DFT samples given in Figure 4(a) are indeed the frequency samples of the frequency response, as expected. As is customary, the horizontal axis in Figure 4(a) has been labeled in terms of the DFT frequency sample (bin) number k, where k is related to the normalized angular frequency ω through Eq. (4). Thus, $\omega = 2\pi \times 8/16 = \pi$ corresponds to $k = 8$, and $\omega = 2\pi \times 15/16 = 1.875\pi$ corresponds to $k = 15$.

From the plot of Figure 4(a), it is difficult to determine whether there is one or more sinusoids in the signal being examined and the exact locations of the sinusoids. To increase the accuracy of the locations of the sinusoids, we increase the size of the DFT to 32 and recompute the DFT, as indicated in Figure 4(c). In this plot, there appear to be some concentrations around $k = 7$ and around $k = 11$ in the normalized frequency range from 0 to 0.5. Figure 4(d) shows the DFT plot obtained for $R = 64$. In this plot, there are two clear peaks occurring at $k = 13$ and $k = 22$ that correspond to the normalized frequencies of 0.2031 and 0.3438, respectively. To improve further the accuracy of the peak location, we compute next a 128-point DFT, as indicated in Figure 4(e), in which the peak occurs around $k = 27$ and $k = 45$, corresponding to the normalized frequencies of 0.2109 and 0.3516, respectively. However, this plot also shows a number of minor peaks, and it is not clear by examining this DFT plot whether additional sinusoids of lesser strengths are present in the original signal or not.

As Example 1 points out, in general, an increase in the DFT length improves the sampling accuracy of the Fourier transform by reducing the spectral separation of adjacent DFT samples.

EXAMPLE 2 Effect of Spectral Separation on the DFT of a Sum of Two Sinusoids

In this example, we compute the DFT of a sum of two finite-length sinusoidal sequences, as given by Eq. (14), with one of the sinusoids at a fixed frequency, while the frequency of the other sinusoid is varied. Specifically, we keep $f_2 = 0.34$ and vary f_1 from 0.28 to 0.31. The length of the signal being analyzed is 16, while the DFT length is 128.

Figure 5 shows the plots of the DFTs computed, along with the frequencies of the sinusoids obtained using Program A-2. As can be seen from these plots, the two sinusoids are clearly resolved in Figures 5(a) and (b), while they cannot be resolved in Figures 5(c) and (d). The reduced resolution occurs when the difference between the two frequencies becomes less than 0.04.

As indicated by Eq. (11), the Fourier transform $\Gamma(e^{j\omega})$ of a length-N sinusoid of normalized angular frequency ω_1 is obtained by frequency translating the Fourier transform $\Psi_R(e^{j\omega})$ of a length-N rectangular window to the frequencies $\pm\omega_1$ and scaling their amplitudes appropriately. In the case of a sum of two length-N sinusoids of normalized angular frequencies ω_1 and ω_2, the Fourier transform is obtained by summing the Fourier transforms of the individual sinusoids. As the difference between the two frequencies becomes smaller, the main lobes of the Fourier transforms of the individual sinusoids get closer and eventually overlap. If there is a significant overlap, it will be difficult to resolve the peaks. It follows therefore that the frequency resolution is essentially determined by the main lobe Δ_{ML} of the Fourier transform of the window.

Now from Table 10.2 of Text, the main lobe width Δ_{ML} of a length-N rectangular window is given by $4\pi/N$. In terms of normalized frequency, the main lobe width of a length-16 rectangular window is 0.125. Hence, two closely spaced sinusoids windowed with a rectangular window of length 16 can be clearly resolved if the difference in their frequencies is about half of the main lobe width, that is, 0.0625.

Even though the rectangular window has the smallest main lobe width, it has the largest relative sidelobe amplitude and, as a consequence, causes considerable leakage. As seen from Examples 1 and 2, the large amount of leakage results in minor peaks that may be falsely identified as sinusoids. We now study the effect of windowing the signal with a Hamming window.[5]

EXAMPLE 3 Minimization of the Leakage Using a Tapered Window

We compute the DFT of a sum of two sinusoids windowed by a Hamming window. The signal being analyzed is $x[n] \cdot w[n]$, where $x[n]$ is given by

$$x[n] = 0.85\sin(2\pi f_1 n) + \sin(2\pi f_2 n),$$

and $w[n]$ is a Hamming window of length N. The two normalized frequencies are $f_1 = 0.22$ and $f_2 = 0.26$.

Figure 6(a) shows the 16-point DFT of the windowed signal with a window length of 16. As can be seen from this plot, the leakage has been reduced considerably, but it is difficult to resolve the two sinusoids. We next increase the DFT length to 64, while keeping the window length fixed at 16. The resulting plot is shown in Figure 6(b), indicating a substantial reduction in the leakage but with no change in the resolution. From Table 10.2, the main lobe width Δ_{ML} of a length-N Hamming window is $8\pi/N$. Thus, for $N = 16$, the normalized main lobe width is 0.25. Hence, with such a window, we can resolve two frequencies if their difference is of the order of half the main lobe width, that is, 0.125 or better. In our example, the difference is 0.04, which is considerably smaller than this value.

In order to increase the resolution, we increase the window length to 32, which reduces the main lobe width by half. Figure 6(c) shows its 32-point DFT. There now appears to be two peaks. Increasing the DFT size to 64 clearly separates the two peaks, as indicated in Figure 6(d). This separation becomes more visible with an increase in the DFT size to 256, as shown in Figure 6(e). Finally, Figure 6(f) shows the result obtained with a window length of 64 and a DFT length of 256.

It is clear from Examples 1 through 3 that performance of the DFT-based spectral analysis depends on several factors, the type of window being used and its length, and the size of the DFT. To improve

[5] For a review of some commonly used windows, see Sections 10.2.4 and 10.2.5 of Text.

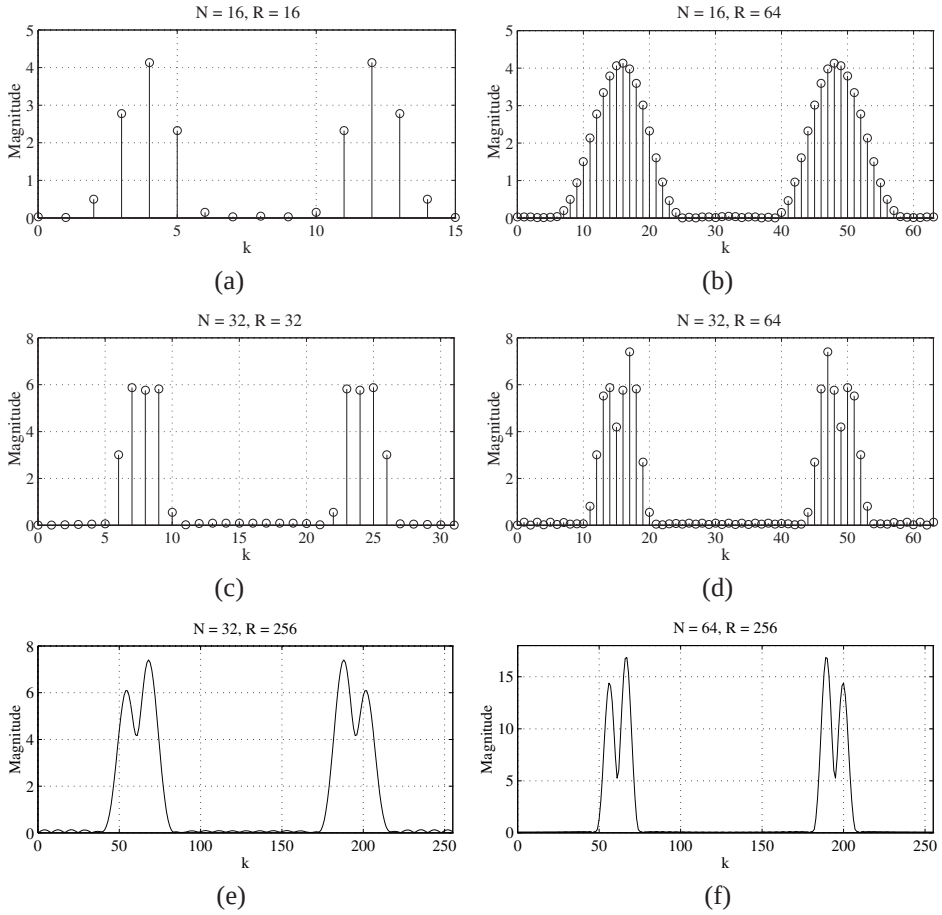


Figure 6: (a)–(f) Spectral analysis using a Hamming window.

the frequency resolution, one must use a window with a very small main lobe width, and to reduce the leakage, the window must have a very small relative sidelobe level. The main lobe width can be reduced by increasing the length of the window. Furthermore, an increase in the accuracy of locating the peaks is achieved by increasing the size of the DFT. To this end, it is preferable to use a DFT length that is a power of 2 so that very efficient FFT algorithms can be employed to compute the DFT. Of course, an increase in the DFT size also increases the computational complexity of the spectral analysis procedure.

3 Analysis of Speech Signals Using the STFT

The short-term Fourier transform described in Section 5.11 of Text is often used in the analysis of speech, since speech signals are generally non-stationary. As indicated in Section 1.3 of Text, the speech signal, generated by the excitation of the vocal tract, is composed of two types of basic waveforms: voiced and unvoiced sounds. A typical speech signal is shown in Figure 1.16 of Text. As can be seen from this figure, a speech segment over a small time interval can be considered as a stationary signal, and as a result, the

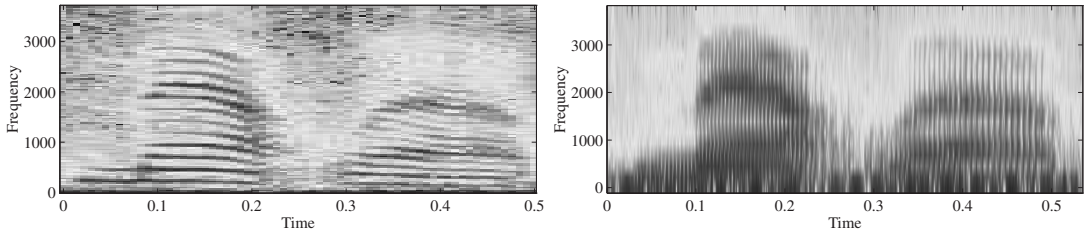


Figure 7: (a) Narrow-band spectrogram and (b) wide-band spectrogram of a speech signal.

DFT of the speech segment can provide a reasonable representation of the frequency domain characteristic of the speech in this time interval.

As in the case of the DFT-based spectral analysis of deterministic signals discussed earlier, in the STFT analysis of non-stationary signals, such as speech, the window also plays an important role. Both the length and shape of the window are critical issues that need to be examined carefully. The function of the window $w[n]$ is to extract a portion of the signal for analysis and ensure that the extracted section of $x[n]$ is approximately stationary. To this end, the window length R should be small, in particular for signals with widely varying spectral parameters. A decrease in the window length increases the time-resolution property of the STFT, whereas the frequency-resolution property of the STFT increases with an increase in the window length. A shorter window thus provides a *wide-band spectrogram*, while a longer window results in a *narrow-band spectrogram*.

A shorter window developing a wide-band spectrogram provides a better time resolution, whereas a longer window developing a narrow-band spectrogram results in an improved frequency resolution. In order to provide a reasonably good estimate of the changes in the vocal tract and the excitation, a wide-band spectrogram is preferable. To this end, the window size is selected to be approximately close to one pitch period, which is adequate for resolving the formants though not adequate to resolve the harmonics of the pitch frequencies. On the other hand, to resolve the harmonics of the pitch frequencies, a narrow-band spectrogram with a window size of several pitch periods is desirable.

The two frequency-domain parameters characterizing the Fourier transform of a window are its main lobe width Δ_{ML} and the relative sidelobe amplitude $A_{s\ell}$. The former parameter determines the ability of the window to resolve two signal components in the vicinity of each other, while the latter controls the degree of leakage of one component into a nearby signal component. It thus follows that in order to obtain a reasonably good estimate of the frequency spectrum of a time-varying signal, the window should be chosen to have a very small relative sidelobe amplitude with a length chosen based on the acceptable accuracy of the frequency and time resolutions.

The following example illustrates the STFT analysis of a speech signal.

EXAMPLE 4 Short-Time Fourier Transform Analysis of a Speech Signal

The `mtlb.mat` file in the *Signal Processing Toolbox* of MATLAB contains a speech signal of duration 4001 samples sampled at 7418 Hz. We compute its STFT using a Hamming window of length 256 with an overlap of 50 samples between consecutive windowed signals using Program 3 in Section 14. Figures 7(b) and (c) show, respectively, a narrow-band spectrogram and a wide-band spectrogram of the speech signal of Figure 7(a). The frequency and time resolution trade-off between the two spectrograms of Figure 7 should be evident.

4 Spectral Analysis of Random Signals

As discussed in Section 2, in the case of a deterministic signal composed of sinusoidal components, a Fourier analysis of the signal can be carried out by taking the discrete Fourier transform (DFT) of a finite-length segment of the signal obtained by appropriate windowing, provided the parameters characterizing the components are time-invariant and independent of the window length. On the other hand, the Fourier analysis of nonstationary signals with time-varying parameters is best carried out using the short-time Fourier transform (STFT) described in Section 3.

Neither the DFT nor the STFT is applicable for the spectral analysis of naturally occurring random signals as here the spectral parameters are also random. These type of signals are usually classified as noiselike random signals such as the unvoiced speech signal generated when a letter such as "/f/" or "/s/" is spoken, and signal-plus-noise random signals, such as seismic signals and nuclear magnetic resonance signals.[6] Spectral analysis of a noiselike random signal is usually carried out by estimating the power density spectrum using Fourier-analysis-based nonparametric methods, whereas a signal-plus-noise random signal is best analyzed using parametric-model-based methods in which the autocovariance sequence is first estimated from the model and then the Fourier transform of the estimate is evaluated. In this section, we review both of these approaches.

4.1 Nonparametric Spectral Analysis

Consider a wide-sense stationary (WSS) random signal $g[n]$ with zero mean. According to the Wiener–Khintchine theorem of Eq. (C.33) in Appendix C of Text, the power spectrum of $g[n]$ is given by

$$\mathcal{P}_{gg}(\omega) = \sum_{\ell=-\infty}^{\infty} \phi_{gg}[\ell] e^{-j\omega\ell}, \tag{15}$$

where $\phi_{gg}[\ell]$ is its autocorrelation sequence, which from Eq. (C.20b) of Appendix C of Text is given by

$$\phi_{gg}[\ell] = E(g[n+\ell]g^*[n]). \tag{16}$$

In Eq. (16), $E(\cdot)$ denotes the expectation operator as defined in Eq. (C.4a) of Appendix C of Text.

Periodogram Analysis

Assume that the infinite-length random discrete-time signal $g[n]$ is windowed by a length-N window sequence $w[n]$, $0 \le n \le N-1$, resulting in the length-N sequence $\gamma[n] = g[n] \cdot w[n]$. The Fourier transform $\Gamma(e^{j\omega})$ of $\gamma[n]$ is given by

$$\Gamma(e^{j\omega}) = \sum_{n=0}^{N-1} \gamma[n] e^{-j\omega n} = \sum_{n=0}^{N-1} g[n] \cdot w[n] e^{-j\omega n}. \tag{17}$$

The estimate $\hat{\mathcal{P}}_{gg}(\omega)$ of the power spectrum $\mathcal{P}_{gg}(\omega)$ is then obtained using

$$\hat{\mathcal{P}}_{gg}(\omega) = \frac{1}{CN} |\Gamma(e^{j\omega})|^2, \tag{18}$$

[6] E.A. Robinson, A historical perspective of spectrum estimation, *Proceedings of the IEEE*, vol. 70, pp. 885-907, 1982.

where the constant C is a normalization factor given by

$$C = \frac{1}{N}\sum_{n=0}^{N-1}|w[n]|^2 \tag{19}$$

and included in Eq. (18) to eliminate any bias in the estimate occurring due to the use of the window $w[n]$. The quantity $\hat{\mathcal{P}}_{gg}(e^{j\omega})$ defined in Eq. (18) is called the *periodogram* when $w[n]$ is a rectangular window and is called a *modified periodogram* for other types of windows.

In practice, the periodogram $\hat{\mathcal{P}}_{gg}(\omega)$ is evaluated at a discrete set of equally spaced R frequencies, $\omega_k = 2\pi k/R$, $0 \le k \le R-1$, by replacing the Fourier transform $\Gamma(e^{j\omega})$ with an R-point DFT $\Gamma[k]$ of the length-N sequence $\gamma[n]$:

$$\hat{\mathcal{P}}_{gg}[k] = \frac{1}{CN}|\Gamma[k]|^2. \tag{20}$$

As in the case of the Fourier analysis of sinusoidal signals discussed earlier, R is usually chosen to be greater than N to provide a finer grid of the samples of the periodogram.

It can be shown that the mean value of the periodogram $\hat{\mathcal{P}}_{gg}(\omega)$ is given by

$$E\left(\hat{\mathcal{P}}_{gg}(\omega)\right) = \frac{1}{2\pi CN}\int_{-\pi}^{\pi}\mathcal{P}_{gg}(\nu)|\Psi(e^{j(\omega-\nu)})|^2\,d\nu, \tag{21}$$

where $\mathcal{P}_{gg}(\omega)$ is the desired power spectrum and $\Psi(e^{j\omega})$ is the Fourier transform of the window sequence $w[n]$. The mean value being nonzero for any finite-length window sequence, the power spectrum estimate given by the periodogram is said to be *biased*. By increasing the window length N, the bias can be reduced.

We illustrate the power spectrum computation in Example 5.

EXAMPLE 5 Power Spectrum of a Noise-Corrupted Sinusoidal Sequence

Let the random signal $g[n]$ be composed of two sinusoidal components of angular frequencies 0.06π and 0.14π radians, corrupted with a Gaussian distributed random signal of zero mean and unity variance, and windowed by a rectangular window of two different lengths: $N = 128$ and 1024. The random signal is generated using the M-file `randn`. Figures 8(a) and (b) show the plots of the estimated power spectrum for the two cases. Ideally the power spectrum should show four peaks at ω equal to 0.06, 0.14, 0.86, and 0.94, respectively, and a flat spectral density at all other frequencies. However, Figure 8(a) shows four large peaks and several other smaller peaks. Moreover, the spectrum shows large amplitude variations throughout the whole frequency range. As N is increased to a much larger value, the peaks get sharper due to increased resolution of the DFT, while the spectrum shows more rapid amplitude variations.

To understand the cause behind the rapid amplitude variations of the computed power spectrum encountered in Example 5, we assume $w[n]$ to be a rectangular window and rewrite the expression for the periodogram given in Eq. (18) using Eq. (17) as

$$\begin{aligned}\hat{\mathcal{P}}_{gg}(\omega) &= \frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} g[m]g^*[n]e^{-j\omega(m-n)} \\ &= \sum_{k=-N+1}^{N-1}\left(\frac{1}{N}\sum_{n=0}^{N-1-|k|} g[n+k]g^*[n]\right)e^{-j\omega k} \\ &= \sum_{k=-N+1}^{N-1}\hat{\phi}_{gg}[k]e^{-j\omega k}.\end{aligned} \tag{22}$$

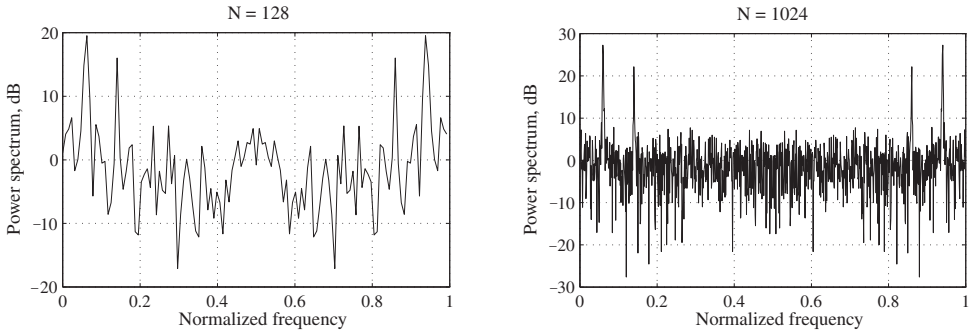


Figure 8: Power spectrum estimate of a signal containing two sinusoidal components corrupted with a white noise sequence of zero mean and unit variance Gaussian distribution: (a) Periodogram with a rectangular window of length $N = 128$ and (b) periodogram with a rectangular window of length $N = 1024$.

Now $\hat{\phi}_{gg}[k]$ is the periodic correlation of $g[n]$ and is an estimate of the true correlation $\phi_{gg}[k]$. Hence, $\hat{\mathcal{P}}_{gg}(\omega)$ is actually the Fourier transform of $\hat{\phi}_{gg}[k]$. A few samples of $g[n]$ are used in the computation of $\hat{\phi}_{gg}[k]$ when k is near N, yielding a poor estimate of the true correlation. This, in turn, results in rapid amplitude variations in the periodogram estimate. A smoother power spectrum estimate can be obtained by the periodogram averaging method discussed next.

Periodogram Averaging

The power spectrum estimation method, originally proposed by Bartlett[7] and later modified by Welch,[8] is based on the computation of the modified periodogram of R overlapping portions of length-N input samples and then averaging these R periodograms. Let the overlap between adjacent segments be K samples. Consider the windowed rth segment of the input data

$$\gamma^{(r)}[n] = g[n + rK]w[n], \qquad 0 \le n \le N-1, \qquad 0 \le r \le R-1, \tag{23}$$

with a Fourier transform given by $\Gamma^{(r)}(e^{j\omega})$. Its periodogram is given by

$$\hat{\mathcal{P}}_{gg}^{(r)}(\omega) = \frac{1}{CN}|\Gamma^{(r)}(e^{j\omega})|^2. \tag{24}$$

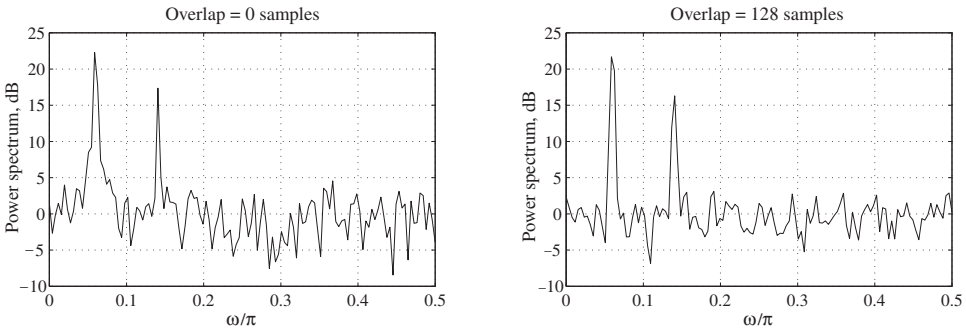
The Welch estimate is then given by the average of all R periodograms $\hat{\mathcal{P}}_{gg}^{(r)}(\omega)$, $0 \le r \le R-1$:

$$\hat{\mathcal{P}}_{gg}^{W}(\omega) = \frac{1}{R}\sum_{r=1}^{R-1}\hat{\mathcal{P}}_{gg}^{(r)}(\omega). \tag{25}$$

It can be shown that the variance of the Welch estimate of Eq. (25) is reduced approximately by a factor R if the R periodogram estimates are assumed to be independent of each other. For a fixed-length input

[7] M.S. Bartlett, Smoothing periodograms from the time series with continuous spectra, *Nature* (*London*), vol. 161, pp. 686-687, 1948.

[8] P.D. Welch, The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms, *IEEE Trans. on Audio and Electroacoustics*, vol. AU-15, pp. 70–73, 1967.



(a) (b)

Figure 9: Power spectrum estimates: (a) Bartlett's method and (b) Welch's method.

sequence, R can be increased by decreasing the window length N which in turn decreases the DFT resolution. On the other hand, an increase in the resolution is obtained by increasing N. Thus, there is a trade-off between resolution and the bias.

It should be noted that if the data sequence is segmented by a rectangular window into contiguous segments with no overlap, the periodiogram estimate given by Eq. (25) reduces to Barlett estimate.

Periodogram Estimate Computation Using MATLAB

The *Signal Processing Toolbox* of MATLAB includes the M-file `psd` for modified periodogram estimate computation. It is available with several options. We illustrate its use in Example 6.

EXAMPLE 6 Estimation of the Power Spectrum of a Noise-Corrupted Sinusoidal Sequence

We consider here the evaluation of the Bartlett and Welch estimates of the power spectrum of the random signal considered in Example 6. To this end, Program 4 in Section 14 can be used. This program is run first with no overlap and with a rectangular window generated using the function `boxcar`. The power spectrum computed by the above program is then the Bartlett estimate, as indicated in Figure 9(a). It is then run with an overlap of 128 samples and a Hamming window. The corresponding power spectrum is then the Welch estimate, as shown in Figure 9(b). It should be noted from Figure 9 that the Welch periodogram estimate is much smoother than the Bartlett periodogram estimate, as expected. Compared to the power spectrums of Figure 8, there is a decrease in the variance in the smoothed power spectrums of Figure 9, but the latter are still biased. Because of the overlap between adjacent data segments, Welch's estimate has a smaller variance than the others. It should be noted that both periodograms of Figure 9 show clearly two distinct peaks at 0.06 and 0.14.

4.2 Parametric Model-Based Spectral Analysis

In the model-based method, a causal LTI discrete-time system with a transfer function

$$H(z) = \sum_{n=0}^{\infty} h[n]z^{-n} = \frac{P(z)}{D(z)} = \frac{\sum_{k=0}^{L} p_k z^{-k}}{1 + \sum_{k=1}^{M} d_k z^{-k}} \tag{26}$$

is first developed, whose output, when excited by a white noise sequence $e[n]$ with zero mean and variance σ_e^2, matches the specified data sequence $g[n]$. An advantage of the model-based approach is that it can extrapolate a short-length data sequence to create a longer data sequence for improved power spectrum estimation. On the other hand, in nonparametric methods, spectral leakages limit the frequency resolution if the data length is short.

