6 SAMPLING AND ANALOG-TO-DIGITAL CONVERSION

As briefly discussed in Chapter 1, analog signals can be digitized through sampling and quantization. This analog-to-digital (A/D) conversion sets the foundation of modern digital communication systems. In the A/D converter, the sampling rate must be large enough to permit the analog signal to be reconstructed from the samples with sufficient accuracy. The **sampling theorem**, which is the basis for determining the proper (lossless) sampling rate for a given signal, has played a huge role in signal processing, communication theory, and A/D circuit design.

6.1 SAMPLING THEOREM

We first show that a signal $g(t)$ whose spectrum is band-limited to B Hz, that is,

$$G(f) = 0 \quad \text{for } |f| > B$$

can be reconstructed exactly (without any error) from its discrete time samples taken uniformly at a rate of R samples per second. The condition is that $R > 2B$. In other words, the minimum sampling frequency for perfect signal recovery is $f_s = 2B$ Hz.

To prove the sampling theorem, consider a signal $g(t)$ (Fig. 6.1a) whose spectrum is band-limited to B Hz (Fig. 6.1b).* For convenience, spectra are shown as functions of f as well as of ω. Sampling $g(t)$ at a rate of f_s Hz means that we take f_s uniform samples per second. This uniform sampling can be accomplished by multiplying $g(t)$ by an impulse train $\delta_{T_s}(t)$ of Fig. 6.1c, consisting of unit impulses repeating periodically every T_s seconds, where $T_s = 1/f_s$. This results in the sampled signal $\overline{g}(t)$ shown in Fig. 6.1d. The sampled signal consists of impulses spaced every T_s seconds (the sampling interval). The nth impulse, located at $t = nT_s$, has a strength $g(nT_s)$ which is the value of $g(t)$ at $t = nT_s$. Thus, the relationship between the

* The spectrum $G(f)$ in Fig. 6.1b is shown as real, for convenience. Our arguments are valid for complex $G(f)$.

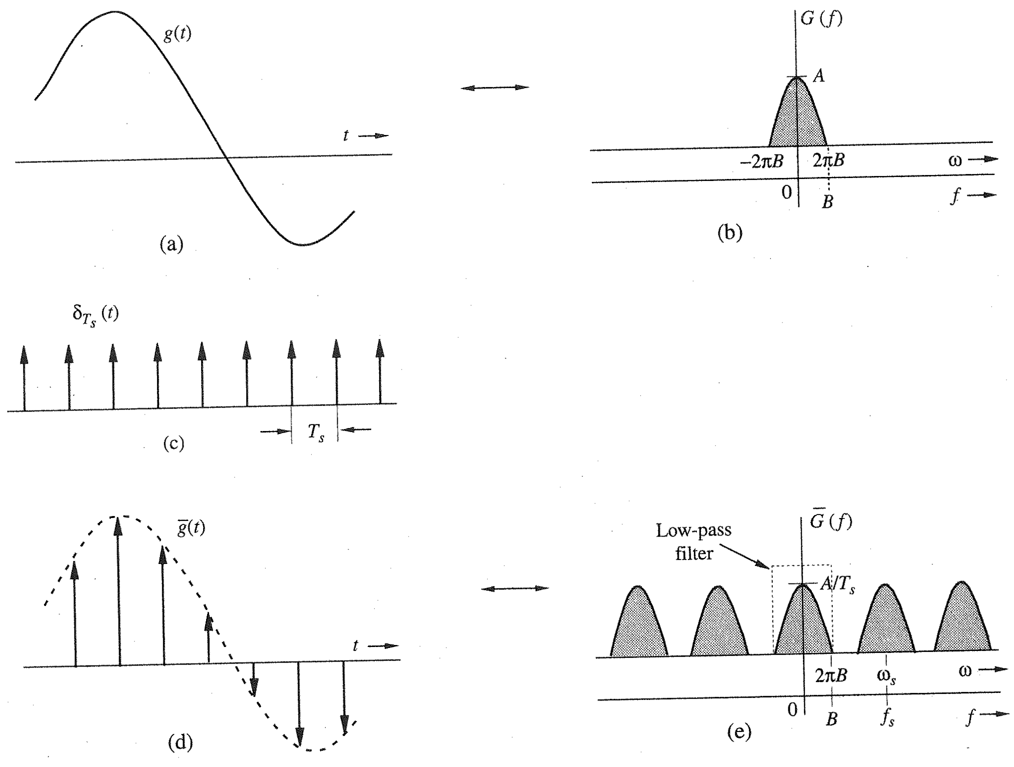


Figure 6.1 Sampled signal and its Fourier spectra.

sampled signal $\overline{g}(t)$ and the original analog signal $g(t)$ is

$$\overline{g}(t) = g(t)\delta_{T_s}(t) = \sum_n g(nT_s)\delta(t - nT_s) \tag{6.1}$$

Because the impulse train $\delta_{T_s}(t)$ is a periodic signal of period T_s, it can be expressed as an exponential Fourier series, already found in Example 3.11 as

$$\delta_{T_s}(t) = \frac{1}{T_s}\sum_{n=-\infty}^{\infty} e^{jn\omega_s t} \qquad \omega_s = \frac{2\pi}{T_s} = 2\pi f_s \tag{6.2}$$

Therefore,

$$\begin{aligned}\overline{g}(t) &= g(t)\delta_{T_s}(t) \\ &= \frac{1}{T_s}\sum_{n=-\infty}^{\infty} g(t)e^{jn2\pi f_s t}\end{aligned} \tag{6.3}$$

To find $\overline{G}(f)$, the Fourier transform of $\overline{g}(t)$, we take the Fourier transform of the summation in Eq. (6.3). Based on the frequency-shifting property, the transform of the *nth* term is shifted by nf_s. Therefore,

$$\overline{G}(f) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} G(f - nf_s) \tag{6.4}$$

This means that the spectrum $\overline{G}(f)$ consists of $G(f)$, scaled by a constant $1/T_s$, repeating periodically with period $f_s = 1/T_s$ Hz, as shown in Fig. 6.1e.

After uniform sampling that generates a set of signal samples $\{g(kT_s)\}$, the vital question becomes: **Can $g(t)$ be reconstructed from $\overline{g}(t)$ without any loss or distortion?** If we are to reconstruct $g(t)$ from $\overline{g}(t)$, equivalently in the frequency domain we should be able to recover $G(f)$ from $\overline{G}(f)$. Graphically from Fig. 6.1, perfect recovery is possible if there is no overlap among the replicas in $\overline{G}(f)$. Figure 6.1e clearly shows that this requires

$$f_s > 2B \tag{6.5}$$

Also, the sampling interval $T_s = 1/f_s$. Therefore,

$$T_s < \frac{1}{2B} \tag{6.6}$$

Thus, as long as the sampling frequency f_s is greater than twice the signal bandwidth B (in hertz), $\overline{G}(f)$ will consist of nonoverlapping repetitions of $G(f)$. When this is true, Fig. 6.1e shows that $g(t)$ can be recovered from its samples $\overline{g}(t)$ by passing the sampled signal $\overline{g}(t)$ through an ideal low-pass filter of bandwidth B Hz. The minimum sampling rate $f_s = 2B$ required to recover $g(t)$ from its samples $\overline{g}(t)$ is called the **Nyquist rate** for $g(t)$, and the corresponding sampling interval $T_s = 1/2B$ is called the **Nyquist interval** for the low-pass signal $g(t)$.*

We need to stress one important point regarding the possibility of $f_s = 2B$ and a particular class of low-pass signals. For a general signal spectrum, we have proved that the sampling rate $f_s > 2B$. However, if the spectrum $G(f)$ has no impulse (or its derivatives) at the highest frequency B, then the overlap is still zero as long as the sampling rate is greater than or equal to the Nyquist rate, that is,

$$f_s \geq 2B$$

If, on the other hand, $G(f)$ contains an impulse at the highest frequency $\pm B$, then the equality must be removed or else overlap will occur. In such case, the sampling rate f_s must be greater than $2B$ Hz. A well-known example is a sinusoid $g(t) = \sin 2\pi B(t - t_0)$. This signal is band-limited to B Hz, but all its samples are zero when uniformly taken at a rate $f_s = 2B$ (starting at $t = t_0$), and $g(t)$ cannot be recovered from its Nyquist samples. Thus, for sinusoids, the condition of $f_s > 2B$ must be satisfied.

6.1.1 Signal Reconstruction from Uniform Samples

The process of reconstructing a continuous time signal $g(t)$ from its samples is also known as **interpolation**. In Fig. 6.1, we used a constructive proof to show that a signal $g(t)$ band-limited

* The theorem stated here (and proved subsequently) applies to low-pass signals. A bandpass signal whose spectrum exists over a frequency band $f_c - B/2 < |f| < f_c + B/2$ has a bandwidth B Hz. Such a signal is also uniquely determined by samples taken at above the Nyquist frequency $2B$. The sampling theorem is generally more complex in such case. It uses two interlaced uniform sampling trains, each at half the overall sampling rate $R_s > B$. See, for example, the Refs. 1 and 2.

to B Hz can be reconstructed (interpolated) exactly from its samples. This means not only that uniform sampling at above the Nyquist rate preserves all the signal information, but also that simply passing the sampled signal through an ideal low-pass filter of bandwidth B Hz will reconstruct the original message. As seen from Eq. (6.3), the sampled signal contains a component $(1/T_s)g(t)$, and to recover $g(t)$ [or $G(f)$], the sampled signal

$$\overline{g}(t) = \sum g(nT_s)\delta(t - nT_s)$$

must be sent through an ideal low-pass filter of bandwidth B Hz and gain T_s. Such an ideal filter response has the transfer function

$$H(f) = T_s \, \Pi\left(\frac{\omega}{4\pi B}\right) = T_s \, \Pi\left(\frac{f}{2B}\right) \tag{6.7}$$

Ideal Reconstruction

To recover the analog signal from its uniform samples, the ideal interpolation filter transfer function found in Eq. (6.7) is shown in Fig. 6.2a. The impulse response of this filter, the inverse Fourier transform of $H(f)$, is

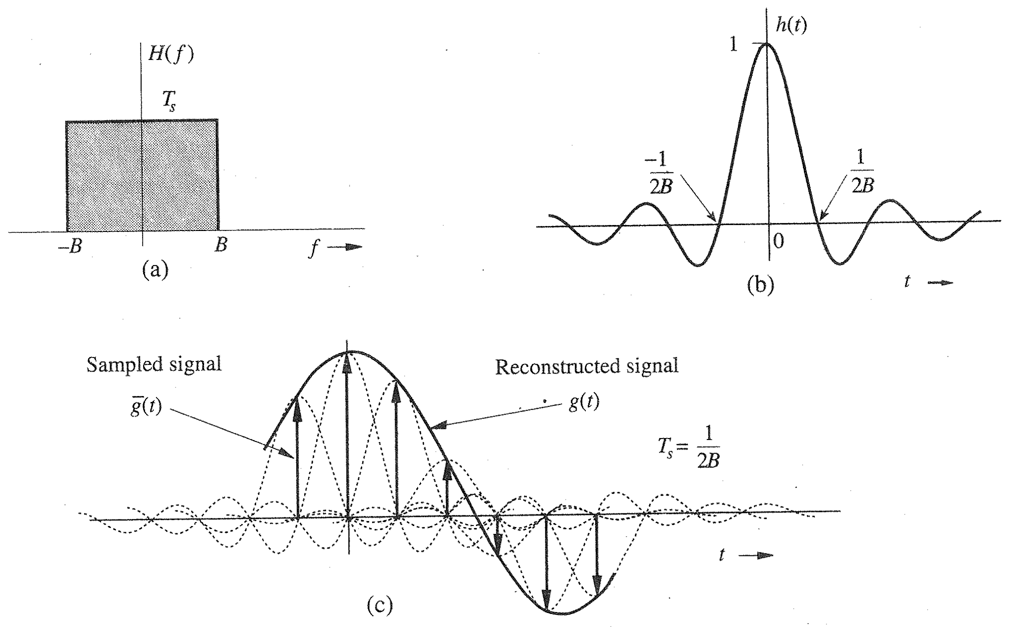
$$h(t) = 2BT_s \operatorname{sinc}(2\pi Bt) \tag{6.8}$$

Assuming the use of Nyquist sampling rate, that is, $2BT_s = 1$, then

$$h(t) = \operatorname{sinc}(2\pi Bt) \tag{6.9}$$

This $h(t)$ is shown in Fig. 6.2b. Observe the very interesting fact that $h(t) = 0$ at all Nyquist sampling instants ($t = \pm n/2B$) except $t = 0$. When the sampled signal $\overline{g}(t)$ is applied at the input of this filter, the output is $g(t)$. Each sample in $\overline{g}(t)$, being an impulse, generates a sinc pulse of height equal to the strength of the sample, as shown in Fig. 6.2c. The process is

Figure 6.2 Ideal interpolation.



identical to that shown in Fig. 6.6, except that $h(t)$ is a sinc pulse instead of a rectangular pulse. Addition of the sinc pulses generated by all the samples results in $g(t)$. The kth sample of the input $\overline{g}(t)$ is the impulse $g(kT_s)\delta(t-kT_s)$; the filter output of this impulse is $g(kT_s)h(t-kT_s)$. Hence, the filter output to $\overline{g}(t)$, which is $g(t)$, can now be expressed as a sum,

$$g(t) = \sum_k g(kT_s)h(t-kT_s)$$

$$= \sum_k g(kT_s)\,\text{sinc}\,[2\pi B(t-kT_s)] \tag{6.10a}$$

$$= \sum_k g(kT_s)\,\text{sinc}\,(2\pi Bt - k\pi) \tag{6.10b}$$

Equation (6.10) is the **interpolation formula**, which yields values of $g(t)$ between samples as a weighted sum of all the sample values.

Example 6.1 Find a signal $g(t)$ that is band-limited to B Hz and whose samples are

$$g(0) = 1 \quad \text{and} \quad g(\pm T_s) = g(\pm 2T_s) = g(\pm 3T_s) = \cdots = 0$$

where the sampling interval T_s is the Nyquist interval for $g(t)$, that is, $T_s = 1/2B$.

We use the interpolation formula (6.10b) to construct $g(t)$ from its samples. Since all but one of the Nyquist samples are zero, only one term (corresponding to $k = 0$) in the summation on the right-hand side of Eq. (6.10b) survives. Thus,

$$g(t) = \text{sinc}\,(2\pi Bt) \tag{6.11}$$

This signal is shown in Fig. 6.3. Observe that this is the only signal that has a bandwidth B Hz and sample values $g(0) = 1$ and $g(nT_s) = 0$ ($n \neq 0$). No other signal satisfies these conditions.

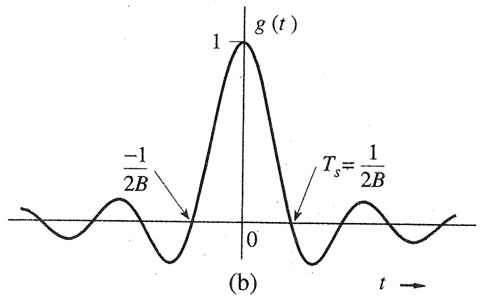


Figure 6.3 Signal reconstructed from the Nyquist samples in Example 6.1.

Practical Signal Reconstruction (Interpolation)

We established in Sec. 3.5 that the ideal low-pass filter is noncausal and unrealizable. This can be equivalently seen from the infinitely long nature of the sinc reconstruction pulse used in the ideal reconstruction of Eq. (6.10). For practical application of signal reconstruction (e.g., a

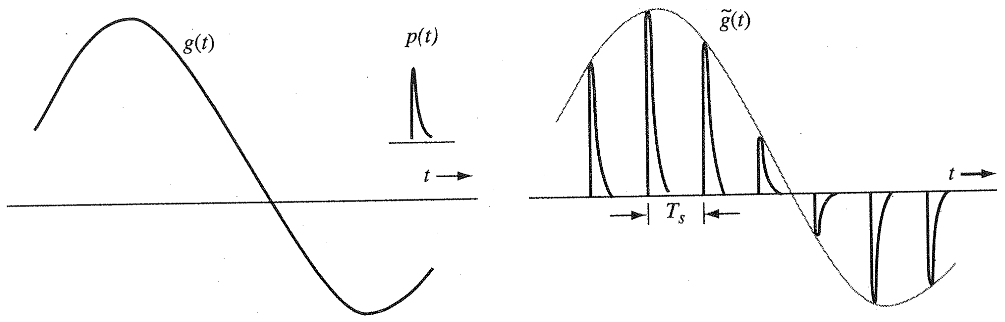


Figure 6.4 Practical reconstruction (interpolation) pulse.

CD player), we need to implement realizable signal reconstruction systems from the uniform signal samples.

For practical implementation, this reconstruction pulse $p(t)$ must be easy to generate. For example, we may apply the reconstruction pulse $p(t)$ as shown in Fig. 6.4. However, we must first use the nonideal interpolation pulse $p(t)$ to analyze the accuracy of the reconstructed signal. Let us denote the new signal from reconstruction as

$$\tilde{g}(t) \triangleq \sum_n g(nT_s)p(t - nT_s) \tag{6.12}$$

To determine its relation to the original analog signal $g(t)$, we can see from the properties of convolution and Eq.(6.1) that

$$\begin{aligned} \tilde{g}(t) &= \sum_n g(nT_s)p(t - nT_s) = p(t) * \left[\sum_n g(nT_s)\delta(t - nT_s)\right] \\ &= p(t) * \overline{g}(t) \end{aligned} \tag{6.13a}$$

In the frequency domain, the relationship between the reconstruction and the original analog signal can rely on Eq. (6.4)

$$\tilde{G}(f) = P(f)\frac{1}{T_s}\sum_n G(f - nf_s) \tag{6.13b}$$

This means that the reconstructed signal $\tilde{g}(t)$ using pulse $p(t)$ consists of multiple replicas of $G(f)$ shifted to the frequency center nf_s and filtered by $P(f)$. To fully recover $g(t)$, further filtering of $\tilde{g}(t)$ becomes necessary. Such filters are often referred to as equalizers.

Denote the equalizer transfer function as $E(f)$. Distortionless reconstruction requires that

$$\begin{aligned} G(f) &= E(f)\tilde{G}(f) \\ &= E(f)P(f)\frac{1}{T_s}\sum_n G(f - nf_s) \end{aligned}$$

This relationship clearly illustrates that the equalizer must remove all the shifted replicas $G(f - nf_s)$ in the summation except for the low-pass term with $n = 0$, that is,

$$E(f)P(f) = 0 \quad |f| > f_s - B \tag{6.14a}$$

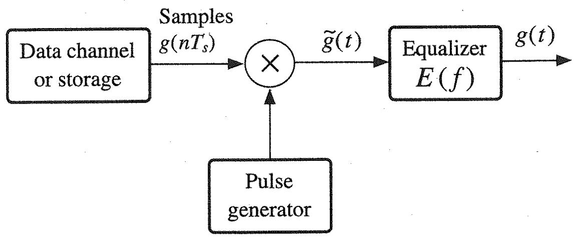


Figure 6.5 Practical signal reconstruction.

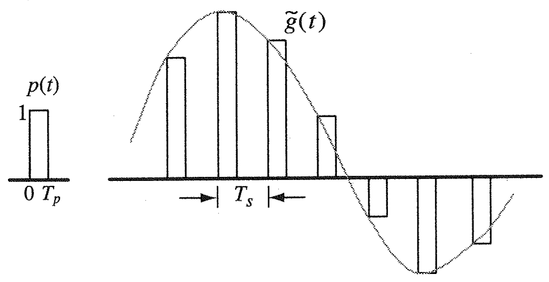


Figure 6.6 Simple interpolation by means of simple rectangular pulses.

Additionally, distortionless reconstruction requires that

$$E(f)P(f) = T_s \quad |f| < B \tag{6.14b}$$

The equalizer filter $E(f)$ must be low-pass in nature to stop all frequency content above $f_s - B$ Hz, and it should be the inverse of $P(f)$ within the signal bandwidth of B Hz. Figure 6.5 demonstrates the diagram of a practical signal reconstruction system utilizing such an equalizer.

Let us now consider a very simple interpolating pulse generator that generates short (zero-order hold) pulses. As shown in Fig. 6.6,

$$p(t) = \Pi\left(\frac{t - 0.5T_p}{T_p}\right)$$

This is a gate pulse of unit height with pulse duration T_p. The reconstruction will first generate

$$\tilde{g}(t) = \sum_n g(nT_s)\,\Pi\left(\frac{t - nT_s - 0.5T_p}{T_p}\right)$$

The transfer function of filter $P(f)$ is the Fourier transform of $\Pi(t/T_p)$ shifted by $0.5T_p$:

$$P(f) = T_p \operatorname{sinc}\left(\pi f T_p\right) e^{-j\pi f T_p} \tag{6.15}$$

As a result, the equalizer frequency response should satisfy

$$E(f) = \begin{cases} T_s/P(f) & |f| \le B \\ \text{Flexible} & B < |f| < (1/T_s - B) \\ 0 & |f| \ge (1/T_s - B) \end{cases}$$

It is important for us to ascertain that the equalizer passband response is realizable. First of all, we can add another time delay to the reconstruction such that

$$E(f) = T_s \cdot \frac{\pi f}{\sin(\pi f T_p)} e^{-j2\pi f t_0} \qquad |f| \le B \tag{6.16}$$

For the passband gain of $E(f)$ to be well defined, it is imperative for us to choose a short pulse width T_p such that

$$\frac{\sin(\pi f T_p)}{\pi f} \neq 0 \quad |f| \le B$$

This means that the equalizer $E(f)$ does not need to achieve infinite gain. Otherwise the equalizer would become unrealizable. Equivalently, this requires that

$$T_p < 1/B$$

Hence, as long as the rectangular reconstruction pulse width is shorter than $1/B$, it may be possible to design an analog equalizer filter to recover the original analog signal $g(t)$ from the nonideal reconstruction pulse train. Of course, this is a requirement for a rectangular reconstruction pulse generator. In practice, T_p can be chosen very small, to yield the following equalizer passband response:

$$E(f) = T_s \cdot \frac{\pi f}{\sin(\pi f T_p)} \approx \frac{T_s}{T_p} \qquad |f| \le B \tag{6.17}$$

This means that very little distortion remains when very short rectangular pulses are used in signal reconstruction. Such cases make the design of the equalizer either unnecessary or very simple. An illustrative example is given as a MATLAB exercise in Sec. 6.9.