The model of Eq. (26) is called an *autoregressive moving-average* (ARMA) process of order (L, M) if $P(z) \neq 1$, an *all-pole* or *autoregressive* (AR) process of order M if $P(z) = 1$, and an all-zero or *moving-average* (MA) process of order L if $D(z) = 1$. For an ARMA or an AR model, for stability, the denominator $D(z)$ must have all its zeros inside the unit circle. In the time domain, the input–output relation of the model is given by

$$g[n] = -\sum_{k=1}^{M} d_k g[n-k] + \sum_{k=0}^{L} p_k e[n-k]. \tag{27}$$

As indicated in Section C.8 of Appendix C of Text, the output $g[n]$ of the model is a WSS random signal. From Eq. (C.85) of Appendix C of Text, it follows that the power spectrum $\mathcal{P}_{gg}(\omega)$ of $g[n]$ can be expressed as

$$\mathcal{P}_{gg}(\omega) = \sigma_e^2 |H(e^{j\omega})|^2 = \sigma_e^2 \frac{|P(e^{j\omega})|^2}{|D(e^{j\omega})|^2}, \tag{28}$$

where $H(e^{j\omega}) = P(e^{j\omega})/D(e^{j\omega})$ is the frequency response of the model, and

$$P(e^{j\omega}) = \sum_{k=0}^{L} p_k e^{-j\omega k}, \qquad D(e^{j\omega}) = 1 + \sum_{k=1}^{M} d_k e^{-j\omega k}.$$

In the case of an AR or an MA model, the power spectrum is thus given by

$$\mathcal{P}_{gg}(\omega) = \begin{cases} \sigma_e^2 |P(e^{j\omega})|^2, & \text{for an MA model,} \\ \frac{\sigma_e^2}{|D(e^{j\omega})|^2}, & \text{for an AR model.} \end{cases} \tag{29}$$

The spectral analysis is carried out by first determining the model and then computing the power spectrum using either Eq. (28) for an ARMA model or using Eq. (29) for an MA or an AR model. To determine the model, we need to decide the type of the model (i.e., pole-zero IIR structure, all-pole IIR structure, or all-zero FIR structure) to be used; determine an appropriate order of its transfer function $H(z)$ (i.e., both L and M for an ARMA model or M for an AR model or L for an MA model); and then, from the specified length-N data $g[n]$, estimate the coefficients of $H(z)$. We restrict our discussion here to the development of the AR model, as it is simpler and often used. Applications of the AR model include spectral analysis, system identification, speech analysis and compression, and filter design.[9]

Relation Between Model Parameters and the Autocorrelation Sequence

The model filter coefficients $\{p_k\}$ and $\{d_k\}$ are related to the autocorrelation sequence $\phi_{gg}[\ell]$ of the random signal $g[n]$. To establish this relation, we obtain from Eq. (27),

$$\phi_{gg}[\ell] = -\sum_{k=1}^{M} d_k \phi_{gg}[\ell-k] + \sum_{k=0}^{L} p_k \phi_{eg}[\ell-k], \qquad -\infty < \ell < \infty, \tag{30}$$

[9] For a discussion on the development of the MA model and the ARMA model, see R. Kumaresan, Spectral analysis, In S.K. Mitra and J.F. Kaiser, editors, *Handbook for Digital Signal Processing*, chapter 16, pages 1143–1242. Wiley-Interscience, New York NY, 1993.

by multiplying both sides of the equation with $g^*[n-\ell]$ and taking the expected values. In Eq. (30), the cross-correlation $\phi_{eg}[\ell]$ between $g[n]$ and $e[n]$ can be written as

$$\begin{aligned}\phi_{eg}[\ell] &= E(g^*[n]e[n+\ell]) \\ &= \sum_{k=0}^{\infty} h^*[k]\ E(e^*[n-k]e[n+\ell]) = \sigma_e^2 h^*[-\ell],\end{aligned} \tag{31}$$

where $h[n]$ is the causal impulse response of the LTI model as defined in Eq. (26) and σ_e^2 is the variance of the white noise sequence $e[n]$ applied to the input of the model.

For an AR model, $L = 0$, and hence Eq. (30) reduces to

$$\phi_{eg}[\ell] = \begin{cases} -\sum_{k=1}^{M} d_k \phi_{gg}[\ell-k], & \text{for } \ell > 0, \\ -\sum_{k=1}^{M} d_k \phi_{gg}[\ell-k] + \sigma_e^2, & \text{for } \ell = 0, \\ \phi_{gg}^*[-\ell], & \text{for } \ell < 0. \end{cases} \tag{32}$$

From Eq. (32), we obtain for $1 \le \ell \le M$, a set of M equations,

$$\sum_{k=1}^{M} d_k \phi_{gg}[\ell-k] = -\phi_{eg}[\ell], \qquad 1 \le \ell \le M, \tag{33}$$

which can be written in matrix form as

$$\begin{bmatrix} \phi_{gg}[0] & \phi_{gg}[-1] & \cdots & \phi_{gg}[-M+1] \\ \phi_{gg}[1] & \phi_{gg}[0] & \cdots & \phi_{gg}[-M+2] \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{gg}[M-1] & \phi_{gg}[M-2] & \cdots & \phi_{gg}[0] \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_M \end{bmatrix} = - \begin{bmatrix} \phi_{gg}[1] \\ \phi_{gg}[2] \\ \vdots \\ \phi_{gg}[M] \end{bmatrix}. \tag{34}$$

For $\ell = 0$, we also get from Eq. (32)

$$\phi_{gg}[0] + \sum_{k=1}^{M} d_k \phi_{gg}[-k] = \sigma_e^2. \tag{35}$$

Combining Eq. (35) with Eq. (34) we arrive at

$$\begin{bmatrix} \phi_{gg}[0] & \phi_{gg}[-1] & \cdots & \phi_{gg}[-M] \\ \phi_{gg}[1] & \phi_{gg}[0] & \cdots & \phi_{gg}[-M+1] \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{gg}[M] & \phi_{gg}[M-1] & \cdots & \phi_{gg}[0] \end{bmatrix} \begin{bmatrix} 1 \\ d_1 \\ d_2 \\ \vdots \\ d_M \end{bmatrix} = \begin{bmatrix} \sigma_e^2 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{36}$$

The matrix equation of Eq. (36) is more commonly known as the *Yule–Walker equation*. It can be seen from Eq. (36) that knowing the $M+1$ autocorrelation samples $\phi_{xx}[\ell]$ for $0 \le \ell \le M$, we can determine the model parameters d_k for $1 \le k \le M$ by solving the matrix equation. The $(M+1) \times (M+1)$ matrix in Eq. (36) is a *Toeplitz matrix*.[10]

[10] A Toeplitz matrix has the same element values along each negative-sloping diagonal.

Because of the structure of the Toeplitz matrix, the matrix equation of Eq. (36) can be solved using the fast *Levinson–Durbin algorithm*.[11,12] This algorithm develops the AR model recursively. Let the filter coefficients at the ith iteration be denoted by $d_k^{(i)}, 0 \leq k \leq i$. Define two other parameters for the ith stage, the *reflection coefficient* K_i and the *prediction error* $\mathcal{E}_i$. The recursion algorithm consists of the following steps:

Step 1: Start the recursion with:

$$K_1 = d_1^{(1)} = \phi_{gg}[1]/\phi_{gg}[0], \qquad \mathcal{E}_1 = \phi_{gg}[0](1 - |K_1|^2).$$

Step 2: For $i > 0$, evaluate the $(i + 1)$th reflection coefficient using

$$K_{i+1} = d_{i+1}^{(i+1)} = \frac{\phi_{gg}[i + 1] + \sum_{r=1}^{i} d_r^{(i)} \phi_{gg}[i + 1 - r]}{\mathcal{E}_i}.$$

Step 3: For $i > 0$, evaluate the rth filter coefficient of the $(i + 1)$-th order model with $r \leq i$ using:

$$d_r^{(i+1)} = d_r^{(i)} + K_{r+1}(d_{i+1-r}^{(i)})^*.$$

Step 4: Determine the $(i + 1)$th prediction error using:

$$\mathcal{E}_{i+1} = \mathcal{E}_i(1 - |K_i|^2).$$

Step 5: If $i + 1 = M$ stop the iteration, otherwise go back to Step 2.

The causal all-pole LTI system $H(z) = 1/D(z)$ resulting from the application of the Levinson–Durbin recursions is guaranteed to be BIBO stable. Moreover, the recursion automatically leads to a realization in the form of a cascaded FIR lattice structure, as shown in Figure 8.40.

Power Spectrum Estimation Using an AR Model

The AR model parameters can be determined using the *Yule–Walker method*, which makes use of the estimates of the autocorrelation sequence samples, as their actual values are not known a priori. The autocorrelation at lag ℓ is determined from the specified data samples $g[n]$ for $0 \leq n \leq N - 1$ using

$$\hat{\phi}_{gg}[\ell] = \frac{1}{N} \sum_{n=0}^{N-1-|\ell|} g^*[n]\, g[n + \ell], \qquad 0 \leq \ell \leq N - 1. \tag{37}$$

[11] N. Levinson, The Wiener RMS criterion in filter design and prediction, *J. Math. Phys.*, vol. 25, pp. 261–278, 1947.

[12] J. Durbin, Efficient estimation of parameters in moving average model, *Biometrika*, vol. 46, pp. 306–316, 1959.

The above estimates are used in Eq. (34) in place of the true autocorrelation samples, with the AR model parameters d_k replaced with their estimates $\hat{d}_k$. The resulting equation is next solved using the Levinson–Durbin algorithm to determine the estimates of the AR model parameters $\hat{d}_k$. The power spectrum estimate is then evaluated using

$$\hat{\mathcal{P}}_{gg}(\omega) = \frac{\hat{\mathcal{E}}_M}{\left|1 + \sum_{k=1}^{M} \hat{d}_k \, e^{-j\omega k}\right|^2}, \tag{38}$$

where $\hat{\mathcal{E}}_M$ is the prediction error for the Mth-order AR model:

$$\hat{\mathcal{E}}_M = \hat{\phi}_{gg}[0] \prod_{i=1}^{M} \left(1 - |\hat{K}_i|^2\right). \tag{39}$$

The Yule–Walker method is related to the linear prediction problem. Here the problem is to predict the N-th sample $g[N]$ from the previous M data samples $g[n], 0 \le n \le M-1$, with the assumption that data samples outside this range are zeros. The predicted value $\hat{g}[n]$ of the data sample $g[n]$ can be found by a linear combination of the previous M data samples as

$$\begin{aligned} \hat{g}[n] &= -\sum_{k=1}^{M} \hat{d}_k g[n-k] \\ &= g[n] - e[n], \end{aligned} \tag{40}$$

where $e[n]$ is the prediction error. For the specified data sequence, Eq. (40) leads to $N + M$ prediction equations given by

$$g[n] + \sum_{k=1}^{M} g[n-k]\hat{d}_k = e[n], \qquad 0 \le n \le N + M - 1. \tag{41}$$

The optimum linear predictor coefficients $\hat{d}_k$ are obtained by minimizing the error

$$\frac{1}{N} \sum_{n=0}^{N+M-1} |e[n]|^2.$$

It can be shown that the solution of the minimization problem is given by Eq. (34). Thus, the best all-pole linear predictor filter is also the AR model resulting from the solution of Eq. (34).

It should be noted that the AR model is guaranteed stable. But the all-pole filter developed may not model an AR process exactly of the same order due to the windowing of the data sequence to a finite length, with samples outside the window range assumed to be zeros.

The function `lpc` in MATLAB finds the AR model using the above method.

EXAMPLE 7 Development of an AR Model of an FIR Filter

We consider the approximation of an FIR digital filter of order 13 with an all-pole IIR digital filter of order 7. The coefficients of the FIR filter are obtained using the function `firpm`, and the all-pole IIR filter is designed using the function `lpc`. Program 5 in Section 14 can be used for the design. The magnitude response plots generated by running this program are shown in Figure 10.

Several comments are in order here. First, the linear predictor coefficients $\{d_i\}$ match the power spectral densities of the all-pole model with that of the sequence $\{g_i\}$. Since, the sequence of the FIR filter

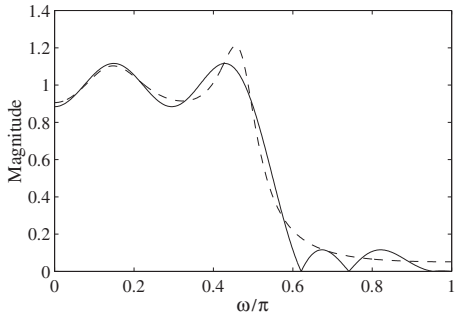


Figure 10: Magnitude response of the FIR filter (shown with solid line) and the all-pole IIR model (shown with dashed line).

coefficients $\{b_i\}$ is not a power signal, and to convert the energy spectrum of the sequence $\{b_i\}$ to a power spectrum, the sequence $\{b_i\}$ needs to be divided by its length N. Hence, to approximate the power spectrum density of the sequence $\{b_i\}$ with that of the AR model, we need to scale the ARMA filter transfer function with the factor $\sqrt{NE}$, where E is the variance of the prediction error. Second, it can be seen from this figure that the AR model has reasonably matched the passband response, with peaks occurring very close to the peaks of the magnitude response of the FIR system. However, there are no nulls in the stopband response of the AR model even though the stopband response of the FIR system has nulls. Since the nulls are generated by the zeros of the transfer function, an all-pole model cannot produce nulls.

In order to apply the above method to power spectrum estimation, it is necessary to estimate first the model order M. A number of formulae have been advanced for order estimation.[13] Unfortunately, none of the these formulae yields a really good estimate of the true model order in many applications.

5 Musical Sound Processing

Recall from our discussion in Section 1.4.1 that almost all musical programs are produced in basically two stages. First, sound from each individual instrument is recorded in an acoustically inert studio on a single track of a multitrack tape recorder. Then, the signals from each track are manipulated by the sound engineer to add special audio effects and are combined in a mix-down system to finally generate the stereo recording on a two-track tape recorder.[14, 15] The audio effects are artificially generated using various signal processing circuits and devices, and they are increasingly being performed using digital signal processing techniques.[16]

Some of the special audio effects that can be implemented digitally are reviewed in this section.

[13] R. Kumaresan, Spectral analysis, In S.K. Mitra and J.F. Kaiser, editors, *Handbook for Digital Signal Processing*, chapter 16, pages 1143–1242. Wiley-Interscience, New York NY, 1993.

[14] B. Blesser and J.M. Kates, Digital processing in audio signals, In A.V. Oppenheim, editor, *Applications of Digital Signal Processing*, chapter 2. Prentice Hall, Englewood Cliffs NJ, 1978.

[15] J.M. Eargle, *Handbook of Recording Engineering*, Van Nostrand Reinhold, New York NY, 1986.

[16] S.J. Orfanidis, *Introduction to Signal Processing*, Prentice Hall, Englewood Cliffs NJ, 1996.

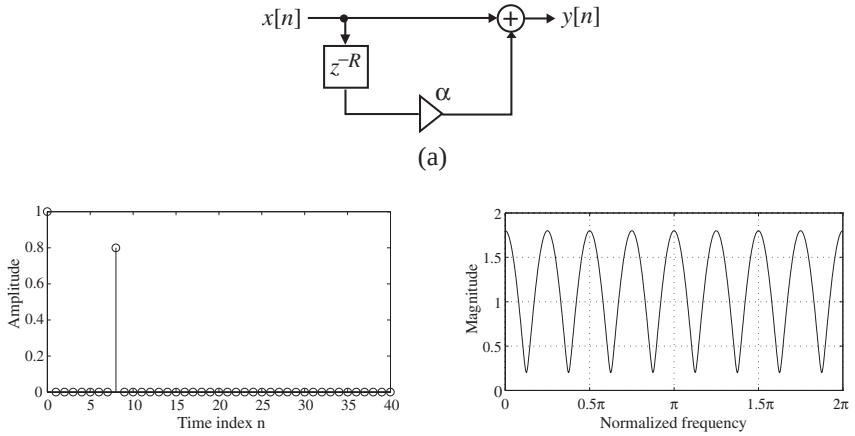


Figure 11: Single echo filter: (a) filter structure, (b) typical impulse response, and (c) magnitude response for $R = 8$ and $\alpha = 0.8$.

5.1 Time-Domain Operations

Commonly used time-domain operations carried on musical sound signals are echo generation, reverberation, flanging, chorus generation, and phasing. In each of these operations, the basic building block is a delay.

Single Echo Filter

Echoes are simply generated by delay units. For example, the direct sound and a single echo appearing R sampling periods later can be simply generated by the FIR filter of Figure 11(a), which is characterized by the difference equation

$$y[n] = x[n] + \alpha x[n - R], \qquad |\alpha| < 1, \tag{42}$$

or, equivalently, by the transfer function

$$H(z) = 1 + \alpha z^{-R}. \tag{43}$$

In Eqs. (42) and (43), the delay parameter R denotes the time the sound wave takes to travel from the sound source to the listener after bouncing back from the reflecting wall, whereas the parameter α, with $|\alpha| < 1$, represents the signal loss caused by propagation and reflection.

The impulse response of the single echo filter is sketched in Figure 11(b). The magnitude response of a single echo FIR filter for $\alpha = 0.8$ and $R = 8$ is shown in Figure 11(c). The magnitude response exhibits R peaks and R dips in the range $0 \le \omega < 2\pi$, with the peaks occurring at $\omega = 2\pi k/R$ and the dips occurring at $\omega = (2k + 1)\pi/R$, $k = 0, 1, \ldots, R - 1$. Because of the comb-like shape of the magnitude response, such a filter is also known as a *comb filter*. The maximum and minimum values of the magnitude response are given by $1 + \alpha = 1.8$ and $1 - \alpha = 0.2$, respectively.

Program A-6[17] can be used to investigate the effect of a single echo on the speech signal shown in Figure 1.16 of Text.

[17] Reproduced with permission of Prof. Dale Callahan, University of Alabama, Birmingham, AL.

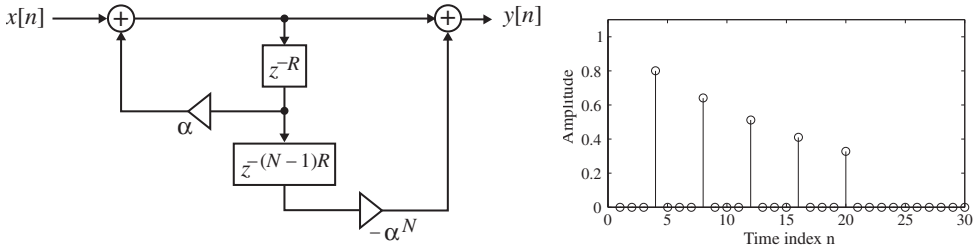


Figure 12: Multiple echo filter generating $N-1$ echoes: (a) filter structure and (b) impulse response with $\alpha = 0.8$ for $N = 6$ and $R = 4$.

Multiple Echo Filter

To generate a fixed number of multiple echoes spaced R sampling periods apart with exponentially decaying amplitudes, one can use an FIR filter with a transfer function of the form

$$H(z) = 1 + \alpha z^{-R} + \alpha^2 z^{-2R} + \cdots + \alpha^{N-1} z^{-(N-1)R} = \frac{1-\alpha^N z^{-NR}}{1-\alpha z^{-R}}. \tag{44}$$

An IIR realization of this filter is sketched in Figure 12(a). The impulse response of a multiple echo filter with $\alpha = 0.8$ for $N = 6$ and $R = 4$ is shown in Figure 12(b).

An infinite number of echoes spaced R sampling periods apart with exponentially decaying amplitudes can be created by an IIR filter with a transfer function of the form

$$\begin{aligned} H(z) &= 1 + \alpha z^{-R} + \alpha^2 z^{-2R} + \alpha^3 z^{-3R} + \cdots \\ &= \frac{1}{1-\alpha z^{-R}}, \qquad |\alpha| < 1. \end{aligned} \tag{45}$$

Figure 13(a) shows one possible realization of the above IIR filter whose first 61 impulse response samples for $R = 4$ are indicated in Figure 13(b). The magnitude response of this IIR filter for $R = 7$ is sketched in Figure 13(c). The magnitude response exhibits R peaks and R dips in the range $0 \le \omega < 2\pi$, with the peaks occurring at $\omega = 2\pi k/R$ and the dips occurring at $\omega = (2k+1)\pi/R$, $k = 0, 1, \ldots, R-1$. The maximum and minimum values of the magnitude response are given by $1/(1-\alpha) = 5$ and $1/(1+\alpha) = 0.5556$, respectively.

The *fundamental repetition frequency* of the IIR multiple echo filter of Eq. (45) is given by $F_R = F_T/R$ Hz, or $\omega_R = 2\pi/R$ radians. In practice, the repetition frequency F_R is often locked to the fundamental frequency of an accompanying musical instrument, such as the drum beat. For a specified F_R, the delay parameter R can be determined from $R = F_R/F_T$, resulting in a time delay of $RT = R/F_T$ seconds.[16]

Program 7[18] can be used to investigate the effect of multiple echos on the speech signal shown in Figure 1.16 of Text.

Reverberation

As indicated in Section 1.4.1, the sound reaching the listener in a closed space, such as a concert hall, consists of several components: direct sound, early reflections, and reverberation. The early reflections

[18]Reproduced with permission of Prof. Dale Callahan, University of Alabama, Birmingham, AL.

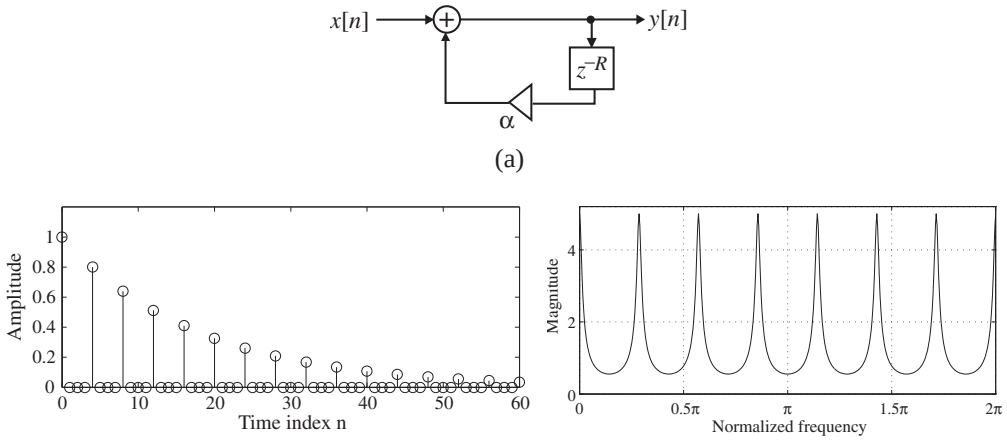


Figure 13: IIR filter generating an infinite number of echoes: (a) filter structure, (b) impulse response with $\alpha = 0.8$ for $R = 4$, and (c) magnitude response with $\alpha = 0.8$ for $R = 7$.

are composed of several closely spaced echoes that are basically delayed and attenuated copies of the direct sound, whereas the reverberation is composed of densely packed echoes. The sound recorded in an inert studio is different from that recorded inside a closed space, and, as a result, the former does not sound "natural" to a listener. However, digital filtering can be employed to convert the sound recorded in an inert studio into a natural-sounding one by artificially creating the echoes and adding them to the original signal.

The IIR comb filter of Figure 13(a) by itself does not provide natural-sounding reverberations for two reasons.[19] First, as can be seen from Figure 13(c), its magnitude response is not constant for all frequencies, resulting in a "coloration" of many musical sounds that are often unpleasant for listening purposes. Second, the output echo density, given by the number of echoes per second, generated by a unit impulse at the input, is much lower than that observed in a real room, thus causing a "fluttering" of the composite sound. It has been observed that approximately 1000 echoes per second are necessary to create a reverberation that sounds free of flutter.[19] To develop a more realistic reverberation, a reverberator with an allpass structure, as indicated in Figure 13(a), has been proposed.[19] Its transfer function is given by

$$H(z) = \frac{\alpha + z^{-R}}{1 + \alpha z^{-R}}, \qquad |\alpha| < 1. \tag{46}$$

In the steady state, the spectral balance of the sound signal remains unchanged due to the unity magnitude response of the allpass reverberator.

Program A-8[20] can be used to investigate the effect of an allpass reverberator on the speech signal shown in Figure 1.16.

The IIR comb filter of Figure 13(a) and the allpass reverberator of Figure 14(a) are basic reverberator units that are suitably interconnected to develop a natural-sounding reverberation. Figure 15 shows one such interconnection composed of a parallel connection of four IIR echo generators in cascade with two

[19] M.R. Schroeder, Natural sounding artificial reverberation, *Journal of the Audio Engineering Society*, vol. 10, pp. 219–223, 1962

[20] Reproduced with permission of Prof. Dale Callahan, University of Alabama, Birmingham, AL.

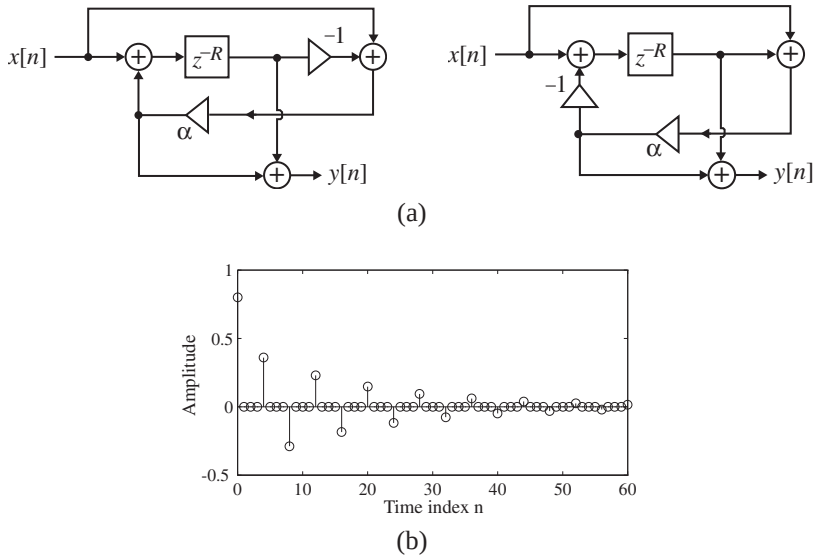


Figure 14: Allpass reverberator: (a) block diagram representation and (b) impulse response with $\alpha = 0.8$ for $R = 4$.

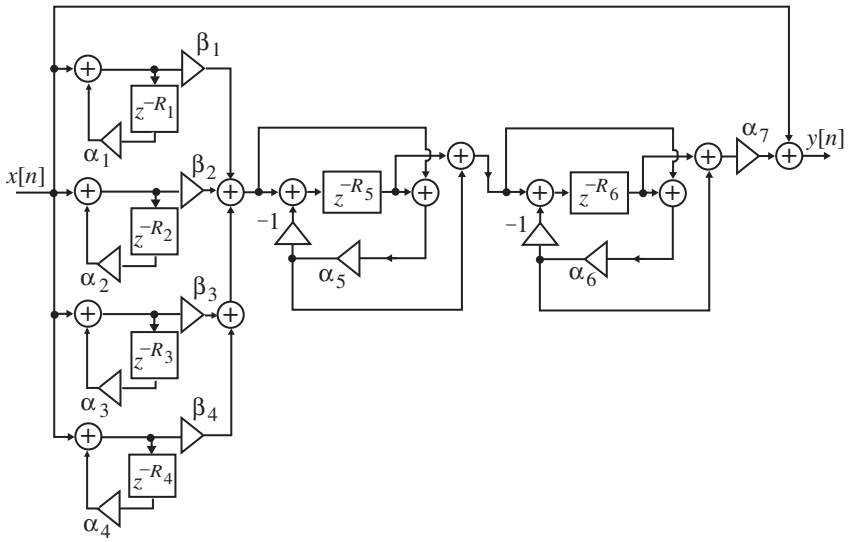


Figure 15: A proposed natural-sounding reverberator scheme.

allpass reverberators.[19] By choosing different values for the delays in each section (obtained by adjusting R_i) and the multiplier constants α_i, it is possible to arrive at a pleasant-sounding reverberation, duplicating that occurring in a specific closed space, such as a concert hall.