We can improve on the zero-order-hold filter by using the **first-order-hold** filter, which results in a linear interpolation instead of the staircase interpolation. The linear interpolator, whose impulse response is a triangle pulse $\Delta(t/2T_s)$, results in an interpolation in which successive sample tops are connected by straight-line segments (Prob. 6.1-7).

6.1.2 Practical Issues in Signal Sampling and Reconstruction

Realizability of Reconstruction Filters

If a signal is sampled at the Nyquist rate $f_s = 2B$ Hz, the spectrum $\overline{G}(f)$ consists of repetitions of $G(f)$ without any gap between successive cycles, as shown in Fig. 6.7a. To recover $g(t)$ from $\overline{g}(t)$, we need to pass the sampled signal $\overline{g}(t)$ through an ideal low-pass filter (dotted area in Fig. 6.7a). As seen in Sec. 3.5, such a filter is unrealizable in practice; it can be closely approximated only with infinite time delay in the response. This means that we can recover the signal $g(t)$ from its samples with infinite time delay.

A practical solution to this problem is to sample the signal at a rate higher than the Nyquist rate ($f_s > 2B$ or $\omega_s > 4\pi B$). This yields $\overline{G}(f)$, consisting of repetitions of $G(f)$ with a finite band gap between successive cycles, as shown in Fig. 6.7b. We can now recover $G(f)$ from $\overline{G}(f)$ [or from $\tilde{G}(f)$] by using a low-pass filter with a gradual cutoff characteristic (dotted area in Fig. 6.7b). But even in this case, the filter gain is required to be zero beyond the first cycle

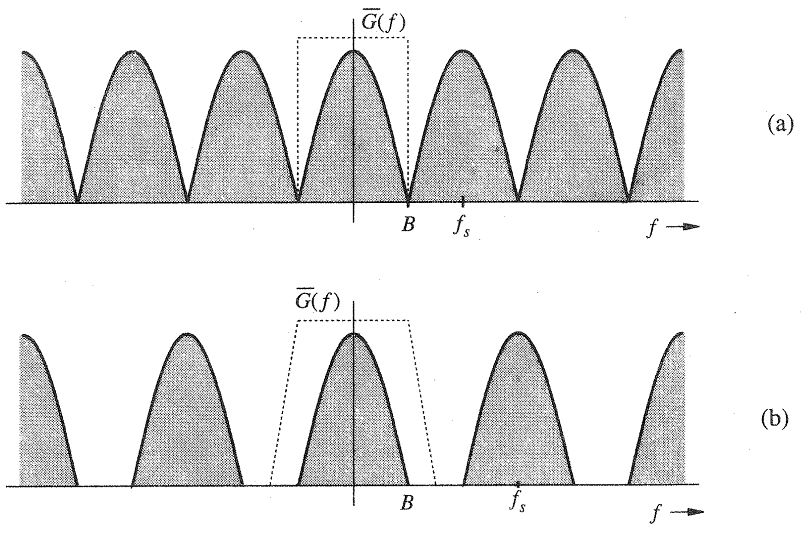


Figure 6.7 Spectra of a sampled signal: (a) at the Nyquist rate; (b) above the Nyquist rate.

of $G(f)$ (Fig. 6.7b). According to the Paley-Wiener criterion, it is impossible to realize even this filter. The only advantage in this case is that the required filter can be better approximated with a smaller time delay. This shows that it is impossible in practice to recover a band-limited signal $g(t)$ exactly from its samples, even if the sampling rate is higher than the Nyquist rate. However, as the sampling rate increases, the recovered signal approaches the desired signal more closely.

The Treachery of Aliasing

There is another fundamental practical difficulty in reconstructing a signal from its samples. The sampling theorem was proved on the assumption that the signal $g(t)$ is band-limited. *All practical signals are time-limited;* that is, they are of finite duration or width. We can demonstrate (Prob. 6.1-8) that a signal cannot be time-limited and band-limited simultaneously. A time-limited signal cannot be band-limited, and vice versa (but a signal can be simultaneously non-time-limited and non-band-limited). Clearly, all practical signals, which are necessarily time-limited, are non-band-limited, as shown in Fig. 6.8a; they have infinite bandwidth, and the spectrum $\overline{G}(f)$ consists of overlapping cycles of $G(f)$ repeating every f_s Hz (the sampling frequency), as illustrated in Fig. 6.8b. Because of the infinite bandwidth in this case, the spectral overlap is unavoidable, regardless of the sampling rate. Sampling at a higher rate reduces but does not eliminate overlapping between repeating spectral cycles. Because of the overlapping tails, $\overline{G}(f)$ no longer has complete information about $G(f)$, and it is no longer possible, even theoretically, to recover $g(t)$ exactly from the sampled signal $\overline{g}(t)$. If the sampled signal is passed through an ideal low-pass filter of cutoff frequency $f_s/2$ Hz, the output is not $G(f)$ but $G_a(f)$ (Fig. 6.8c), which is a version of $G(f)$ distorted as a result of two separate causes:

1. The loss of the tail of $G(f)$ beyond $|f| > f_s/2$ Hz.
2. The reappearance of this tail inverted or folded back onto the spectrum.

Note that the spectra cross at frequency $f_s/2 = 1/2T$ Hz, which is called the *folding* frequency. The spectrum may be viewed as if the lost tail is folding back onto itself at the folding frequency. For instance, a component of frequency $(f_s/2) + f_z$ shows up as, or "impersonates," a component of lower frequency $(f_s/2) - f_z$ in the reconstructed signal. Thus, the components of frequencies above $f_s/2$ reappear as components of frequencies below $f_s/2$. This tail inversion,

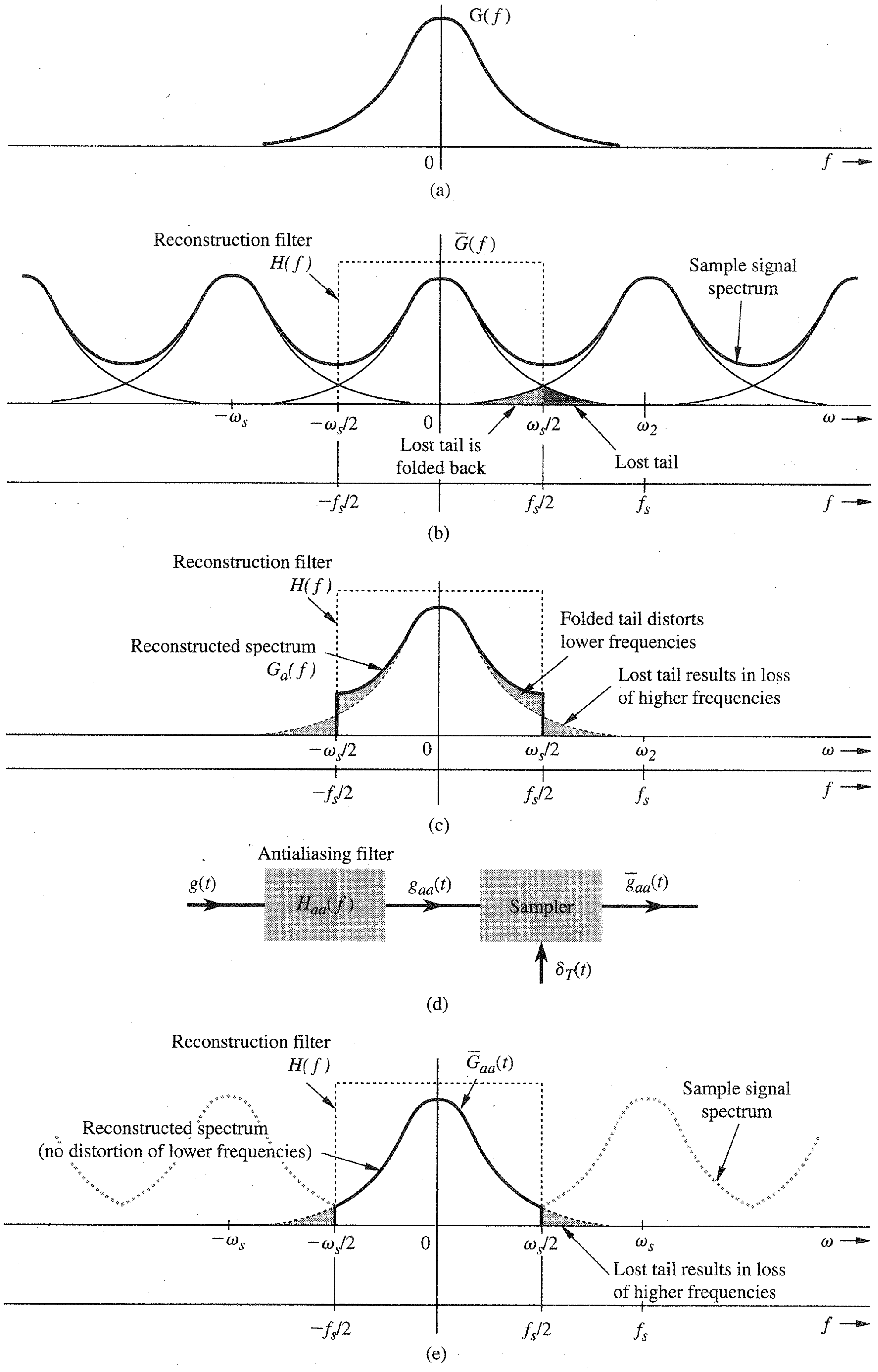


Figure 6.8 Aliasing effect. (a) Spectrum of a practical signal $g(t)$. (b) Spectrum of sampled $g(t)$. (c) Reconstructed signal spectrum. (d) Sampling scheme using antialiasing filter. (e) Sampled signal spectrum (dotted) and the reconstructed signal spectrum (solid) when antialiasing filter is used.

known as *spectral folding* or *aliasing,* is shown shaded in Fig. 6.8b and also in Fig. 6.8c. In the process of aliasing, not only are we losing all the components of frequencies above the folding frequency $f_s/2$ Hz, but these very components reappear (aliased) as lower frequency components in Fig. 6.8b or c. Such aliasing destroys the integrity of the frequency components below the folding frequency $f_s/2$, as depicted in Fig. 6.8c.

The problem of aliasing is analogous to that of an army when a certain platoon has secretly defected to the enemy side but remains nominally loyal to their army. The army is in double jeopardy. First, it has lost the defecting platoon as an effective fighting force. In addition, during actual fighting, the army will have to contend with sabotage caused by the defectors and will have to use loyal platoon to neutralize the defectors. Thus, the army has lost two platoons to nonproductive activity.

Defectors Eliminated: The Antialiasing Filter

If you were the commander of the betrayed army, the solution to the problem would be obvious. As soon as you got wind of the defection, you would incapacitate, by whatever means, the defecting platoon. By taking this action *before the fighting begins*, you lose only one (the defecting)* platoon. This is a partial solution to the double jeopardy of betrayal and sabotage, a solution that partly rectifies the problem and cuts the losses in half.

We follow exactly the same procedure. The potential defectors are all the frequency components beyond the folding frequency $f_s/2 = 1/2T$ Hz. We should eliminate (suppress) these components from $g(t)$ *before sampling* $g(t)$. Such suppression of higher frequencies can be accomplished by an ideal low-pass filter of cutoff $f_s/2$ Hz, as shown in Fig. 6.8d. This is called the *antialiasing filter*. Figure 6.8d also shows that antialiasing filtering is performed before sampling. Figure 6.8e shows the sampled signal spectrum and the reconstructed signal $G_{aa}(f)$ when the antialiasing scheme is used. An antialiasing filter essentially band-limits the signal $g(t)$ to $f_s/2$ Hz. This way, we lose only the components beyond the folding frequency $f_s/2$ Hz. These suppressed components now cannot reappear, corrupting the components of frequencies below the folding frequency. Clearly, use of an antialiasing filter results in the reconstructed signal spectrum $G_{aa}(f) = G(f)$ for $|f| < f_s/2$. Thus, although we lost the spectrum beyond $f_s/2$ Hz, the spectrum for all the frequencies below $f_s/2$ remains intact. The effective aliasing distortion is cut in half owing to elimination of folding. We stress again that the antialiasing operation must be performed *before the signal is sampled.*

An antialiasing filter also helps to reduce noise. Noise, generally, has a wideband spectrum, and without antialiasing, the aliasing phenomenon itself will cause the noise components outside the desired signal band to appear in the signal band. Antialiasing suppresses the entire noise spectrum beyond frequency $f_s/2$.

The antialiasing filter, being an ideal filter, is unrealizable. In practice we use a steep-cutoff filter, which leaves a sharply attenuated residual spectrum beyond the folding frequency $f_s/2$.

Sampling Forces Non-Band-Limited Signals to Appear Band-Limited

Figure 6.8b shows the spectrum of a signal $\overline{g}(t)$ consists of overlapping cycles of $G(f)$. This means that $\overline{g}(t)$ are sub-Nyquist samples of $g(t)$. However, we may also view the spectrum in Fig. 6.8b as the spectrum $G_a(f)$ (Fig. 6.8c), repeating periodically every f_s Hz without overlap. The spectrum $G_a(f)$ is band-limited to $f_s/2$ Hz. Hence, these (sub-Nyquist) samples of $g(t)$

* Figure 6.8b shows that from the infinite number of repeating cycles, only the neighboring spectral cycles overlap. This is a somewhat simplified picture. In reality, all the cycles overlap and interact with every other cycle because of the infinite width of all practical signal spectra. Fortunately, all practical spectra also must decay at higher frequencies. This results in an insignificant amount of interference from cycles other than the immediate neighbors. When such an assumption is not justified, aliasing computations become little more involved.

are actually the Nyquist samples for signal $g_a(t)$. In conclusion, sampling a non-band-limited signal $g(t)$ at a rate f_s Hz makes the samples appear to be the Nyquist samples of some signal $g_a(t)$, band-limited to $f_s/2$ Hz. In other words, sampling makes a non-band-limited signal appear to be a band-limited signal $g_a(t)$ with bandwidth $f_s/2$ Hz. A similar conclusion applies if $g(t)$ is band-limited but sampled at a sub-Nyquist rate.

6.1.3 Maximum Information Rate: Two Pieces of Information per Second per Hertz

A knowledge of the maximum rate at which information can be transmitted over a channel of bandwidth B Hz is of fundamental importance in digital communication. We now derive one of the basic relationships in communication, which states that *a maximum of* 2*B independent pieces of information per second can be transmitted, error free, over a noiseless channel of bandwidth B Hz*. The result follows from the sampling theorem.

First, the sampling theorem shows that a low-pass signal of bandwidth B Hz can be fully recovered from samples uniformly taken at the rate of $2B$ samples per second. Conversely, we need to show that any sequence of independent data at the rate of $2B$ Hz can come from uniform samples of a low-pass signal with bandwidth B. Moreover, we can construct this low-pass signal from the independent data sequence.

Suppose a sequence of independent data samples is denoted as $\{g_n\}$. Its rate is $2B$ samples per second. Then there always exists a (not necessarily band-limited) signal $g(t)$ such that

$$g_n = g(nT_s) \qquad T_s = \frac{1}{2B}$$

In Figure 6.9a we illustrate again the effect of sampling the non-band-limited signal $g(t)$ at sampling rate $f_s = 2B$ Hz. Because of aliasing, the ideal sampled signal

$$\begin{aligned}\overline{g}(t) &= \sum_n g(nT_s)\delta(t - nT_s)\\ &= \sum_n g_a(nT_s)\delta(t - nT_s)\end{aligned}$$

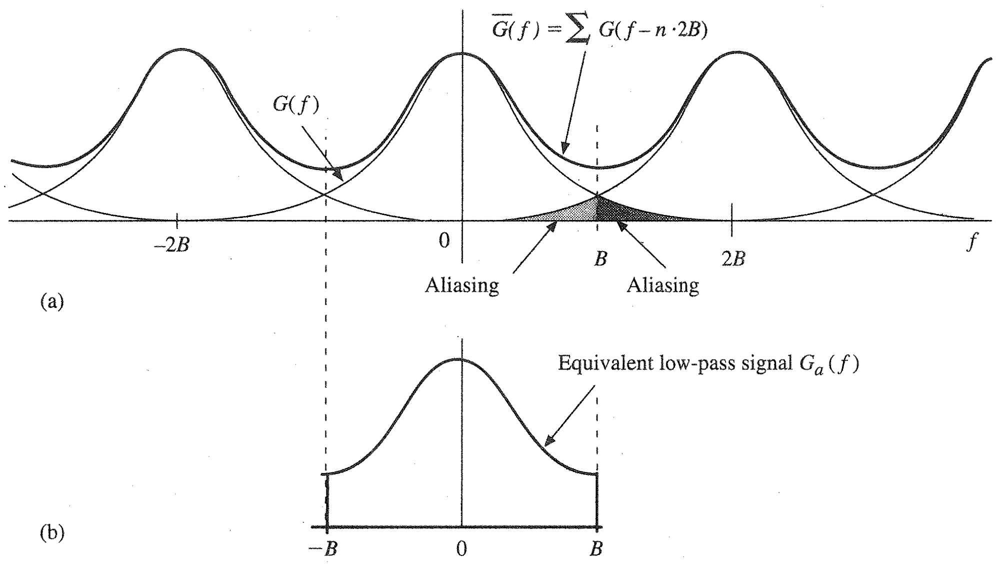
where $g_a(t)$ is the aliased low-pass signal whose samples $g_a(nT_s)$ equal to the samples of $g(nT_s)$. In other words, sub-Nyquist sampling of a signal $g(t)$ generates samples that can be equally well obtained by Nyquist sampling of a band-limited signal $g_a(t)$. Thus, through Figure 6.9, we demonstrate that sampling $g(t)$ and $g_a(t)$ at the rate of $2B$ Hz will generate the same independent information sequence $\{g_n\}$:

$$g_n = g(nT_s) = g_a(nT_s) \qquad T_s = \frac{1}{2B} \tag{6.18}$$

Also from the sampling theorem, a low-pass signal $g_a(t)$ with bandwidth B can be reconstructed from its uniform samples [Eq. (6.10)]

$$g_a(t) = \sum_n g_n \operatorname{sinc}(2\pi Bt - k\pi)$$

Assuming no noise, this signal can be transmitted over a distortionless channel of bandwidth B Hz, error free. At the receiver, the data sequence $\{g_n\}$ can be recovered from the Nyquist samples of the distortionless channel output $g_a(t)$ as the desired information data.



$\bar{G}(f) = \sum G(f - n \cdot 2B)$
$G(f)$
$-2B$
0
B
$2B$
f
Aliasing
Aliasing
(a)
Equivalent low-pass signal $G_a(f)$
(b)
$-B$
0
B

Figure 6.9 (a) Non-band-limited signal spectrum and its sampled spectrum $\bar{G}(f)$. (b) Equivalent low-pass signal spectrum $G_a(f)$ constructed from uniform samples of $g(t)$ at sampling rate $2B$.

This theoretical rate of communication assumes a noise-free channel. In practice, channel noise is unavoidable, and consequently, this rate will cause some detection errors. In Chapter 14, we shall present the Shannon capacity which determines the theoretical error-free communication rate in the presence of noise.

6.1.4 Nonideal Practical Sampling Analysis

Thus far, we have mainly focused on ideal uniform sampling that can use an ideal impulse sampling pulse train to precisely extract the signal value $g(kT_s)$ at the precise instant of $t = kT_s$. In practice, no physical device can carry out such a task. Consequently, we need to consider the more practical implementation of sampling. This analysis is important to the better understanding of errors that typically occur during practical A/D conversion and their effects on signal reconstruction.

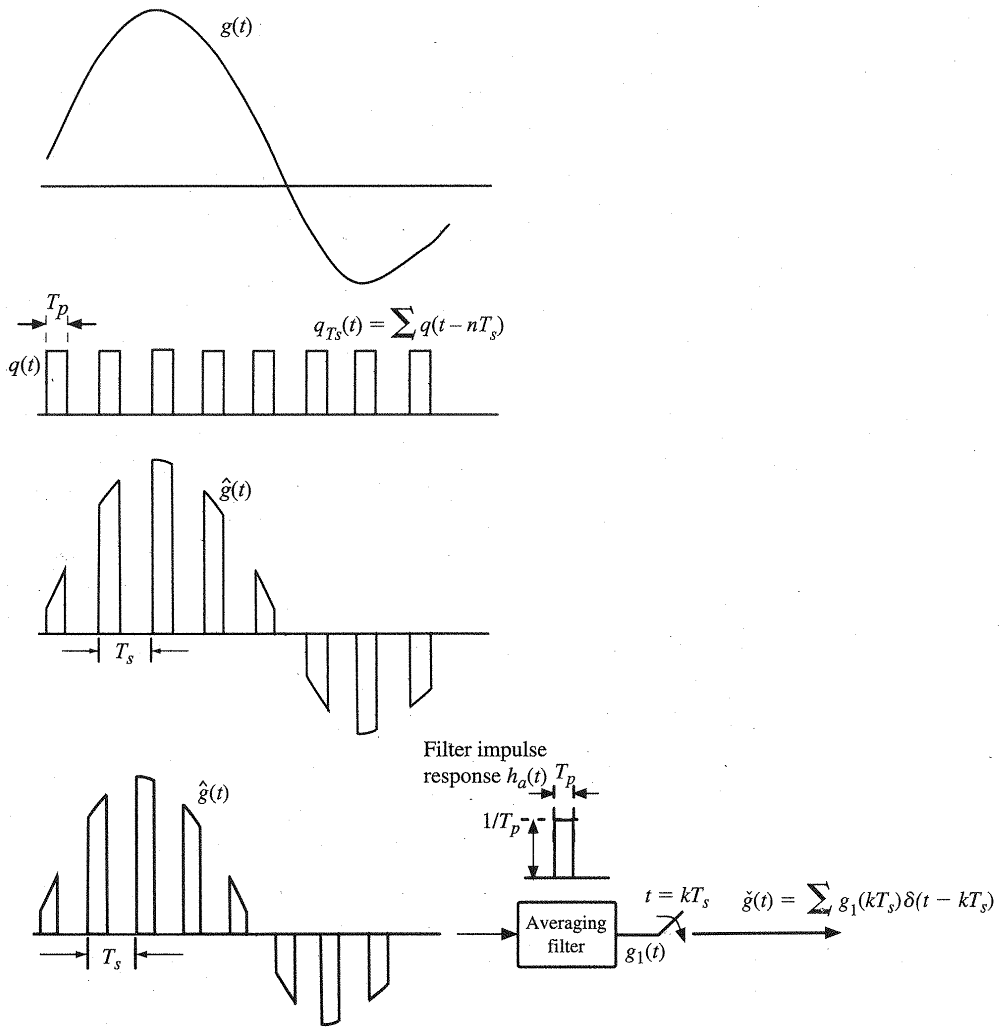
Practical samplers take each signal sample over a short time interval T_p around $t = kT_s$. In other words, every T_s seconds, the sampling device takes a short snapshot of duration T_p from the signal $g(t)$ being sampled. This is just like taking a sequence of still photographs of a sprinter during an 100-meter Olympic race. Much like a regular camera that generates a still picture by averaging the picture scene over the window T_p, the practical sampler would generate a sample value at $t = kT_s$ by averaging the values of signal $g(t)$ over the window T_p, that is,

$$g_1(kT_s) = \frac{1}{T_p} \int_{-T_p/2}^{T_p/2} g(kT_s + t)\, dt \tag{6.19a}$$

Depending on the actual device, this averaging may be weighted by a device-dependent averaging function $q(t)$ such that

$$g_1(kT_s) = \frac{1}{T_p} \int_{-T_p/2}^{T_p/2} q(t) g(kT_s + t)\, dt \tag{6.19b}$$

Figure 6.10 Illustration of practical sampling.


$g(t)$
T_p
$q_{T_s}(t) = \sum q(t - nT_s)$
$q(t)$
$\hat{g}(t)$
T_s
Filter impulse response $h_a(t)$
T_p
$1/T_p$
$\hat{g}(t)$
T_s
Averaging filter
$g_1(t)$
$t = kT_s$
$\breve{g}(t) = \sum g_1(kT_s)\delta(t - kT_s)$

Thus we have used the camera analogy to establish that practical samplers in fact generate sampled signal of the form

$$\breve{g}(t) = \sum g_1(kT_s)\delta(t - kT_s) \tag{6.20}$$

We will now show the relationship between the practically sampled signal $\breve{g}(t)$ and the original low-pass analog signal $g(t)$ in the frequency domain.

We will use Fig. 6.10 to illustrate the relationship between $\breve{g}(t)$ and $g(t)$ for the special case of uniform weighting. This means that

$$q(t) = \begin{cases} 1 & |t| \le 0.5T_p \\ 0 & |t| > 0.5T_p \end{cases}$$

As shown in Fig. 6.10, $g_1(t)$ can be equivalently obtained by first using "natural gating" to generate the signal *snapshots*

$$\widehat{g}(t) = g(t) \cdot q_{T_s}(t) \tag{6.21}$$

where

$$q_{T_s}(t) = \sum_{n=-\infty}^{\infty} q(t - nT_s)$$

Figure 6.10b illustrates the snapshot signal $\hat{g}(t)$. We can then define an averaging filter with impulse response

$$h_a(t) = \begin{cases} \dfrac{1}{T_p} & -\dfrac{T_p}{2} \leq t < \dfrac{T_p}{2} \\ 0 & \text{elsewhere} \end{cases}$$

or transfer function

$$H_a(f) = \text{sinc}\left(\pi f T_p\right)$$

Sending the naturally gated snapshot signal $\hat{g}(t)$ into the averaging filter generates the output signal

$$g_1(t) = h_a(t) * \hat{g}(t)$$

As illustrated in Fig. 6.10c, the practical sampler generate a sampled signal $\check{g}(t)$ by sampling the averaging filter output $g_1(kT_s)$. Thus we have used Fig. 6.10c to establish the equivalent process of taking snapshots, averaging, and sampling in generating practical samples of $g(t)$. Now we can examine the frequency domain relationships to analyze the distortion generated by practical samplers.

In the following analysis, we will consider a general weighting function $q(t)$ whose only constraint is that

$$q(t) = 0, \qquad t \notin (-0.5T_p, 0.5T_p)$$

To begin, note that $q_{T_s}(t)$ is periodic. Therefore, its Fourier series can be written as

$$q_{T_s}(t) = \sum_{n=-\infty}^{\infty} Q_n e^{jn\omega_s t}$$

where

$$Q_n = \frac{1}{T_s} \int_{-0.5T_p}^{0.5T_p} q(t) e^{-jn\omega_s t} dt$$

Thus, the averaging filter output signal is

$$\begin{aligned} g_1(t) &= h_a(t) * \left[g(t) q_{T_s}(t)\right] \\ &= h_a(t) * \sum_{n=-\infty}^{\infty} Q_n g(t) e^{jn\omega_s t} \qquad (6.22) \end{aligned}$$

In the frequency domain, we have

$$G_1(f) = H(f) \sum_{n=-\infty}^{\infty} Q_n G(f - nf_s)$$
$$= \text{sinc}\left(\pi f T_p\right) \sum_{n=-\infty}^{\infty} Q_n G(f - nf_s) \tag{6.23}$$

Because

$$\breve{g}(t) = \sum_k g_1(kT_s)\delta(t - kT_s)$$

we can apply the sampling theorem to show that

$$\breve{G}(f) = \frac{1}{T_s}\sum_m G_1(f + mf_s)$$
$$= \frac{1}{T_s}\sum_m \text{sinc}\left[\frac{(2\pi f + m2\pi f_s)T_p}{2}\right]\sum_n Q_n G(f + mf_s - nf_s)$$
$$= \sum_\ell \left(\frac{1}{T_s}\sum_n Q_n \,\text{sinc}\left[(\pi f + (n+\ell)\pi f_s)T_p\right]\right) G(f + \ell f_s) \tag{6.24}$$

The last equality came from the change of the summation index $\ell = m - n$.

We can define frequency responses

$$F_\ell(f) = \frac{1}{T_s}\sum_n Q_n \,\text{sinc}\left[(\pi f + (n+\ell)\pi f_s)T_p\right]$$

This definition allows us to conveniently write

$$\breve{G}(f) = \sum_\ell F_\ell(f) G_1(f + \ell f_s) \tag{6.25}$$

For the low-pass signal $G(f)$ with bandwidth B Hz, applying an ideal low-pass (interpolation) filter will generate a distorted signal

$$F_0(f)G(f) \tag{6.26a}$$

in which

$$F_0(f) = \frac{1}{T_s}\sum_n Q_n \,\text{sinc}\left[\pi(f + nf_s)T_p\right] \tag{6.26b}$$

It can be seen from Eqs. (6.25) and (6.26) that the practically sampled signal already contains a known distortion $F_0(f)$.

Moreover, the use of a practical reconstruction pulse $p(t)$ as in Eq. (6.12) will generate additional distortion. Let us reconstruct $g(t)$ by using the practical samples to generate

$$\check{g}(t) = \sum_n g_1(nT_s)p(t - nT_s)$$

Then from Eq. (6.13) we obtain the relationship between the spectra of the reconstruction and the original message $G(f)$ as

$$\check{G}(f) = P(f)\sum_n F_n(f)G(f + nf_s) \tag{6.27}$$

Since $G(f)$ has bandwidth B Hz, we will need to design a new equalizer with transfer function $E(f)$ such that the reconstruction is distortionless within the bandwidth B, that is,

$$E(f)P(f)F_0(f) = \begin{cases} 1 & |f| < B \\ \text{Flexible} & B < |f| < f_s - B \\ 0 & |f| > f_s - B \end{cases} \tag{6.28}$$

This single equalizer can be designed to compensate for two sources of distortion: nonideal sampling effect in $F_0(f)$ and nonideal reconstruction effect in $P(f)$. The equalizer design is made practically possible because both distortions are known in advance.

6.1.5 Some Applications of the Sampling Theorem

The sampling theorem is very important in signal analysis, processing, and transmission because it allows us to replace a continuous time signal by a discrete sequence of numbers. Processing a continuous time signal is therefore equivalent to processing a discrete sequence of numbers. This leads us directly into the area of digital filtering. In the field of communication, the transmission of a continuous time message reduces to the transmission of a sequence of numbers. This opens doors to many new techniques of communicating continuous time signals by pulse trains. The continuous time signal $g(t)$ is sampled, and sample values are used to modify certain parameters of a periodic pulse train. We may vary the amplitudes (Fig. 6.11b), widths (Fig. 6.11c), or positions (Fig. 6.11d) of the pulses in proportion to the sample values of the signal $g(t)$. Accordingly, we can have **pulse amplitude modulation** (PAM), **pulse width modulation** (PWM), or **pulse position modulation** (PPM). The most important form of pulse modulation today is **pulse code modulation** (PCM), introduced in Sec. 1.2. In all these cases, instead of transmitting $g(t)$, we transmit the corresponding pulse-modulated signal. At the receiver, we read the information of the pulse-modulated signal and reconstruct the analog signal $g(t)$.

One advantage of using pulse modulation is that it permits the simultaneous transmission of several signals on a time-sharing basis—**time division multiplexing (TDM)**. Because a pulse-modulated signal occupies only a part of the channel time, we can transmit several pulse-modulated signals on the same channel by interweaving them. Figure 6.12 shows the TDM of two PAM signals. In this manner we can multiplex several signals on the same channel by reducing pulse widths.

Another method of transmitting several baseband signals simultaneously is frequency division multiplexing (FDM), briefly discussed in Chapter 4. In FDM, various signals are multiplexed by sharing the channel bandwidth. The spectrum of each message is shifted to a specific band not occupied by any other signal. The information of various signals is located in nonoverlapping frequency bands of the channel. In a way, TDM and FDM are duals of each other.

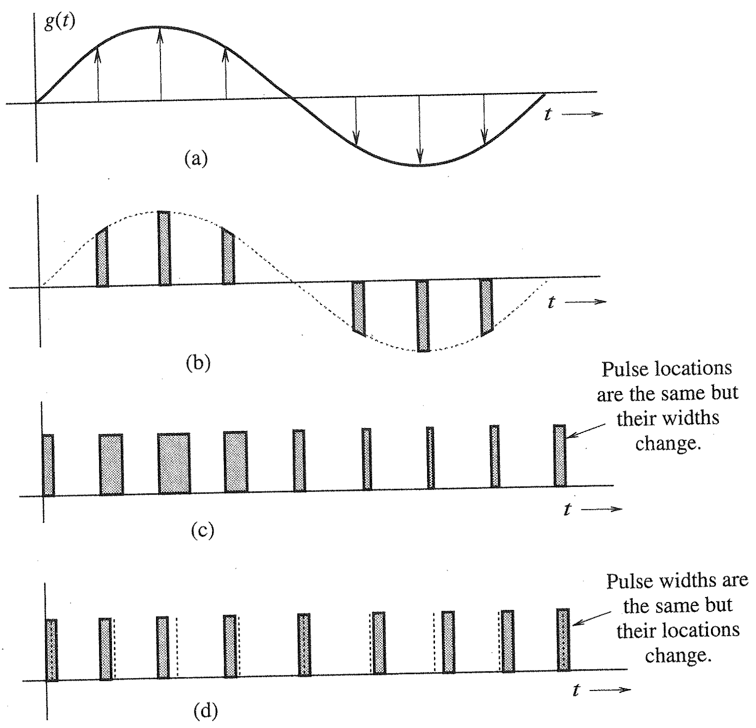


Figure 6.11 Pulse-modulated signals. (a) The unmodulated signal. (b) The PAM signal. (c) The PWM (PDM) signal. (d) The PPM signal.

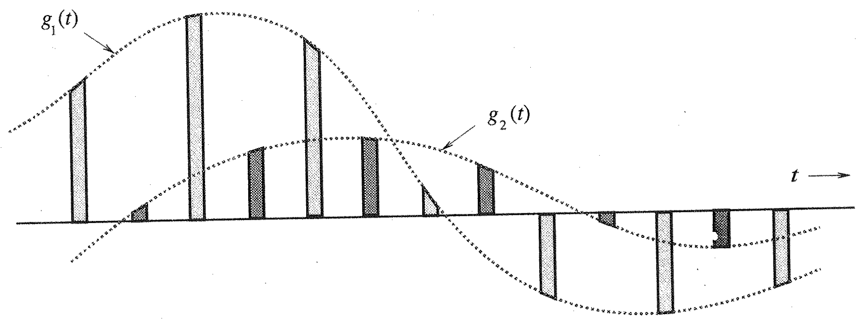


Figure 6.12 Time division multiplexing of two signals.

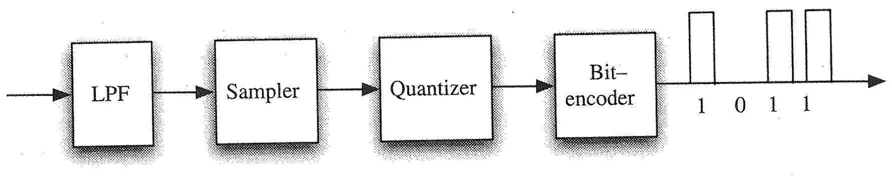


Figure 6.13 PCM system diagram.

6.2 PULSE CODE MODULATION (PCM)

PCM is the most useful and widely used of all the pulse modulations mentioned. As shown in Fig. 6.13, PCM basically is a tool for converting an analog signal into a digital signal (A/D conversion). An **analog** signal is characterized by an amplitude that can take on any value over a continuous range. This means that it can take on an infinite number of values. On the other hand, **digital** signal amplitude can take on only a finite number of values. An analog signal can

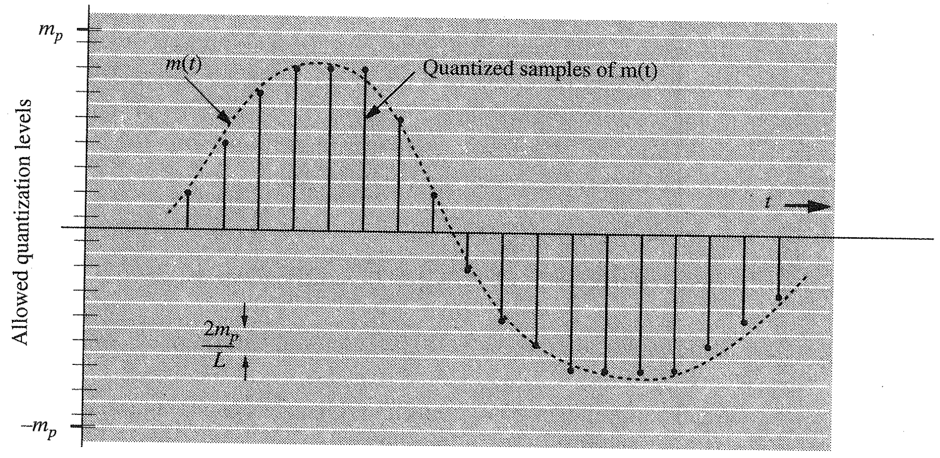


Figure 6.14 Quantization of a sampled analog signal.

be converted into a digital signal by means of sampling and **quantizing**, that is, rounding off its value to one of the closest permissible numbers (or **quantized levels**), as shown in Fig. 6.14. The amplitudes of the analog signal $m(t)$ lie in the range $(-m_p, m_p)$, which is partitioned into L subintervals, each of magnitude $\Delta v = 2m_p/L$. Next, each sample amplitude is approximated by the midpoint value of the subinterval in which the sample falls (see Fig. 6.14 for $L = 16$). Each sample is now approximated to one of the L numbers. Thus, the signal is digitized, with quantized samples taking on any one of the L values. Such a signal is known as an ***L*-ary digital signal**.

From practical viewpoint, a binary digital signal (a signal that can take on only two values) is very desirable because of its simplicity, economy, and ease of engineering. We can convert an L-ary signal into a binary signal by using pulse coding. Such a coding for the case of $L = 16$ was shown in Fig. 1.5. This code, formed by binary representation of the 16 decimal digits from 0 to 15, is known as the **natural binary code (NBC)**. Other possible ways of assigning a binary code will be discussed later. Each of the 16 levels to be transmitted is assigned one binary code of four digits. The analog signal $m(t)$ is now converted to a (binary) digital signal. A **bi**nary digi**t** is called a **bit** for convenience. This contraction of "binary digit" to "bit" has become an industry standard abbreviation and is used throughout the book.

Thus, each sample in this example is encoded by four bits. To transmit this binary data, we need to assign a distinct pulse shape to each of the two bits. One possible way is to assign a negative pulse to a binary **0** and a positive pulse to a binary **1** (Fig. 1.5) so that each sample is now transmitted by a group of four binary pulses (pulse code). The resulting signal is a binary signal.

The audio signal bandwidth is about 15 kHz. However, for speech, subjective tests show that signal articulation (intelligibility) is not affected if all the components above 3400 Hz are suppressed.*[,3] Since the objective in telephone communication is intelligibility rather than high fidelity, the components above 3400 Hz are eliminated by a low-pass filter. The resulting signal is then sampled at a rate of 8000 samples per second (8 kHz). This rate is intentionally kept higher than the Nyquist sampling rate of 6.8 kHz so that realizable filters can be applied for signal reconstruction. Each sample is finally quantized into 256 levels ($L = 256$), which requires a group of eight binary pulses to encode each sample ($2^8 = 256$). Thus, a telephone signal requires $8 \times 8000 = 64,000$ binary pulses per second.

* Components below 300 Hz may also be suppressed without affecting the articulation.

The compact disc (CD) is a more recent application of PCM. This is a high-fidelity situation requiring the audio signal bandwidth to be 20 kHz. Although the Nyquist sampling rate is only 40 kHz, the actual sampling rate of 44.1 kHz is used for the reason mentioned earlier. The signal is quantized into a rather large number ($L = 65,536$) of quantization levels, each of which is represented by 16 bits to reduce the quantizing error. The binary-coded samples (1.4 million bit/s) are then recorded on the compact disc.

6.2.1 Advantages of Digital Communication

Here are some of the advantages of digital communication over analog communication.

1. Digital communication, which can withstand channel noise and distortion much better than analog as long as the noise and the distortion are within limits, is more rugged than analog communication. With analog messages, on the other hand, any distortion or noise, no matter how small, will distort the received signal.
2. The greatest advantage of digital communication over analog communication, however, is the viability of regenerative repeaters in the former. In an analog communication system, a message signal becomes progressively weaker as it travels along the channel, whereas the cumulative channel noise and the signal distortion grow progressively stronger. Ultimately the signal is overwhelmed by noise and distortion. Amplification offers little help because it enhances the signal and the noise by the same proportion. Consequently, the distance over which an analog message can be transmitted is limited by the initial transmission power. For digital communications, a long transmission path may also lead to overwhelming noise and interferences. The trick, however, is to set up repeater stations along the transmission path at distances short enough to be able to detect signal pulses before the noise and distortion have a chance to accumulate sufficiently. At each repeater station the pulses are detected, and new, clean pulses are transmitted to the next repeater station, which, in turn, duplicates the same process. If the noise and distortion are within limits (which is possible because of the closely spaced repeaters), pulses can be detected correctly.* This way the digital messages can be transmitted over longer distances with greater reliability. The most significant error in PCM comes from quantizing. This error can be reduced as much as desired by increasing the number of quantizing levels, the price of which is paid in an increased bandwidth of the transmission medium (channel).
3. Digital hardware implementation is flexible and permits the use of microprocessors, digital switching, and large-scale integrated circuits.
4. Digital signals can be coded to yield extremely low error rates and high fidelity as well as for privacy.
5. It is easier and more efficient to multiplex several digital signals.
6. Digital communication is inherently more efficient than analog in exchanging SNR for bandwidth.
7. Digital signal storage is relatively easy and inexpensive. It also has the ability to search and select information from distant electronic database.
8. Reproduction with digital messages can be extremely reliable without deterioration. Analog messages such as photocopies and films, for example, lose quality at each successive stage of reproduction and must be transported physically from one distant place to another, often at relatively high cost.

* The error in pulse detection can be made negligible.

9. The cost of digital hardware continues to halve every two or three years, while performance or capacity doubles over the same time period. And there is no end in sight yet to this breathtaking and relentless exponential progress in digital technology. As a result, digital technologies today dominate in any given area of communication or storage technologies.

A Historical Note

The ancient Indian writer Pingala applied what turns out to be advanced mathematical concepts for describing prosody, and in doing so presented the first known description of a binary numeral system, possibly as early as the eighth century BCE.[6] Others, like R. Hall in Mathematics of Poetry place him later, circa 200 BCE. Gottfried Wilhelm Leibniz (1646–1716) was the first mathematician in the West to work out systematically the binary representation (using 1s and 0s) for any number. He felt a spiritual significance in this discovery, believing that **1**, representing unity, was clearly a symbol for God, while **0** represented nothingness. He reasoned that if all numbers can be represented merely by the use of **1** and **0**, this surely proves that God created the universe out of nothing!

6.2.2 Quantizing

As mentioned earlier, digital signals come from a variety of sources. Some sources such as computers are inherently digital. Some sources are analog, but are converted into digital form by a variety of techniques such as PCM and delta modulation (DM), which will now be analyzed. The rest of this section provides quantitative discussion of PCM and its various aspects, such as quantizing, encoding, synchronizing, the required transmission bandwidth and SNR.

For quantization, we limit the amplitude of the message signal $m(t)$ to the range $(-m_p, m_p)$, as shown in Fig. 6.14. Note that m_p is not necessarily the peak amplitude of $m(t)$. The amplitudes of $m(t)$ beyond $\pm m_p$ are simply chopped off. Thus, m_p is not a parameter of the signal $m(t)$; rather, it is the limit of the quantizer. The amplitude range $(-m_p, m_p)$ is divided into L uniformly spaced intervals, each of width $\Delta v = 2m_p/L$. A sample value is approximated by the midpoint of the interval in which it lies (Fig. 6.14). The quantized samples are coded and transmitted as binary pulses. At the receiver some pulses may be detected incorrectly. Hence, there are two sources of error in this scheme: *quantization error* and *pulse detection error*. In almost all practical schemes, the pulse detection error is quite small compared to the quantization error and can be ignored. In the present analysis, therefore, we shall assume that the error in the received signal is caused exclusively by quantization.