Program A-9[21] can be used to investigate the effect of the above natural-sounding reverberator on the speech signal shown in Figure 1.16.

[21] Reproduced with permission of Prof. Dale Callahan, University of Alabama, Birmingham, AL.

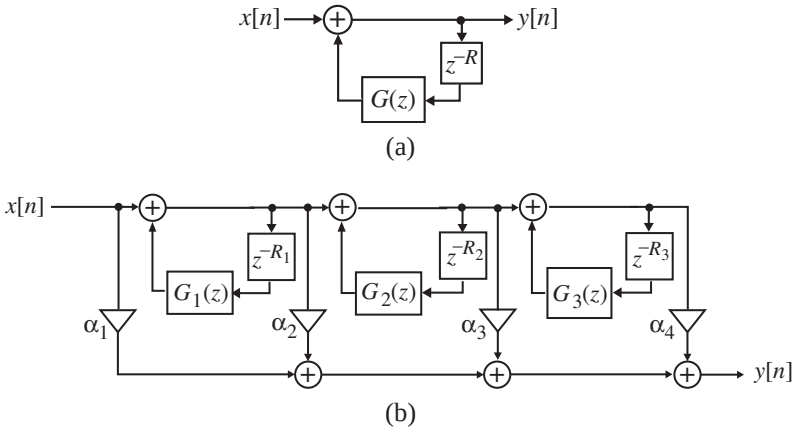


Figure 16: (a) Lowpass reverberator and (b) a multitap reverberator structure.

An interesting modification of the basic IIR comb filter of Figure 13(a) is obtained by replacing the multiplier α with a lowpass FIR or IIR filter $G(z)$, as indicated in Figure 16(a). It has a transfer function given by

$$H(z) = \frac{1}{1 - z^{-R}G(z)}, \tag{47}$$

obtained by replacing α in Eq. (45) with $G(z)$. This structure has been referred to as the *teeth filter* and has been introduced to provide a natural tonal character to the artificial reverberation generated by it.[22] This type of reverberator should be carefully designed to avoid the stability problem. To provide a reverberation with a higher echo density, the teeth filter has been used as a basic unit in a more complex structure such as that indicated in Figure 14(b).

Additional details concerning these and other such composite reverberator structures can be found in literature.[19,23]

Flanging

There are a number of special sound effects that are often used in the mix-down process. One such effect is called *flanging*. Originally, it was created by feeding the same musical piece to two tape recorders and then combining their delayed outputs while varying the difference Δt between their delay times. One way of varying Δt is to slow down one of the tape recorders by placing the operator's thumb on the flange of the feed reel, which led to the name flanging.[15] The FIR comb filter of Figure 11(a) can be modified to create the flanging effect. In this case, the unit generating the delay of R samples, or equivalently, a delay of RT seconds, where T is the sampling period, is made a time-varying delay $\beta(n)$, as indicated in Figure 17. The corresponding input–output relation is then given by

$$y[n] = x[n] + \alpha x[n - \beta(n)]. \tag{48}$$

[22] L.D.J. Eggermont and P.J. Berkhout, Digital audio circuits: Computer simulations and listening tests, *Philips Technical Review*, vol. 41, No. 3, pp. 99–103, 1983/84.

[23] J.A. Moorer, About this reverberation business, *Computer Music Journal*, vol. 3, No. 2, pp. 13–28, 1979.

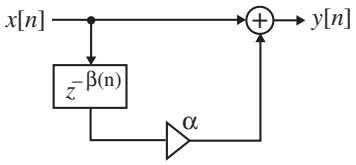


Figure 17: Generation of a flanging effect.

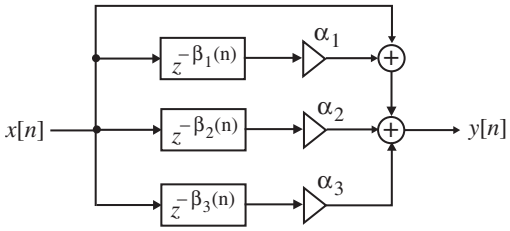


Figure 18: Generation of a chorus effect.

Periodically varying the delay $\beta(n)$ between 0 and R with a low frequency ω_o such as

$$\beta(n) = \frac{R}{2}\,(1 - \cos(\omega_o n)) \tag{49}$$

generates a flanging effect on the sound. It should be noted that, as the value of $\beta(n)$ at an instant n, in general, has a non-integer value, in an actual implementation, the output sample value $y[n]$ should be computed using some type of interpolation method such as that outlined in Section 13.5 of Text.

Program A-10[24] can be used to investigate the effect of flanging on the musical sound signal dt.wav.

Chorus Generator

The *chorus* effect is achieved when several musicians are playing the same musical piece at the same time but with small changes in the amplitudes and small timing differences between their sounds. Such an effect can also be created synthetically by a *chorus generator* from the music of a single musician. A simple modification of the digital filter of Figure 17 leads to a structure that can be employed to simulate this sound effect. For example, the structure of Figure 18 can effectively create a chorus of four musicians from the music of a single musician. To achieve this effect, the delays $\beta_i(n)$ are randomly varied with very slow variations.

The *phasing* effect is produced by processing the signal through a narrowband notch filter with variable notch characteristics and adding a scaled portion of the notch filter output to the original signal, as indicated in Figure 19.[16] The phase of the signal at the notch filter output can dramatically alter the phase of the combined signal, particularly around the notch frequency when it is varied slowly. The tunable notch filter can be implemented using the technique described in Section 8.7.2 of Text. The notch filter in Figure 19 can be replaced with a cascade of tunable notch filters to provide an effect similar to flanging. However, in flanging, the swept notch frequencies are always equally spaced, whereas in phasing, the locations of the notch frequencies and their corresponding 3-dB bandwidths are varied independently.

[24]Reproduced with permission of Prof. Dale Callahan, University of Alabama, Birmingham, AL.

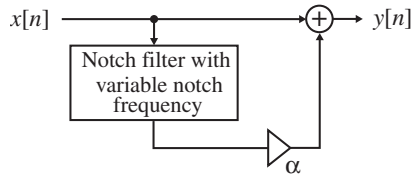


Figure 19: Generation of the phasing effect.

5.2 Frequency-Domain Operations

The frequency responses of individually recorded instruments or musical sounds of performers are frequently modified by the sound engineer during the mix-down process. These effects are achieved by passing the original signals through an equalizer, briefly reviewed in Section 1.4.1 of Text. The purpose of the equalizer is to provide "presence" by peaking the midfrequency components in the range of 1.5 to 3 kHz and to modify the bass–treble relationships by providing "boost" or "cut" to components outside this range. It is usually formed from a cascade of first-order and second-order filters with adjustable frequency responses. Many of the low-order digital filters employed for implementing these functions have been obtained by applying the bilinear transformation to analog filters. We first review the analog filters and then develop their digital equivalents. In addition, we describe some new structures with more flexible frequency responses.

Analog Filters

Simple lowpass and highpass analog filters with a Butterworth magnitude response are usually employed in analog mixers. The transfer functions of first-order analog lowpass and highpass Butterworth filters were given in Eq. (9.22) and (9.27) of Text, respectively. The transfer functions of higher-order lowpass and highpass analog Butterworth filters can be derived using the method outlined in Section A.2 of Appendix A in Text. Also used in analog mixers are second-order analog bandpass and bandstop filters whose transfer functions were given in Eq. (9.29) and Eq. (9.34) of Text, respectively.

A first-order lowpass analog shelving filter for boost has a transfer function given by[25]

$$H_{LP}^{(B)}(s) = \frac{s + K\Omega_c}{s + \Omega_c}, \qquad K > 1. \tag{50}$$

It follows from Eq. (50) that

$$H_{LP}^{(B)}(0) = K, \qquad H_{LP}^{(B)}(\infty) = 1. \tag{51}$$

The transfer function $H_{LP}^{(B)}(s)$ of Eq. (50) can also be used for cut if $K < 1$. However, in this case, $H_{LP}^{(B)}(s)$ has a magnitude response that is not symmetrical to that for the case of $K > 1$ (boost) with respect to the frequency axis without changing the cutoff frequency.[25] The first-order lowpass analog

[25] P.A. Regalia and S.K. Mitra, Tunable digital frequency response equalization filters, *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. ASSP-35, pp. 118–120, January 1987

shelving filter providing cut that retains the cutoff frequency has a transfer function given by[26]

$$H_{LP}^{(C)}(s) = \frac{s + \Omega_c}{s + \Omega_c/K}, \qquad K < 1, \tag{52}$$

for which

$$H_{LP}^{(C)}(0) = K, \qquad H_{LP}^{(C)}(\infty) = 1. \tag{53}$$

The first-order highpass analog shelving filter $H_{HP}^{(B)}(s)$ for the boost and cut can be derived by applying a lowpass-to-highpass transformation to the transfer functions of Eqs. (50) and (52), respectively. The transfer function for boost is given by

$$H_{HP}^{(B)}(s) = \frac{Ks + \Omega_c}{s + \Omega_c}, \qquad K > 1, \tag{54}$$

for which

$$H_{HP}^{(B)}(0) = 1, \qquad H_{HP}^{(B)}(\infty) = K. \tag{55}$$

Likewise, the transfer function for cut is given by

$$H_{HP}^{(C)}(s) = K\left(\frac{s + \Omega_c}{s + K\Omega_c}\right), \qquad K < 1, \tag{56}$$

for which

$$H_{HP}^{(C)}(0) = 1, \qquad H_{HP}^{(C)}(\infty) = K. \tag{57}$$

The *peak filter* is used for boost or cut at a finite frequency Ω_o. The transfer function of an analog second-order peak filter is given by

$$H_{BP}^{(BC)}(s) = \frac{s^2 + KBs + \Omega_o^2}{s^2 + Bs + \Omega_o^2}, \tag{58}$$

for which the maximum (minimum) value of the magnitude response, determined by K, occurs at the center frequency Ω_o. The above peak filter operates as a bandpass filter for $K > 1$ and as a bandstop filter for $K < 1$. The 3-dB bandwidth of the passband for a bandpass response and the 3-dB bandwidth of the stopband for a bandstop response is given by $B = \Omega_o/Q_o$.

First-Order Digital Filters and Equalizers

The analog filters can be converted into their digital equivalents by applying the Type 1 bilinear transformation of Eq. (9.14) of Text to their corresponding transfer functions. The design of first-order Butterworth digital lowpass and highpass filters derived via bilinear transformation of corresponding analog transfer functions has been treated in Section 9.2.2 of Text. The relevant transfer functions are given in Eqs. (9.24) and (9.28), respectively, of Text.

The transfer functions of the first-order digital lowpass and highpass filters given by Eqs. (9.24) and (9.28) can be alternatively expressed as

$$G_{LP}(z) = \tfrac{1}{2}\{1 - \mathcal{A}_1(z)\}, \tag{59a}$$

$$G_{HP}(z) = \tfrac{1}{2}\{1 + \mathcal{A}_1(z)\}, \tag{59b}$$

[26] U. Zölzer, *Digital Audio Signal Processing*, Wiley, New York NY, 1997.

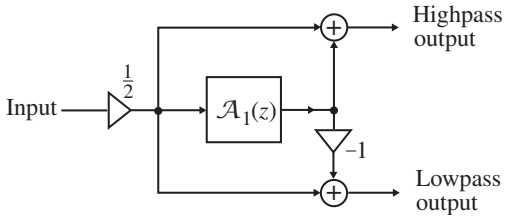


Figure 20: A parametrically tunable first-order lowpass/highpass filter.

where $\mathcal{A}_1(z)$ is a first-order allpass transfer function given by

$$\mathcal{A}_1(z) = \frac{\alpha - z^{-1}}{1 - \alpha z^{-1}}. \tag{60}$$

A composite realization of the above two transfer functions is sketched in Figure 20, where the first-order allpass digital transfer function $\mathcal{A}_1(z)$ can be realized using any one of the single multiplier structures of Figure 8.24 of Text. Note that in this structure, the 3-dB cutoff frequency ω_c of both digital filters is independently controlled by the multiplier constant α of the allpass section.

To derive the transfer function $G_{LP}^{(B)}(z)$ of a first-order digital low-frequency shelving filter for boost, we first observe that Eq. (54) can be rewritten as a sum of a first-order analog lowpass and a first-order analog highpass transfer function[23]

$$H_{LP}^{(B)}(s) = K\left(\frac{\Omega_c}{s + \Omega_c}\right) + \left(\frac{s}{s + \Omega_c}\right). \tag{61}$$

Applying the bilinear transformation to the transfer function of Eq. (61) and making use of Eqs. (59a) and (59b), we arrive at

$$G_{LP}^{(B)}(z) = \tfrac{K}{2}\left[1 - \mathcal{A}_B(z)\right] + \tfrac{1}{2}\left[1 + \mathcal{A}_B(z)\right], \tag{62}$$

where, to emphasize the fact that the above shelving filter is for boost, we have replaced $\mathcal{A}_1(z)$ with $\mathcal{A}_B(z)$, with the latter rewritten as

$$\mathcal{A}_B(z) = \frac{\alpha_B - z^{-1}}{1 - \alpha_B z^{-1}}. \tag{63}$$

From Eq. (9.32) of Text the tuning parameter α_B is given by

$$\alpha_B = \frac{1 - \tan(\omega_c T/2)}{1 + \tan(\omega_c T/2)}. \tag{64}$$

Likewise, the transfer function of a first-order digital low-frequency shelving filter for cut is obtained by applying the bilinear transformation to $H_{LP}^{(C)}(s)$ of Eq. (56).[24] To this end, we first rewrite $H_{LP}^{(C)}(s)$ as a sum of a lowpass and a highpass transfer functions as indicated below:

$$H_{LP}^{(C)}(s) = \left(\frac{\Omega_c}{s + \Omega_c/K}\right) + \left(\frac{s}{s + \Omega_c/K}\right), \tag{65}$$

which, after a bilinear transformation, leads to the transfer function of a first-order low-frequency digital shelving filter for cut as given by

$$G_{LP}^{(C)}(z) = \tfrac{K}{2}[1 - \mathcal{A}_C(z)] + \tfrac{1}{2}[1 + \mathcal{A}_C(z)], \tag{66}$$

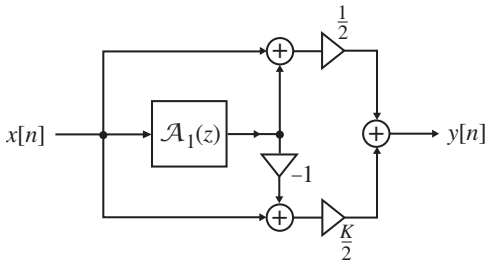


Figure 21: Low-frequency shelving filter where $\mathcal{A}_1(z) = \mathcal{A}_B(z)$ for boost and $\mathcal{A}_1(z) = \mathcal{A}_C(z)$ for cut.

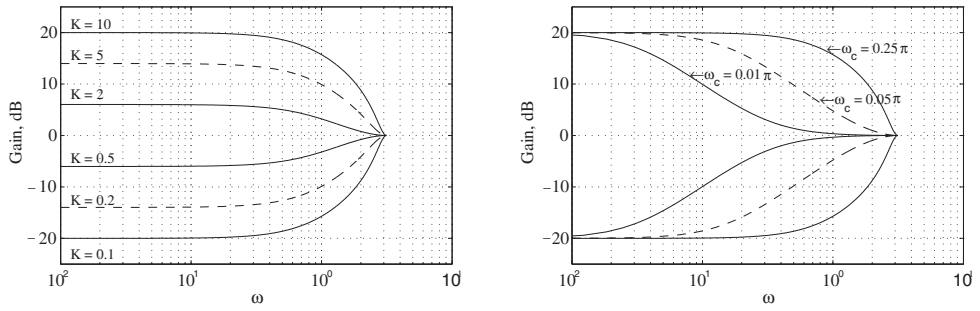


Figure 22: Gain responses of the low-frequency digital shelving filter (a) for six values of K with $\omega_c = 0.25\pi$ and $T = 1$ and (b) for three values of ω_c with $T = 1$ and $K = 10$ for boost and $K = 0.1$ for cut.

where

$$\mathcal{A}_C(z) = \frac{\alpha_C - z^{-1}}{1 - \alpha_C z^{-1}}, \tag{67}$$

with

$$\alpha_C = \frac{K - \tan(\omega_c T/2)}{K + \tan(\omega_c T/2)}. \tag{68}$$

It should be noted that $G_{LP}^{(C)}(z)$ of Eq. (66) is identical in form to $G_{LP}^{(B)}(z)$ of Eq. (62). Hence, the digital filter structure shown in Figure 21 can be used for both boost and cut, except for boost $\mathcal{A}_1(z) = \mathcal{A}_B(z)$ and for cut $\mathcal{A}_1(z) = \mathcal{A}_C(z)$.

Figures 22(a) and (b) show the gain responses of the first-order lowpass digital shelving filter obtained by varying the multiplier constant K and ω_c. Note that the parameter K controls the amount of boost or cut at low frequencies, while the parameters α_B and α_C control the boost bandwidth and cut bandwidth, respectively.

To derive the transfer function $G_{HP}^{(B)}(z)$ of a first-order high-frequency shelving filter for boost, we first express Eq. (54) as a sum of a first-order analog lowpass and highpass transfer function and then apply the bilinear transformation to the resulting expression, arriving at

$$G_{HP}^{(B)}(z) = \tfrac{1}{2}\left[1 - \mathcal{A}_B(z)\right] + \tfrac{K}{2}\left[1 + \mathcal{A}_B(z)\right], \tag{69}$$

where $\mathcal{A}_B(z)$ is as given by Eq. (63), with the multiplier constant α_B given by Eq. (64).

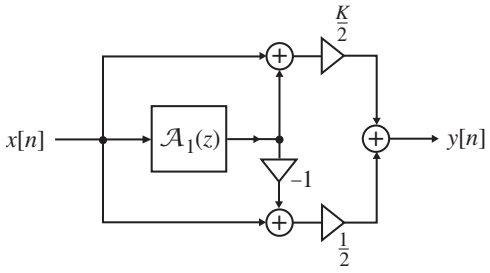


Figure 23: High-frequency shelving filter.

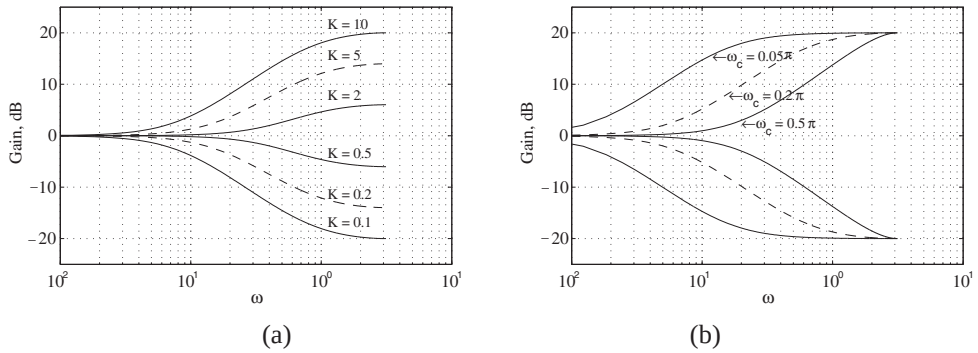


Figure 24: Gain responses of the high-frequency shelving filter of Figure 23 (a) for three values of the parameter K, with $\omega_c = 0.5\pi$ and $T = 1$, and (b) for three values of the parameter ω_c, with $K = 10$ and $T = 1$.

Likewise, the transfer function $G_{HP}^{(C)}(z)$ of a first-order high-frequency shelving filter for cut is obtained by expressing Eq. (56) as a sum of a first-order analog lowpass and highpass transfer function and then applying the bilinear transformation resulting in

$$G_{HP}^{(C)}(z) = \tfrac{1}{2}\left[1 - \mathcal{A}_C(z)\right] + \tfrac{K}{2}\left[1 + \mathcal{A}_C(z)\right], \tag{70}$$

where $\mathcal{A}_C(z)$ is as given by Eq. (66), with the multiplier constant α_C given by

$$\alpha_C = \frac{1 - K\tan(\omega_c T/2)}{1 + K\tan(\omega_c T/2)}. \tag{71}$$

As $G_{HP}^{(B)}(z)$ of Eq. (69) and $G_{HP}^{(C)}(z)$ of Eq. (70) are identical in form, the digital filter structure of Figure 23 can be employed for both boost and cut, except for boost $\mathcal{A}_1(z) = \mathcal{A}_B(z)$ and for cut $\mathcal{A}_1(z) = \mathcal{A}_C(z)$.

Figures 24(a) and (b) show the gain responses of the first-order high-frequency shelving filter obtained by varying the multiplier constant K and ω_c. Note that, as in the case of the low-frequency shelving filter, here the parameter K controls the amount of boost or cut at high frequencies, while the parameters α_B and α_C control the boost bandwidth and cut bandwidth, respectively.

Second-Order Digital Filters and Equalizers

The design of second-order digital bandpass and bandstop filters derived via bilinear transformation of corresponding analog transfer functions has been treated in Section 9.2.3 of Text. The relevant transfer

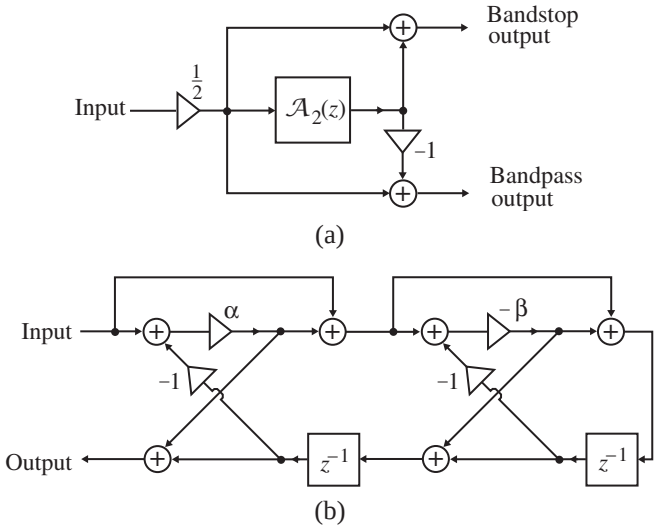


Figure 25: A parametrically tunable second-order digital bandpass/bandstop filter: (a) overall structure and (b) allpass section realizing $\mathcal{A}_2(z)$.

functions, given in Eqs. (9.40) and (9.44) of Text, can be alternatively expressed as

$$G_{BP}(z) = \tfrac{1}{2}\,[1 - \mathcal{A}_2(z)]\,, \tag{72a}$$

$$G_{BS}(z) = \tfrac{1}{2}\,[1 + \mathcal{A}_2(z)]\,, \tag{72b}$$

where $\mathcal{A}_2(z)$ is a second-order allpass transfer function given by

$$\mathcal{A}_2(z) = \frac{\alpha - \beta(1+\alpha)z^{-1} + z^{-2}}{1 - \beta(1+\alpha)z^{-1} + \alpha z^{-2}}. \tag{73}$$

A composite realization of both filters is indicated in Figure 25(a), where the second-order allpass section is realized using the cascaded lattice structure of Figure 25(b) for independent tuning of the center (notch) frequency ω_o and the 3-dB bandwidth B_w.

The transfer function $G_{BP}^{(B)}(z)$ of a second-order peak filter for boost can be derived by applying the simplified lowpass-to-bandpass spectral transformation of Eq. (9.47) of Text to the lowpass shelving filter of Eq. (62), resulting in[16]

$$G_{BP}^{(B)}(z) = G_{LP}^{(B)}(z)\,|_{z^{-1}\to -z^{-1}\left(\frac{z^{-1}+\beta}{1+\beta z^{-1}}\right)} = \tfrac{K}{2}[1 - \mathcal{A}_{2B}(z)] + \tfrac{1}{2}[1 + \mathcal{A}_{2B}(z)], \tag{74}$$

where

$$\beta = \cos(\omega_o), \tag{75}$$

determines the center angular frequency ω_o where the bandpass response peaks, and

$$\mathcal{A}_{2B}(z) = \frac{\alpha_B - \beta(1+\alpha_B)z^{-1} + z^{-2}}{1 - \beta(1+\alpha_B)z^{-1} + \alpha_B z^{-2}} \tag{76}$$

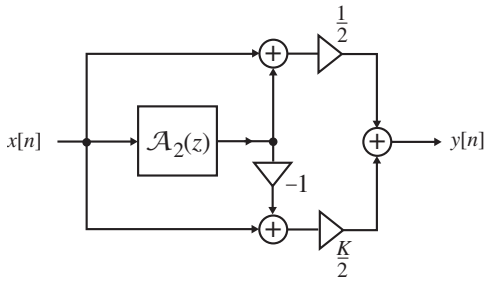


Figure 26: A parametrically tunable second-order peak filter for boost and cut.

is a second-order allpass transfer function obtained by applying the lowpass-to-bandpass transformation of Eq. (9.47) of Text to the first-order allpass transfer function $\mathcal{A}_{(B)}(z)$ of Eq. (63). Here, the parameter α_B is now related to the 3-dB bandwidth B_w of the bandpass response through

$$\alpha_B = \frac{1 - \tan(B_w T/2)}{1 + \tan(B_w T/2)}. \tag{77}$$

Likewise, the transfer function $G_{BP}^{(C)}(z)$ of a second-order peak filter for cut is obtained by applying the lowpass-to-bandpass transformation to the lowpass shelving filter for cut of Eq. (66), resulting in

$$G_{BP}^{(C)}(z) = G_{LP}^{(C)}(z)\Big|_{z^{-1}\to -z^{-1}\left(\frac{z^{-1}+\beta}{1+\beta z^{-1}}\right)} = \tfrac{K}{2}[1 - \mathcal{A}_{2C}(z)] + \tfrac{1}{2}[1 + \mathcal{A}_{2C}(z)]. \tag{78}$$

In Eq. (78), the center angular frequency ω_o, where the bandstop response dips, is related to the parameter β through Eq. (75) and

$$\mathcal{A}_{2C}(z) = \frac{\alpha_C - \beta(1+\alpha_C)z^{-1} + z^{-2}}{1 - \beta(1+\alpha_C)z^{-1} + \alpha_C z^{-2}} \tag{79}$$

is a second-order allpass transfer function obtained by applying the lowpass-to-bandpass transformation of Eq. (9.56) to the first-order allpass transfer function $\mathcal{A}_{(C)}(z)$ of Eq. (66). Here, the parameter α_C is now related to the 3-dB bandwidth B_w of the bandstop response through

$$\alpha_C = \frac{K - \tan(B_w T/2)}{K + \tan(B_w T/2)}. \tag{80}$$

Since both $G_{BP}^{(B)}(z)$ and $G_{BP}^{(C)}(z)$ are identical in form, the digital filter structure of Figure 26 can be employed for both boost and cut, except for boost $\mathcal{A}_2(z) = \mathcal{A}_{2B}(z)$ and for cut $\mathcal{A}_2(z) = \mathcal{A}_{2C}(z)$.