If $m(kT_s)$ is the kth sample of the signal $m(t)$, and if $\hat{m}(kT_s)$ is the corresponding quantized sample, then from the interpolation formula in Eq. (6.10),

$$m(t) = \sum_k m(kT_s) \text{ sinc } (2\pi Bt - k\pi)$$

and

$$\hat{m}(t) = \sum_k \hat{m}(kT_s) \text{ sinc } (2\pi Bt - k\pi)$$

where $\hat{m}(t)$ is the signal reconstructed from quantized samples. The distortion component $q(t)$ in the reconstructed signal is $q(t) = \hat{m}(t) - m(t)$. Thus,

$$\begin{aligned} q(t) &= \sum_k [\hat{m}(kT_s) - m(kT_s)] \operatorname{sinc}(2\pi Bt - k\pi) \\ &= \sum_k q(kT_s) \operatorname{sinc}(2\pi Bt - k\pi) \end{aligned}$$

where $q(kT_s)$ is the quantization error in the kth sample. The signal $q(t)$ is the undesired signal, and, hence, acts as noise, known as **quantization noise**. To calculate the power, or the mean square value of $q(t)$, we have

$$\begin{aligned} \widetilde{q^2(t)} &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} q^2(t)\, dt \\ &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[\sum_k q(kT_s) \operatorname{sinc}(2\pi Bt - k\pi) \right]^2 dt \end{aligned} \tag{6.29a}$$

We can show that (see Prob. 3.7-4) the signals $\operatorname{sinc}(2\pi Bt - m\pi)$ and $\operatorname{sinc}(2\pi Bt - n\pi)$ are orthogonal, that is,

$$\int_{-\infty}^{\infty} \operatorname{sinc}(2\pi Bt - m\pi) \operatorname{sinc}(2\pi Bt - n\pi)\, dt = \begin{cases} 0 & m \neq n \\ \dfrac{1}{2B} & m = n \end{cases} \tag{6.29b}$$

Because of this result, the integrals of the cross-product terms on the right-hand side of Eq. (6.29a) vanish, and we obtain

$$\begin{aligned} \widetilde{q^2(t)} &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} \sum_k q^2(kT_s) \operatorname{sinc}^2(2\pi Bt - k\pi)\, dt \\ &= \lim_{T\to\infty} \frac{1}{T} \sum_k q^2(kT_s) \int_{-T/2}^{T/2} \operatorname{sinc}^2(2\pi Bt - k\pi)\, dt \end{aligned}$$

From the orthogonality relationship (6.29b), it follows that

$$\widetilde{q^2(t)} = \lim_{T\to\infty} \frac{1}{2BT} \sum_k q^2(kT_s) \tag{6.30}$$

Because the sampling rate is $2B$, the total number of samples over the averaging interval T is $2BT$. Hence, the right-hand side of Eq. (6.30) represents the average, or the mean of the square of the quantization error. The quantum levels are separated by $\Delta v = 2m_p/L$. Since a sample value is approximated by the midpoint of the subinterval (of height Δv) in which the sample falls, the maximum quantization error is $\pm\Delta v/2$. Thus, the quantization error lies in the range $(-\Delta v/2, \Delta v/2)$, where

$$\Delta v = \frac{2m_p}{L} \tag{6.31}$$

Assuming that the error is equally likely to lie anywhere in the range $(-\Delta v/2,\ \Delta v/2)$, the mean square quantizing error $\widetilde{q^2}$ is given by*

$$\widetilde{q^2} = \frac{1}{\Delta v}\int_{-\Delta v/2}^{\Delta v/2} q^2\,dq$$

$$= \frac{(\Delta v)^2}{12} \tag{6.32}$$

$$= \frac{m_p^2}{3L^2} \tag{6.33}$$

Because $\widetilde{q^2(t)}$ is the mean square value or power of the quantization noise, we shall denote it by N_q,

$$N_q = \widetilde{q^2(t)} = \frac{m_p^2}{3L^2}$$

Assuming that the pulse detection error at the receiver is negligible, the reconstructed signal $\hat{m}(t)$ at the receiver output is

$$\hat{m}(t) = m(t) + q(t)$$

The desired signal at the output is $m(t)$, and the (quantization) noise is $q(t)$. Since the power of the message signal $m(t)$ is $\widetilde{m^2(t)}$, then

$$S_o = \widetilde{m^2(t)}$$

$$N_o = N_q = \frac{m_p^2}{3L^2}$$

and

$$\frac{S_o}{N_o} = 3L^2\frac{\widetilde{m^2(t)}}{m_p^2} \tag{6.34}$$

In this equation, m_p is the peak amplitude value that a quantizer can accept, and is therefore a parameter of the quantizer. This means S_o/N_o, the SNR, is a linear function of the message signal power $\widetilde{m^2(t)}$ (see Fig. 6.18 with $\mu = 0$).

---

* Those who are familiar with the theory of probability can derive this result directly by noting that the probability density of the quantization error q is $1/(2m_p/L) = L/2m_p$ over the range $|q| \le m_p/L$ and is zero elsewhere. Hence,

$$\widetilde{q^2} = \int_{-m_p/L}^{m_p/L} q^2 p(q)\,dq = \int_{-m_p/L}^{m_p/L} \frac{L}{2m_p} q^2\,dq = \frac{m_p^2}{3L^2}$$

6.2.3 Principle of Progressive Taxation: Nonuniform Quantization

Recall that S_o/N_o, the SNR, is an indication of the quality of the received signal. Ideally we would like to have a constant SNR (the same quality) for all values of the message signal power $m^2(t)$. Unfortunately, the SNR is directly proportional to the signal power $m^2(t)$, which varies from speaker to speaker by as much as 40 dB (a power ratio of 10^4). The signal power can also vary because of the different lengths of the connecting circuits. This indicates that the SNR in Eq. (6.34) can vary widely, depending on the speaker and the length of the circuit. Even for the same speaker, the quality of the received signal will deteriorate markedly when the person speaks softly. Statistically, it is found that smaller amplitudes predominate in speech and larger amplitudes are much less frequent. This means the SNR will be low most of the time.

The root of this difficulty lies in the fact that the quantizing steps are of uniform value $\Delta v = 2m_p/L$. The quantization noise $N_q = (\Delta v)^2/12$ [Eq. (6.32)] is directly proportional to the square of the step size. The problem can be solved by using smaller steps for smaller amplitudes (nonuniform quantizing), as shown in Fig. 6.15a. The same result is obtained by first compressing signal samples and then using a uniform quantization. The input-output characteristics of a compressor are shown in Fig. 6.15b. The horizontal axis is the normalized input signal (i.e., the input signal amplitude m divided by the signal peak value m_p). The vertical axis is the output signal y. The compressor maps input signal increments Δm into larger increments Δy for small input signals, and vice versa for large input signals. Hence, a given interval Δm contains a larger number of steps (or smaller step size) when m is small. The quantization noise is lower for smaller input signal power. An approximately logarithmic compression characteristic yields a quantization noise nearly proportional to the signal power $m^2(t)$, thus making the SNR practically independent of the input signal power over a large dynamic range[5] (see later Fig. 6.18). This approach of equalizing the SNR appears similar to the use of progressive income tax to equalize incomes. The loud talkers and stronger signals are penalized with higher noise steps Δv to compensate the soft talkers and weaker signals.

Among several choices, two compression laws have been accepted as desirable standards by the ITU-T:[6] the μ-law used in North America and Japan, and the A-law used in Europe and the rest of the world and on international routes. Both the μ-law and the A-law curves have odd symmetry about the vertical axis. The μ-law (for positive amplitudes) is given by

$$y = \frac{1}{\ln(1+\mu)} \ln\left(1 + \frac{\mu m}{m_p}\right) \qquad 0 \le \frac{m}{m_p} \le 1 \tag{6.35a}$$

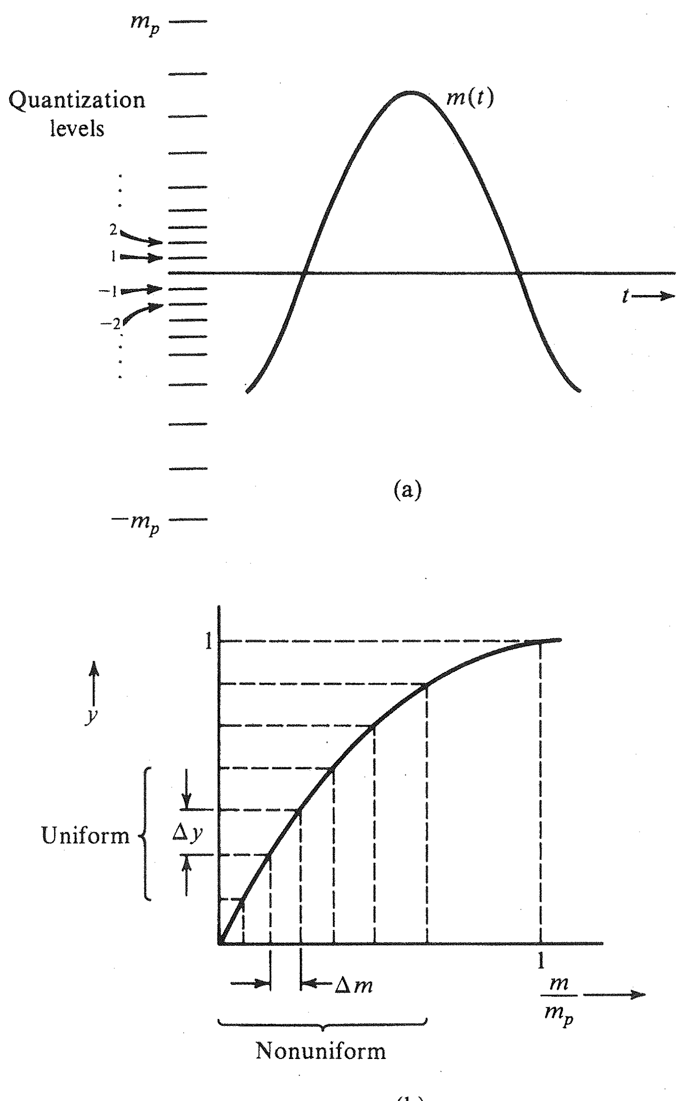
The A-law (for positive amplitudes) is

$$y = \begin{cases} \dfrac{A}{1+\ln A}\left(\dfrac{m}{m_p}\right) & 0 \le \dfrac{m}{m_p} \le \dfrac{1}{A} \\[2ex] \dfrac{1}{1+\ln A}\left(1 + \ln\dfrac{Am}{m_p}\right) & \dfrac{1}{A} \le \dfrac{m}{m_p} \le 1 \end{cases} \tag{6.35b}$$

These characteristics are shown in Fig. 6.16.

The compression parameter μ (or A) determines the degree of compression. To obtain a nearly constant S_o/N_o over a dynamic range of for input signal power 40 dB, μ should be greater than 100. Early North American channel banks and other digital terminals used a value of $\mu = 100$, which yielded the best results for 7-bit (128-level) encoding. An optimum value

Figure 6.15 Nonuniform quantization.



of $\mu = 255$ has been used for all North American 8-bit (256-level) digital terminals, and the earlier value of μ is now almost extinct. For the A-law, a value of $A = 87.6$ gives comparable results and has been standardized by the ITU-T.[6]

The compressed samples must be restored to their original values at the receiver by using an expander with a characteristic complementary to that of the compressor. The compressor and the expander together are called the **compandor**. Figure 6.17 describes the use of compressor and expander along with a uniform quantizer to achieve nonuniform quantization.

Generally speaking, time compression of a signal increases its bandwidth. But in PCM, we are compressing not the signal $m(t)$ in time but its sample values. Because neither the time scale not the number of samples changes, the problem of bandwidth increase does not arise here. It happens that when a μ-law compandor is used, the output SNR is

$$\frac{S_o}{N_o} \simeq \frac{3L^2}{[\ln(1+\mu)]^2} \qquad \mu^2 \gg \frac{m_p^2}{\widetilde{m^2(t)}} \tag{6.36}$$

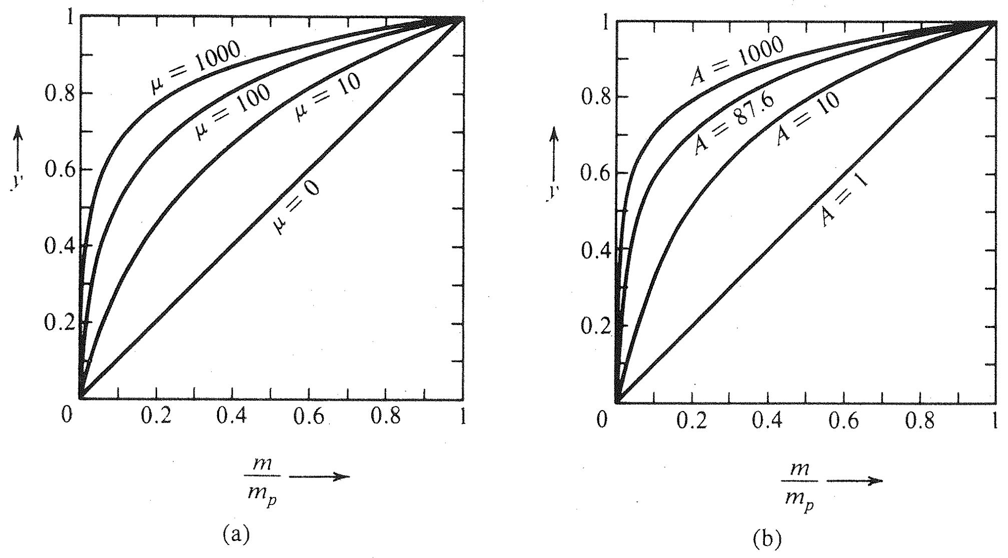


Figure 6.16 (a) μ-Law characteristic. (b) A-Law characteristic.

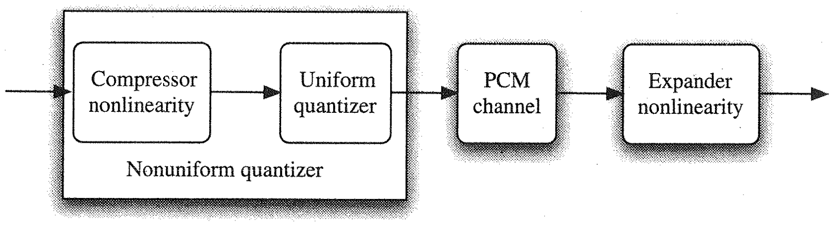


Figure 6.17 Utilization of compressor and expander for nonuniform quantization.

The output SNR for the cases of $\mu = 255$ and $\mu = 0$ (uniform quantization) as a function of $\widetilde{m^2(t)}$ (the message signal power) is shown in Fig. 6.18.

The Compandor

A logarithmic compressor can be realized by a semiconductor diode, because the *V–I* characteristic of such a diode is of the desired form in the first quadrant:

$$V = \frac{KT}{q} \ln \left(1 + \frac{I}{I_s}\right)$$

Two matched diodes in parallel with opposite polarity provide the approximate characteristic in the first and third quadrants (ignoring the saturation current). In practice, adjustable resistors are placed in series with each diode and a third variable resistor is added in parallel. By adjusting various resistors, the resulting characteristic is made to fit a finite number of points (usually seven) on the ideal characteristics.

An alternative approach is to use a piecewise linear approximation to the logarithmic characteristics. A 15-segmented approximation (Fig. 6.19) to the eighth bit ($L = 256$) with $\mu = 255$ law is widely used in the D2 channel bank that is used in conjunction with the T1 carrier system. The segmented approximation is only marginally inferior in terms of SNR.[8] The piecewise linear approximation has almost universally replaced earlier logarithmic approximations to the true $\mu = 255$ characteristic and is the method of choice in North American standards.

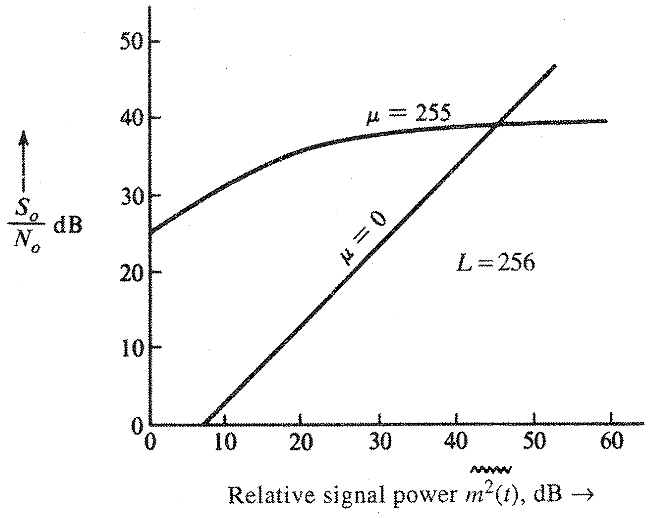


Figure 6.18 Ratio of signal to quantization noise in PCM with and without compression.

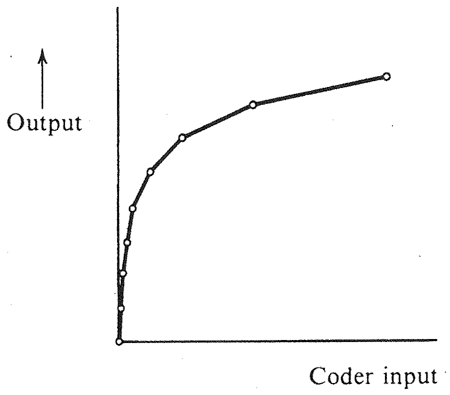


Figure 6.19 Piecewise linear compressor characteristic.

Though a true $\mu = 255$ compressor working with a $\mu = 255$ expander will be superior to similar piecewise linear devices, a digital terminal device exhibiting the true characteristic in today's network must work end-to-end against other network elements that use the piecewise linear approximation. Such a combination of differing characteristics is inferior to either of the characteristics obtained when the compressor and the expander operate using the same compression law.

In the standard audio file format used by Sun, Unix and Java, the audio in "au" files can be pulse-code-modulated or compressed with the ITU-T G.711 standard through either the μ-law or the A-law.[6] The μ-law compressor ($\mu = 255$) converts 14-bit signed linear PCM samples to logarithmic 8-bit samples, leading to storage saving. The A-law compressor ($A = 87.6$) converts 13-bit signed linear PCM samples to logarithmic 8-bit samples. In both cases, sampling at the rate of 8000 Hz, a G.77 encoder thus creates from audio signals bit streams at 64 kilobits per second (kbit/s). Since the A-law and the μ-law are mutually compatible, audio recoded into "au" files can be decoded in either format. It should be noted that the Microsoft WAV audio format also has compression options that use μ-law and A-law.

The PCM Encoder

The multiplexed PAM output is applied at the input of the encoder, which quantizes and encodes each sample into a group of n binary digits. A variety of encoders is available.[7, 10] We shall discuss here the **digit-at-a-time** encoder, which makes n sequential comparisons to generate an n-bit codeword. The sample is compared with a voltage obtained by a combination of reference voltages proportional to $2^7, 2^6, 2^5, \ldots, 2^0$. The reference voltages are conveniently generated by a bank of resistors $R, 2R, 2^2R, \ldots, 2^7R$.

The encoding involves answering successive questions, beginning with whether the sample is in the upper or lower half of the allowed range. The first code digit **1** or **0** is generated, depending on whether the sample is in the upper or the lower half of the range. In the second step, another digit **1** or **0** is generated, depending on whether the sample is in the upper or the lower half of the subinterval in which it has been located. This process continues until the last binary digit in the code has been generated.

Decoding is the inverse of encoding. In this case, each of the n digits is applied to a resistor of different value. The kth digit is applied to a resistor 2^kR. The currents in all the resistors are added. The sum is proportional to the quantized sample value. For example, a binary code word **10010110** will give a current proportional to $2^7 + 0 + 0 + 2^4 + 0 + 2^2 + 2^1 + 0 = 150$. This completes the D/A conversion.

6.2.4 Transmission Bandwidth and the Output SNR

For a binary PCM, we assign a distinct group of n binary digits (bits) to each of the L quantization levels. Because a sequence of n binary digits can be arranged in 2^n distinct patterns,

$$L = 2^n \qquad \text{or} \qquad n = \log_2 L \tag{6.37}$$

each quantized sample is, thus, encoded into n bits. Because a signal $m(t)$ band-limited to B Hz requires a minimum of $2B$ samples per second, we require a total of $2nB$ bit/s, that is, $2nB$ pieces of information per second. Because a unit bandwidth (1 Hz) can transmit a maximum of two pieces of information per second (Sec. 6.1.3), we require a minimum channel of bandwidth B_T Hz, given by

$$B_T = nB \text{ Hz} \tag{6.38}$$

This is the theoretical minimum transmission bandwidth required to transmit the PCM signal. In Secs. 7.2 and 7.3, we shall see that for practical reasons we may use a transmission bandwidth higher than this minimum.

Example 6.2 A signal $m(t)$ band-limited to 3 kHz is sampled at a rate $33\frac{1}{3}\%$ higher than the Nyquist rate. The maximum acceptable error in the sample amplitude (the maximum quantization error) is 0.5% of the peak amplitude m_p. The quantized samples are binary coded. Find the minimum bandwidth of a channel required to transmit the encoded binary signal. If 24 such signals are time-division-multiplexed, determine the minimum transmission bandwidth required to transmit the multiplexed signal.

The Nyquist sampling rate is $R_N = 2 \times 3000 = 6000$ Hz (samples per second). The actual sampling rate is $R_A = 6000 \times (1\frac{1}{3}) = 8000$ Hz.

The quantization step is Δv, and the maximum quantization error is $\pm\Delta v/2$.

Therefore, from Eq. (6.31),

$$\frac{\Delta v}{2} = \frac{m_p}{L} = \frac{0.5}{100} m_p \Longrightarrow L = 200$$

For binary coding, L must be a power of 2. Hence, the next higher value of L that is a power of 2 is $L = 256$.

From Eq. (6.37), we need $n = \log_2 256 = 8$ bits per sample. We require to transmit a total of $C = 8 \times 8000 = 64,000$ bit/s. Because we can transmit up to 2 bit/s per hertz of bandwidth, we require a minimum transmission bandwidth $B_T = C/2 = 32$ kHz.

The multiplexed signal has a total of $C_M = 24 \times 64,000 = 1.536$ Mbit/s, which requires a minimum of $1.536/2 = 0.768$ MHz of transmission bandwidth.

Exponential Increase of the Output SNR

From Eq. (6.37), $L^2 = 2^{2n}$, and the output SNR in Eq. (6.34) or Eq. (6.36) can be expressed as

$$\frac{S_o}{N_o} = c(2)^{2n} \tag{6.39}$$

where

$$c = \begin{cases} \dfrac{3\ \widetilde{m^2(t)}}{m_p^2} & \text{[uncompressed case, in Eq. (6.34)]} \\[2ex] \dfrac{3}{[\ln(1+\mu)]^2} & \text{[compressed case, in Eq. (6.36)]} \end{cases}$$

Substitution of Eq. (6.38) into Eq. (6.39) yields

$$\frac{S_o}{N_o} = c(2)^{2B_T/B} \tag{6.40}$$

From Eq. (6.40) we observe that the SNR increases exponentially with the transmission bandwidth B_T. This trade of SNR for bandwidth is attractive and comes close to the upper theoretical limit. A small increase in bandwidth yields a large benefit in terms of SNR. This relationship is clearly seen by using the decibel scale to rewrite Eq. (6.39) as

$$\begin{aligned} \left(\frac{S_o}{N_o}\right)_{\text{dB}} &= 10 \log_{10}\left(\frac{S_o}{N_o}\right) \\ &= 10 \log_{10}[c(2)^{2n}] \\ &= 10 \log_{10} c + 2n \log_{10} 2 \\ &= (\alpha + 6n) \quad \text{dB} \end{aligned} \tag{6.41}$$

where $\alpha = 10\ \log_{10} c$. This shows that increasing n by 1 (increasing one bit in the codeword) quadruples the output SNR (a 6 dB increase). Thus, if we increase n from 8 to 9, the SNR quadruples, but the transmission bandwidth increases only from 32 kHz to 36 kHz (an increase of only 12.5%). This shows that in PCM, SNR can be controlled by transmission bandwidth. We shall see later that frequency and phase modulation also do this. But it requires a doubling of the bandwidth to quadruple the SNR. In this respect, PCM is strikingly superior to FM or PM.

Example 6.3 A signal $m(t)$ of bandwidth $B = 4$ kHz is transmitted using a binary companded PCM with $\mu = 100$. Compare the case of $L = 64$ with the case of $L = 256$ from the point of view of transmission bandwidth and the output SNR.

For $L = 64$, $n = 6$, and the transmission bandwidth is $nB = 24$ kHz,

$$\frac{S_o}{N_o} = (\alpha + 36) \text{ dB}$$

$$\alpha = 10 \log \frac{3}{[\ln (101)]^2} = -8.51$$

Hence,

$$\frac{S_o}{N_o} = 27.49 \text{ dB}$$

For $L = 256$, $n = 8$, and the transmission bandwidth is 32 kHz,

$$\frac{S_o}{N_o} = \alpha + 6n = 39.49 \text{ dB}$$

The difference between the two SNRs is 12 dB, which is a ratio of 16. Thus, the SNR for $L = 256$ is 16 times the SNR for $L = 64$. The former requires just about 33% more bandwidth compared to the latter.

Comments on Logarithmic Units

Logarithmic units and logarithmic scales are very convenient when a variable has a large dynamic range. Such is the case with frequency variables or SNRs. A logarithmic unit for the power ratio is the decibel (dB), defined as $10 \log_{10}$ (power ratio). Thus, an SNR is x dB, where

$$x = 10 \log_{10} \frac{S}{N}$$

We use the same unit to express power gain or loss over a certain transmission medium. For instance, if over a certain cable the signal power is attenuated by a factor of 15, the cable gain is

$$G = 10 \log_{10} \frac{1}{15} = -11.76 \text{ dB}$$

or the cable attenuation (loss) is 11.76 dB.

Although the decibel is a measure of power ratios, it is often used as a measure of power itself. For instance, "100 watt" may be considered to be a power ratio of 100 with respect to 1-watt power, and is expressed in units of dBW as

$$P_{\text{dBW}} = 10 \log_{10} 100 = 20 \text{ dBW}$$

Thus, 100-watt power is 20 dBW. Similarly, power measured with respect to 1 mW power is dBm. For instance, 100-watt power is

$$P_{\text{dBm}} = 10 \log \frac{100\text{ W}}{1\text{ mW}} = 50 \text{ dBm}$$

6.3 DIGITAL TELEPHONY: PCM IN T1 CARRIER SYSTEMS

A Historical Note

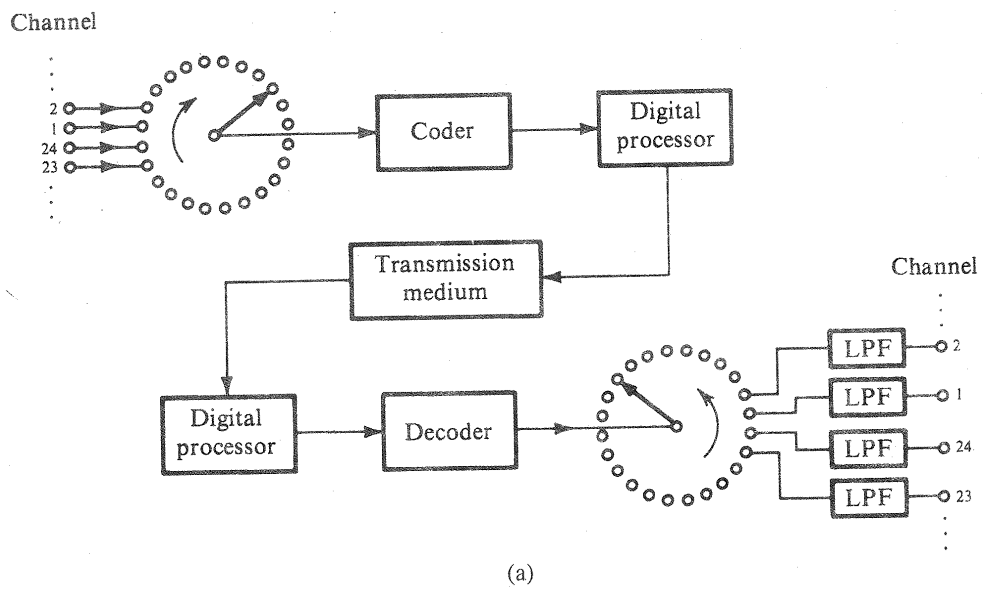
Because of the unavailability of suitable switching devices, more than 20 years elapsed between the invention of PCM and its implementation. Vacuum tubes, used before the invention of the transistor, were not only bulky, but they were poor switches and dissipated a lot of heat. Systems having vacuum tubes as switches were large, rather unreliable, and tended to overheat. PCM was just waiting for the invention of the transistor, which happens to be a small device that consumes little power and is a nearly ideal switch.

Coincidentally, at about the time the transistor was invented, the demand for telephone service had become so heavy that the existing system was overloaded, particularly in large cities. It was not easy to install new underground cables because space available under the streets in many cities was already occupied by other services (water, gas, sewer, etc.). Moreover, digging up streets and causing many dislocations was not very attractive. An attempt was made on a limited scale to increase the capacity by frequency-division-multiplexing several voice channels through amplitude modulation. Unfortunately, the cables were primarily designed for the audio voice range (0–4 kHz) and suffered severely from noise. Furthermore, cross talk between pairs of channels on the same cable was unacceptable at high frequencies. Ironically, PCM—requiring a bandwidth several times larger than that required for FDM signals—offered the solution. This is because digital systems with closely spaced regenerative repeaters can work satisfactorily on noisy lines that give poor high-frequency performance.[9] The repeaters, spaced approximately 6000 feet apart, clean up the signal and regenerate new pulses before the pulses get too distorted and noisy. This is the history of the Bell System's T1 carrier system.[3, 10] A pair of wires that used to transmit one audio signal of bandwidth 4 kHz is now used to transmit 24 time-division-multiplexed PCM telephone signals with a total bandwidth of 1.544 MHz.

T1 Time Division Multiplexing

A schematic of a T1 carrier system is shown in Fig. 6.20a. All 24 channels are sampled in a sequence. The sampler output represents a time-division-multiplexed PAM signal. The multiplexed PAM signal is now applied to the input of an encoder that quantizes each sample and encodes it into eight binary pulses—a binary codeword* (see Fig. 6.20b). The signal, now converted to digital form, is sent over the transmission medium. Regenerative repeaters spaced approximately 6000 feet apart detect the pulses and retransmit new pulses. At the receiver, the decoder converts the binary pulses into samples (decoding). The samples are then demultiplexed (i.e., distributed to each of the 24 channels). The desired audio signal is reconstructed by passing the samples through a low-pass filter in each channel.

* In an earlier version, each sample was encoded by seven bits. An additional bit was added for signaling.



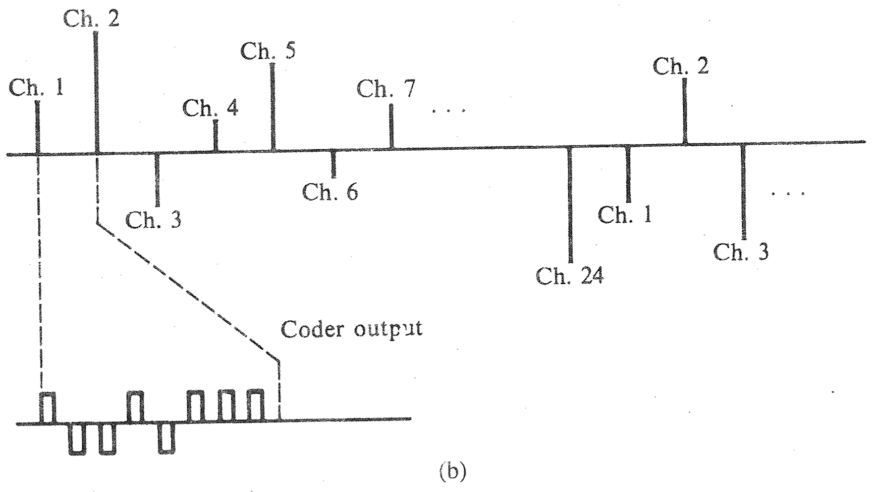


Figure 6.20 T1 carrier system.

The commutators in Fig. 6.20 are not mechanical but are high-speed electronic switching circuits. Several schemes are available for this purpose.[11] Sampling is done by electronic gates (such as a bridge diode circuit, as shown in Fig. 4.5a) opened periodically by narrow pulses of 2 μs duration. The 1.544 Mbit/s signal of the T1 system, called **digital signal level 1 (DS1)**, is used further to multiplex into progressively higher level signals DS2, DS3, and DS4, as described next, in Sec. 6.4

After the Bell System introduced the T1 carrier system in the United States, dozens of variations were proposed or adopted elsewhere before the ITU-T standardized its 30-channel PCM system with a rate of 2.048 Mbit/s (in contrast to T1, with 24 channels and 1.544 Mbit/s). The 30-channel system is used all over the world, except in North America and Japan. Because of the widespread adoption of the T1 carrier system in the United States and Japan before the ITU-T standardization, the two standards continue to be used in different parts of the world, with appropriate interfaces in international connections.

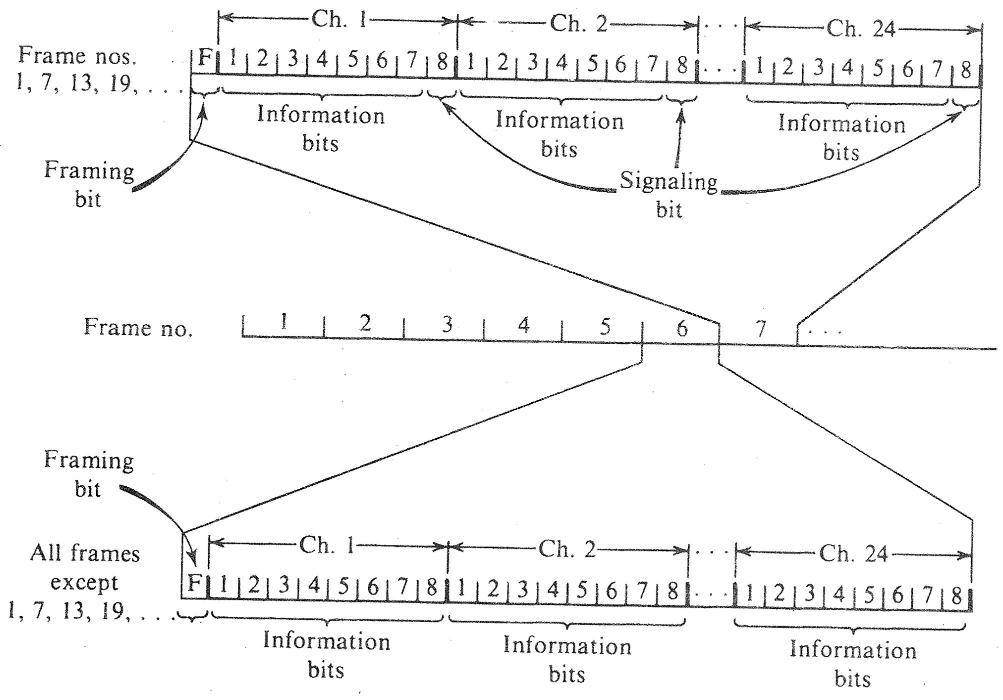

Ch. 1
Ch. 2
Ch. 24
Frame nos. 1, 7, 13, 19, . . .
F 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 . . . 1 2 3 4 5 6 7 8
Information bits
Information bits
Information bits
Framing bit
Signaling bit
Frame no. 1 2 3 4 5 6 7 . . .
Framing bit
All frames except 1, 7, 13, 19, . . .
Information bits
Information bits
Information bits

Figure 6.21 T1 system signaling format.

Synchronizing and Signaling

Binary codewords corresponding to samples of each of the 24 channels are multiplexed in a sequence, as shown in Fig. 6.21. A segment containing one codeword (corresponding to one sample) from each of the 24 channels is called a **frame**. Each frame has $24 \times 8 = 192$ information bits. Because the sampling rate is 8000 samples per second, each frame takes 125 μs. To separate information bits correctly at the receiver, it is necessary to be sure where each frame begins. Therefore, a **framing bit** is added at the beginning of each frame. This makes a total of 193 bits per frame. Framing bits are chosen so that a sequence of framing bits, one at the beginning of each frame, forms a special pattern that is unlikely to be formed in a speech signal.

The sequence formed by the first bit from each frame is examined by the logic of the receiving terminal. If this sequence does not follow the given code pattern (framing bit pattern), a synchronization loss is detected, and the next position is examined to determine whether it is actually the framing bit. It takes about 0.4 to 6 ms to detect and about 50 ms (in the worst possible case) to reframe.

In addition to information and framing bits, we need to transmit signaling bits corresponding to dialing pulses, as well as telephone on-hook/off-hook signals. When channels developed by this system are used to transmit signals between telephone switching systems, the switches must be able to communicate with each other to use the channels effectively. Since all eight bits are now used for transmission instead of the seven bits used in the earlier version,* the signaling channel provided by the eighth bit is no longer available. Since only a rather low-speed signaling channel is required, rather than create extra time slots for this information, we use one information bit (the least significant bit) of every sixth sample of a signal

* In the earlier version of T1, quantizing levels $L = 128$ required only seven information bits. The eighth bit was used for signaling.

to transmit this information. This means that every sixth sample of each voice signal will have a possible error corresponding to the least significant digit. Every sixth frame, therefore, has $7 \times 24 = 168$ information bits, 24 signaling bits, and 1 framing bit. In all the remaining frames, there are 192 information bits and 1 framing bit. This technique is called **$7\frac{5}{6}$** bit encoding, and the signaling channel so derived is called **robbed-bit signaling**. The slight SNR degradation suffered by impairing one out of six frames is considered to be an acceptable penalty. The signaling bits for each signal occur at a rate of $8000/6 = 1333$ bit/s. The frame format is shown in Fig. 6.21.

The older seven-bit framing format required only that frame boundaries be identified so that the channels could be located in the bit stream. When signaling is superimposed on the channels in every sixth frame, it is necessary to identify, at the receiver, which frames are the signaling frames. A new framing structure, called the **superframe**, was developed to take care of this. The framing bits are transmitted at 8 kbit/s as before and occupy the first bit of each frame. The framing bits form a special pattern, which repeats in 12 frames: **100011011100**. The pattern thus allows the identification of frame boundaries as before, but also allows the determination of the locations of the sixth and twelfth frames within the superframe. Note that the superframe described here is 12 frames in length. Since two bits per superframe are available for signaling for each channel, it is possible to provide four-state signaling for a channel by using the four possible patterns of the two signaling bits. **00, 01, 10,** and **11**. Although most switch-to-switch applications in the telephone network require only two-state signaling, three- and four-state signaling techniques are used in certain special applications.

Advances in digital electronics and in coding theory have made it unnecessary to use the full 8 kbit/s of the framing channel in a DS1 signal to perform the framing task. A new superframe structure, called the **extended superframe (ESF)** format, was introduced during the 1970s to take advantage of the reduced framing bandwidth requirement. An ESF is 24 frames in length and carries signaling bits in the eighth bit of each channel in frames 6, 12, 18, and 24. Sixteen-state signaling is thus possible and is sometimes used although, as with the superframe format, most applications require only two-state signaling.

The 8 kbit/s overhead (framing) capacity of the ESF signal is divided into three channels: 2 kbit/s for framing, 2 kbit/s for a cyclic redundancy check (CRC-6) error detection channel, and 4 kbit/s for a data channel. The highly reliable error checking provided by the CRC-6 pattern and the use of the data channel to transport information on signal performance as received by the distant terminal make ESF much more attractive to service providers than the older superframe format. More discussions on CRC error detection can be found in Chapter 14.

The 2 kbit/s framing channel of the ESF format carries the repetitive pattern **001011**. . . , a pattern that repeats in 24 frames and is much less vulnerable to counterfeiting than the patterns associated with the earlier formats.

For various reasons, including the development of intelligent network-switching nodes, the function of signaling is being transferred out from the channels that carry the messages or data signals to separate signaling networks called **common channel interoffice signaling (CCIS)** systems. The universal deployment of such systems will significantly decrease the importance of robbed-bit signaling, and all eight bits of each message (or sample) will be transmitted in most applications.

The Conference on European Postal and Telegraph Administration (CEPT) has standardized a PCM with 256 time slots per frame. Each frame has $30 \times 8 = 240$ information bits, corresponding to 30 speech channels (with eight bits each). The remaining 16 bits per frame are used for frame synchronization and signaling. Therefore, although the bit rate is 2.048 Mbit/s, corresponding to 32 voice channels, only 30 voice channels are transmitted.

6.4 DIGITAL MULTIPLEXING

Several low-bit-rate signals can be multiplexed, or combined, to form one high-bit-rate signal, to be transmitted over a high-frequency medium. Because the medium is time-shared by various incoming signals, this is a case of TDM (time division multiplexing). The signals from various incoming channels, or tributaries, may be as diverse as a digitized voice signal (PCM), a computer output, telemetry data, and a digital facsimile. The bit rates of various tributaries need not be the same.

To begin with, consider the case of all tributaries with identical bit rates. Multiplexing can be done on a bit-by-bit basis (known as bit or **digit interleaving**) as shown in Fig. 6.22a, or on a word-by-word basis (known as byte or **word interleaving**). Figure 6.22b shows the interleaving of words, formed by four bits. The North American digital hierarchy uses bit interleaving (except at the lowest level), where bits are taken one at a time from the various signals to be multiplexed. Byte interleaving, used in building the DS1 signal and SONET-formatted signals, involves inserting bytes in succession from the channels to be multiplexed.

The T1 carrier, discussed in Sec. 6.3, uses eight-bit word interleaving. When the bit rates of incoming channels are not identical, the high-bit-rate channel is allocated proportionately more slots. Four-channel multiplexing consists of three channels B, C, and D of identical bit rate R and one channel (channel A) with a bit rate of $3R$. (Fig. 6.22c,d). Similar results can be attained by combining words of different lengths. It is evident that the minimum length of the multiplex frame must be a multiple of the lowest common multiple of the incoming channel bit rates, and, hence, this type of scheme is practical only when some fairly simple relationship exists among these rates. The case of completely asynchronous channels is discussed later.