It follows from the above discussion that the peak or the dip of the gain response occurs at the frequency ω_o, which is controlled independently by the parameter β according to Eq. (75), and the 3-dB bandwidth B_w of the gain response is determined solely by the parameter α_B of Eq. (77) for boost or by the parameter α_C of of Eq. (80) for cut. Moreover, the height of the peak of the magnitude response for boost is given by $K = G_{BP}^{(B)}(e^{j\omega_o})$ and the height of the dip of the magnitude response for cut is given by $K = G_{BP}^{(C)}(e^{j\omega_o})$. Figures 27(a), (b), and (c) show the gain responses of the second-order peak filter obtained by varying the parameters K, ω_o, and B_w.

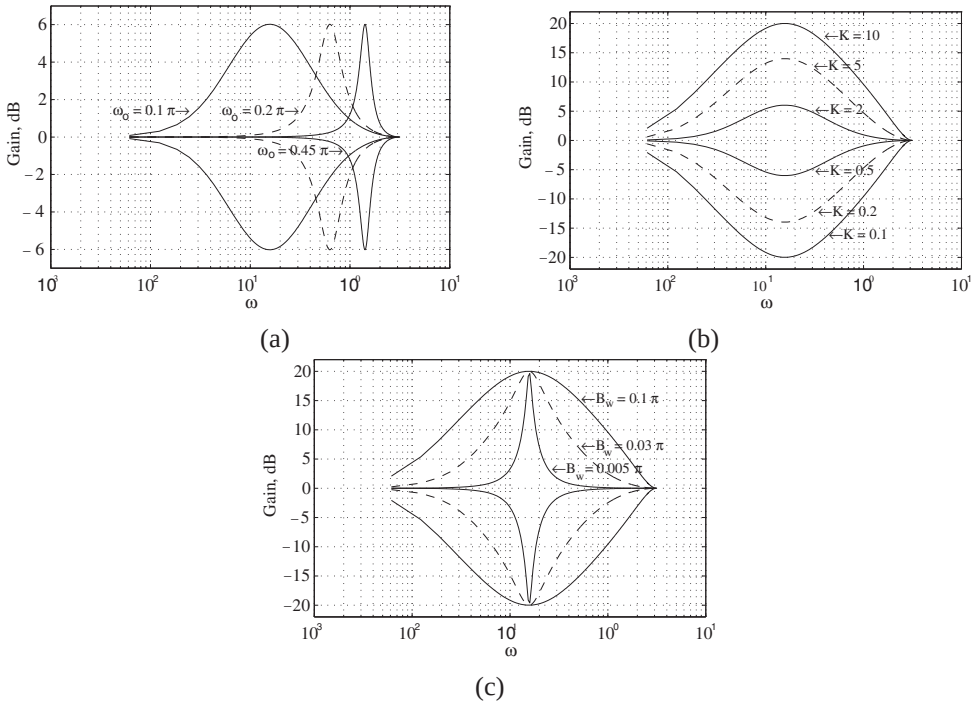


Figure 27: Gain responses of the second-order peak filter (a) for various values of the center frequency ω_o, with $B_w = 0.1\pi$, $T = 1$, and $K = 10$ for boost and $K = 0.1$ for cut; (b) for various values of the parameter K, with $\omega_o = 0.45\pi$, $B_w = 0.1\pi$, and $T = 1$; and (c) for various values of the parameter B_w, with $\omega_0 = 0.45\pi$, $T = 1$ and $K = 10$ for boost and $K = 0.1$ for cut.

Higher-Order Equalizers

A graphic equalizer with tunable gain response can be built using a cascade of first-order and second-order equalizers with external control of the maximum gain values of each section in the cascade. Figure 28(a) shows the block diagram of a cascade of one first-order and three second-order equalizers with nominal frequency response parameters as indicated. Figure 28(b) shows its gain response for some typical values of the parameter K (maximum gain values) of the individual sections.

6 Digital Music Synthesis

As mentioned in Section 1.4.4 of Text that there are basically four methods of musical sound synthesis: (1) *wavetable synthesis*, (2) *spectral modeling synthesis*, (3) *nonlinear synthesis*, and (4) *physical modeling synthesis*.[27] [28]

[27] R. Rabenstein and L. Trautmann, Digital sound synthesis by physical modeling, In *Proc. 2nd International Symp. on Image and Signal Processing and Analysis*, pages 12–23, Pula, Croatia, June 2001.

[28] J.O. Smith III, Viewpoints on the history of digital synthesis, In *Proc. International Computer Music Conference*, pages 1–10, Montreal, Que., Canada, October 1991.

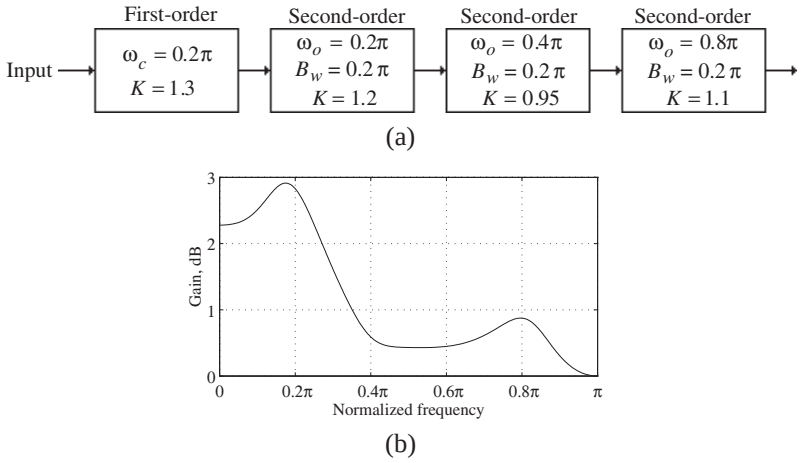


Figure 28: (a) Block diagram of a typical graphic equalizer and (b) its gain response for the section parameter values shown.

A detailed discussion of all these methods is beyond the scope of this book. In this section, we outline a simple wavetable synthesis-based method for generating the sounds of plucked-string instruments.[29]

The basic idea behind the wavetable synthesis method is to store one period of a desired musical tone and repeat it over and over to generate a periodic signal. Such a signal can be generated by running the IIR digital filter structure of Figure 13(a) with specified initial conditions, called *wavetable*, stored in the delay register z^{-R} and with no input. Mathematically, the generated periodic note can be expressed as

$$y[n] = y[n - R], \tag{81}$$

where R, called the *wavetable length*, is the period. The frequency of the tone is F_T/R, where F_T is the sampling frequency. Usually, samples of simple waveforms are used as initial conditions.

A simple modification of the algorithm has been used to generate plucked-string tones. The modified algorithm is given by

$$y[n] = \frac{\alpha^R}{2}(y[n - R] + y[n - R - 1]). \tag{82}$$

The corresponding *plucked-string filter* structure is shown in Figure 29(a). It should be noted that this structure has been derived from the IIR filter structure of Figure 13(a) by inserting a lowpass filter $G(z)$ consisting of a 2-point moving average filter in cascade with a gain block α^R in the feedback path.

The initial sound of a plucked guitar string contains many high-frequency components. To simulate this effect, the plucked-string filter structure is run with zero input and with zero-mean random numbers initially stored in the delay block z^{-R}. The high-frequency components of the stored data get repeatedly lowpass filtered by $G(z)$ as they circulate around the feedback loop of the filter structure of Figure 29(a) and decay faster than the low-frequency components. Since the 2-point moving average filter has a group delay of $\frac{1}{2}$ samples, the pitch period of the tone is $R + \frac{1}{2}$ samples.

It is instructive to examine the gain response of the plucked-string filter.[30] The transfer function of the

[29] K. Karplus and A. Strong, Digital synthesis of plucked-string and drum timbres, *Computer Music Journal*, vol. 7, pp. 43–55, Summer 1983.

[30] K. Steiglitz, *A Digital Signal Processing Primer*, Addison Wesley, Menlo Park CA, 1996.

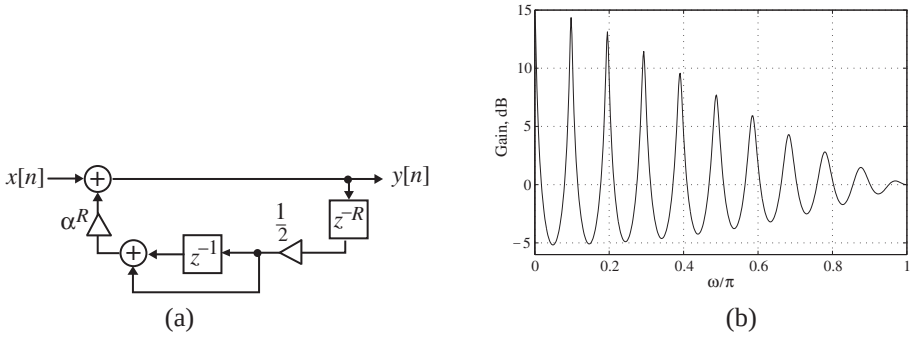


Figure 29: (a) Basic plucked-string filter structure and (b) its gain response for $R = 20$ and $\alpha = 0.99$.

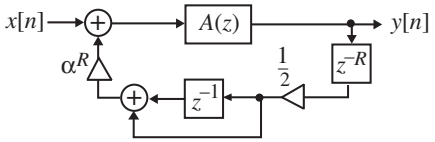


Figure 30: Modified plucked-string filter structure.

the filter structure of Fig. 29(a) is given by

$$H(z) = \frac{1}{1 - \frac{\alpha^R}{2}(1 + z^{-1})z^{-R}}. \tag{83}$$

As the loop delay is 20.5 samples, the resonance frequencies are expected to occur at integer multiples of the pitch frequency $F_T/20.5$, where F_T is the sampling frequency. It can be seen from the gain response plot shown in Figure 29(b) for $R = 20$ and $\alpha = 0.99$, the resonance peaks occur at frequencies very close to the expected values. In addition, the amplitudes of the peaks decrease with increasing frequencies as desired. Moreover, the widths of the resonance peaks increase with increasing frequency, as expected.

For better control of the pitch frequency, an allpass filter $A(z)$ is inserted in the feedback loop, as indicated in Figure 30.[31] The fractional group delay of the allpass filter can be adjusted to tune the overall loop delay of the modified structure. A detailed discussion on the design of the modified plucked-string filter structure for the generation of a sound with a given fundamental frequency can be found in Steiglitz.[30]

7 Discrete-Time Analytic Signal Generation

As discussed in Section **??**, an analytic continuous-time signal has a zero-valued spectrum for all negative frequencies. Such a signal finds applications in single-sideband analog communication systems and analog frequency-division multiplex systems. A discrete-time signal with a similar property finds applications in digital communication systems and is the subject of this section. We illustrate here the generation of an analytic signal $y[n]$ from a discrete-time real signal $x[n]$ and describe some of its applications.

[31] D.A. Jaffe and J.O. Smith, Extensions of the Karplus-Strong plucked-string algorithm, *Computer Music Journal*, vol. 9, pp. 26–23, 1983.

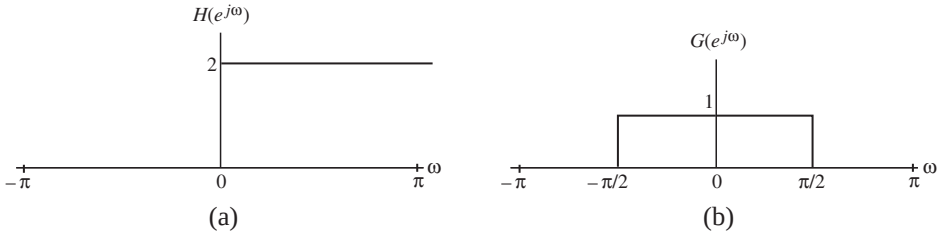


Figure 31: (a) Frequency response of the discrete-time filter generating an analytic signal and (b) half-band lowpass filter.

Now, the Fourier transform $X(e^{j\omega})$ of a real signal $x[n]$, if it exists, is nonzero for both positive and negative frequencies. On the other hand, a signal $y[n]$ with a single-sided spectrum $Y(e^{j\omega})$ that is zero for negative frequencies must be a complex signal. Consider the complex analytic signal

$$y[n] = x[n] + j\,\hat{x}[n], \tag{84}$$

where $x[n]$ and $\hat{x}[n]$ are real. Its Fourier transform $Y(e^{j\omega})$ is given by

$$Y(e^{j\omega}) = X(e^{j\omega}) + j\,\hat{X}(e^{j\omega}), \tag{85}$$

where $\hat{X}(e^{j\omega})$ is the Fourier transform of $\hat{x}[n]$. Now, $x[n]$ and $\hat{x}[n]$ being real, their corresponding Fourier transforms are conjugate symmetric; that is, $X(e^{j\omega}) = X^*(e^{-j\omega})$ and $\hat{X}(e^{j\omega}) = \hat{X}^*(e^{-j\omega})$. Hence, from Eq. (85), we obtain

$$X(e^{j\omega}) = \tfrac{1}{2}\left[Y(e^{j\omega}) + Y^*(e^{-j\omega})\right], \tag{86a}$$

$$j\,\hat{X}(e^{j\omega}) = \tfrac{1}{2}\left[Y(e^{j\omega}) - Y^*(e^{-j\omega})\right]. \tag{86b}$$

Since, by assumption, $Y(e^{j\omega}) = 0$ for $-\pi \le \omega < 0$, we obtain from Eq. (86a)

$$Y(e^{j\omega}) = \begin{cases} 2X(e^{j\omega}), & 0 \le \omega < \pi, \\ 0, & -\pi \le \omega < 0. \end{cases} \tag{87}$$

Thus, the analytic signal $y[n]$ can be generated by passing $x[n]$ through a linear discrete-time system, with a frequency response $H(e^{j\omega})$ given by

$$H(e^{j\omega}) = \begin{cases} 2, & 0 \le \omega < \pi\,, \\ 0, & -\pi \le \omega < 0, \end{cases} \tag{88}$$

as indicated in Figure 31(a).

7.1 The Discrete-Time Hilbert Transformer

We now relate the imaginary part $\hat{x}[n]$ of the analytic signal $y[n]$ to its real part $x[n]$. From Eq. (86b), we get

$$\hat{X}(e^{j\omega}) = \tfrac{1}{2j}\left[Y(e^{j\omega}) - Y^*(e^{-j\omega})\right]. \tag{89}$$

For $0 \le \omega < \pi$, $Y(e^{-j\omega}) = 0$, and for $-\pi \le \omega < 0$, $Y(e^{j\omega}) = 0$. Using this property and Eq. (87) in Eq. (89), it can be easily shown that

$$\hat{X}(e^{j\omega}) = \begin{cases} -jX(e^{j\omega}), & 0 \le \omega < \pi, \\ jX(e^{j\omega}), & -\pi \le \omega < 0. \end{cases} \tag{90}$$

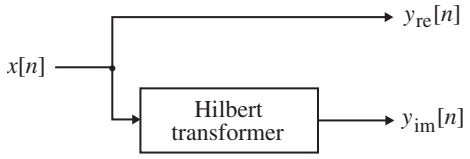


Figure 32: Generation of an analytic signal using a Hilbert transformer.

Thus, the imaginary part $\hat{x}[n]$ of the analytic signal $y[n]$ can be generated by passing its real part $x[n]$ through a linear discrete-time system, with a frequency response $H_{HT}(e^{j\omega})$ given by

$$H_{HT}(e^{j\omega}) = \begin{cases} -j, & 0 \le \omega < \pi, \\ j, & -\pi \le \omega < 0. \end{cases} \tag{91}$$

The linear system defined by Eq. (91) is usually referred to as the ideal *Hilbert transformer*. Its output $\hat{x}[n]$ is called the *Hilbert transform* of its input $x[n]$. The basic scheme for the generation of an analytic signal $y[n] = y_{re}[n] + jy_{im}[n]$ from a real signal $x[n]$ is thus as indicated in Figure 32. Observe that $|H_{HT}(e^{j\omega})| = 1$ for all frequencies and has a -90-degree phase-shift for $0 \le \omega < \pi$ and a $+90$-degree phase-shift for $-\pi \le \omega < 0$. As a result, an ideal Hilbert transformer is also called a 90-*degree phase-shifter*.

The impulse response $h_{HT}[n]$ of the ideal Hilbert transformer is obtained by taking the inverse Fourier transform of $H_{HT}(e^{j\omega})$ and can be shown to be

$$h_{HT}[n] = \begin{cases} 0, & \text{for } n \text{ even}, \\ \frac{2}{\pi n}, & \text{for } n \text{ odd}. \end{cases} \tag{92}$$

Since the ideal Hilbert transformer has a two-sided infinite-length impulse response defined for $-\pi < n < \pi$, it is an unrealizable system. Moreover, its transfer function $H_{HT}(z)$ exists only on the unit circle. We describe later two approaches for developing a realizable approximation.

7.2 Relation with Half-Band Filters

Consider the filter with a frequency response $G(e^{j\omega})$ obtained by shifting the frequency response $H(e^{j\omega})$ of Eq. (88) by $\pi/2$ radians and scaling by a factor $\frac{1}{2}$ (see Figure 31):

$$G(e^{j\omega}) = \tfrac{1}{2}H(e^{j(\omega+\pi/2)}) = \begin{cases} 1, & 0 < |\omega| < \frac{\pi}{2}, \\ 0, & \frac{\pi}{2} < |\omega| < \pi. \end{cases} \tag{93}$$

From our discussion in Section 13.6.2 of Text, we observe that $G(e^{j\omega})$ is a half-band lowpass filter. Because of the relation between $H(e^{j\omega})$ of Eq. (88) and the real coefficient half-band lowpass filter $G(e^{j\omega})$ of Eq. (93), the filter $H(e^{j\omega})$ has been referred to as a *complex half-band filter*.[32]

7.3 Design of the Hilbert Transformer

It also follows from the above relation that a complex half-band filter can be designed simply by shifting the frequency response of a half-band lowpass filter by $\pi/2$ radians and then scaling by a factor 2. Equivalently, the relation between the transfer functions of a complex half-band filter $H(z)$ and a real half-band

[32] P.A. Regalia, Special filter designs, In S.K. Mitra and J.F. Kaiser, editors, *Handbook for Digital Signal Processing*, chapter 13, pages 967–980. Wiley-Interscience, New York, NY, 1993.

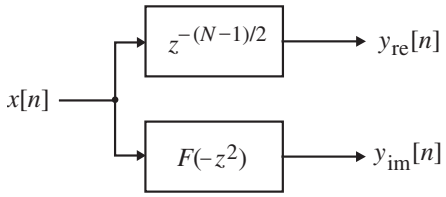


Figure 33: FIR realization of a complex half-band filter.

lowpass filter $G(z)$ is given by

$$H(z) = j2G(-jz). \tag{94}$$

Three methods of the design of the real half-band filter have been presented in Section 13.6 of Text. We adopt two of these methods here for the design of complex half-band filters.

FIR Complex Half-Band Filter

Let $G(z)$ be the desired FIR real half-band linear-phase lowpass filter of even degree N, with the passband edge at ω_p, stopband edge at ω_s, and passband and stopband ripples of δ_p, with $\omega_p + \omega_s = \pi$. The half-band filter $G(z)$ is then designed by first designing a wide-band linear-phase filter $F(z)$ of degree $N/2$ with a passband from 0 to $2\omega_p$, a transition band from $2\omega_p$ to π, and a passband ripple of 2δ. The desired half-band filter $G(z)$ is then obtained by forming

$$G(z) = \tfrac{1}{2}\left[z^{-N/2} + F(z^2)\right]. \tag{95}$$

Substituting Eq. (95) in Eq. (94), we obtain

$$H(z) = j\left[(-jz)^{-N/2} + F(-z^2)\right] = z^{-N/2} + jF(-z^2). \tag{96}$$

An FIR implementation of the complex half-band filter based on the above decomposition is indicated in Figure 33. The linear-phase FIR filter $F(-z^2)$ is thus an approximation to a Hilbert transformer.

We illustrate the above approach in Example 8.

EXAMPLE 8 FIR Complex Half-Band Filter Design

Using MATLAB, we design a wide-band FIR filter $F(z)$ of degree `13` with a passband from `0` to `0.85`π and an extremely small stopband from `0.9`π to π. We use the function `remez` with a magnitude vector `m = [1 1 0 0]`. The weight vector used to weigh the passband and the stopband is `wt = [2 0.05]`. The magnitude responses of the wide-band filter $F(z)$ and the Hilbert transformer $F(-z^2)$ are shown in Figures 34(a) and (b).

The FIR Hilbert transformer can be designed directly using the function `remez`. Example 9 illustrates this approach.

EXAMPLE 9 Direct Design of FIR Complex Half-Band Filter Using MATLAB

We design a 26th-order FIR Hilbert transformer with a passband from 0.1π to 0.9π. It should be noted that for the design of a Hilbert transformer, the first frequency point in the vector `f` containing the specified bandedges cannot be a 0. The magnitude response of the designed Hilbert transformer obtained using the program statement `b =remez(26, [0.1 0.9], [1 1], 'Hilbert')` is indicated in Figure 35.

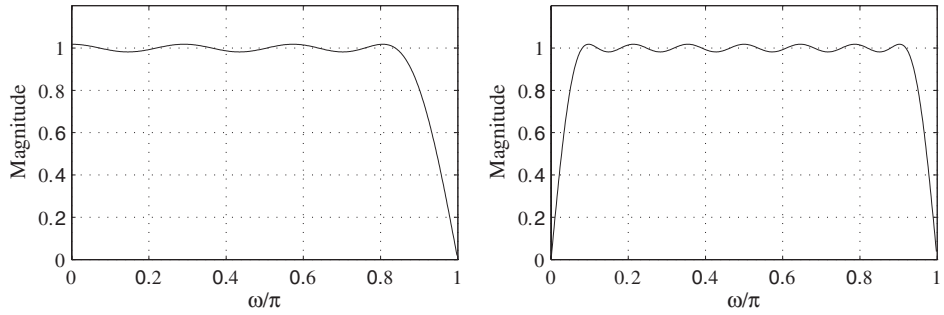


Figure 34: Magnitude responses of (a) the wide-band FIR filter $F(z)$ and (b) the approximate Hilbert transformer $F(-z^2)$.

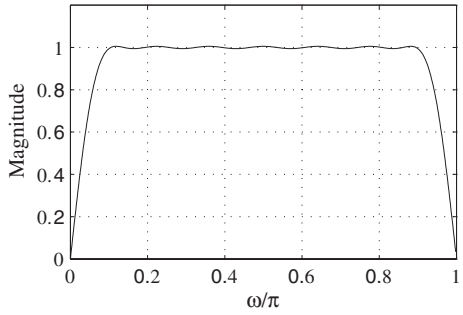


Figure 35: The magnitude response of the Hilbert transformer designed directly using MATLAB.

It should be noted that due to numerical round-off problems, unlike the design of Example 8, the odd impulse response coefficients of the Hilbert transformer here are not exactly zero.

IIR Complex Half-Band Filter

We outlined in Section 13.6.5 of Text a method to design stable IIR real coefficient half-band filters of odd order in the form[33]

$$G(z) = \tfrac{1}{2}[\mathcal{A}_0(z^2) + z^{-1}\mathcal{A}_1(z^2)], \tag{97}$$

where $\mathcal{A}_0(z)$ and $\mathcal{A}_1(z)$ are stable allpass transfer functions. Substituting Eq. (97) in Eq. (94), we therefore arrive at

$$H(z) = \mathcal{A}_0(-z^2) + jz^{-1}\mathcal{A}_1(-z^2). \tag{98}$$

A realization of the complex half-band filter based on the above decomposition is thus as shown in Figure 36.

We illustrate the above approach to Hilbert transformer design in Example 10.

[33] P.P. Vaidyanathan, P.A. Regalia, and S.K. Mitra, Design of doubly-complementary IIR digital filters using a single complex allpass filter, with multirate applications, *IEEE Trans. on Circuits and Systems*, vol. CAS-34, pp. 378–389, April 1987.

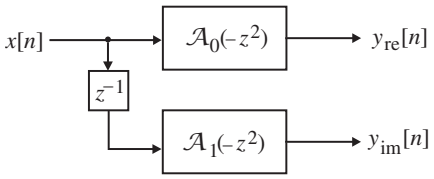


Figure 36: IIR realization of a complex half-band filter.

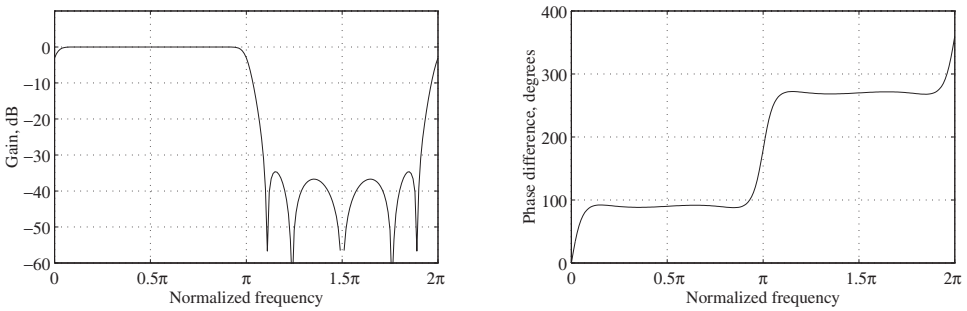


Figure 37: (a) Gain response of the complex half-band filter (normalized to 0 dB maximum gain) and (b) phase difference between the two allpass sections of the complex half-band filter.

EXAMPLE 10 IIR Complex Half-Band Filter Design

In Example 13.24, we designed a real half-band elliptic filter with the following frequency response specifications: $\omega_s = 0.6\pi$ and $\delta_s = 0.016$. The transfer function of the real half-band filter $G(z)$ can be expressed as in Eq. (97), where the transfer functions of the two allpass sections $\mathcal{A}_0(z^2)$ and $\mathcal{A}_1(z^2)$ are given in Eq. (13.116). The gain response of the complex half-band filter $H(z)$ obtained using Eq. (98) is sketched in Figure 37(a). Figure 37(b) shows the phase difference between the two allpass functions $\mathcal{A}_0(-z^2)$ and $z^{-1}\mathcal{A}_1(-z^2)$ of the complex half-band filter. Note that, as expected, the phase difference is 90 degrees for most of the positive frequency range and 270 degrees for most of the negative frequency range. In plotting the gain response of the complex half-band filter and the phase difference between its constituent two allpass sections, the M-file `freqz(num,den,n,'whole')` has been used to compute the pertinent frequency response values over the whole normalized frequency range from 0 to 2π.

7.4 Single-Sideband Modulation

For efficient transmission over long distances, a real low-frequency band-limited signal $x[n]$, such as speech or music, is modulated by a very high frequency sinusoidal carrier signal $\cos \omega_c n$, with the carrier frequency ω_c being less than half of the sampling frequency. The spectrum $V(e^{j\omega})$ of the resulting signal $v[n] = x[n] \cos \omega_c n$ is given by

$$V(e^{j\omega}) = \frac{1}{2}\left[X(e^{j(\omega-\omega_c)}) + X(e^{j(\omega+\omega_c)})\right]. \tag{99}$$

As indicated in Figure 38, if $X(e^{j\omega})$ is band-limited to ω_M, the spectrum $V(e^{j\omega})$ of the modulated signal $v[n]$ has a bandwidth of $2\omega_M$ centered at $\pm\omega_c$. By choosing widely separated carrier frequencies, one

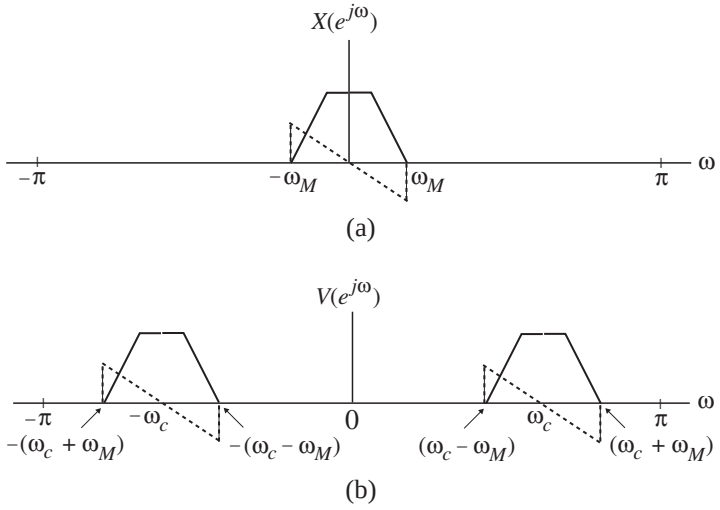


Figure 38: Spectra of a real signal and its modulated version. (Solid lines represent the real parts, and dashed lines represent the imaginary parts.)

can modulate a number of low-frequency signals to high-frequency signals, combine them by frequency-division multiplexing, and transmit over a common channel. The carrier frequencies are chosen appropriately to ensure that there is no overlap in the spectra of the modulated signals when combined by frequency-division multiplexing. At the receiving end, each of the modulated signals is then separated by a bank of bandpass filters of center frequencies corresponding to the different carrier frequencies.

It is evident from Figure 38 that, for a real low-frequency signal $x[n]$, the spectrum of its modulated version $v[n]$ is symmetric with respect to the carrier frequency ω_c. Thus, the portion of the spectrum in the frequency range from ω_c to $(\omega_c + \omega_M)$, called the *upper sideband*, has the same information content as the portion in the frequency range from $(\omega_c - \omega_M)$ to ω_c, called the *lower sideband*. Hence, for a more efficient utilization of the channel bandwidth, it is sufficient to transmit either the upper or the lower sideband signal. A conceptually simple way of eliminating one of the sidebands is to pass the modulated signal $v[n]$ through a sideband filter whose passband covers the frequency range of one of the sidebands.

An alternative, often preferred, approach for single-sideband signal generation is by modulating the analytic signal whose real and imaginary parts are, respectively, the real signal and its Hilbert transform. To illustrate this approach, let $y[n] = x[n] + j\hat{x}[n]$, where $\hat{x}[n]$ is the Hilbert transform of $x[n]$. Consider

$$\begin{aligned} s[n] &= y[n]e^{j\omega_c n} = (y_{\text{re}}[n] + jy_{\text{im}}[n])\,(\cos\omega_c n + j\,\sin\omega_c n) \\ &= (x[n]\cos\omega_c n - \hat{x}[n]\sin\omega_c n) \\ &\quad + j\,(x[n]\sin\omega_c n + \hat{x}[n]\cos\omega_c n)\,. \end{aligned} \tag{100}$$

From Eq. (100), the real and imaginary parts of $s[n]$ are thus given by

$$s_{\text{re}}[n] = x[n]\cos\omega_c n - \hat{x}[n]\sin\omega_c n, \tag{101a}$$

$$s_{\text{im}}[n] = x[n]\sin\omega_c n + \hat{x}[n]\cos\omega_c n. \tag{101b}$$

Figure 39 shows the spectra of $x[n]$, $\hat{x}[n]$, $y[n]$, $s[n]$, $s_{\text{re}}[n]$, and $s_{\text{im}}[n]$. It therefore follows from these plots that a single-sideband signal can be generated using either one of the modulation schemes described

by Eqs. (101a) and (101b), respectively. A block diagram representation of the scheme of Eq. (101a) is sketched in Figure 40.

8 Signal Compression

As mentioned earlier, signals carry information, and the objective of signal processing is to preserve the information contained in the signal and extract and manipulate it when necessary. Most digital signals encountered in practice contain a huge amount of data. For example, a gray-level image of size 512×512 with 8-bits per pixel contains $(512)^2 \cdot 8 = 2,097,152$ bits. A color image of the same size contains 3 times as many bits. For efficient storage of digital signals, it is often necessary to compress the data into a smaller size requiring significantly fewer number of bits. A signal in compressed form also requires less bandwidth for transmission. Roughly speaking, signal compression is concerned with the reduction of the amount of data, while preserving the information content of the signal with some acceptable fidelity.

Most practical signals exhibit *data redundancy*, as they contain some amount of data with no relevant information. Three types of data redundancy are usually encountered in practice: *coding redundancy*, *intersample redundancy*, and *psychovisual redundancy*.[34] Signal compression methods exploit one or more of these redundancies to achieve data reduction.

A signal coding system consists of an encoder and a decoder. The input to the encoder is the signal $\mathbf{x}$ to be compressed, and its output is the compressed bit stream $\mathbf{d}$. The decoder performs the reverse operation. Its input is the compressed bit stream $\mathbf{d}$ developed by the encoder, and its output $\hat{\mathbf{x}}$ is a reasonable replica of the original input signal of the encoder. The basic components of the encoder and the decoder are shown in Figure 41.

The *energy compression* block transforms the input sequence $\mathbf{x}$ into another sequence $\mathbf{y}$ with the same total energy, while packing most of the energy in very few of samples $\mathbf{y}$. The *quantizer* block develops an approximate representation of $\mathbf{y}$ for a given level of accuracy in the form of an integer-valued sequence $\mathbf{q}$ by adjusting the quantizer step size to control the trade-off between distortion and bit rate. The *entropy coding* block uses variable-length entropy coding to encode the integers in the sequence $\mathbf{q}$ into a binary bitstream $\mathbf{d}$, with the aim of minimizing the total number of bits in $\mathbf{d}$ by making use of the statistics of the class of samples in $\mathbf{q}$.

The *entropy decoding* block regenerates the integer-valued sequence $\mathbf{q}$ from the binary bit stream $\mathbf{d}$. The *inverse quantizer* develops $\hat{\mathbf{y}}$, a best estimate of $\mathbf{y}$ from $\mathbf{q}$. Finally, the *reconstruction* block develops $\hat{\mathbf{x}}$, the best approximation of the original input sequence $\mathbf{x}$ from $\hat{\mathbf{y}}$.

The signal compression methods can be classified into two basic groups: *lossless* and *lossy.* In the lossless compression methods, no information is lost due to compression, and the original signal can be recovered exactly from the compressed data by the decoder. On the other hand, in the lossy compression methods, some amount of information (usually less relevant) is lost, due to compression and the signal reconstructed by the decoder is not a perfect replica of the original signal but is still an acceptable approximation for the application at hand. Naturally, the latter method can result in a significant reduction in the number of bits necessary to represent the signal and is considered here. Moreover, for conciseness, we discuss image compression methods that exploit only the coding redundancy. A detailed exposition of compression methods exploiting all types of data redundancies is beyond the scope of this book.

[34] R.C. Gonzalez and P. Wintz, *Digital Image Processing*, Second Edition, Prentice-Hall, Upper Saddle River NJ, 2002.

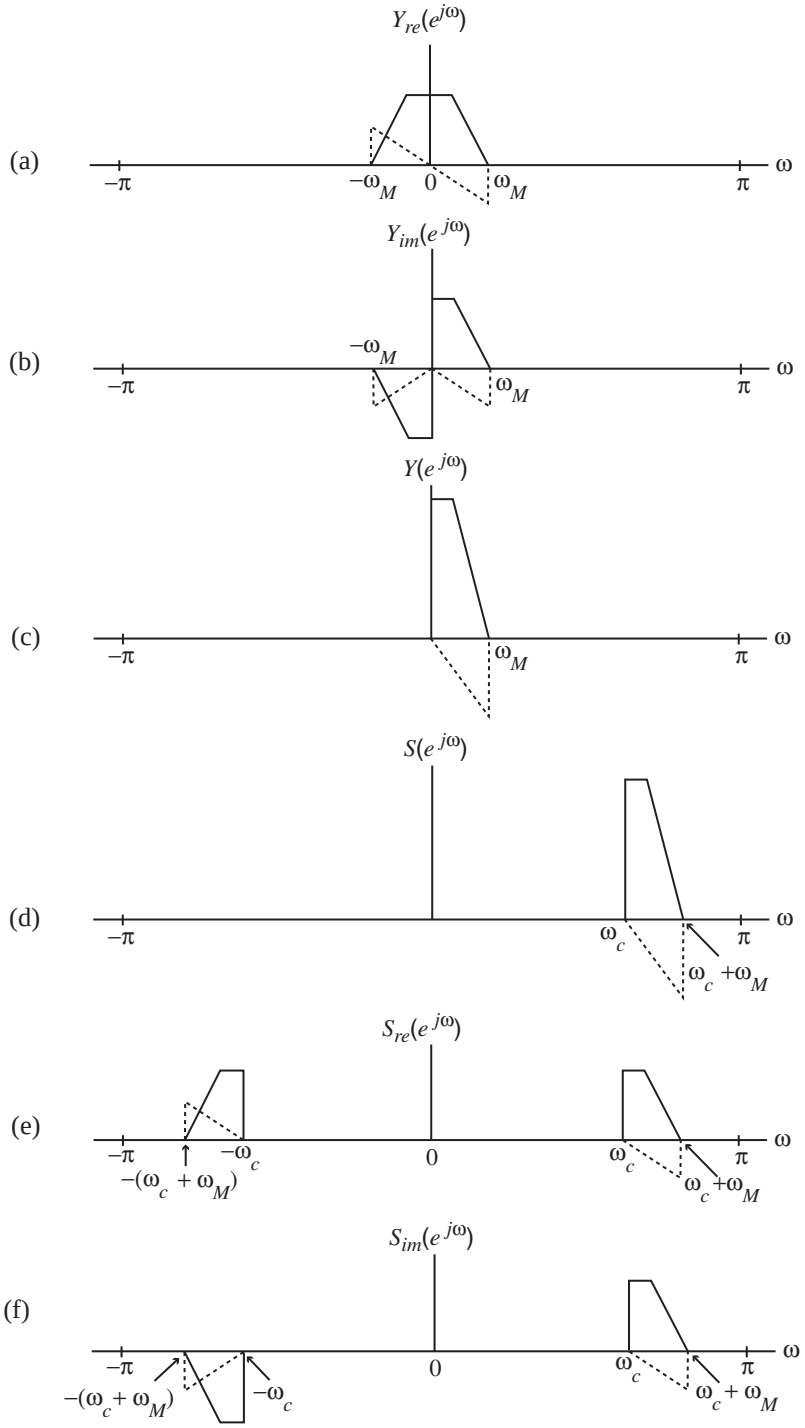


Figure 39: (a)–(f) Illustration of the generation of single-sideband signals via the Hilbert transform. (Solid lines represent the real parts, and dashed lines represent the imaginary parts.)

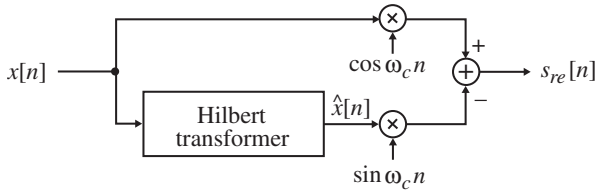


Figure 40: Schematic of the single-sideband generation scheme of Eq. (101a).

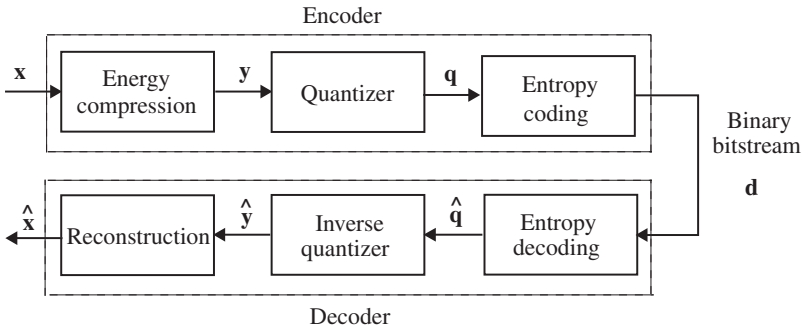


Figure 41: The block diagram representation of the signal compression system.

8.1 Coding Redundancy

We assume that each sample of the discrete-time signal $\{x[n]\}$ is a random variable r_i taking one of Q distinct values with a probability $p_i, 0 \leq i \leq Q-1$, where $p_i \leq 1$ and $\sum_{i=0}^{Q-1} p_i = 1$. Each possible value r_i is usually called a *symbol*. The probability p_i of each symbol r_i can be estimated from the histogram of the signal. Thus, if the signal contains a total of N samples, with m_i denoting the total number of samples taking the value r_i, then

$$p_i = \frac{m_i}{N}, \qquad 0 \leq i \leq Q-1. \tag{102}$$

Let b_i denote the length of the i-th codeword, that is, the total number of bits necessary to represent the value of the random variable r_i. A measure of the coding redundancy is then given by the average number of bits needed to represent each sample of the signal $\{x[n]\}$:

$$B_{\text{av}} = \sum_{i=0}^{Q-1} b_i\, p_i \text{ bits}. \tag{103}$$

As a result, the total number of bits required to represent the signal is $N \cdot B_{\text{av}}$.

8.2 Entropy

The goal of the compression method is to reduce the volume of data while retaining the information content of the original signal with some acceptable fidelity. The information content represented by a symbol can be informally related to its unexpectedness; that is, if a symbol that arrives is the one that was

expected, it does not convey very much information. On the other hand, if an unexpected symbol arrives, it conveys much more information. Thus, the information content of a particular symbol can be related to its probability of occurrence, as described next.

For a discrete-time sequence $\{x[n]\}$ with samples taking one of Q distinct symbols r_i with a probability $p_i, 0 \leq i \leq Q-1$, a measure of the information content I_i of the i-th symbol r_i is defined by[35]

$$I_i = -\log_2 p_i. \tag{104}$$

It follows from the above definition that $I_i \geq 0$. Moreover, it also can be seen that I_i is very large when p_i is very small.

A measure of the average information content of the signal $\{x[n]\}$ is then given by its *entropy*, which is defined by

$$\mathcal{H}_x = \sum_{i=0}^{Q-1} p_i I_i = -\sum_{i=0}^{Q-1} p_i \log_2 p_i \text{ bits/symbol.} \tag{105}$$

The coding redundancy is defined as the difference between the actual data rate and the entropy of a data stream.

8.3 A Signal Compression Example[36]

We now consider the compression of a gray level image to illustrate the various concepts introduced earlier in this section. For the energy compression stage, we make use of the Haar wavelets of Section 14.6.2. Since the image is a two-dimensional sequence, the wavelet decomposition is first applied row-wise and then column-wise. Applying the Haar transform $\mathbf{H}$ to the input image $\mathbf{x}$, first row-wise and then column-wise, we get

$$\mathbf{y} = \mathbf{H}\mathbf{x}\mathbf{H}^T, \tag{106}$$

where

$$\mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{107}$$

To understand the effect of the decomposition on the image, consider a 2×2 two-dimensional sequence given by

$$\mathbf{x} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}. \tag{108}$$

Then,

$$\mathbf{y} = \frac{1}{2} \begin{bmatrix} a+b+c+d & a-b+c-d \\ a+b-c-d & a-b-c+d \end{bmatrix}. \tag{109}$$

The element $a+b+c+d$ at the top left position of $\mathbf{y}$ is the 4-point average of $\mathbf{x}$ and therefore contains only the vertical and horizontal low-frequency components of $\mathbf{x}$. It is labeled as the $\mathbf{L}L$ part of $\mathbf{x}$. The element $a-b+c-d$ at the top right position of $\mathbf{y}$ is obtained by forming the differences of the horizontal components and the sum of the vertical components and hence contains the vertical low- and horizontal high-frequency components. It is labeled as the $\mathbf{H}L$ part of $\mathbf{x}$. The element $a+b-c-d$ at the bottom left position of $\mathbf{y}$ is obtained by forming the sum of the horizontal components and the differences of the

[35] A.K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs NJ, 1989.

[36] Portions of this section have been adapted from N. Kingsbury, *Image Coding Course Notes*, Department of Engineering, University of Cambridge, Cambridge, U.K., July 13, 2001, by permission of the author.

vertical components and hence contains the horizontal low- and vertical high-frequency components. It is labeled as the $\mathbf{L}H$ part of $\mathbf{x}$. Finally, the element $a - b - c + d$ at the bottom right is obtained by forming the differences between both the horizontal and vertical components and therefore contains only the vertical and horizontal high-frequency components of $\mathbf{x}$. It is labeled as the $\mathbf{H}H$ part of $\mathbf{x}$.

Applying a one-level Haar wavelet decomposition to the image "Goldhill" of Figure 42(a), downsampling the outputs of all filters by a factor-of-2 in both horizontal and vertical directions, we arrive at the four subimages shown in Figure 42(b). The original image is of size 512×512 pixels. The subimages of Figure 42(b) are of size 256×256 pixels each. The total energies of the subimages and their percentages of the total energy of all subimages are now as follows:

LL:	**HL**:	**LH**:	**HH**:
3919.91×10^6	6.776×10^6	7.367×10^6	1.483×10^6
99.603 %	0.172 %	0.187 %	0.038 %

The sum of the energies of all subimages is equal to 3935.54×10^6, which is also the total energy of the original image of Figure 42(a). As can be seen from the above energy distribution data and Figure 42(b), the $\mathbf{L}L$ subimage contains most of the energy of the original image, whereas, the **HH** subimage contains least of the energy. Also, the **HL** subimage has mostly the near-horizontal edges, whereas the **LH** subimage has mostly the near-vertical edges.

To evaluate the entropies, we use uniform scalar quantizers for all signals with a quantization step size $Q = 15$. The entropy of the original image computed from its histogram, after compression with Q = 15, is $\mathcal{H}_x = 3.583$ bits/pixel. The entropies of the sub-images after a one-level Haar decomposition are as given below:

LL:	**HL**:	**LH**:	**HH**:
$4.549/4$	$1.565/4$	$1.375/4$	$0.574/4$
$= 1.1370$	$= 0.3911$	0.3438	0.1436

The entropy of the wavelet representation is $\mathcal{H}_y = 2.016$ bits/pixel, obtained by adding the entropies of the subimages given above. Hence, the compression ratio is 1.78-to-1.0. Figures 43(a) and (b) show, respectively, the reconstructed "Goldhill" image after direct quantization of the original pixels and after quantization of the wavelet coefficients.

A commonly used measure of the quality of the reconstructed image compared with the original image is the *peak-signal-to-noise ratio* (PSNR). Let $x[m,n]$ denote the (m,n)th pixel of an original image $\mathbf{x}$ of size $M \times N$, and, $y[m,n]$ denote the (m,n)th pixel of the reconstructed image $\mathbf{y}$ of the same size, with 8-bits per pixel. Then, the PSNR is defined by

$$\text{PSNR} = 20 \log_{10} \left(\frac{255}{\text{RMSE}} \right) \text{ dB}, \tag{110}$$

where RMSE is the *root mean square error,* which is the square root of the *mean square error* (MSE), given by

$$\text{MSE} = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} (x^2[m,n] - y^2[m,n])}{MN}. \tag{111}$$

The PSNR of the reconstructed image of Figure 43(a) is 35.42 dB and that of Figure 43(b) is 35.72 dB.

We next apply a two-level Haar wavelet decomposition to "Goldhill" image. The process is equivalent to applying a one-level decomposition to the **LL**-subimage of Figure 43(b). Figure 44(a) shows the seven

(a) (b)

Figure 42: (a) Original "Goldhill" image and (b) subimages after one-level Haar wavelet decomposition.

(a) (b)

Figure 43: Reconstructed "Goldhill" image: (a) after direct quantization of the original pixels and (b) after quantization of the wavelet coefficients.

subimages. The subimage at the top left is of size 128×128 and contains only the low frequencies and is labeled **LLL**. The remaining 3 subimages obtained after the second-level decomposition are labeled accordingly.

The total energies of the four subimages of size 128×128 at the top left corner and their percentages of the total energy of all subimages are now as follows:

LLL:	**LHL**:	**LLH**:	**LHH**:
3898.26×10^6	9.412×10^6	10.301×10^6	1.940×10^6
99.053 %	0.239 %	0.262 %	0.049 %

The total energies of the remaining three subimages of size 256×256 at the top right and bottom left

(a) (b)

Figure 44: (a) Subimages after a two-level Haar wavelet decomposition and (b) reconstructed image after quantization of the two-level wavelet coefficients.

and right corners remain the same as given earlier. The sum of the energies of all subimages is again equal to 3935.54×10^6. The entropy of the wavelet representation after a two-level decomposition is now $\mathcal{H}_y = 1.620$ bits/pixel, obtained by adding the entropies of the subimages given above and is seen to be much smaller than that obtained after a one-level decomposition. The compression ratio is 2.2-to-1.0. Figure 44(b) shows the reconstructed "Goldhill" image after quantization of the wavelet coefficients at the second level. The PSNR of the reconstructed image is now 35.75 dB.

The compression ratio advantage of the wavelet decomposition is a consequence of the entropy difference between the quantization of the image in the space domain and the separate quantization of each subimage. The wavelet decomposition allows for an entropy reduction since most of the signal energy is allocated to low-frequency subimages with a smaller number of pixels. If the quantization step does not force the entropy reduction to be very large, then the wavelet-reconstructed image can still have better quality than that obtained from space-domain coding, which can seen from the compression examples given in Figures 43 and 44.

In order to exploit the entropy of the image representation after quantization, a lossless source coding scheme such as Huffman or arithmetic coding is required.[37] The design of these codes goes beyond the scope of this book and is therefore not included here. It is enough to state that lossless codes allow the image in these examples to be compressed in practice at rates (number of bits/pixel) arbitrarily close to those expressed by the entropy values that have been shown.

The histograms of all subimages, except the one with lowest frequency content (i.e., the top left subimage), have only one mode and are centered at zero, with tails that usually decay with exponential behavior. This means that many of the subimage pixels are assigned to the quantized interval centered at zero. Run-length coding is a lossless coding scheme that encodes sequences of zeros by a special symbol denoting the beginning of such sequences, followed by the length of the sequence.[37] This different representation allows for further reduction of the subimages entropy after the quantization, and thus, run-length coding can be used to improve, without any loss of quality, the compression ratios shown in the examples of this section.

[37] N.S. Jayant and P. Knoll, *Digital Coding of Waveforms*, Prentice Hall, Englewood Cliffs NJ, 1984.

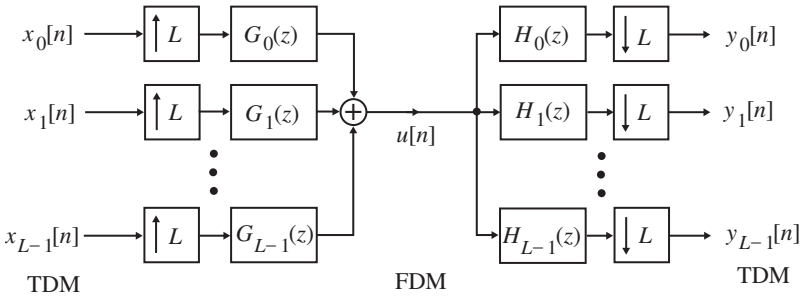


Figure 45: The basic L-channel transmultiplexer structure.

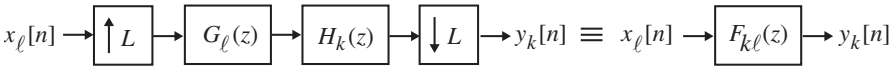


Figure 46: The k, ℓ-path of the L-channel transmultiplexer structure.

9 Transmultiplexers

In the United States and most other countries, the telephone service employs two types of multiplexing schemes to transmit multiple low-frequency voice signals over a wide-band channel. In the *frequency-division multiplex* (FDM) telephone system, multiple analog voice signals are first modulated by single-sideband (SSB) modulators onto several subcarriers, combined, and transmitted simultaneously over a common wide-band channel. To avoid *cross-talk*, the subcarriers are chosen to ensure that the spectra of the modulated signals do not overlap. At the receiving end, the modulated subcarrier signals are separated by analog bandpass filters and demodulated to reconstruct the individual voice signals. On the other hand, in the *time-division multiplex* (TDM) telephone system, the voice signals are first converted into digital signals by sampling and A/D conversion. The samples of the digital signals are time-interleaved by a digital multiplexer, and the combined signal is transmitted. At the receiving end, the digital voice signals are separated by a digital demultiplexer and then passed through a D/A converter and an analog reconstruction filter to recover the original analog voice signals.

The TDM system is usually employed for short-haul communication, while the FDM scheme is preferred for long-haul transmission. Until the telephone service becomes all digital, it is necessary to translate signals between the two formats. This is achieved by the transmultiplexer system discussed next.