At the receiving terminal, the incoming digit stream must be divided and distributed to the appropriate output channel. For this purpose, the receiving terminal must be able to correctly identify each bit. This requires the receiving system to uniquely synchronize in time with the beginning of each frame, with each slot in a frame, and with each bit within a slot. This is accomplished by adding framing and synchronization bits to the data bits. These bits are part of the so-called **overhead bits.**

6.4.1 Signal Format

Figure 6.23 illustrates a typical format, that of the DM1/2 multiplexer. We have here bit-by-bit interleaving of four channels each at a rate of 1.544 Mbit/s. The main frame (multiframe) consists of four subframes. Each subframe has six overhead bits: for example the subframe 1 (first line in Fig. 6.23) has overhead bits M_0, C_A, F_0, C_A, C_A, and F_1. In between these overhead bits are 48 interleaved data bits from the four channels (12 data bits from each channel). We begin with overhead bit M_0, followed by 48 multiplexed data bits, then add a second overhead bit C_A followed by the next 48 multiplexed bits, and so on. Thus, there are a total of $48 \times 6 \times 4 = 1152$ data bits and $6 \times 4 = 24$ overhead bits making a total 1176 bits/frame. The efficiency is $1152/1176 \simeq 98\%$. The overhead bits with subscript 0 are always **0** and those with subscript 1 are always **1**. Thus, M_0, F_0 are all **0**s and M_1 and F_1 are all **1**s. The F digits are periodic **010101** ... and provide the main framing pattern, which the multiplexer uses to synchronize on the frame. After locking onto this pattern, the demultiplexer searches for the **0111** pattern formed by overhead bits $M_0M_1M_1M_1$. This further identifies the four subframes, each corresponding to a line in Fig. 6.23. It is possible, although unlikely, that signal bits will also have a pattern **101010**. . . . The receiver could lock onto this

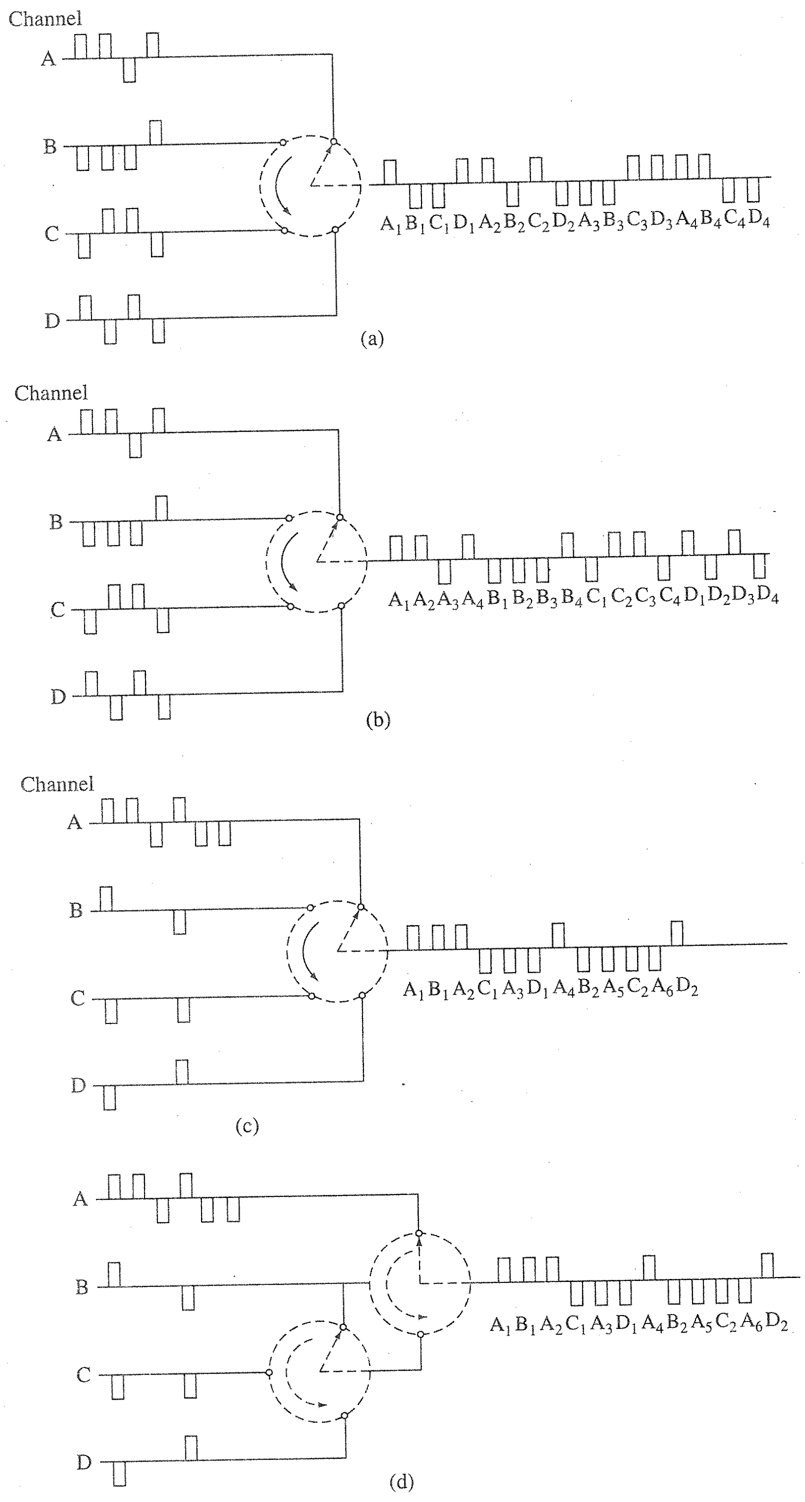

Channel
A
B
C
D
$A_1 B_1 C_1 D_1 A_2 B_2 C_2 D_2 A_3 B_3 C_3 D_3 A_4 B_4 C_4 D_4$
(a)
Channel
A
B
C
D
$A_1 A_2 A_3 A_4 B_1 B_2 B_3 B_4 C_1 C_2 C_3 C_4 D_1 D_2 D_3 D_4$
(b)
Channel
A
B
C
D
$A_1 B_1 A_2 C_1 A_3 D_1 A_4 B_2 A_5 C_2 A_6 D_2$
(c)
A
B
C
D
$A_1 B_1 A_2 C_1 A_3 D_1 A_4 B_2 A_5 C_2 A_6 D_2$
(d)

Figure 6.22 Time division multiplexing of digital signals: (a) digit interleaving; (b) word (or byte) interleaving; (c) interleaving channel having different bit rate; (d) alternate scheme for (c).

Figure 6.23 DM1/2 multiplexer format.

M_0	[48]	C_A	[48]	F_0	[48]	C_A	[48]	C_A	[48]	F_1	[48]
M_1	[48]	C_B	[48]	F_0	[48]	C_B	[48]	C_B	[48]	F_1	[48]
M_1	[48]	C_C	[48]	F_0	[48]	C_C	[48]	C_C	[48]	F_1	[48]
M_1	[48]	C_D	[48]	F_0	[48]	C_D	[48]	C_D	[48]	F_1	[48]

wrong sequence. The presence of $M_0M_1M_1M_1$ provides verification of the genuine $F_0F_1F_0F_1$ sequence. The C bits are used to transmit additional information about bit stuffing, as discussed later.

In the majority of cases, not all incoming channels are active all the time: some transmit data, and some are idle. This means the system is underutilized. We can, therefore, accept more input channels to take advantage of the inactivity, at any given time, of at least one channel. This obviously involves much more complicated switching operations, and also rather careful system planning. In any random traffic situation we cannot guarantee that the number of transmission channels demanded will not exceed the number available; but by taking account of the statistics of the signal sources, it is possible to ensure an acceptably low probability of this occurring. Multiplex structures of this type have been developed for satellite systems and are known as **time division multiple-access (TDMA) systems**.

In TDMA systems employed for telephony, the design parameters are chosen so that any overload condition lasts only a fraction of a second, which leads to acceptable performance for speech communication. For other types of data and telegraphy, transmission delays are unimportant. Hence, in overload condition, the incoming data can be stored and transmitted later.

6.4.2 Asynchronous Channels and Bit Stuffing

In the preceding discussion, we assumed synchronization between all the incoming channels and the multiplexer. This is difficult even when all the channels are nominally at the same rate. For example, consider a 1000 km coaxial cable carrying 2×10^8 pulses per second. Assuming the nominal propagation speed in the cable to be 2×10^8 m/s, it takes 1/200 second of transit time and 1 million pulses will be in transit. If the cable temperature increases by 1°F, the propagation velocity will increase by about 0.01%. This will cause the pulses in transit to arrive sooner, thus producing a temporary increase in the rate of pulses received. Because the extra pulses cannot be accommodated in the multiplexer, they must be temporarily stored at the receiver. If the cable temperature drops, the rate of received pulses will drop, and the multiplexer will have vacant slots with no data. These slots need to be stuffed with dummy digits (**pulse stuffing**).

DS1 signals in the North American network are often generated by crystal oscillators in individual channel banks or other digital terminal equipment. Although the oscillators are quite stable, they will not oscillate at exactly the same frequency, leading to another cause of asynchronicity in the network.

This shows that even in synchronously multiplexed systems, the data are rarely received at a synchronous rate. We always need a storage (known as an **elastic store**) and pulse stuffing (also known as **justification**) to accommodate such an situation. Obviously, this method of an elastic store and pulse stuffing will work even when the channels are asynchronous.

Three variants of the pulse stuffing scheme exist: (1) positive pulse stuffing, (2) negative pulse stuffing, and (3) positive/negative pulse stuffing. In positive pulse stuffing, the multiplexer

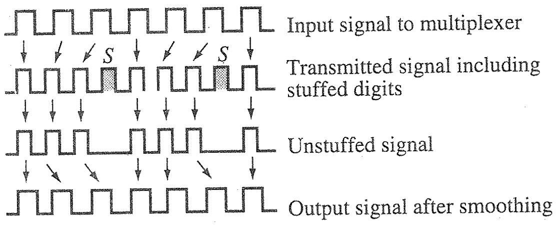

Input signal to multiplexer
S
S
Transmitted signal including stuffed digits
Unstuffed signal
Output signal after smoothing

Figure 6.24 Pulse stuffing.

rate is higher than that required to accommodate all incoming tributaries at their maximum rate. Hence, the time slots in the multiplexed signal will become available at a rate exceeding that of the incoming data so that the tributary data will tend to lag (Fig. 6.24). At some stage, the system will decide that this lag has become great enough to require pulse stuffing. The information about the stuffed-pulse positions is transmitted through overhead bits. From the overhead bits, the receiver knows the stuffed-pulse position and eliminates that pulse.

Negative pulse stuffing is a complement of positive pulse stuffing. The time slots in the multiplexed signal now appear at a slightly slower rate than those of the tributaries, and thus the multiplexed signal cannot accommadate all the tributary pulses. Information about any left-out pulse and its position is transmitted through overhead bits. The positive/negative pulse stuffing is a combination of the first two schemes. The nominal rate of the multiplexer is equal to the nominal rate required to accommodate all incoming channels. Hence, we may need positive pulse stuffing at some times and negative stuffing at others. All this information is sent through overhead bits.

The C digits in Fig. 6.23 are used to transmit stuffing information. Only one stuffed bit per input channel is allowed per frame. This is sufficient to accommodate expected variations in the input signal rate. The bits C_A convey information about stuffing in channel A, bits C_B convey information about stuffing in channel B, and so on. The insertion of any stuffed pulse in any one subframe is denoted by setting all the three Cs in that line to **1**. No stuffing is indicated by using **0**s for all the three Cs. If a bit has been stuffed, the stuffed bit is the first information bit associated with the immediate channel following the F_1 bit, that is, the first such bit in the last 48-bit sequence in that subframe. For the first subframe, the stuffed bit will immediately follow the F_1 bit. For the second subframe, the stuffed bit will be the second bit following the F_1 bit, and so on.

6.4.3 Plesiochronous (almost Synchronous) Digital Hierarchy

We now present the digital hierarchy developed by the Bell System and currently included in the ANSI standards for telecommunications (Fig. 6.25). The North American hierarchy is implemented in North America and Japan.

Two major classes of multiplexers are used in practice. The first category is used for combining low-data-rate channels. It multiplexes channels of rates of up to 9600 bit/s into a signal of data rate of up to 64 kbit/s. The multiplexed signal, called **"digital signal level 0" (DS0)** in the North American hierarchy, is eventually transmitted over a voice-grade channel. The second class of multiplexers is at a much higher bit rate.

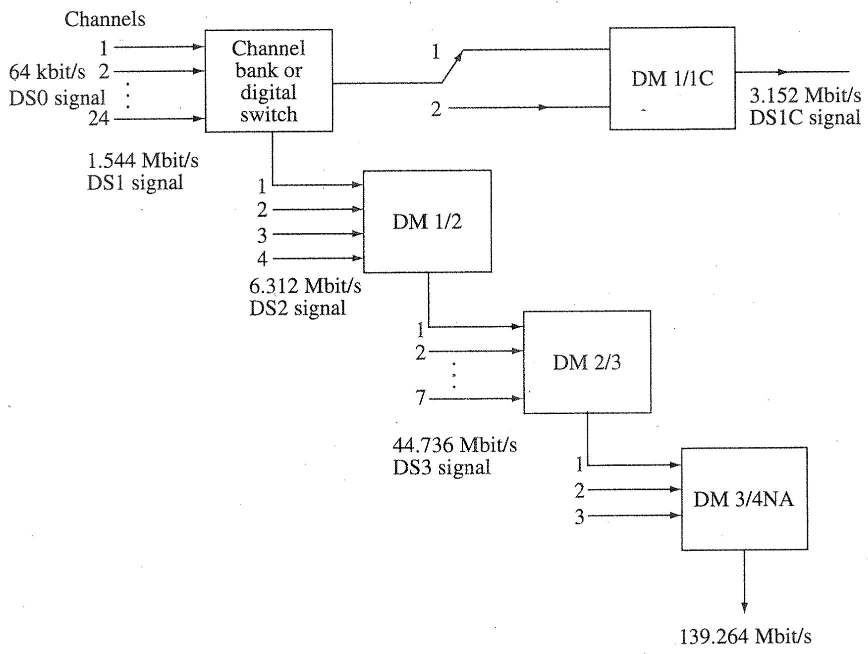


Figure 6.25 North American digital hierarchy (AT&T system).

There are four orders, or levels, of multiplexing. The first level is the **T1 multiplexer** or **channel bank**, consisting of 24 channels of 64 kbit/s each. The output of this multiplexer is a **DS1 (digital level 1)** signal at a rate of 1.544 Mbit/s. Four DS1 signals are multiplexed by a DM1/2 multiplexer to yield a DS2 signal at a rate 6.312 Mbit/s. Seven DS2 signals are multiplexed by a DM2/3 multiplexer to yield a DS3 signal at a rate of 44.736 Mbit/s. Finally, three DS3 signals are multiplexed by a DM3/4NA multiplexer to yield a DS4NA signal at a rate 139.264 Mbit/s. There is also a lower rate multiplexing hierarchy, known as the **digital data system (DDS)**, which provides standards for multiplexing digital signals with rates as low as 2.4 kbit/s into a DS0 signal for transmission through the network.

The inputs to a T1 multiplexer need not be restricted only to digitized voice channels alone. Any digital signal of 64 kbit/s of appropriate format can be transmitted. The case of the higher levels is similar. For example, all the incoming channels of the DM1/2 multiplexer need not be DS1 signals obtained by multiplexing 24 channels of 64 kbit/s each. Some of them may be 1.544 Mbit/s digital signals of appropriate format, and so on.

In Europe and many other parts of the world, another hierarchy, recommended by the ITU as an standard, has been adopted. This hierarchy, based on multiplexing 30 telephone channels of 64 kbit/s (E-0 channels) into an E-1 carrier at 2.048 Mbit/s (30 channels) is shown in Fig. 6.26. Starting from the base level of E-1, four lower level lines form one higher level line progressively, generating an E-2 line with data throughput of 8.448 Mbit/s, an E-3 line with data throughput of 34.368 Mbit/s, an E-4 line line with data throughput of 139.264 Mbit/s, and an E-5 line with data throughput of 565.148 Mbit/s. Because different networks must be able to interface with one another across the three different systems (North American, Japanese, and other) in the world, Fig. 6.26 demonstrates the relative relationship and the points of their common interface.

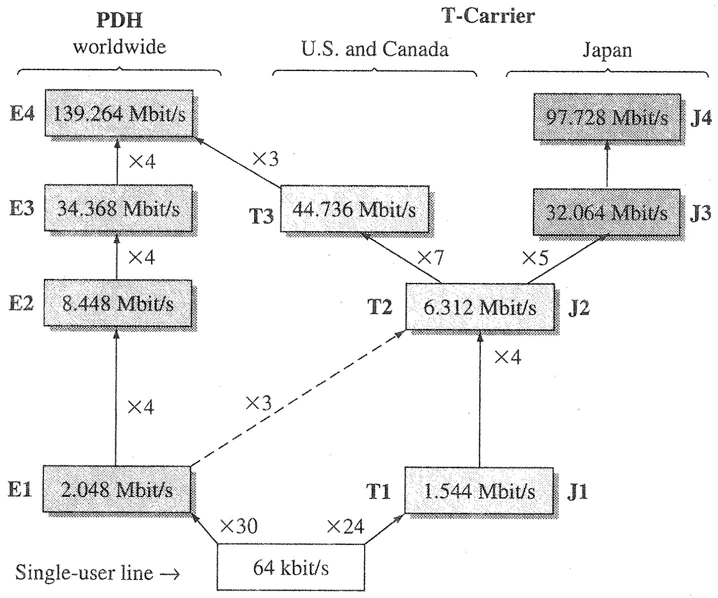


Figure 6.26 Plesiochronous digital hierarchy (PDH) according to ITU-T Recommendation G.704.

6.5 DIFFERENTIAL PULSE CODE MODULATION (DPCM)

PCM is not a very efficient system because it generates so many bits and requires so much bandwidth to transmit. Many different ideas have been proposed to improve the encoding efficiency of A/D conversion. In general, these ideas exploit the characteristics of the source signals. DPCM is one such scheme.

In analog messages we can make a good guess about a sample value from knowledge of past sample values. In other words, the sample values are not independent, and generally there is a great deal of redundancy in the Nyquist samples. Proper exploitation of this redundancy leads to encoding a signal with fewer bits. Consider a simple scheme; instead of transmitting the sample values, we transmit the difference between the successive sample values. Thus, if $m[k]$ is the kth sample, instead of transmitting $m[k]$, we transmit the difference $d[k] = m[k]-m[k-1]$. At the receiver, knowing $d[k]$ and several previous sample value $m[k-1]$, we can reconstruct $m[k]$. Thus, from knowledge of the difference $d[k]$, we can reconstruct $m[k]$ iteratively at the receiver. Now, the difference between successive samples is generally much smaller than the sample values. Thus, the peak amplitude m_p of the transmitted values is reduced considerably. Because the quantization interval $\Delta v = m_p/L$, for a given L (or n), this reduces the quantization interval Δv, thus reducing the quantization noise, which is given by $\Delta v^2/12$. This means that for a given n (or transmission bandwidth), we can increase the SNR, or for a given SNR, we can reduce n (or transmission bandwidth).

We can improve upon this scheme by estimating (predicting) the value of the kth sample $m[k]$ from a knowledge of several previous sample values. If this estimate is $\hat{m}[k]$, then we transmit the difference (prediction error) $d[k] = m[k]-\hat{m}[k]$. At the receiver also, we determine the estimate $\hat{m}[k]$ from the previous sample values, and then generate $m[k]$ by adding the received $d[k]$ to the estimate $\hat{m}[k]$. Thus, we reconstruct the samples at the receiver iteratively. If our prediction is worth its salt, the predicted (estimated) value $\hat{m}[k]$ will be close to $m[k]$, and their difference (prediction error) $d[k]$ will be even smaller than the difference between the successive samples. Consequently, this scheme, known as the **differential PCM (DPCM)**, is superior to the naive prediction described in the preceding paragraph, which is a special case of DPCM, where the estimate of a sample value is taken as the previous sample value, that is, $\hat{m}[k] = m[k-1]$.

Spirits of Taylor, Maclaurin, and Wiener

Before describing DPCM, we shall briefly discuss the approach to signal prediction (estimation). To the uninitiated, future prediction seems like mysterious stuff, fit only for psychics, wizards, mediums, and the like, who can summon help from the spirit world. Electrical engineers appear to be hopelessly outclassed in this pursuit. Not quite so! We can also summon the spirits of Taylor, Maclaurin, Wiener, and the like to help us. What is more, unlike Shakespeare's spirits, our spirits come when called.* Consider, for example, a signal $m(t)$, which has derivatives of all orders at t. Using the Taylor series for this signal, we can express $m(t+T_s)$ as

$$m(t+T_s) = m(t) + T_s\dot{m}(t) + \frac{T_s{}^2}{2!}\ddot{m}(t) + \frac{T_s{}^3}{3!}\dddot{m}(t) + \cdots \tag{6.42a}$$

$$\approx m(t) + T_s\dot{m}(t) \qquad \text{for small } T_s \tag{6.42b}$$

Equation (6.42a) shows that from a knowledge of the signal and its derivatives at instant t, we can predict a future signal value at $t + T_s$. In fact, even if we know just the first derivative, we can still predict this value approximately, as shown in Eq. (6.42b). Let us denote the kth sample of $m(t)$ by $m[k]$, that is, $m(kT_s) = m[k]$, and $m(kT_s \pm T_s) = m[k \pm 1]$, and so on. Setting $t = kT_s$ in Eq. (6.42b), and recognizing that $\dot{m}(kT_s) \approx [m(kT_s) - m(kT_s - T_s)]/T_s$, we obtain

$$\begin{aligned} m[k+1] &\approx m[k] + T_s\left[\frac{m[k]-m[k-1]}{T_s}\right] \\ &= 2m[k] - m[k-1] \end{aligned}$$

This shows that we can find a crude prediction of the $(k+1)$th sample from the two previous samples. The approximation in Eq. (6.42b) improves as we add more terms in the series on the right-hand side. To determine the higher order derivatives in the series, we require more samples in the past. The larger the number of past samples we use, the better will be the prediction. Thus, in general, we can express the prediction formula as

$$m[k] \approx a_1 m[k-1] + a_2 m[k-2] + \cdots + a_N m[k-N] \tag{6.43}$$

The right-hand side is $\hat{m}[k]$, the predicted value of $m[k]$. Thus,

$$\hat{m}[k] = a_1 m[k-1] + a_2 m[k-2] + \cdots + a_N m[k-N] \tag{6.44}$$

This is the equation of an Nth-order predictor. Larger N would result in better prediction in general. The output of this filter (predictor) is $\hat{m}[k]$, the predicted value of $m[k]$. The input consists of the previous samples $m[k-1]$, $m[k-2]$, ..., $m[k-N]$, although it is customary to say that the input is $m[k]$ and the output is $\hat{m}[k]$. Observe that this equation reduces to

* From Shakespeare, Henry IV, Part 1, Act III, Scene 1:
Glendower: *I can call the spirits from vasty deep.*
Hotspur: *Why, so can I, or so can any man;*
But will they come when you do call for them?

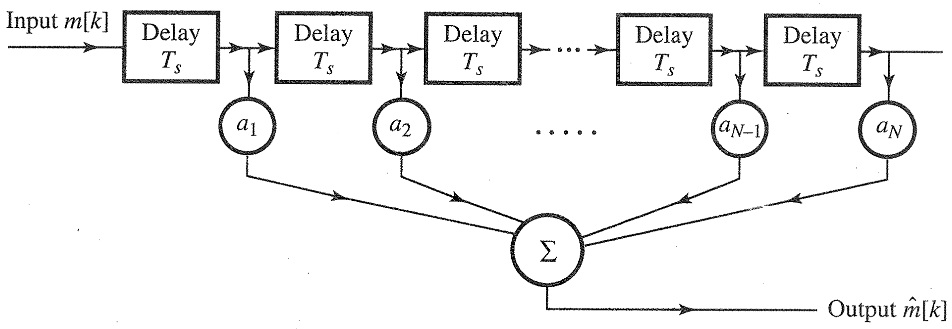


Figure 6.27 Transversal filter (tapped delay line) used as a linear predictor.

$\hat{m}[k] = m[k-1]$ in the case of the first-order prediction. It follows from Eq. (6.42b), where we retain only the first term on the right-hand side. This means that $a_1 = 1$, and the first-order predictor is a simple time delay.