The *transmultiplexer* is a multi-input, multi-output, multirate structure, as shown in Figure 45. It is exactly the opposite to that of the L-channel QMF bank of Figure 14.18 of Text and consists of an L-channel synthesis filter bank at the input end, followed by an L-channel analysis filter bank at the output end. To determine the input–output relation of the transmultiplexer, consider one typical path from the kth input to the ℓth output as indicated in Figure 46(a).[38] A polyphase representation of the structure of Figure 45 is shown in Figure 47(a). Invoking the identity of Section 13.4.5, we note that the structure of Figure 46(a) is equivalent to that shown in Figure 46(b), consisting of an LTI branch with a transfer function $F_{k\ell}(z)$ that is the zeroth polyphase component of $H_k(z)G_\ell(z)$. The input–output relation of the

[38] P.P. Vaidyanathan. *Multirate Systems and Filter Banks*, Prentice Hall, Englewood Cliffs NJ, 1993.

transmultiplexer is therefore given by

$$Y_k(z) = \sum_{\ell=0}^{L-1} F_{k\ell}(z) X_\ell(z), \quad 0 \le k \le L-1. \tag{112}$$

Denoting

$$\mathbf{Y}(z) = [Y_0(z) \quad Y_1(z) \quad \cdots \quad Y_{L-1}(z)]^t, \tag{113a}$$

$$\mathbf{X}(z) = [X_0(z) \quad X_1(z) \quad \cdots \quad X_{L-1}(z)]^t, \tag{113b}$$

we can rewrite Eq. (112) as

$$\mathbf{Y}(z) = \mathbf{F}(z)\mathbf{X}(z), \tag{114}$$

where $\mathbf{F}(z)$ is an $L \times L$ matrix whose (k,ℓ)th element is given by $F_{k\ell}(z)$. The objective of the transmultiplexer design is to ensure that $y_k[n]$ is a reasonable replica of $x_k[n]$. If $y_k[n]$ contains contributions from $x_r[n]$ with $r \neq n$, then there is *cross-talk* between these two channels. It follows from Eq. (114) that cross-talk is totally absent if $F(z)$ is a diagonal matrix, in which case Eq. (114) reduces to

$$Y_k(z) = F_{kk}(z) X_k(z), \quad 0 \le k \le L-1. \tag{115}$$

As in the case of the QMF bank, we can define three types of transmultiplexer systems. It is a phase-preserving system if $F_{kk}(z)$ is a linear-phase transfer function for all values of k. Likewise, it is a magnitude-preserving system if $F_{kk}(z)$ is an allpass function. Finally, for a perfect reconstruction transmultiplexer,

$$F_{kk}(z) = \alpha_k z^{-n_k}, \quad 0 \le k \le L-1, \tag{116}$$

where n_k is an integer and α_k is a nonzero constant. For a perfect reconstruction system, $y_k[n] = \alpha_k x_k[n-n_k]$.

The perfect reconstruction condition can also be derived in terms of the polyphase components of the synthesis and analysis filter banks of the transmultiplexer of Figure 45, as shown in Figure 47(a).[39] Using the cascade equivalences of Figure 13.14, we arrive at the equivalent representation indicated in Figure 47(b). Note that the structure in the center part of this figure is a special case of the system of Figure 45, where $G_\ell(z) = z^{-(L-1-\ell)}$ and $H_k(z) = z^{-k}$, with $\ell, k = 0, 1, \ldots, L-1$. Here the zeroth polyphase component of $H_{\ell+1}(z)G_\ell(z)$ is z^{-1} for $\ell = 0, 1, \ldots, L-2$, the zeroth polyphase component of $H_0(z)G_{L-1}(z)$ is 1, and the zeroth polyphase component of $H_k(z)G_\ell(z)$ is 0 for all other cases. As a result, a simplified equivalent representation of Figure 47(b) is as shown in Figure 48.

The transfer matrix characterizing the transmultiplexer is thus given by

$$\mathbf{F}(z) = \mathbf{E}(z) \begin{bmatrix} \mathbf{0} & 1 \\ z^{-1}\mathbf{I}_{L-1} & \mathbf{0} \end{bmatrix} \mathbf{R}(z), \tag{117}$$

where $\mathbf{I}_{L-1}$ is an $(L-1)\times(L-1)$ identity matrix. Now, for a perfect reconstruction system, it is sufficient to ensure that

$$\mathbf{F}(z) = dz^{-n_o}\mathbf{I}_L, \tag{118}$$

where n_o is a positive integer. From Eqs. (117) and (118) we arrive at the condition for perfect reconstruction in terms of the polyphase components as

$$\mathbf{R}(z)\mathbf{E}(z) = dz^{-m_o} \begin{bmatrix} \mathbf{0} & \mathbf{I}_{L-1} \\ z^{-1} & 0 \end{bmatrix}, \tag{119}$$

[39] R.D. Koilpillai, T.Q. Nguyen, and P.P. Vaidyanathan, Some results in the theory of crosstalk-free transmultiplexers, *IEEE Trans. on Signal Processing*, 39:2174–2183, October 1991.

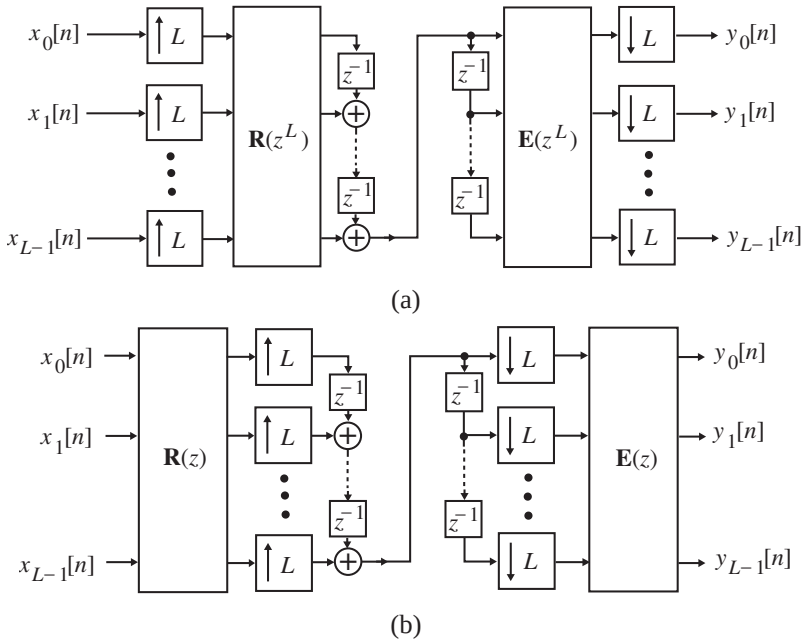


Figure 47: (a) Polyphase representation of the L-channel transmultiplexer and (b) its computationally efficient realization.

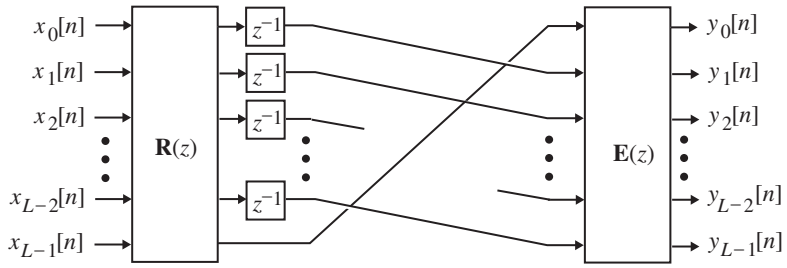


Figure 48: Simplified equivalent circuit of Figure 47.

where m_o is a suitable positive integer.

It is possible to develop a perfect reconstruction transmultiplexer from a perfect reconstruction QMF bank with analysis filters $H_\ell(z)$ and synthesis filters $G_\ell(z)$, with a distortion transfer function given by $T(z) = dz^{-K}$, where d is a nonzero constant and K is a positive integer. It can be shown that a perfect reconstruction transmultiplexer can then be designed using the analysis filters $H_\ell(z)$ and synthesis filters $z^{-R}G_\ell(z)$, where R is a positive integer less than L such that $R + K$ is a multiple of L. We illustrate this approach in Example 11.[37]

EXAMPLE 11 Design of a Perfect Reconstruction Transmultiplexer

Consider the perfect reconstruction analysis/synthesis filter bank of Example 14.8 with an input–output

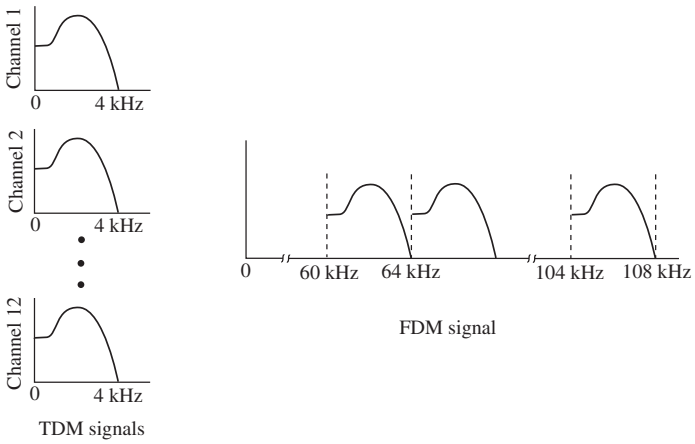


Figure 49: Spectrums of TDM signals and the FDM signal.

relation $y[n] = 4x[n-2]$. In this case, the analysis and synthesis filters are given by

$$\begin{aligned} H_0(z) &= 1 + z^{-1} + z^{-2}, \quad & H_1(z) &= 1 - z^{-1} + z^{-2}, \quad & H_2(z) &= 1 - z^{-2}, \\ G_0(z) &= 1 + 2z^{-1} + z^{-2}, \quad & G_1(z) &= 1 - 2z^{-1} + z^{-2}, \quad & G_2(z) &= -2 + 2z^{-2}. \end{aligned}$$

Here, $d = 4$ and $K = 2$. We thus choose $R = 1$ so that $R + K = 3$. The synthesis filters of the transmultiplexer are thus given by $z^{-1}G_\ell(z)$.

We now examine the products $z^{-1}G_\ell(z)H_k(z)$, for $\ell, k = 0, 1, 2$, and determine their zeroth polyphase components. Thus,

$$z^{-1}G_0(z)H_0(z) = z^{-1} + 3z^{-2} + 4z^{-3} + 3z^{-4} + z^{-5},$$

whose zeroth polyphase component is given by $4z^{-1}$, and hence, $y_0[n] = 4x_0[n-1]$. Likewise,

$$z^{-1}G_1(z)H_1(z) = z^{-1} - 3z^{-2} + 4z^{-3} - 3z^{-4} + z^{-5},$$

with a zeroth polyphase component $4z^{-1}$, resulting in $y_1[n] = 4x_1[n-1]$. Similarly,

$$z^{-1}G_2(z)H_2(z) = -2z^{-1} + 4z^{-3} - 2z^{-5},$$

whose zeroth polyphase component is again $4z^{-1}$, implying $y_2[n] = 4x_2[n-1]$. It can be shown that the zeroth polyphase components for all other products $z^{-1}G_\ell(z)H_k(z)$, with $\ell \neq k$, is 0, indicating a total absence of cross-talk between channels.

In a typical TDM-to-FDM format translation, 12 digitized speech signals are interpolated by a factor of 12, modulated by single-sideband modulation, digitally summed, and then converted into an FDM analog signal by D/A conversion. At the receiving end, the analog signal is converted into a digital signal by A/D conversion and passed through a bank of 12 single-sideband demodulators whose outputs are then decimated, resulting in the low-frequency speech signals. The speech signals have a bandwidth of 4 kHz and are sampled at an 8-kHz rate. The FDM analog signal occupies the band 60 kHz to 108 kHz, as illustrated in Figure 49. The interpolation and the single-sideband modulation can be performed by up-sampling and appropriate filtering. Likewise, the single-sideband demodulation and the decimation can be implemented by appropriate filtering and down-sampling.

10 Discrete Multitone Transmission of Digital Data

Binary data are normally transmitted serially as a pulse train, as indicated in Figure 50(a). However, in order to faithfully extract the information transmitted, the receiver requires complex equalization procedures to compensate for channel imperfection and to make full use of the channel bandwidth. For example, the pulse train of Figure 50(a) arriving at the receiver may appear as indicated in Figure 50(b). To alleviate the problems encountered with the transmission of data as a pulse train, frequency-division multiplexing with overlapping subchannels has been proposed. In such a system, each binary digit a_r, $r = 0, 1, 2, \ldots, N-1$, modulates a subcarrier sinusoidal signal $\cos(2\pi rt/T)$, as indicated in Figure 50(c), for the transmission of the data of Figure 50(a), and then the modulated subcarriers are summed and transmitted as one composite analog signal. At the receiver, the analog signal is passed through a bank of coherent demodulators whose outputs are tested to determine the digits transmitted. This is the basic idea behind the multicarrier modulation/demodulation scheme for digital data transmission.

A widely used form of the multicarrier modulation is the discrete multitone transmission (DMT) scheme in which the modulation and demodulation processes are implemented via the discrete Fourier transform (DFT), efficiently realized using fast Fourier transform (FFT) methods. This approach leads to an all-digital system, eliminating the arrays of sinusoidal generators and the coherent demodulators.[40][41]

We outline here the basic idea behind the DMT scheme. Let $\{a_k[n]\}$ and $\{b_k[n]\}$, $0 \le k \le M-1$, be two $M-1$ real-valued data sequences operating at a sampling rate of F_T that are to be transmitted. Define a new set of complex sequences $\{\alpha_k[n]\}$ of length $N = 2M$ according to

$$\alpha_k[n] = \begin{cases} a_0[n], & k = 0, \\ a_k[n] + jb_k[n], & 1 \le k \le \frac{N}{2} - 1, \\ b_0[n], & k = \frac{N}{2}, \\ a_{N-k}[n] - jb_{N-k}[n], & \frac{N}{2} + 1 \le k \le N - 1. \end{cases} \tag{120}$$

We apply an inverse DFT, and the above set of N sequences is transformed into another new set of N signals $\{u_\ell[n]\}$, given by

$$u_\ell[n] = \frac{1}{N} \sum_{k=0}^{N-1} \alpha_k[n] W_N^{-\ell k}, \qquad 0 \le \ell \le N-1, \tag{121}$$

where $W_N = e^{-j2\pi/N}$. Note that the method of generation of the complex sequence set $\{\alpha_k[n]\}$ ensures that its IDFT $\{u_\ell[n]\}$ will be a real sequence. Each of these N signals is then upsampled by a factor of N and time-interleaved, generating a composite signal $\{x[n]\}$ operating at a rate of NF_T that is assumed to be equal to $2F_c$. The composite signal is converted into an analog signal $x_a(t)$ by passing it through a D/A converter, followed by an analog reconstruction filter. The analog signal $x_a(t)$ is then transmitted over the channel.

At the receiver, the received analog signal $y_a(t)$ is passed through an analog anti-aliasing filter and then converted into a digital signal $\{y[n]\}$ by an S/H circuit, followed by an A/D converter operating at a rate of $NF_T = 2F_c$. The received digital signal is then deinterleaved by a delay chain containing $N-1$ unit delays, whose outputs are next down-sampled by a factor of N, generating the set of signals $\{v_\ell[n]\}$.

[40] A. Peled and A. Ruiz, Frequency domain data transmission using reduced computational complexity algorithms, In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 964–967, Denver CO, April 1980.

[41] J.M. Cioffi, *A multicarrier primer*, ANSI T1E1.4 Committee Contribution, Boca Raton FL, November 1991.

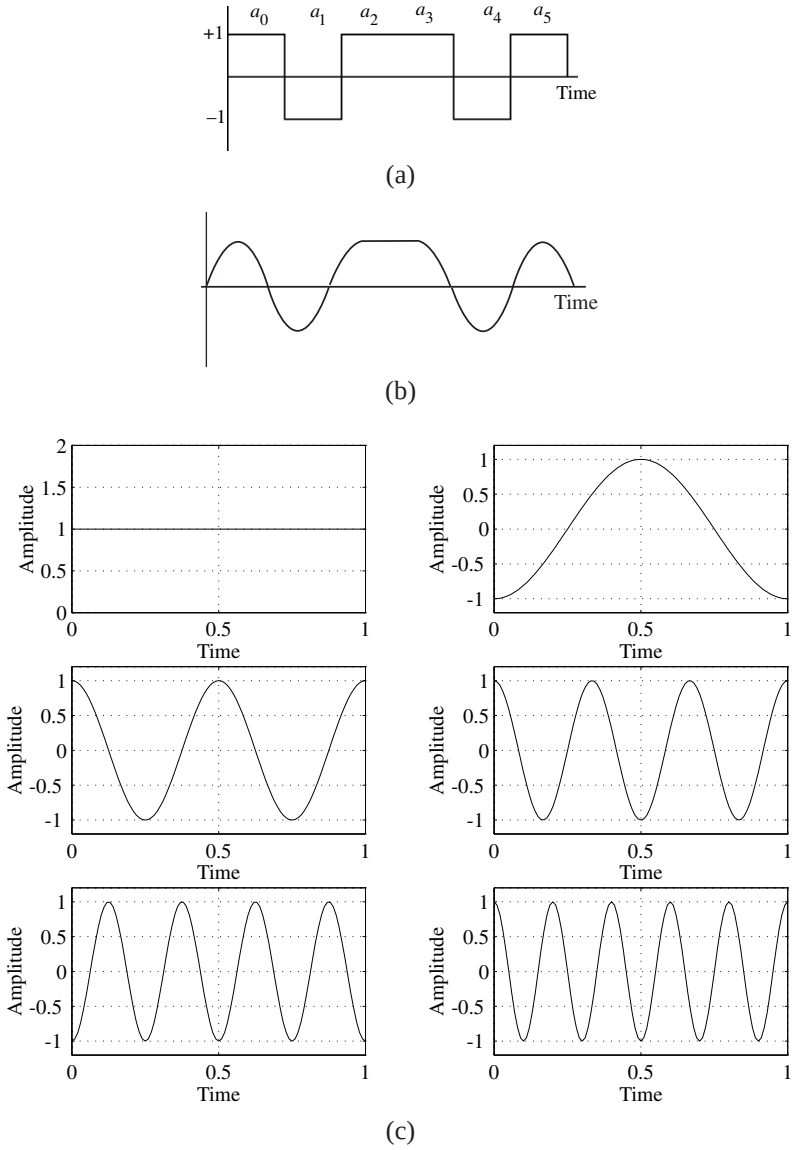


Figure 50: (a) Serial binary data stream, (b) baseband serially transmitted signal at the receiver, and (c) signals generated by modulating a set of subcarriers by the digits of the pulse train in (a).

Applying the DFT to these N signals, we finally arrive at N signals $\{\beta_k[n]\}$

$$\beta_k[n] = \sum_{\ell=0}^{N-1} v_\ell[n] W_N^{\ell k}, \qquad 0 \le k \le N-1. \tag{122}$$

Figure 51 shows schematically the overall DMT scheme. If we assume the frequency response of the channel to have a flat passband and assume the analog reconstruction and anti-aliasing filters to be ideal

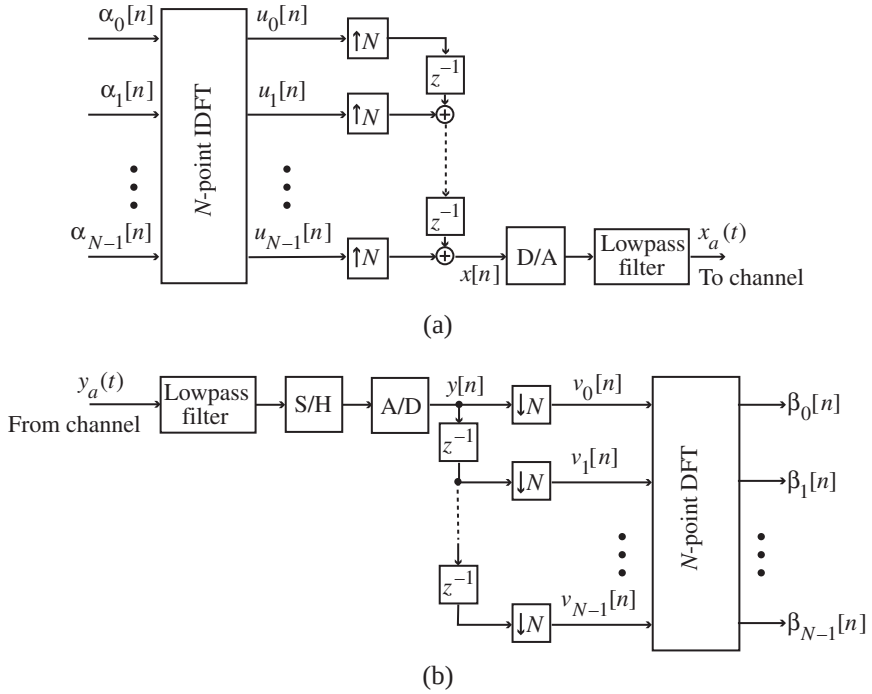


Figure 51: The DMT scheme: (a) transmitter and (b) receiver.

lowpass filters, then neglecting the nonideal effects of the D/A and the A/D converters, we can assume $y[n] = x[n]$. Hence, the interleaving circuit of the DMT structure at the transmitting end connected to the deinterleaving circuit at the receiving end is identical to the same circuit in the transmultiplexer structure of Figure 47(b) (with $L = N$). From the equivalent representation given in Figure 48, it follows that

$$\begin{aligned} v_k[n] &= u_{k-1}[n-1], \qquad 0 \le k \le N-2, \\ v_0[n] &= u_{N-1}[n], \end{aligned} \tag{123}$$

or, in other words,

$$\begin{aligned} \beta_k[n] &= \alpha_{k-1}[n-1], \qquad 0 \le k \le N-2, \\ \beta_0[n] &= \alpha_{N-1}[n]. \end{aligned} \tag{124}$$

Transmission channels, in general, have a bandpass frequency response $H_{\text{ch}}(f)$, with a magnitude response dropping to zero at some frequency F_c. In some cases, in the passband of the channel, the magnitude response, instead of being flat, drops very rapidly outside its passband, as indicated in Figure 52. For reliable digital data transmission over such a channel and its recovery at the receiving end, the channel's frequency response needs to be compensated by essentially a highpass equalizer at the receiver. However, such an equalization also amplifies high-frequency noise that is invariably added to the data signal as it passes through the channel.

For a large value of the DFT length N, the channel can be assumed to be composed of a series of contiguous narrow-bandwidth bandpass subchannels. If the bandwidth is reasonably narrow, the corresponding bandpass subchannel can be considered to have an approximately flat magnitude response, as

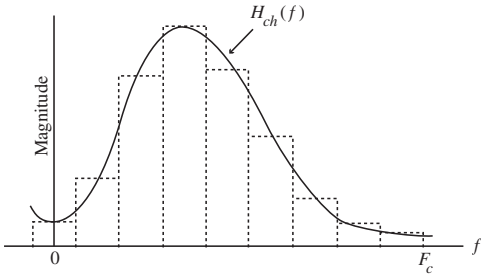


Figure 52: Frequency response of a typical band-limited channel.

indicated by the dotted lines in Figure 52, and the channel can be approximately characterized by a single complex number given by the value of its frequency response at $\omega = 2\pi k/N$. The values can be determined by first transmitting a known training signal of unmodulated carriers and generating the respective channel frequency response samples. The real data samples are then divided by these complex numbers at the receiver to compensate for channel distortion.

Further details on the performance of the above DMT scheme under nonideal conditions can be found in the literature.[39, 42, 43]

11 Oversampling A/D Converter

For the digital processing of an analog continuous-time signal, the signal is first passed through a sample-and-hold circuit whose output is then converted into a digital form by means of an A/D converter. However, according to the sampling theorem, discussed in Section 3.8.1 of Text, a band-limited continuous-time signal with a lowpass spectrum can be fully recovered from its uniformly sampled version if it is sampled at a sampling frequency that is at least twice the highest frequency contained in the analog signal. If this condition is not satisfied, the original continuous-time signal cannot be recovered from its sampled version because of aliasing. To prevent aliasing, the analog signal is thus passed through an analog anti-aliasing lowpass filter prior to sampling, which enforces the condition of the sampling theorem. The passband cutoff frequency of the lowpass filter is chosen equal to the frequency of the highest signal frequency component that needs to be preserved at the output. The anti-aliasing filter also cuts off all out-of-band signal components and any high-frequency noise that may be present in the original analog signal, which otherwise would alias into the baseband after sampling. The filtered signal is then sampled at a rate that is at least twice that of the cutoff frequency.

Let the signal band of interest be the frequency range $0 \leq f \leq F_m$. Then, the Nyquist rate is given by $F_N = 2F_m$. Now, if the sampling rate F_T is the same as the Nyquist rate, we need to use before the sampler an anti-aliasing lowpass filter with a very sharp cutoff in its frequency response, satisfying the requirements as given by Eq. (A.35) in Appendix A of Text.[44] This requires the design of a very high-order anti-aliasing filter structure built with high-precision analog components, and it is usually difficult to implement such a filter in VLSI technology. Moreover, such a filter also introduces undesirable phase

[42] J.A.C. Bingham, Multicarrier modulation for data transmission: An idea whose time has come, *IEEE Communications Magazine*, pages 5–14, May 1990.

[43] K. Shenoi, *Digital Signal Processing in Telecommunication*, Prentice Hall, Englewood Cliffs NJ, 1995.

[44] Recall that $F_T = 1/T$, where T is the sampling period.

distortion in its output. An alternative approach is to sample the analog signal at a rate much higher than the Nyquist rate, use a fast low-resolution A/D converter, and then decimate the digital output of the converter to the Nyquist rate. This approach relaxes the sharp cutoff requirements of the analog anti-aliasing filter, resulting in a simpler filter structure that can be built using low-precision analog components while requiring fast, more complex digital signal processing hardware at later stages. The overall structure is not only amenable to VLSI fabrication but also can be designed to provide linear-phase response in the signal band of interest.

The oversampling approach is an elegant application of multirate digital signal processing and is increasingly being employed in the design of high-resolution A/D converters for many practical systems.[45,46] In this section, we analyze the quantization noise performance of the conventional A/D converter and show analytically how the oversampling approach decreases the quantization noise power in the signal band of interest.[44] We then show that further improvement in the noise performance of an oversampling A/D converter can be obtained by employing a sigma-delta ($\Sigma\Delta$) quantization scheme. For simplicity, we restrict our discussion to the case of a basic first-order sigma-delta quantizer.

To illustrate the noise performance improvement property, consider a b-bit A/D converter operating at F_T Hz. Now, for a full-scale peak-to-peak input analog voltage of $R_{\rm FS}$, the smallest voltage step represented by b bits is

$$\Delta V = \frac{R_{\rm FS}}{2^b - 1} \cong \frac{R_{\rm FS}}{2^b}. \tag{125}$$

From Eq. (12.70) of Text, the rms quantization noise power σ_e^2 of the error voltage, assuming a uniform distribution of the error between $-\Delta V/2$ and $\Delta V/2$, is given by

$$\sigma_e^2 = \frac{(\Delta V)^2}{12}. \tag{126}$$

The rms noise voltage, given by σ_e, therefore has a flat spectrum in the frequency range from 0 to $F_T/2$.

The noise power per unit bandwidth, called the *noise density*, is then given by

$$P_{e,n} = \frac{(\Delta V)^2/12}{F_T/2} = \frac{(\Delta V)^2}{6F_T}. \tag{127}$$

A plot of the noise densities for two different sampling rates is shown in Figure 53, where the shaded portion indicates the signal band of interest. As can be seen from this figure, the total amount of noise in the signal band of interest for the high sampling rate case is smaller than that for the low sampling rate case. The total noise in the signal band of interest, called the *in-band noise power*, is given by

$$P_{\rm total} = \frac{(R_{\rm FS}/2^b)^2}{12} \cdot \frac{F_m}{F_T/2}. \tag{128}$$

It is interesting to compute the needed wordlength β of the A/D converter operating at the Nyquist rate in order that its total noise in the signal band of interest be equal to that of a b-bit A/D converter operating at a higher rate. Substituting $F_T = 2F_m$ and replacing b with β in Eq. (128), we arrive at

$$P_{\rm total} = \frac{(R_{\rm FS}/2^\beta)^2}{12} = \frac{(R_{\rm FS}/2^b)^2}{12} \cdot \frac{F_m}{F_T/2}, \tag{129}$$

[45] J.C. Candy and G.C. Temes. Oversampling methods for A/D and D/A conversion. In J.C. Candy and G.C. Temes, editors, *Oversampling Delta-Sigma Data Converters*, pages 1–25, IEEE Press, New York NY, 1992.

[46] M.E. Frerking. *Digital Signal Processing in Communication Systems*, Van Nostrand Reinhold, New York NY, 1994.

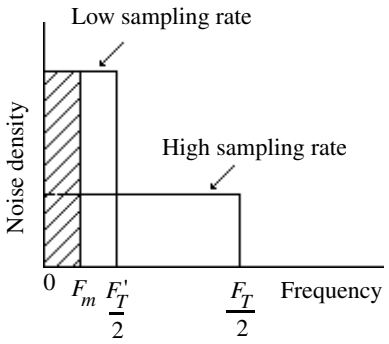


Figure 53: A/D converter noise density.

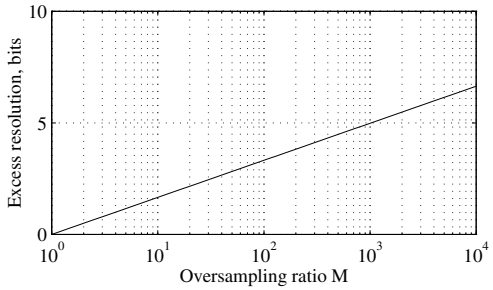


Figure 54: Excess resolution as a function of the oversampling ratio M.

which leads to the desired relation

$$\beta = b + \frac{1}{2}\log_2 M, \tag{130}$$

where $M = F_T/2F_m$ denotes the *oversampling ratio* (OSR). Thus, $\beta - b$ denotes the increase in the resolution of a b-bit converter whose oversampled output is filtered by an ideal brick-wall lowpass filter. A plot of the increase in resolution as a function of the oversampling ratio is shown in Figure 54. For example, for an OSR of $M = 1000$, an 8-bit oversampling A/D converter has an effective resolution equal to that of a 13-bit A/D converter operating at the Nyquist rate. Note that Eq. (130) implies that the increase in the resolution is $\frac{1}{2}$-bit per doubling of the OSR.

We now illustrate the improvement in the noise performance obtained by employing a sigma-delta ($\Sigma\Delta$) quantization scheme. The sigma-delta A/D converter is shown in block-diagram form in Figure 55 for convenience. This figure also indicates the sampling rates at various stages of the structure. It should be noted here that the 1-bit output samples of the quantizer after decimation become b-bit samples at the output of the sigma-delta A/D converter due to the filtering operations involving b-bit multiplier coefficients of the Mth-band digital lowpass filter.

Since the oversampling ratio M is typically very large in practice, the sigma-delta A/D converter is most useful in low-frequency applications such as digital telephony, digital audio, and digital spectrum analyzers. For example, Figure 56 shows the block diagram of a typical compact disk encoding system used to convert the input analog audio signal into a digital bit stream that is then applied to generate the master disk.[47] Here, the oversampling sigma-delta A/D converter employed has a typical input sampling

[47] J.P.J. Heemskerk and K.A.S. Immink. Compact disc: System aspects and modulation. *Philips Technical Review*, 40(6):157–

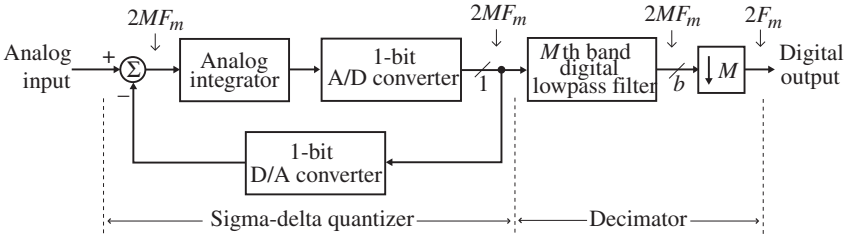


Figure 55: Oversampling sigma-delta A/D converter structure.

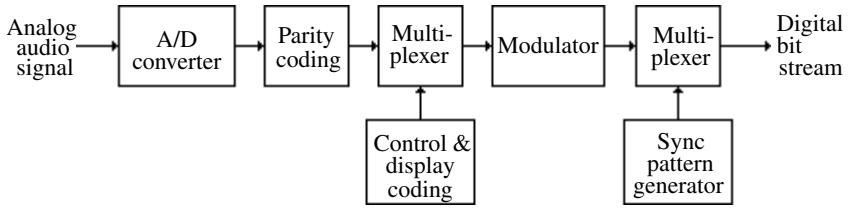


Figure 56: Compact disk encoding system for one channel of a stereo audio.

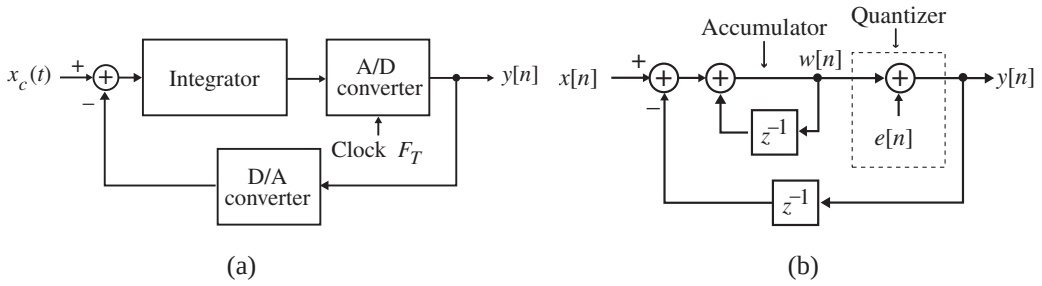


Figure 57: Sigma-delta quantization scheme: (a) quantizer and (b) its discrete-time equivalent.

rate of 3175.2 kHz and an output sampling rate of 44.1 kHz.[48]

To understand the operation of the sigma-delta A/D converter of Figure 55, we need to study the operation of the sigma-delta quantizer shown in Figure 57(a). To this end, it is convenient to use the discrete-time equivalent circuit of Figure 57(b), where the integrator has been replaced with an accumulator.[49] Here, the input $x[n]$ is a discrete-time sequence of analog samples developing an output sequence of binary-valued samples $y[n]$. From this diagram, we observe that, at each discrete instant of time, the circuit forms the difference (Δ) between the input and the delayed output, which is accumulated by a summer (Σ) whose output is then quantized by a one-bit A/D converter, that is., a comparator.

Even though the input–output relation of the sigma-delta quantizer is basically nonlinear, the low-

165, 1982.

[48] J.J. Van der Kam. A digital "decimating" filter for analog-to-digital conversion of hi-fi audio signals. *Philips Technical Review*, 42:230–238, 1986.

[49] In practice, the integrator is implemented as a discrete-time switched-capacitor circuit.

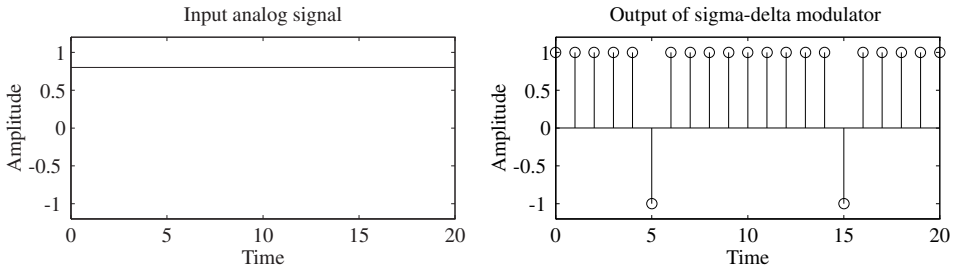


Figure 58: (a) Input and (b) output waveforms of the sigma-delta quantizer of Figure 57(a) for a constant input.

frequency content of the input $x_c(t)$ can be recovered from the output $y[n]$ by passing it through a digital lowpass filter. This property can be easily shown for a constant input analog signal $x_a(t)$ with a magnitude less than $+1$. In this case, the output $w[n]$ of the accumulator is a bounded sequence with sample values equal to either -1 or $+1$. This can happen only if the input to the accumulator has an average value of zero. Or in other words, the average value of $w[n]$ must be equal to the average value of the input $x[n]$.[50] Examples 12 and 13 illustrate the operation of a sigma-delta quantizer.

EXAMPLE 12 Sigma-Delta Quantization of a Constant Amplitude Signal

We first consider the operation for the case of a constant input signal using MATLAB. To this end, we can use Program 11 in Section 14. The plots generated by this program are the input and output waveforms of the sigma-delta quantizer of Figure 57(a) and are shown in Figure 58. The program also prints the average value of the output as indicated below:

```
Average value of output is =
    0.8095
```

which is very close to the amplitude 0.8 of the constant input. It can be easily verified that the average value of the output gets closer to the amplitude of the constant input as the length of the input increases.

EXAMPLE 13 Sigma-Delta Quantization of a Sinusoidal Signal

We now verify the operation of the sigma-delta A/D converter for a sinusoidal input of frequency 0.01 Hz using MATLAB. To this end, we make use of the Program 12 in Section 14. Because of the short length of the input sequence, the filtering operation is performed here in the DFT domain.[48] Figure 59 shows the input and output waveforms of the sigma-delta quantizer of Figure 57(a) for a sinusoidal input. Figure 60 depicts the lowpass filtered version of the output signal shown in Figure 59(b). As can be seen from these figures, the filtered output is nearly an exact replica of the input.

It follows from Figure 57(b) that the output $y[n]$ of the quantizer is given by

$$y[n] = w[n] + e[n], \tag{131}$$

where

$$w[n] = x[n] - y[n-1] + w[n-1]. \tag{132}$$

[50] R. Schreier. *Noise-Shaped Coding.* PhD thesis, University of Toronto, Toronto Canada, 1991.

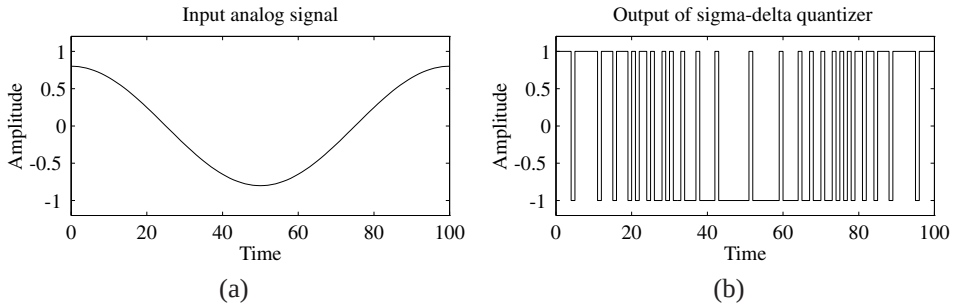


Figure 59: Input and output waveforms of the sigma-delta quantizer of Figure 57(a) with a sine wave input.

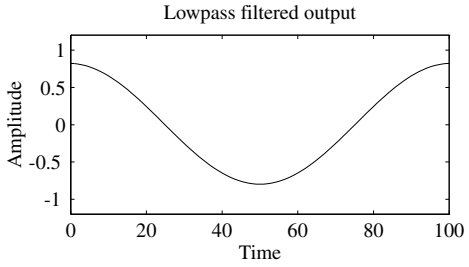


Figure 60: The lowpass filtered version of the waveform of Figure 59(b).

From Eqs. (131) and (132), we obtain, after some algebra,

$$y[n] = x[n] + (e[n] - e[n-1]), \tag{133}$$

where the quantity inside the parentheses represents the noise due to sigma-delta modulation. The noise transfer function is simply $G(z) = (1 - z^{-1})$. The power spectral density of the modulation noise is therefore given by

$$P_y(f) = \left|G(e^{j2\pi fT})\right|^2 P_e(f) = 4\sin^2\left(\frac{2\pi fT}{2}\right) P_e(f), \tag{134}$$

where we have assumed the power spectral density $P_e(\omega)$ of the quantization noise to be the one-sided power spectral density defined for positive frequencies only. For a random signal input $x[n]$, $P_e(f)$ is constant for all frequencies and is given by

$$P_e(f) = \frac{(\Delta V)^2/12}{F_T/2}. \tag{135}$$

Substituting Eq. (135) in Eq. (134), we arrive at the power spectral density of the output noise, given by

$$P_y(f) = \frac{2}{3}\frac{(\Delta V)^2}{F_T}\sin^2(\pi fT). \tag{136}$$

The noise-shaping provided by the sigma-delta quantizer is similar to that encountered in the first-order error-feedback structures of Section 9.10.1 and shown in Figure 9.42. For a very large OSR, as is usually

the case, the frequencies in the signal band of interest are much smaller than F_T, the sampling frequency. Thus, we can approximate $P_y(f)$ of Eq. (136) as

$$P_y(f) \cong \tfrac{2}{3}\frac{(\Delta V)^2}{F_T}(\pi f T)^2 = \tfrac{2}{3}\pi^2(\Delta V)^2 T^3 f^2, \qquad f << F_T. \tag{137}$$

From Eq. (137), the in-band noise power of the sigma-delta A/D converter is thus given by

$$P_{\text{total,sd}} = \int_0^{F_m} P_y(f)\,df = \tfrac{2}{3}\pi^2(\Delta V)^2 T^3 \int_0^{F_m} f^2\,df = \frac{2}{9}\pi^2(\Delta V)^2 T^3 (F_m)^3. \tag{138}$$

It is instructive to compare the noise performance of the sigma-delta A/D converter with that of a direct oversampling A/D converter operating at a sampling rate of F_T with a signal band of interest from dc to F_m. From Eq. (129), the in-band noise power of the latter is given by

$$P_{\text{total,os}} = \tfrac{1}{6}(\Delta V)^2 T F_m. \tag{139}$$

The improvement in the noise performance is therefore given by

$$10\log_{10}\left(\frac{P_{\text{total,os}}}{P_{\text{total,sd}}}\right) = 10\log_{10}\left(\frac{3M^2}{\pi^2}\right) = -5.1718 + 20\log_{10}(M)\,\text{dB}, \tag{140}$$

where we have used $M = F_T/2F_m$ to denote the OSR. For example, for an OSR of $M = 1000$, the improvement in the noise performance using the sigma-delta modulation scheme is about 55 dB. In this case, the increase in the resolution is about 1.5 bits per doubling of the OSR.

The improved noise performance of the sigma-delta A/D converter results from the shape of $\left|G(e^{j2\pi fT})\right|$, which decreases the noise power spectral density in-band ($0 \le f \le F_m$), while increasing it outside the signal band of interest ($f > F_m$). Since this type of converter also employs oversampling, it requires a less stringent analog anti-aliasing filter.

The A/D converter of Figure 55 employs a single-loop feedback and is often referred to as a first-order sigma-delta converter. Multiple feedback loop modulation schemes have been advanced to reduce the in-band noise further. However, the use of more than two feedback loops may result in unstable operation of the system, and care must be taken in the design to ensure stable operation.[43]

As indicated in Figure 55, the quantizer output is passed through an Mth-band lowpass digital filter whose output is then down-sampled by a factor of M to reduce the sampling rate to the desired Nyquist rate. The function of the digital lowpass filter is to eliminate the out-of-band quantization noise and the out-of-band signals that would be aliased into the passband by the down-sampling operation. As a result, the filter must exhibit a very sharp cutoff frequency response with a passband edge at F_m. This necessitates the use of a very high-order digital filter. In practice, it is preferable to use a filter with a transfer function having simple integer-valued coefficients to reduce the cost of hardware implementation and to permit all multiplication operations to be carried out at the down-sampled rate. In addition, most applications require the use of linear-phase digital filters, which can be easily implemented using FIR filters.

Further details on first- and higher-order sigma-delta converters can be found in Candy and Temes.[41]

12 Oversampling D/A Converter

As indicated earlier in Section 3.8 of Text, the digital-to-analog conversion process consists of two steps: the conversion of input digital samples into a staircase continuous-time waveform by means of a D/A

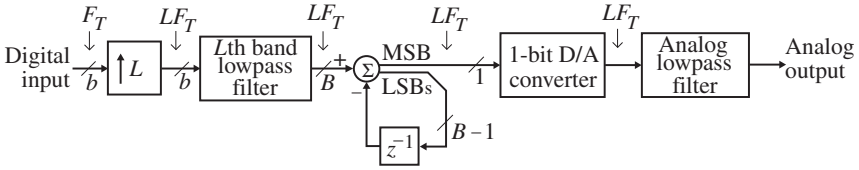


Figure 61: Block diagram representation of an oversampling sigma-delta D/A converter.

converter with a zero-order hold at its output, followed by an analog lowpass reconstruction filter. If the sampling rate F_T of the input digital signal is the same as the Nyquist rate, the analog lowpass reconstruction filter must have a very sharp cutoff in its frequency response, satisfying the requirements of Eq. (A.38) in Appendix A of Text. As in the case of the anti-aliasing filter, this involves the design of a very high-order analog reconstruction filter requiring high-precision analog circuit components. To get around the above problem, here also an oversampling approach is often used, in which case a wide transition band can be tolerated in the frequency response of the reconstruction filter allowing its implementation using low-precision analog circuit components, while, requiring a more complex digital interpolation filter at the front end.

Further improvement in the performance of an oversampling D/A converter is obtained by employing a digital sigma-delta 1-bit quantizer at the output of the digital interpolator, as indicated in Figure 61 for convenience.[51,52] The quantizer extracts the MSB from its input and subtracts the remaining LSBs, the quantization noise, from its input. The MSB output is then fed into a 1-bit D/A converter and passed through an analog lowpass reconstruction filter to remove all frequency components beyond the signal band of interest. Since the signal band occupies a very small portion of the baseband of the high-sample-rate signal, the reconstruction filter in this case can have a very wide transition band, permitting its realization with a low-order filter that, for example, can be implemented using a Bessel filter to provide an approximately linear phase in the signal band.[53]

The spectrum of the quantized 1-bit output of the digital sigma-delta quantizer is nearly the same as that of its input. Moreover, it also shapes the quantization noise spectrum by moving the noise power out of the signal band of interest. To verify this result analytically, consider the sigma-delta quantizer shown separately in Figure 62. It follows from this figure that the input–output relation of the quantizer is given by

$$y[n] - e[n] = x[n] - e[n-1],$$

or, equivalently, by

$$y[n] = x[n] + e[n] - e[n-1], \tag{141}$$

where $y[n]$ is the MSB of the nth sample of the adder output, and $e[n]$ is the nth sample of the quantization noise composed of all bits except the MSB. From Eq. (141), it can be seen that the transfer function of the quantizer with no quantization noise is simply unity, and the noise transfer function is given by $G(z) = 1 - z^{-1}$, which is the same as that for the first-order sigma-delta modulator employed in the oversampling A/D converter discussed in the previous section.

[51] J.C. Candy and A-N. Huynh. Double interpolation for digital-to-analog conversion. *IEEE Trans. on Communications*, COM-34:77–81, January 1986.

[52] L.E. Larson and G.C. Temes. Signal conditioning and interface circuits. In S.K. Mitra and J.F. Kaiser, editors, *Handbook for Digital Signal Processing*, chapter 10, pages 677–720. Wiley-Interscience, New York NY, 1993.

[53] See Section **??**.

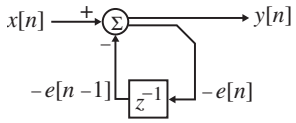


Figure 62: The sigma-delta quantizer.

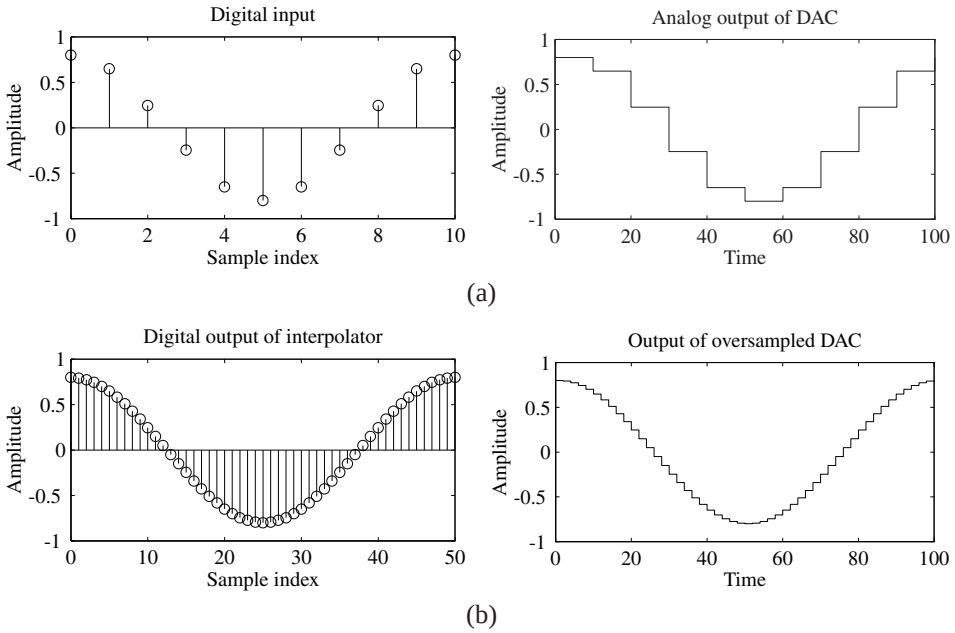


Figure 63: Input and output signals of (a) lower-rate D/A converter and (b) oversampling D/A converter.

Examples 14 and 15 illustrate the operation of a sigma-delta D/A converter for a discrete-time sinusoidal input sequence.

EXAMPLE 14 Illustration of the Oversampling D/A Conversion

Let the input to the D/A converter be a sinusoidal sequence of frequency 100 Hz operating at a sampling rate F_T of 1 kHz. Figure 63(a) shows the digital input sequence and the analog output generated by a D/A converter operating at F_T from this input. Figure 63(b) depicts the interpolated sinusoidal sequence operating at a higher sampling rate of 5 kHz, obtained by passing the low-sampling-rate sinusoidal signal through a factor-of-5 digital interpolator and the corresponding analog output generated by a D/A converter operating at $5F_T$ rate. If we compare the two D/A converter outputs, we can see that the staircase waveform of the oversampling D/A converter output is much smoother with smaller jumps than that of the lower-rate D/A converter output. Thus, the oversampling D/A converter output has considerably smaller high-frequency components in contrast to the lower-rate D/A converter. This fact can be easily verified by examining their spectra.

The high-frequency components in the baseband outside the signal band of interest can be removed by passing the D/A converter output through an analog lowpass filter, which also eliminates any leftover replicas of the baseband not completely removed by the zero-order hold in the D/A converter. Since

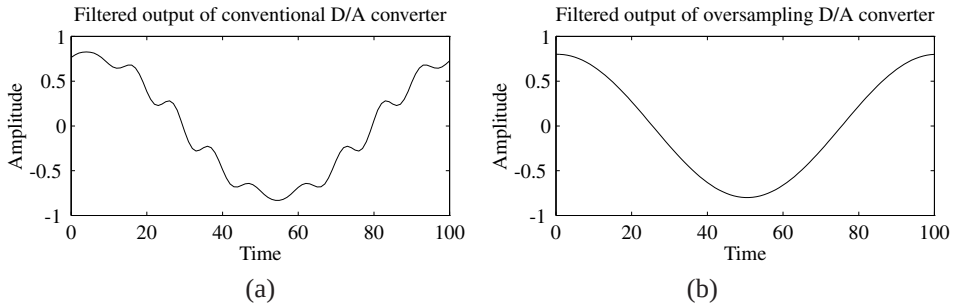


Figure 64: Lowpass filtered output signals of (a) conventional D/A converter and (b) oversampling D/A converter.

the signal band of interest occupies a small portion of the baseband, the replicas of the signal band immediately outside the baseband are widely separated from the signal band inside the baseband. Hence, the lowpass filter can be designed with a very wide transition band. Moreover, due to reduced high-frequency components in the D/A converter output caused by oversampling, the stopband attenuation also does not have to be very large. On the other hand, the replicas of the signal band in the spectrum of the output of the low-rate D/A converter are closely spaced, and the high-frequency components are relatively large in amplitudes. In this case, the lowpass filter must have a sharp cutoff with much larger stopband attenuation to effectively remove the undesired components in the D/A converter output.

Figure 64 shows the filtered outputs of the conventional lower-rate and oversampled D/A converters when the same lowpass filter with a wide transition band is used in both cases. As can be seen from this figure, the analog output in the case of the low-rate D/A converter still contains some high-frequency components, while that in the case of the oversampled D/A converter is very close to a perfect sinusoidal signal. A much better output response is obtained in the case of a conventional D/A converter if a sharp cutoff lowpass filter is employed, as indicated in Figure 65.

EXAMPLE 15 Illustration of the Operation of the Sigma-Delta D/A Converter Using MATLAB

In this example, we verify using MATLAB the operation of the sigma-delta D/A converter for a sinusoidal input sequence of frequency 100 Hz operating at a sampling rate F_T of 5 kHz. The signal is clearly oversampled since the sampling rate is much higher than the Nyquist rate of 200 Hz. Program 13 in Section 14 first generates the input digital signal, then generates a two-valued digital signal by quantizing the output of the sigma-delta quantizer, and finally, develops the output of the D/A converter by lowpass filtering the quantized output. As in the case of the sigma-delta converter of Example 14, the filtering operation here has also been performed in the DFT domain due to the short length of the input sequence.[48]

Figure 66 shows the digital input signal, the quantized digital output of the sigma-delta quantizer, and the filtered output of the D/A converter generated by this program. As can be seen from these plots, the lowpass filtered output is nearly a scaled replica of the desired sinusoidal analog signal.

One of the most common applications of the oversampling sigma-delta D/A converter is in the compact disk (CD) player. Figure 67 shows the block diagram of the basic components in the signal processing part of a CD player, where typically a factor-of-4 oversampling D/A converter is employed for each audio channel.[54] Here, the 44.1-kHz input digital audio signal is interpolated first by a factor of 4 to the 176.4-kHz rate and then converted into an analog audio signal.

[54] D. Goedhart, R.J. Van de Plassche, and E.F. Stikvoort. Digital-to-analog conversion in playing a compact disc. *Philips Technical Review*, 40(6):174–179, 1982.