We have outlined here a very simple procedure for predictor design. In a more sophisticated approach, discussed in Sec. 8.5, where we use the minimum mean squared error criterion for best prediction, the **prediction coefficients** a_j in Eq. (6.44) are determined from the statistical correlation between various samples. The predictor described in Eq. (6.44) is called a *linear predictor*. It is basically a transversal filter (a tapped delay line), where the tap gains are set equal to the prediction coefficients, as shown in Fig. 6.27.

Analysis of DPCM

As mentioned earlier, in DPCM we transmit not the present sample $m[k]$, but $d[k]$ (the difference between $m[k]$ and its predicted value $\hat{m}[k]$). At the receiver, we generate $\hat{m}[k]$ from the past sample values to which the received $d[k]$ is added to generate $m[k]$. There is, however, one difficulty associated with this scheme. At the receiver, instead of the past samples $m[k-1]$, $m[k-2]$, ..., as well as $d[k]$, we have their quantized versions $m_q[k-1]$, $m_q[k-2]$, ... Hence, we cannot determine $\hat{m}[k]$. We can determine only $\hat{m}_q[k]$, the estimate of the quantized sample $m_q[k]$, in terms of the quantized samples $m_q[k-1]$, $m_q[k-2]$, ... This will increase the error in reconstruction. In such a case, a better strategy is to determine $\hat{m}_q[k]$, the estimate of $m_q[k]$ (instead of $m[k]$), at the transmitter also from the quantized samples $m_q[k-1]$, $m_q[k-2]$, ... The difference $d[k] = m[k] - \hat{m}_q[k]$ is now transmitted via PCM. At the receiver, we can generate $\hat{m}_q[k]$, and from the received $d[k]$, we can reconstruct $m_q[k]$.

Figure 6.28a shows a DPCM transmitter. We shall soon show that the predictor input is $m_q[k]$. Naturally, its output is $\hat{m}_q[k]$, the predicted value of $m_q[k]$. The difference

$$d[k] = m[k] - \hat{m}_q[k] \tag{6.45}$$

is quantized to yield

$$d_q[k] = d[k] + q[k] \tag{6.46}$$

where $q[k]$ is the quantization error. The predictor output $\hat{m}_q[k]$ is fed back to its input so that the predictor input $m_q[k]$ is

$$\begin{aligned} m_q[k] &= \hat{m}_q[k] + d_q[k] \\ &= m[k] - d[k] + d_q[k] \\ &= m[k] + q[k] \end{aligned} \tag{6.47}$$

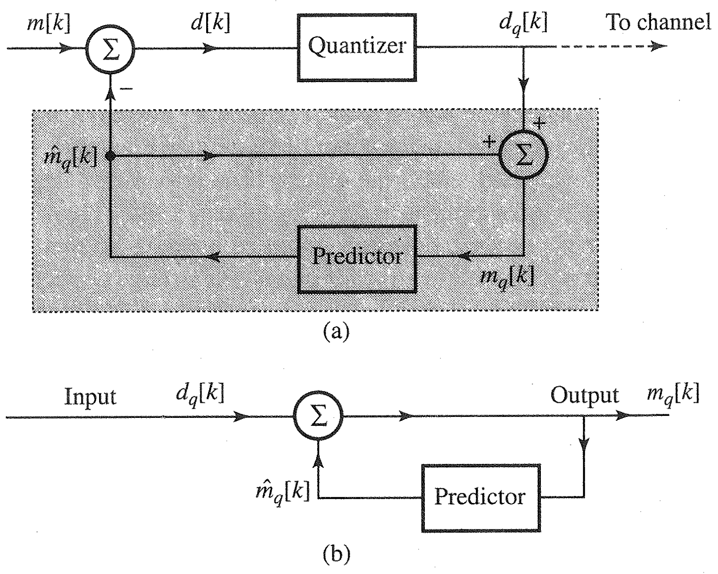


Figure 6.28 DPCM system: (a) transmitter; (b) receiver.

This shows that $m_q[k]$ is a quantized version of $m[k]$. The predictor input is indeed $m_q[k]$, as assumed. The quantized signal $d_q[k]$ is now transmitted over the channel. The receiver shown in Fig. 6.28b is identical to the shaded portion of the transmitter. The inputs in both cases are also the same, namely, $d_q[k]$. Therefore, the predictor output must be $\hat{m}_q[k]$ (the same as the predictor output at the transmitter). Hence, the receiver output (which is the predictor input) is also the same, viz., $m_q[k] = m[k] + q[k]$, as found in Eq. (6.47). This shows that we are able to receive the desired signal $m[k]$ plus the quantization noise $q[k]$. This is the quantization noise associated with the difference signal $d[k]$, which is generally much smaller than $m[k]$. The received samples $m_q[k]$ are decoded and passed through a low-pass filter for D/A conversion.

SNR Improvement

To determine the improvement in DPCM over PCM, let m_p and d_p be the peak amplitudes of $m(t)$ and $d(t)$, respectively. If we use the same value of L in both cases, the quantization step Δv in DPCM is reduced by the factor d_p/m_p. Because the quantization noise power is $(\Delta v)^2/12$, the quantization noise in DPCM is reduced by the factor $(m_p/d_p)^2$, and the SNR is increased by the same factor. Moreover, the signal power is proportional to its peak value squared (assuming other statistical properties invariant). Therefore, G_p (SNR improvement due to prediction) is at least

$$G_p = \frac{P_m}{P_d}$$

where P_m and P_d are the powers of $m(t)$ and $d(t)$, respectively. In terms of decibel units, this means that the SNR increases by $10 \log_{10}(P_m/P_d)$ dB. Therefore, Eq. (6.41) applies to DPCM also with a value of α that is higher by $10 \log_{10}(P_m/P_d)$ dB. In Example 8.24, a second-order predictor processor for speech signals is analyzed. For this case, the SNR improvement is found to be 5.6 dB. In practice, the SNR improvement may be as high as 25 dB in such cases as short-term voiced speech spectra and in the spectra of low-activity images.[12] Alternately, for the same SNR, the bit rate for DPCM could be lower than that for PCM by 3 to 4 bits per sample. Thus, telephone systems using DPCM can often operate at 32 or even 24 kbit/s.

6.6 ADAPTIVE DIFFERENTIAL PCM (ADPCM)

Adaptive DPCM (ADPCM) can further improve the efficiency of DPCM encoding by incorporating an adaptive quantizer at the encoder. Figure 6.29 illustrates the basic configuration of ADPCM. For practical reasons, the number of quantization level L is fixed. When a fixed quantization step Δv is applied, either the quantization error is too large because Δv is too big or the quantizer cannot cover the necessary signal range when Δv is too small. Therefore, it would be better for the quantization step Δv to be adaptive so that Δv is large or small depending on whether the prediction error for quantizing is large or small.

It is important to note that the quantized prediction error $d_q[k]$ can be a good indicator of the prediction error size. For example, when the quantized prediction error samples vary close to the largest positive value (or the largest negative value), it indicates that the prediction error is large and Δv needs to grow. Conversely, if the quantized samples oscillate near zero, then the prediction error is small and Δv needs to decrease. It is important that both the modulator and the receiver have access to the same quantized samples. Hence, the adaptive quantizer and the receiver reconstruction can apply the same algorithm to adjust the Δv identically.

Compared with DPCM, ADPCM can further compress the number of bits needed for a signal waveform. For example, it is very common in practice for an 8-bit PCM sequence to be encoded into a 4-bit ADPCM sequence at the same sampling rate. This easily represents a 2:1 bandwidth or storage reduction with virtually no loss.

ADPCM encoder has many practical applications. The ITU-T standard G.726 specifies an ADPCM speech coder and decoder (called **codec**) for speech signal samples at 8 kHz.[7] The G.726 ADPCM predictor uses an eighth-order predictor. For different quality levels, G.726 specifies four different ADPCM rates at 16, 24, 32, and 40 kbit/s. They correspond to four different bit sizes for each speech sample at 2 bits, 3 bits, 4 bits, and 5 bits, respectively, or equivalently, quantization levels of 4, 8, 16, and 32, respectively.

The most common ADPCM speech encoders use 32 kbit/s. In practice, there are multiple variations of ADPCM speech codec. In addition to the ITU-T G.726 specification,[7] these include the OKI ADPCM codec, the Microsoft ADPCM codec supported by WAVE players, and the Interactive Multimedia Association (IMA) ADPCM, also known as the DVI ADPCM. The 32 kbit/s ITU-T G.726 ADPCM speech codec is widely used in the DECT (digital enhanced cordless telecommunications) system, which itself is widely used for residential and business cordless phone communications. Designed for short-range use as an access mechanism to the main networks, DECT offers cordless voice, fax, data, and multimedia communications. DECT is now in use in over 100 countries worldwide. Another major user of the 32 kbit/s ADPCM codec is the Personal Handy-phone System (or PHS), also marketed as the Personal Access System (PAS) and known as Xiaolingtong in China.

PHS is a mobile network system similar to a cellular network, operating in the 1880 to 1930 MHz frequency band, used mainly in Japan, China, Taiwan, and elsewhere in Asia. Originally developed by the NTT Laboratory in Japan in 1989, PHS is much simpler to implement and

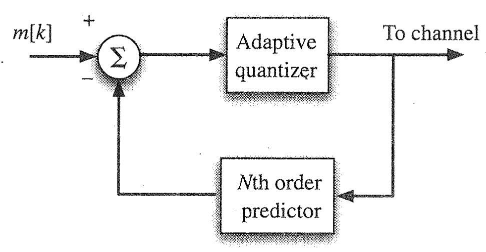


Figure 6.29 ADPCM encoder uses an adaptive quantizer controlled only by the encoder output bits.

deploy. Unlike cellular networks, PHS phones and base stations are low-power, short-range facilities. The service is often pejoratively called the "poor man's cellular" because of its limited range and poor roaming ability. PHS first saw limited deployment (NTT-Personal, DDI-Pocket, and ASTEL) in Japan in 1995 but has since nearly disappeared. Surprisingly, PHS has seen a resurgence in markets like China, Taiwan, Vietnam, Bangladesh, Nigeria, Mali, Tanzania, and Honduras, where its low cost of deployment and hardware costs offset the system's disadvantages. In China alone, there was an explosive expansion of subscribers, reaching nearly 80 million in 2006.

6.7 DELTA MODULATION

Sample correlation used in DPCM is further exploited in **delta modulation (DM)** by oversampling (typically four times the Nyquist rate) the baseband signal. This increases the correlation between adjacent samples, which results in a small prediction error that can be encoded using only one bit ($L = 2$). Thus, DM is basically a 1-bit DPCM, that is, a DPCM that uses only two levels ($L = 2$) for quantization of $m[k] - \hat{m}_q[k]$. In comparison to PCM (and DPCM), it is a very simple and inexpensive method of A/D conversion. A 1-bit codeword in DM makes word framing unnecessary at the transmitter and the receiver. This strategy allows us to use fewer bits per sample for encoding a baseband signal.

In DM, we use a first-order predictor, which, as seen earlier, is just a time delay of T_s (the sampling interval). Thus, the DM transmitter (modulator) and receiver (demodulator) are identical to those of the DPCM in Fig. 6.28, with a time delay for the predictor, as shown in Fig. 6.30, from which we can write

$$m_q[k] = m_q[k-1] + d_q[k] \tag{6.48}$$

Hence,

$$m_q[k-1] = m_q[k-2] + d_q[k-1]$$

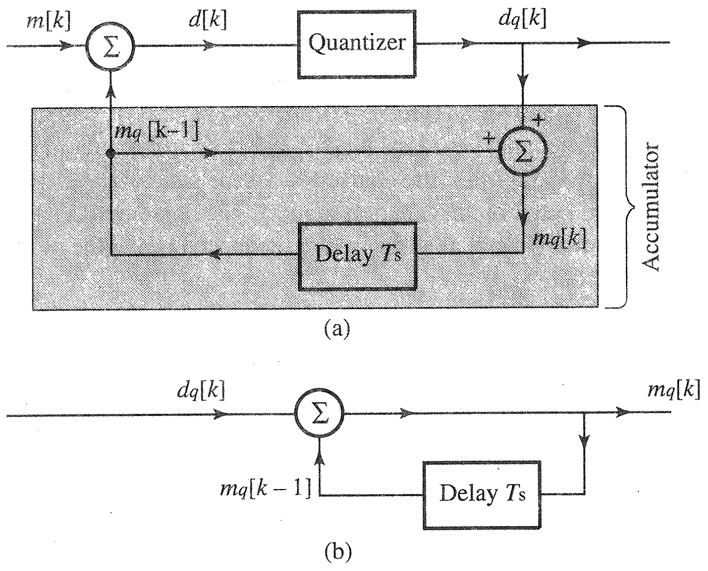


Figure 6.30 Delta modulation is a special case of DPCM.

Substituting this equation into Eq. (6.48) yields

$$m_q[k] = m_q[k-2] + d_q[k] + d_q[k-1]$$

Proceeding iteratively in this manner, and assuming zero initial condition, that is, $m_q[0] = 0$, we write

$$m_q[k] = \sum_{m=0}^{k} d_q[m] \tag{6.49}$$

This shows that the receiver (demodulator) is just an accumulator (adder). If the output $d_q[k]$ is represented by impulses, then the accumulator (receiver) may be realized by an integrator because its output is the sum of the strengths of the input impulses (sum of the areas under the impulses). We may also replace with an integrator the feedback portion of the modulator (which is identical to the demodulator). The demodulator output is $m_q[k]$, which when passed through a low-pass filter yields the desired signal reconstructed from the quantized samples.

Figure 6.31 shows a practical implementation of the delta modulator and demodulator. As discussed earlier, the first-order predictor is replaced by a low-cost integrator circuit (such as an *RC* integrator). The modulator (Fig. 6.31a) consists of a comparator and a sampler in the direct path and an integrator-amplifier in the feedback path. Let us see how this delta modulator works.

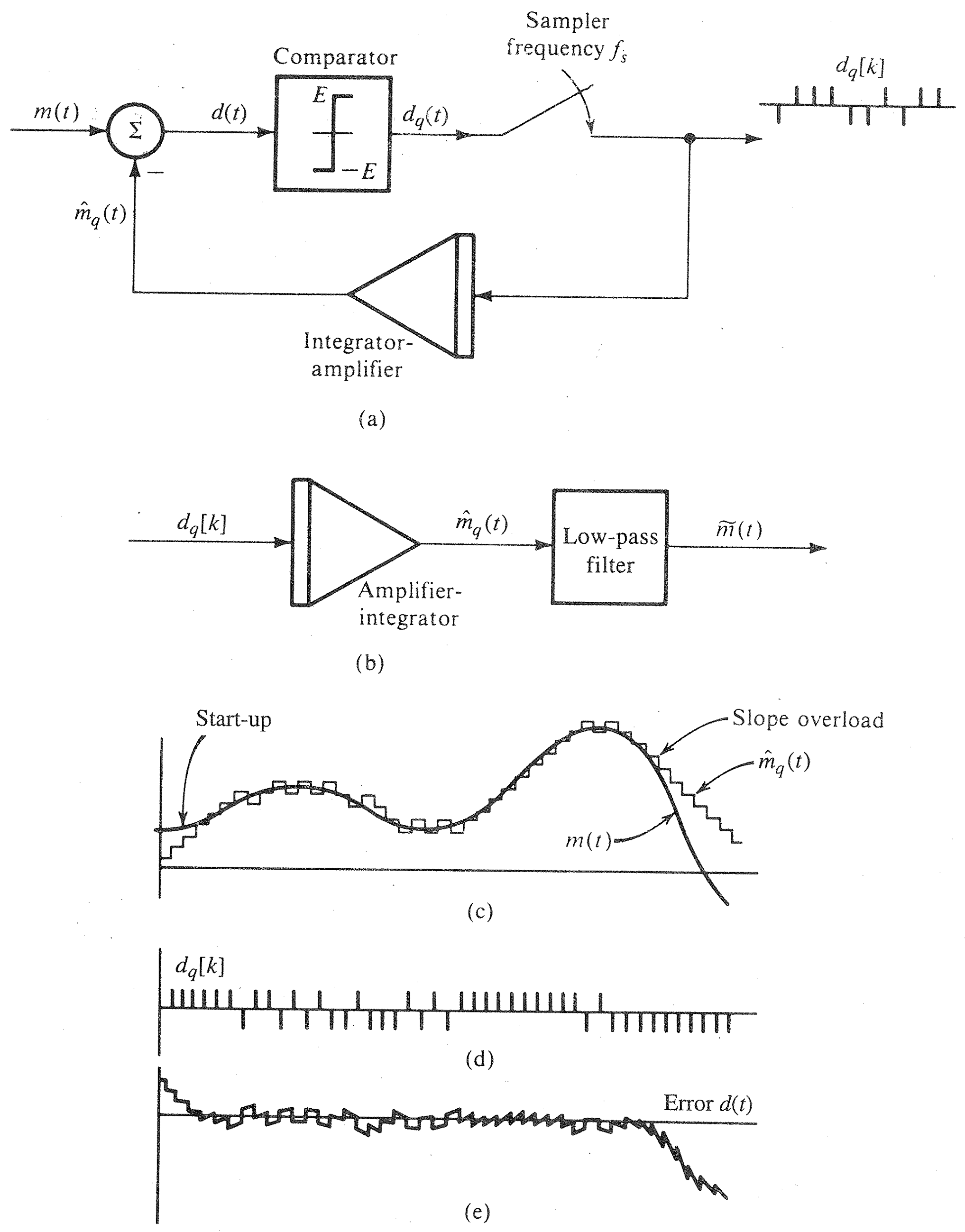
The analog signal $m(t)$ is compared with the feedback signal (which serves as a predicted signal) $\hat{m}_q(t)$. The error signal $d(t) = m(t) - \hat{m}_q(t)$ is applied to a comparator. If $d(t)$ is positive, the comparator output is a constant signal of amplitude E, and if $d(t)$ is negative, the comparator output is $-E$. Thus, the difference is a binary signal ($L = 2$) that is needed to generate a 1-bit DPCM. The comparator output is sampled by a sampler at a rate of f_s samples per second, where f_s is typically much higher than the Nyquist rate. The sampler thus produces a train of narrow pulses $d_q[k]$ (to simulate impulses) with a positive pulse when $m(t) > \hat{m}_q(t)$ and a negative pulse when $m(t) < \hat{m}_q(t)$. Note that each sample is coded by a single binary pulse (1-bit DPCM), as required. The pulse train $d_q[k]$ is the delta-modulated pulse train (Fig. 6.31d). The modulated signal $d_q[k]$ is amplified and integrated in the feedback path to generate $\hat{m}_q(t)$ (Fig. 6.31c), which tries to follow $m(t)$.

To understand how this works, we note that each pulse in $d_q[k]$ at the input of the integrator gives rise to a step function (positive or negative, depending on the pulse polarity) in $\hat{m}_q(t)$. If, for example, $m(t) > \hat{m}_q(t)$, a positive pulse is generated in $d_q[k]$, which gives rise to a positive step in $\hat{m}_q(t)$, trying to equalize $\hat{m}_q(t)$ to $m(t)$ in small steps at every sampling instant, as shown in Fig. 6.31c. It can be seen that $\hat{m}_q(t)$ is a kind of staircase approximation of $m(t)$. When $\hat{m}_q(t)$ is passed through a low-pass filter, the coarseness of the staircase in $\hat{m}_q(t)$ is eliminated, and we get a smoother and better approximation to $m(t)$. The demodulator at the receiver consists of an amplifier-integrator (identical to that in the feedback path of the modulator) followed by a low-pass filter (Fig. 6.31b).

DM Transmits the Derivative of $m(t)$

In PCM, the analog signal samples are quantized in L levels, and this information is transmitted by n pulses per sample ($n = \log_2 L$). A little reflection shows that in DM, the modulated signal carries information not about the signal samples but about the difference between successive samples. If the difference is positive or negative, a positive or a negative pulse (respectively) is generated in the modulated signal $d_q[k]$. Basically, therefore, DM carries the information about the derivative of $m(t)$, hence, the name "delta modulation." This can also be seen from

Figure 6.31 Delta modulation: (a) and (b) delta demodulators; (c) message signal versus integrator output signal (d) delta-modulated pulse trains; (e) modulation errors.



the fact that integration of the delta-modulated signal yields $\hat{m}_q(t)$, which is an approximation of $m(t)$.

In PCM, the information of each quantized sample is transmitted by an n-bit code word, whereas in DM the information of the difference between successive samples is transmitted by a 1-bit code word.

Threshold of Coding and Overloading

Threshold and overloading effects can be clearly seen in Fig. 6.31c. Variations in $m(t)$ smaller than the step value (threshold of coding) are lost in DM. Moreover, if $m(t)$ changes too fast, that is, if $\dot{m}(t)$ is too high, $\hat{m}_q(t)$ cannot follow $m(t)$, and overloading occurs. This is the so-called **slope overload**, which gives rise to the slope overload noise. This noise is one of the basic limiting factors in the performance of DM. We should expect slope overload rather than amplitude overload in DM, because DM basically carries the information about $\dot{m}(t)$. The granular nature of the output signal gives rise to the granular noise similar to the quantization noise. The slope overload noise can be reduced by increasing E (the step size). This unfortunately increases the granular noise. There is an optimum value of E, which yields the best compromise giving the minimum overall noise. This optimum value of E depends on the sampling frequency f_s and the nature of the signal.[12]

The slope overload occurs when $\hat{m}_q(t)$ cannot follow $m(t)$. During the sampling interval T_s, $\hat{m}_q(t)$ is capable of changing by E, where E is the height of the step. Hence, the maximum slope that $\hat{m}_q(t)$ can follow is E/T_s, or Ef_s, where f_s is the sampling frequency. Hence, no overload occurs if

$$|\dot{m}(t)| < Ef_s$$

Consider the case of tone modulation (meaning a sinusoidal message):

$$m(t) = A \cos \omega t$$

The condition for no overload is

$$|\dot{m}(t)|_{\max} = \omega A < Ef_s \tag{6.50}$$

Hence, the maximum amplitude $A_{\max}$ of this signal that can be tolerated without overload is given by

$$A_{\max} = \frac{Ef_s}{\omega} \tag{6.51}$$

The overload amplitude of the modulating signal is inversely proportional to the frequency ω. For higher modulating frequencies, the overload occurs for smaller amplitudes. For voice signals, which contain all frequency components up to (say) 4 kHz, calculating $A_{\max}$ by using $\omega = 2\pi \times 4000$ in Eq. (6.51) will give an overly conservative value. It has been shown by de Jager[13] that $A_{\max}$ for voice signals can be calculated by using $\omega_r \simeq 2\pi \times 800$ in Eq. (6.51),

$$[A_{\max}]_{\text{voice}} \simeq \frac{Ef_s}{\omega_r} \tag{6.52}$$

Thus, the maximum voice signal amplitude $A_{\max}$ that can be used without causing slope overload in DM is the same as the maximum amplitude of a sinusoidal signal of reference frequency f_r ($f_r \simeq 800$ Hz) that can be used without causing slope overload in the same system.