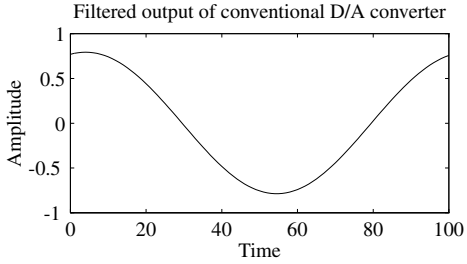


Figure 65: Filtered output signals of the conventional D/A converter employing a sharp cutoff lowpass filter.

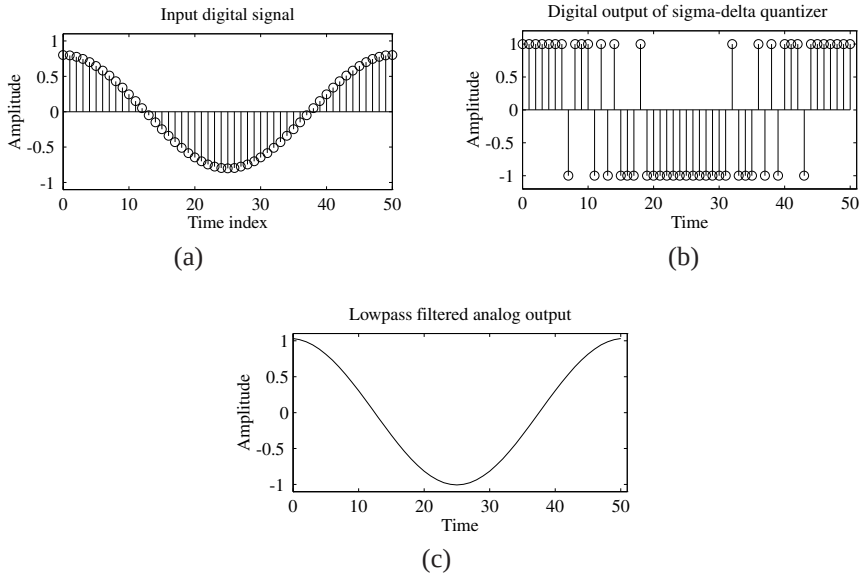


Figure 66: Input and output waveforms of the sigma-delta quantizer of Figure 62.

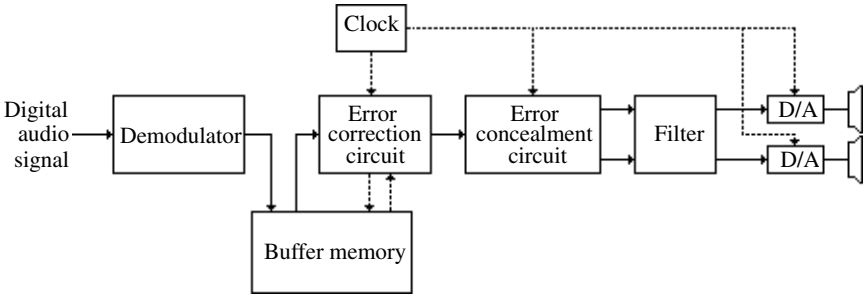


Figure 67: Signal processing part of a CD player.

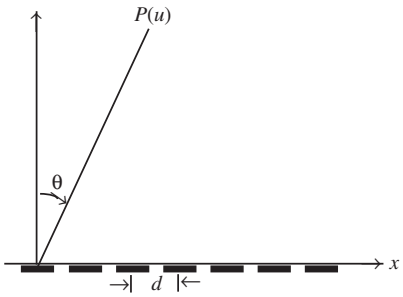


Figure 68: Uniform linear antenna array.

13 Sparse Antenna Array Design

Linear-phased antenna arrays are used in radar, sonar, ultrasound imaging, and seismic signal processing. Sparse arrays with certain elements removed are economical and, as a result, are of practical interest. There is a mathematical similarity between the far-field radiation pattern for a linear antenna array of equally spaced elements and the frequency response of an FIR filter. This similarity can be exploited to design sparse arrays with specific beam patterns. In this section, we point out this similarity and outline a few simple designs of sparse arrays. We restrict our attention here on the design of sparse arrays for ultrasound scanners.

Consider a linear array of N isotropic, equispaced elements with inter-element spacing d and located at $x_n = n \cdot d$ for $0 \leq n \leq N-1$, as shown in Figure 68. The far-field radiation pattern at an angle θ away from the broadside (i.e., the normal to the array), is given by

$$P(u) = \sum_{n=0}^{N-1} w[n] e^{j[2\pi(u/\lambda)d]n}, \tag{142}$$

where $w[n]$ is the complex excitation or weight of the nth element, λ is the wavelength, and $u = \sin\theta$. The function $P(u)$ thus can be considered as the discrete-time Fourier transform of $w[n]$, with the frequency variable given by $2\pi(u/\lambda)d$. The array element weighting $w[n]$ as a function of the element position is called the *aperture function*. For a uniformly excited array, $w[n] =$ a constant, and the grating lobes in the radiation pattern are avoided if $d \leq \lambda/2$. Typically, $d = \lambda/2$, in which case the range of u is between $-\pi$ and π. From Eq. (142), it can be seen that the expression for $P(u)$ is identical to the frequency response of an FIR filter of length N. An often used element weight is $w[n] = 1$ whose radiation pattern is this same as the frequency response of a running-sum or boxcar FIR filter.

Sparse arrays with fewer elements are obtained by removing some of the elements, which increases the interelement spacing between some consecutive pairs of elements to more than $\lambda/2$. This usually results in an increase of sidelobe levels and can possibly cause the appearance of grating lobes in the radiation pattern. However, these unwanted lobes can be reduced significantly by selecting array element locations appropriately. In the case of ultrasound scanners, a two-way radiation pattern is generated by a transmit array and a receive array. The design of such arrays is simplified by treating the problem as the design of an "effective aperture function" $w_{eff}[n]$, which is given by the convolution of the transmit

aperture function $w_T[n]$ and the receive aperture function $w_T[n]$:[55]

$$w_{eff}[n] = w_T[n] \circledast w_R[n]. \tag{143}$$

If the number of elements (including missing elements) in the the transmit and receive arrays are, respectively, L and M, then the number of elements N in a single array with an effective aperture function $w_{eff}[n]$ is $L + M - 1$. The design problem is thus to determine $w_T[n]$ and $w_T[n]$ for a desired $w_{eff}[n]$.

13.1 The Polynomial Factorization Approach

In the z-domain, Eq. (143) is equivalent to

$$P_{eff}(z) = P_T(z)P_R(z), \tag{144}$$

where

$$P_{eff}(z) = \sum_{n=0}^{N-1} w_{eff}[n]z^{-n}, \qquad P_T(z) = \sum_{n=0}^{L-1} w_T[n]z^{-n}, \qquad P_R(z) = \sum_{n=0}^{M-1} w_R[n]z^{-n}. \tag{145}$$

As a result, the sparse antenna array design problem can be formulated as the factorization of the polynomial $P_{eff}(z)$ into factors $P_T(z)$ and $P_R(z)$ with missing coefficients. We first consider the design of a uniform array for which $w_{eff}[n] = 1$. To this end, we can make use of the following factorization of $P_{eff}(z)$ for values of N that are powers-of-2:[56]

$$P_{eff}(z) = (1 + z^{-1})(1 + z^{-2}) \cdots (1 + z^{-2^{K-1}}), \tag{146}$$

where $N = 2^K$.

13.2 Uniform Effective Aperture Function

We now illustrate the application of the above factorization approach to sparse array design for the case $N = 16$; that is, $K = 4$. From Eq. (146) we then have

$$P_{eff}(z) = (1 + z^{-1})(1 + z^{-2})(1 + z^{-4})(1 + z^{-8}).$$

Three possible choices for $P_T(z)$ and $P_R(z)$ are as follows:

$$\begin{aligned}
\text{Design \#1}: \quad & P_T(z) = 1, \\
& P_R(z) = (1 + z^{-1})(1 + z^{-2})(1 + z^{-4})(1 + z^{-8}) \\
& \qquad = 1 + z^{-1} + z^{-2} + z^{-3} + z^{-4} + z^{-5} + z^{-6} + z^{-7} \\
& \qquad + z^{-8} + z^{-9} + z^{-10} + z^{-11} + z^{-12} + z^{-13} + z^{-14} + z^{-15}, \\
\text{Design \#2}: \quad & P_T(z) = 1 + z^{-1}, \\
& P_R(z) = (1 + z^{-2})(1 + z^{-4})(1 + z^{-8}) \\
& \qquad = 1 + z^{-2} + z^{-4} + z^{-6} + z^{-8} + z^{-10} + z^{-12} + z^{-14}, \\
\text{Design \#3}: \quad & P_T(z) = (1 + z^{-1})(1 + z^{-8}) = 1 + z^{-1} + z^{-8} + z^{-9}, \\
& P_R(z) = (1 + z^{-2})(1 + z^{-4}) = 1 + z^{-2} + z^{-4} + z^{-6}.
\end{aligned}$$

[55] G.R. Lockwood, P-C. Li, M. O'Donnell, and F.S. Foster. Optimizing the radiation pattern of sparse periodic linear arrays. *IEEE Trans. on Ultrasonics Ferroelectrics, and Frequency Control*, 43:7-14, January 1996.

[56] S.K. Mitra, M.K. Tchobanou, and G. Jovanovic-Dolecek. A simple approach to the design of one-dimensional sparse antenna arrays. In *Proc. IEEE International Symposium on Circuits & Systems*, May 2004, pages III-541–III-544, Vancouver, B.C., Canada.

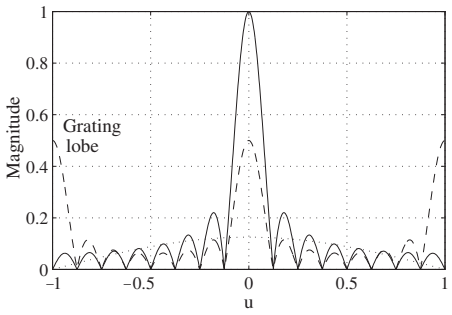


Figure 69: Radiation patterns of transmit array (dotted line), receive array (dashed line), and two-way radiation pattern (solid line). The radiation patterns have been scaled by a factor of 16 to make the value of the two-way radiation pattern at $u = 0$ unity.

Additional choices for $P_T(z)$ and $P_R(z)$ can be found elsewhere.[57]

Design #1 consists of a single-element transmit array and a 16-element nonsparse receive array and thus requires a total of 17 elements. The remaining designs given above result in sparse transmit and/or receive arrays. For example, the transmit and receive aperture functions for Design #2 are given by[56]

$$w_T[n] = \{1 \quad 1\}, \qquad w_R[n] = \{1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1\},$$

where 0 in $w_R[n]$ indicates the absence of an element and requires a total of 10 elements. Figure 69 shows the radiation patterns of the individual arrays and the two-way radiation pattern of the composite array. Note that the grating lobes in the radiation pattern of the receive array are being suppressed by the radiation pattern of the transmit array.

Most economic sparse array design with eight elements is obtained with the Design #3, requiring a total of eight elements. For example, the transmit and receive aperture functions for Design #3 are given by:[56]

$$w_T[n] = \{1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1\}, \qquad w_R[n] = \{1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1\}.$$

13.3 Linearly Tapered Effective Aperture Function

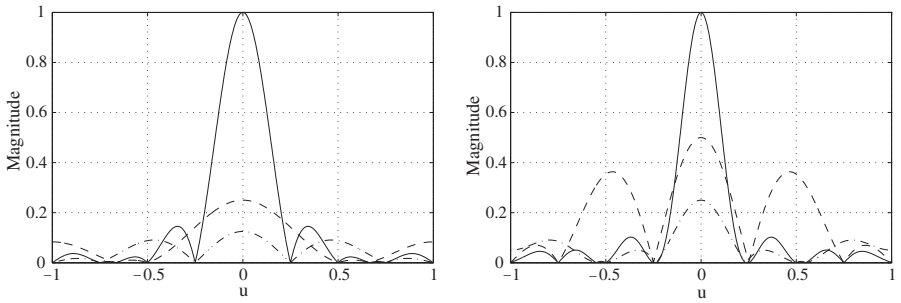
The shape of the effective aperture function can be made smoother to reduce the grating lobes by controlling the shape of the transmit and receive aperture functions. For the design of a sparse array pair with a linearly tapered effective aperture function $P_{eff}(z)$, one can choose[57]

$$P_{eff}(z) = P_1(z)P_2(z), \tag{147}$$

where

$$P_1(z) = \frac{1}{R}\sum_{n=0}^{R-1} z^{-n}, \qquad P_2(z) = \sum_{n=0}^{S-1} z^{-n}. \tag{148}$$

[57] S.K. Mitra, G. Jovanovic-Dolecek, and M.K. Tchobanou. On the design of one-dimensional sparse arrays with apodized end elements. In *Proc. 12th European Signal Processing Conference*, pages 2239-2242, Vienna, Austria, September 2004.



(a) (b)

Figure 70: Illustration of effective aperture smoothing by shaping transmit and receive aperture functions. The radiation patterns have been scaled to make the value of the two-way radiation pattern at $u = 0$ unity.

The number of elements in the effective aperture function is then $N = R + S - 1$. The number of apodized elements in the beginning and at the end of the effective aperture function is $(R-1)$ each. The values of the apodized elements are $\frac{1}{R}, \frac{2}{R}, \ldots, \frac{R-1}{R}$. Moreover, the parameter S must satisfy the condition $S > R - 1$. For a sparse antenna pair design, the value of either R or S or both must be power-of-2.

We consider the design of a linearly tapered array for $R = 3$ and $S = 8$, which results in an effective aperture function given by

$$w_{eff}[n] = \{\tfrac{1}{3} \quad \tfrac{2}{3} \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad \tfrac{2}{3} \quad \tfrac{1}{3}\}.$$

A possible design for the transmit and receive arrays is given by

$$w_T[n] = \{1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1\},$$
$$w_R[n] = \{\tfrac{1}{3} \quad \tfrac{1}{3} \quad \tfrac{2}{3} \quad \tfrac{1}{3} \quad \tfrac{1}{3}\}.$$

The corresponding scaled radiation patterns are shown in Figure 70(a).

13.4 Staircase Effective Aperture Function

Sparse antenna array pairs with a staircase effective aperture function also exhibit reduced grating lobes. For designing such arrays, there are two possible forms of the factor $P_1(z)$ in Eq. (147) [Mit2004b]. One form is for an even number of steps in the effective aperture function, and the other form is for an odd number of steps. We consider here the first form for which

$$P_1(z) = \tfrac{1}{2\ell+1}[1 + z^{-k_1}(1 + z^{-k_2}(1 + \ldots + z^{-k_\ell}(1 + \ldots + z^{-k_2}(1 + z^{-k_1})\ldots)))]. \tag{149}$$

The number R of elements (including zero-valued ones) in $P_1(z)$ is given by $R = 2\sum_{i=1}^{\ell} k_i + 1$. Moreover, for a staircase effective aperture function, the number S of elements in $P_2(z)$ of Eq. (147) must satisfy the condition $S > 2\sum_{i=1}^{\ell} k_i$. The number of apodized elements in the beginning and at the end of the effective aperture function is $2\sum_{i=1}^{\ell} k_i$ each. The values of the apodized elements are $\frac{1}{2\ell+1}, \frac{2}{2\ell+1}, \ldots, \frac{2\ell}{2\ell+1}$. For a sparse antenna pair design, the value of S must be a power-of-2.

For example, consider the design of an array with $k_1 = 1, k_2 = 2$, and $S = 8$. Here

$$P_1(z) = \tfrac{1}{5}[1 + z^{-1}(1 + z^{-2}(1 + z^{-2}(1 + z^{-1})))] = \tfrac{1}{5}[1 + z^{-1} + z^{-3} + z^{-5} + z^{-6}],$$
$$P_2(z) = 1 + z^{-1} + z^{-2} + z^{-3} + z^{-4} + z^{-5} + z^{-6} + z^{-7}.$$

The effective aperture function is then of the form

$$w_{eff}[n] = \{0.2 \quad 0.4 \quad 0.4 \quad 0.6 \quad 0.6 \quad 0.8 \quad 1 \quad 1 \quad 0.8 \quad 0.6 \quad 0.6 \quad 0.4 \quad 0.4 \quad 0.2\}.$$

One possible choice for the transmit and the receive aperture functions is given by

$$w_T[n] = \{\tfrac{1}{5} \quad \tfrac{1}{5} \quad \tfrac{1}{5} \quad \tfrac{2}{5} \quad 0 \quad \tfrac{2}{5} \quad \tfrac{1}{5} \quad \tfrac{1}{5} \quad \tfrac{1}{5}\},$$
$$w_R[n] = \{1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1\}.$$

The corresponding scaled radiation patterns are shown in Figure 70(b).

14 Programs

Program 1—Dual-Tone Multifrequency Tone Detection Using the DFT

```
clf;
d = input('Type in the telephone digit = ', 's');
symbol = abs(d);
tm = [49 50 51 65;52 53 54 66;55 56 57 67;42 48 35 68];
for p = 1:4;
for q = 1:4;
    if tm(p,q) == abs(d);break,end
end
    if tm(p,q) == abs(d);break,end
end
f1 = [697 770 852 941];
f2 = [1209 1336 1477 1633];
n = 0:204;
x = sin(2*pi*n*f1(p)/8000) + sin(2*pi*n*f2(q)/8000);
k = [18 20 22 24 31 34 38 42];
val = zeros(1,8);
for m = 1:8;
    Fx(m) = goertzel(x,k(m)+1);
end
val = abs(Fx);
stem(k,val);grid; xlabel('k');ylabel('|X[k]|');
limit = 80;
for s = 5:8;
    if val(s) > limit,break,end
end
for r = 1:4;
    if val(r) > limit,break,end
end
disp(['Touch-Tone Symbol = ',setstr(tm(r,s-4))])
```

Program 2—Spectral Analysis of a Sum of Two Sinusoids Using the DFT

```
clf;
N = input('Signal length = ');
R = input('DFT length = ');
fr = input('Type in the sinusoid frequencies = ');
n = 0:N-1;
x = 0.5*sin(2*pi*n*fr(1)) + sin(2*pi*n*fr(2));
Fx = fft(x,R);
k = 0:R-1;
stem(k,abs(Fx));grid
xlabel('k'); ylabel('Magnitude');
title(['N = ',num2str(N),', R = ',num2str(R)]);
```

Program 3—Spectrogram of a Speech Signal

```
load mtlb
n = 1:4001;
plot(n-1,mtlb);
xlabel('Time index n');ylabel('Amplitude');
pause
nfft = input('Type in the window length = ');
ovlap = input('Type in the desired overlap = ');
specgram(mtlb,nfft,7418,hamming(nfft),ovlap)
```

Program 4—Power Spectrum Estimation Using Welch's Method

```
n = 0:1000;
g = 2*sin(0.12*pi*n) + sin(0.28*pi*n) + randn(size(n));
nfft = input('Type in the fft size = ');
window = hamming(256);
noverlap =input('Type in the amount of overlap = ');
[Pxx, f] = psd(g,nfft,2,window,noverlap);
plot(f/2,10*log10(Pxx));grid
xlabel('\omega/\pi');ylabel('Power Spectrum, dB');
title(['Overlap = ',num2str(noverlap),' samples']);
```

Program 5—Development of an AR Model of an FIR Filter

```
b = remez(13, [0 0.5 0.6 1], [1 1 0 0]);
[h,w] = freqz(b,1,512);
[d,E] = lpc(b,7);
[h1,w] = freqz(sqrt(E*length(b)),d,512);
plot(w/pi,abs(h),'-',w/pi,abs(h1),'--');
xlabel('\omega/\pi');ylabel('Magnitude');
```

Program 6—Single Echo

```
%Delay Function
% y =  singleecho(x, R, a);
%
% Parameters:
%  x is the input audio signal
%  R is the delay in number of samples
%  a specifies the attenuation in the echo
%
% Return value:
%  y is the output signal
%
% Copyright 2004 Vincent Wan
% Credits: Vikas Sahdev, Rajesh Samudrala, Rajani Sadasivam
%
% Example:
%  [x,fs,nbits] = wavread('dsp01.wav');
%  y = singleecho(x,8000,0.5);
%  wavplay(y,fs);

function y =  singleecho(x, R, a);
xlen=length(x); %Calc. the number of samples in the file

y=zeros(size(x));

% filter the signal

for i=1:1:R+1
   y(i) = x(i);
end

for i=R+1:1:xlen
  y(i)= x(i)+ a*x(i-R);
end;
```

Program 7—Multiple Echo

```
% y = multiecho(x,R,a,N);
%
% Generates multiple echos R samples apart with exponentially decaying amplitude
% Parameters:
%  x is the input audio signal
%  R is the delay in number of samples
%  a specifies the attenuation in the echos
%  N-1 is the total number of echos (If N = 0, an infinite number of echos is produced)
%
```

```
% Return value:
%  y is the output signal
%
% Copyright 2004 Vincent Wan
% Credits: Vikas Sahdev, Rajesh Samudrala, Rajani Sadasivam
%
% Example:
%  [x,fs,nbits] = wavread('dsp01.wav');
%  y = multiecho(x,8000,0.5,3);
%  wavplay(y,fs);

function y = multiecho(x,R,a,N);

if (N == 0)
   num=[zeros(1,R),1];
   den=[1,zeros(1,R-1),-a];
else
   num=[1,zeros(1,N*R-1),-a^N];
   den=[1,zeros(1,R-1),-a];
end
y=filter(num,den,x);
```

Program 8—Allpass Reverberator

```
%Allpass reverberator
% y = alpas(x,R,a)
%
% Parameters:
%  x is the input audio signal
%  R is the delay in allpass structure
%  a specifies the allpass filter coefficient
%
% Return value:
%  y is the output signal
%
% Copyright 2004 Vincent Wan
% Credits: Vikas Sahdev, Rajesh Samudrala, Rajani Sadasivam
%
% Example:
%  [x,fs,nbits] = wavread('dsp01.wav');
%  y = alpas(x,8000,0.5);
%  wavplay(y,fs);

function y = alpas(x,R,a)

num=[a,zeros(1,R-1),1];
```

```
den=fliplr(num);

y=filter(num,den,x);
```

Program 9—Natural Sounding Reverberator

```
%A proposed natural sounding reverberator (The SchroederÕs Reverberator)
% y = reverb(x,R,a)
%
% Parameters:
%  x is the input audio signal
%  R is a 6-element vector describing the delays in allpass structure
%  a is a 7-element vector describing multiplier values in the reverberator
%
% Return value:
%  y is the output signal
%
% Copyright 2004 Vincent Wan
% Credits: Vikas Sahdev, Rajesh Samudrala, Rajani Sadasivam
%
% Example:
%  a = [0.6 0.4 0.2 0.1 0.7 0.6 0.8];
%  R = [700 900 600 400 450 390];
%  [x,fs,nbits] = wavread('dsp01.wav');
%  y = reverb(x,R,a);
%  wavplay(y,fs);

function y = reverb(x,R,a)

d1 = multiecho(x, R(1), a(1), 0);
d2 = multiecho(x, R(2), a(2), 0);
d3 = multiecho(x, R(3), a(3), 0);
d4 = multiecho(x, R(4), a(4), 0);
d_IIR = d1 + d2 + d3 + d4;  %output of IIR echo generators

d_ALL1 = alpas(d_IIR, R(5), a(5));
d_ALL2 = alpas(d_ALL1, R(6), a(6));

y = x + a(7)*d_ALL2;
```

14.1 Program 10—Flanger

```
% flang(x,R,a,omega,fs)
%
% Parameters:
%  x is the input audio signal; R is the maximum delay value
%  a specifies the attenuation in the echo, and can be set between [-1,1]
```

```
%  omega is a low angular frequency over which the delay varies sinusoidally
%  fs is the sampling frequency
%
% Return value:  y is the output signal
%
% Copyright 2004 Vincent Wan
% Credits: Vikas Sahdev, Rajesh Samudrala, Rajani Sadasivam
%
% Example:
%  [x,fs,nbits] = wavread('dsp01.wav');
%  y = flang(x,1000,0.5,2*pi*6,fs);
%  wavplay(y,fs);

function y = flang(x,R,a,omega,fs)
y=zeros(size(x));

% filter the signal
max_length = length(x);
for i=1:max_length
   delay = R/2*(1-cos(omega*i/fs));
   delay_ceiling = ceil(delay);
   y(i) = x(i);
   if (delay <= (i - 1))
      %Use linear interpolation
      y(i) = y(i)+a*( x(i-delay_ceiling) + (x(i-delay_ceiling+1) - x(i-delay_ceiling
   end
end
```

Program 11—Sigma–Delta Quantizer Operation

```
   N = input('Type in the length of input sequence = ');
   n = 1:1:N;
   m = n-1;
   A = input('Type in the input amplitude = ');
   x = A*ones(1,N);
   plot(m,x);
   axis([0 N-1 -1.2 1.2]);
   xlabel('Time'); ylabel('Amplitude');
   title('Input analog signal');
   pause
   y = zeros(1,N+1);
   v0 = 0;
      for k = 2:1:N+1;
           v1 = x(k-1) - y(k-1) + v0;
           y(k) = sign(v1);
           v0 = v1;
      end
```

```
yn = y(2:N+1);
axis([0 N-1 -1.2 1.2]);
stem(m, yn);
xlabel('Time'); ylabel('Amplitude');
title('Output of sigma-delta modulator');
ave = sum(yn)/N;
disp('Average value of output is = ');disp(ave);
```

Program 12—Sigma–Delta A/D Converter Operation

```
wo = 2*pi*0.01;
N = input('Type in the length of input sequence = ');
n = 1:1:N;
m = n-1;
A = input('Type in the amplitude of the input = ');
x = A*cos(wo*m);
axis([0 N-1 -1.2 1.2]);
plot(m,x);
xlabel('Time'); ylabel('Amplitude');
title('Input analog signal');
pause
y = zeros(1,N+1);
v0 = 0;
for k = 2:1:N+1;
v1 = x(k-1) - y(k-1) + v0;
    if v1 >= 0;
      y(k) = 1;
    else
      y(k) = -1;
    end
v0 = v1;
end
yn = y(2:N+1);
axis([0 N-1 -1.2 1.2]);
stairs(m, yn);
xlabel('Time'); ylabel('Amplitude');
title('Output of sigma-delta quantizer');
Y = fft(yn);
pause
H = [1 1 0.5 zeros(1,N-5) 0.5 1];
YF = Y.*H;
out = ifft(YF);
axis([0 N-1 -1.2 1.2]);
plot(m,out);
xlabel('Time'); ylabel('Amplitude');
title('Lowpass filtered output');
```

Bibliography

[Kar83] K. Karplus and A. Strong. Digital synthesis of plucked-string and drum timbres. *Computer Music Journal*, 7:43–55, Summer 1983.

[Mit2004b] S.K. Mitra, G. Jovanovic-Dolecek, and M.K. Tchobanou. On the design of one-dimensional sparse arrays with apodized end elements. In *Proc. 12th European Signal Processing Conference*, pages 2239-2242, Vienna, Austria, September 2004.