Fortunately, the voice spectrum (as well as the television video signal) also decays with frequency and closely follows the overload characteristics (curve c, Fig. 6.32). For this reason, DM is well suited for voice (and television) signals. Actually, the voice signal spectrum (curve b) decreases as $1/\omega$ up to 2000 Hz, and beyond this frequency, it decreases as $1/\omega^2$. If we had used a double integration in the feedback circuit instead of a single integration, $A_{\max}$ in Eq. (6.51) would be proportional to $1/\omega^2$. Hence, a better match between the voice spectrum and the overload characteristics is achieved by using a single integration up to 2000 Hz and a double integration beyond 2000 Hz. Such a circuit (the double integration) responds fast

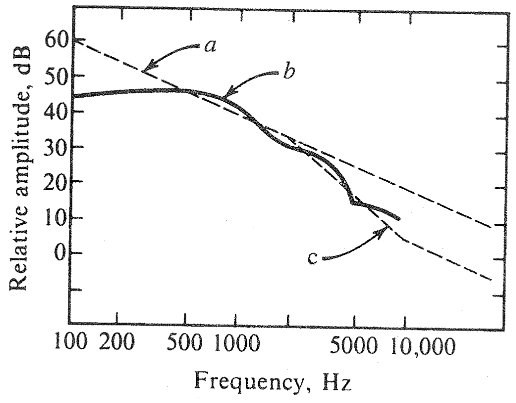


Figure 6.32 Voice signal spectrum.

but has a tendency to instability, which can be reduced by using some low-order prediction along with double integration. A double integrator can be built by placing in cascade two low-pass *RC* integrators with time constants $R_1C_1 = 1/200\pi$ and $R_2C_2 = 1/4000\pi$, respectively. This results in single integration from 100 to 2000 Hz and double integration beyond 2000 Hz.

Sigma-Delta Modulation

While discussing the threshold of coding and overloading, we illustrated that the essence of the conventional DM is to encode and transmit the derivative of the analog message signal. Hence, the receiver of DM requires an integrator as shown in Fig. 6.31 and also, equivalently, in Fig. 6.33a. Since signal transmission inevitably is subjected to channel noise, such noise will be integrated and will accumulate at the receiver output, which is a highly undesirable phenomenon that is a major drawback of DM.

To overcome this critical drawback of DM, a small modification can be made. First, we can view the overall DM system consisting of the transmitter and the receiver as approximately distortionless and linear. Thus, one of its serial components, the receiver integrator $1/s$, may be moved to the front of the transmitter (encoder) without affecting the overall modulator and demodulator response, as shown in Fig. 6.33b. Finally, the two integrators can be merged into a single one after the subtractor, as shown in Fig. 6.33c. This modified system is known as the sigma-delta modulation (Σ-ΔM).

As we found in the study of preemphasis and deemphasis filters in FM, because channel noise and the message signal do not follow the same route, the order of serial components in the overall modulation-demodulation system can have different effects on the SNR. The seemingly minor move of the integrator $1/s$ in fact has several major advantages:

- The channel noise no longer accumulates at the demodulator.
- The important low-frequency content of the message $m(t)$ is preemphasized by the integrator $1/j\omega$. This helps many practical signals (such as speech) whose low-frequency components are more important.
- The integrator effectively smooths the signal for encoding (Fig. 6.33b). Hence, overloading becomes less likely.
- The low-pass nature of the integrator increases the correlation between successive samples, leading to smaller encoding error.
- The demodulator is simplified.

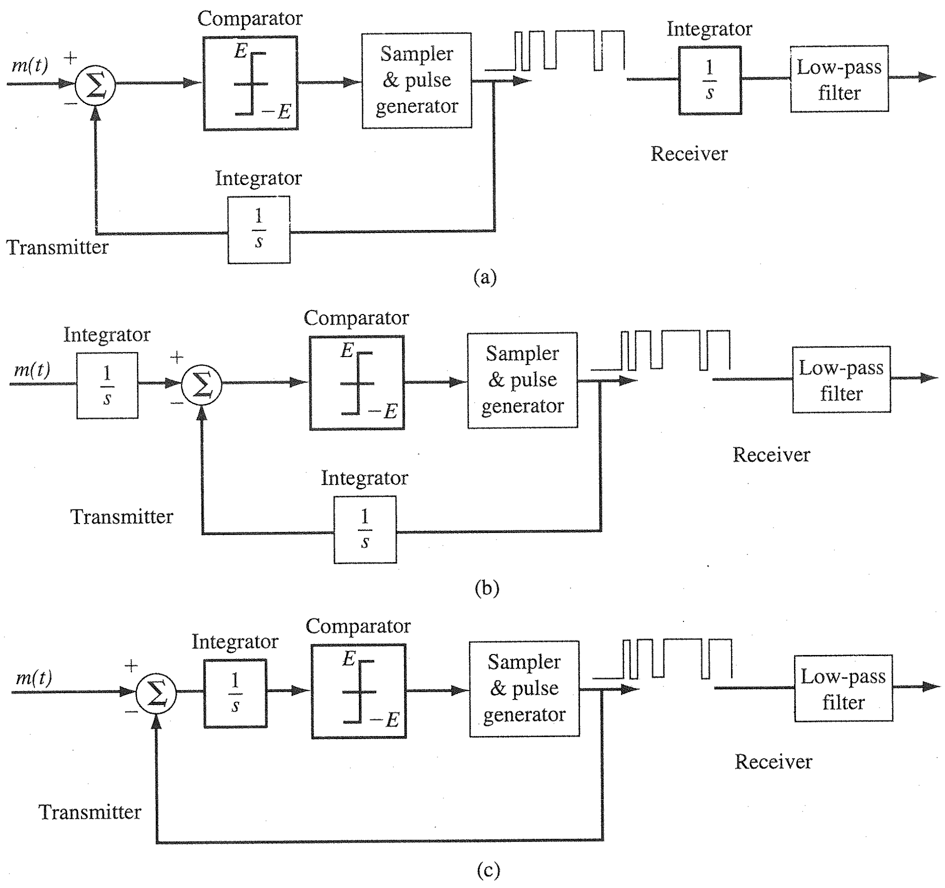


Figure 6.33 (a) Conventional delta modulator. (b) Σ-Δ modulator. (c) Simpler Σ-Δ modulator.

Adaptive Delta Modulation (ADM)

The DM discussed so far suffers from one serious disadvantage. The dynamic range of amplitudes is too small because of the threshold and overload effects discussed earlier. To address this problem, some type of signal compression is necessary. In DM, a suitable method appears to be the adaptation of the step value E according to the level of the input signal derivative. For example, in Fig. 6.31, when the signal $m(t)$ is falling rapidly, slope overload occurs. If we can increase the step size during this period, the overload could be avoided. On the other hand, if the slope of $m(t)$ is small, a reduction of step size will reduce the threshold level as well as the granular noise. The slope overload causes $d_q[k]$ to have several pulses of the same polarity in succession. This calls for increased step size. Similarly, pulses in $d_q[k]$ alternating continuously in polarity indicates small-amplitude variations, requiring a reduction in step size. In ADM we detect such pulse patterns and automatically adjust the step size.[14] This results in a much larger dynamic range for DM.

6.8 VOCODERS AND VIDEO COMPRESSION

PCM, DPCM, ADPCM, DM, and Σ-ΔM are all examples of what are known as waveform source encoders. Basically, waveform encoders do not take into consideration how the signals for digitization are generated. Hence, the amount of compression achievable by waveform encoders is highly limited by the degree of correlation between successive signal samples.

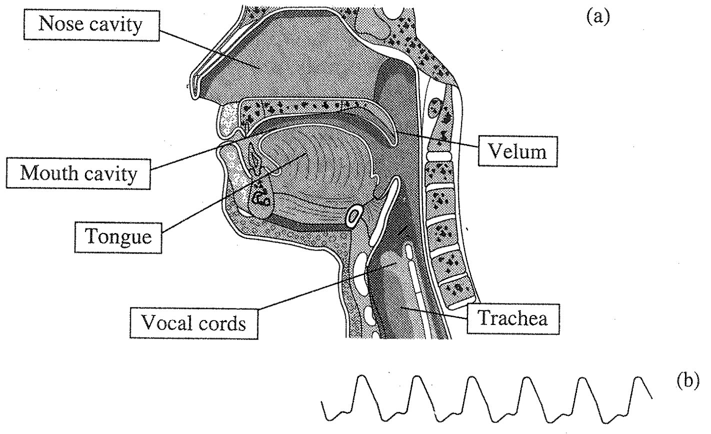


Figure 6.34 (a) The human speech production mechanism. (b) Typical pressure impulses.

For a low-pass source signal with finite bandwidth B Hz, even if we apply the minimum Nyquist sampling rate $2B$ Hz and 1-bit encoding, the bit rate cannot be lower than $2B$ bit/s. There have been many successful methods introduced to drastically reduce the source coding rates of speech and video signals, very important to our daily communication needs. Unlike waveform encoders, the most successful speech and video encoders are based on the human physiological models involved in speech generation and in video perception. Here we describe the basic principles of the linear prediction voice coders (known as vocoders) and the video compression method proposed by the Moving Picture Experts Group (MPEG).

6.8.1 Linear Prediction Coding Vocoders

Voice Models and Model-Based Vocoders

Linear prediction coding (LPC) vocoders are model-based systems. The model, in turn, is based on a good understanding of the human voice mechanism. Fig. 6.34a provides a cross-sectional illustration of the human speech apparatus. Briefly, human speech is produced by the joint interaction of lungs, vocal cords, and the articulation tract, consisting of the mouth and the nose cavity. Based on this physiological speech model, human voices can be divided into *voiced* and the *unvoiced* sound categories. Voiced sounds are those made while the vocal cords are vibrating. Put a finger on your Adam's apple* while speaking, and you can feel the vibration the vocal cords when you pronounce all the vowels and some consonants, such as *g* as in *gut*, *b* as in *but*, and *n* as in *nut*. Unvoiced sounds are made while the vocal cords are not vibrating. Several consonants such as *k*, *p*, and *t* are unvoiced. Examples of unvoiced sounds include *h* in *hut*, *c* in *cut*, and *p* in *put*.

For the production of voiced sounds, the lungs expel air through the epiglottis, causing the vocal cords to vibrate. The vibrating vocal cords interrupt the airstream and produce a quasi-periodic pressure wave consisting of impulses. The pressure wave impulses are commonly called pitch impulses, and the frequency of the pressure signal is the pitch frequency or fundamental frequency as shown in Fig. 6.34b. This is the part of the voice signal that defines the speech tone. Speech that is uttered in a constant pitch frequency sounds monotonous. In ordinary cases, the pitch frequency of a speaker varies almost constantly, often from syllable to syllable.

* The slight projection at the front of the throat formed by the largest cartilage of the larynx, usually more prominent in men than in women.

For voiced sound, the pitch impulses stimulate the air in the vocal tract (mouth and nasal cavities). For unvoiced sounds, the excitation comes directly from the air flow. Extensive studies[15–17] have shown that for unvoiced sounds, the excitation to the vocal tract is more like a broadband noise. When cavities in the vocal tract resonate under excitation, they radiate a sound wave, which is the speech signal. Both cavities form resonators with characteristic resonance frequencies (formant frequencies). Changing the shape (hence the resonant characteristics) of the mouth cavity allows different sounds to be pronounced. Amazingly, this (vocal) articulation tract can be approximately modeled by a simple linear digital filter with an all-pole transfer function

$$H(z) = \frac{g}{A(z)} = g \cdot \left(1 - \sum_{i=1}^{p} a_i z^{-i}\right)^{-1}$$

where g is a gain factor and $A(z)$ is known as the prediction filter, much like the feedback filter used in DPCM and ADPCM. One can view the function of the vocal articulation apparatus as a spectral shaping filter $H(z)$.

LPC Models

Based on this human speech model, a voice encoding approach different from waveform coding can be established. Instead of sending actual signal samples, the model-based vocoders *analyze* the voice signals segment by segment to determine the best-fitting speech model parameters. As shown in Fig. 6.35, after speech analysis, the transmitter sends the necessary speech model parameters (formants) for each voice segment to the receiver. The receiver then uses the parameters for the speech model to set up a voice synthesizer to regenerate the respective voice segments. In other words, what a user hears at the receiver actually consists of signals reproduced by an artificial voice *synthesizing machine*!

In the analysis of a sampled voice segment (consisting of multiple samples), the pitch analysis will first determine whether the speech is a voiced or an unvoiced piece. If the signal is classified as "voiced," the pitch analyzer will estimate pitch frequency (or equivalently the pitch period). In addition, the LPC analyzer will estimate the all-pole filter coefficients in $A(z)$. Because the linear prediction error indicates how well the linear prediction filter fits the voice samples, the LPC analyzer can determine the optimum filter coefficients by minimizing the mean square error (MSE) of the linear prediction error.[18, 19]

Directly transmitting the linear prediction (LP) filter parameters is unsound because the filter is very sensitive to parameter errors due to quantization and channel noises. Worse yet, the LP filter may even become unstable because of small coefficient errors. In practice, the stability of this all-pole linear prediction (LP) filter can be ensured by utilizing the modular lattice filter structure through the well-known Levinson-Durbin algorithm.[20, 21] Lattice filter parameters, known as reflection coefficients $\{r_k\}$, are less sensitive to quantization errors and

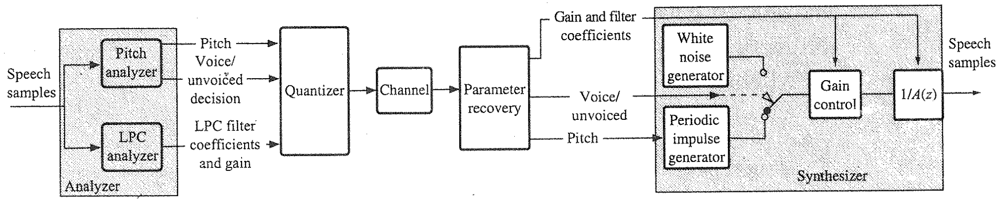


Figure 6.35 Analysis and synthesis of voice signals in an LPC encoder and decoder.

TABLE 6.1
Quantization Bit Allocation in LPC-10 Vocoder

Pitch Period	Voiced/Unvoiced	Gain g	10 LP Filter Parameters, bits/coefficient				
			r_1—r_4	r_5—r_8	r_9	r_{10}	
			5 bits	4 bits	3 bits	2 bits	Voiced
6 bits	1 bit	5 bits	5 bits	*Not used*			Unvoiced

noise. Transmission is further improved by sending their log-area ratios (LAR), defined as

$$o_k \triangleq \log \frac{1 + r_k}{1 - r_k}$$

or by sending intermediate values from the Levinson-Durbin recursion known as the partial reflection coefficients (PARCOR). Another practical approach is to find the equivalent *line spectral pairs (LSP)* as representation of the LPC filter coefficients for transmission over channels. LSP has the advantage of low sensitivity to quantization noise.[22, 23] As long as the pth-order all-pole LP filter is stable, it can be represented by p real-valued, line spectral frequencies. In every representation, however, a pth-order synthesizer filter can be obtained by the LPC decoder from the quantization of p real-valued coefficients. In general 8 to 14 LP parameters are sufficient for vocal tract representation.

We can now use a special LPC example to illustrate the code efficiency of such model-based vocoders. In the so-called LPC-10 vocoder,* the speech is sampled at 8 kHz. 180 samples (22.5 ms) form an LPC frame for transmission.[24] The bits per speech frame are allocated to quantize the pitch period, the voiced/unvoiced flag, the filter gain, and the 10 filter coefficients, according to Table 6.1. Thus, each frame requires between 32 (unvoiced) and 53 (voiced) bits. Adding frame control bits results an average coded stream of 54 bits per speech frame, or an overall rate of 2400 bit/s.[24] Based on subjective tests, this rather minimal LPC-10 codec has low mean opinion score (MOS) but does provide highly intelligible speech connections. LPC-10 is part of the FS-1015, a low-rate secure telephony codec standard developed by the U.S. Department of Defense in 1984. A later enhancement to LPC-10 is known as the LPC-10(e).

Compared with the 64 kbit/s PCM or the 32 kbit/s ADPCM waveform codec, LPC vocoders are much more efficient and can achieve speech code rates below 9.6 kbit/s. The 2.4 kbit/s LPC-10 example can provide speech digitization at a rate much lower than even the speech waveform sampling rate of 8 kHz. The loss of speech quality is a natural trade-off. To better understand the difference between waveform vocoders and the model-based vocoders such as LPC, we can use the analogy of a food delivery service. Imagine a family living Alaska that wishes to order a nice meal from a famous restaurant in New York City. For practical reasons, the restaurant would have to send prepared dishes uncooked and frozen; then the family would follow the cooking directions. The food would probably taste fine, but the meal would be missing the finesse of the original chef. This option is like speech transmission via PCM. The receiver has the basic ingredients but must tolerate the quantization error (manifested by the lack of the chef's cooking finesse). To reduce transportation weight, another option is for the family to order the critical ingredients only. The heavier but common ingredients (such as rice and potatoes) can be acquired locally. This approach is like DPCM or ADPCM, in which only the unpredictable part of the voice is transmitted. Finally, the family can simply go online to

* So-called because it uses order $p = 10$. The idea is to allocate two parameters for each possible formant frequency peak.

order the chef's recipe. All the ingredients are purchased locally and the cooking is also done locally. The Alaskan family can satisfy their gourmet craving without receiving a single food item form New York! Clearly, the last scenario captures the idea of model-based vocoders. LPC vocoders essentially deliver the recipe (i.e., the LPC parameters) for voice synthesis at the receiver end.

Practical High-Quality LP Vocoders

The simple dual-state LPC synthesis of Fig. 6.35 describes no more than the basic idea behind model-based voice codecs. The quality of LP vocoders has been greatly improved by a number of more elaborate codecs in practice. By adding a few bits, these LP-based vocoders attempt to improve the speech quality in two ways: by encoding the residual prediction error and by enhancing the excitation signal.

The most successful methods belong to the class known as code-excited linear prediction (CELP) vocoders. CELP vocoders use a codebook, a table of typical LP error (or residue) signals, which is set up a priori by designers. At the transmitter, the analyzer compares the actual prediction residue to all the entries in the codebook, chooses the entry that is the closest match, and just adds the address (code) for that entry to the bits for transmission. The synthesizer receives this code, retrieves the corresponding residue from the codebook, and uses it to modify the synthesizing output. For CELP to work well, the codebook must be big enough, requiring more transmission bits. The FS-1016 vocoder is an improvement over FS-1015 and provides good quality, natural-sounding speech at 4.8 kbit/s.[25] More modern variants include the RPE-LTP (regular pulse excitation, long-term prediction) LPC codec used in GSM cellular systems, the algebraic CELP (ACELP), the relaxed CELP (RCELP), the Qualcomm CELP (QCELP) in CDMA cellular phones, and vector-sum excited linear prediction (VSELP). Their data rates range from as low as 1.2 kbit/s to 13 kbit/s (full-rate GSM). These vocoders form the basis of many modern cellular vocoders, voice over Internet Protocol (VoIP), and other ITU-T G-series standards.

Video Compression

For video and television to go digital we face a tremendous the challenge. Because of the high video bandwidth (approximately 4.2 MHz), use of direct sampling and quantization leads to an uncompressed digital video signal of roughly 150 Mbit/s. Thus, the modest compression afforded by techniques such as ADPCM and subband coding[26, 27] is insufficient. The key to video compression, as it turns out, has to do with human visual perception.

A great deal of research and development has resulted in methods to drastically reduce the digital bandwidth required for video transmission. Early compression techniques compressed video signals to approximately 45 Mbit/s (DS3). For the emerging video delivery technologies of HFC, ADSL, HDTV, and so on, however, much greater compression was required. MPEG approached this problem and developed new compression techniques, which provide network or VCR quality video at much greater levels of compression. MPEG is a joint effort of the International Standards Organizations (ISO), the International Electrotechnical Committee (IEC), and the American National Standards Institute (ANSI) X3L3 Committee.[28, 29] MPEG has a very informative website that provides extensive information on MPEG and JPEG technologies and standards (http://www.mpeg.org/index.html/). MPEG also has an industrial forum promoting the organization's products (http://www.m4if.org/).

The concept of digital video compression is based on the fact that, on the average, a relatively small number of pixels change from frame to frame. Hence, if only the changes are transmitted, the transmission bandwidth can be reduced significantly. Digitizing allows the noise-free recovery of analog signals and improves the picture quality at the receiver.

Compression reduces the bandwidth required for transmission and the amount of storage for a video program and, hence, expands channel capacity. Without compression, a 2-hour digitized NTSC video program would require roughly 100 gigabytes of storage, far exceeding the capacity of any DVD disc.

There are three primary MPEG standards in use:

MPEG-1:	Used for VCR-quality video and storage on video CD (or VCD) at a data rate of 1.5 Mbit/s. These VCDs were quite popular throughout Asia (except Japan). MPEG-1 decoders are available on most computers. VCD is also a very popular format for karaoke.
MPEG-2:	Supports diverse video coding applications for transmissions ranging in quality from VCR to high-definition TV (HDTV), depending on data rate. It offers 50:1 compression of raw video. MPEG-2 is a highly popular format used in DVD, HDTV, terrestrial digital video broadcasting (DVB-T), and digital video broadcasting by satellite (DVB-S).
MPEG-4	Provides multimedia (audio, visual, or audiovisual) content streaming over different bandwidths including internet. MPEG-4 is supported by Microsoft Windows Media Player, Real Networks, and Apple's Quicktime and iPod. MPEG-4 recently converged with an ITU-T standard known as H.264, to be discussed later.

The power of video compression is staggering. By comparison, NTSC broadcast television in digital form requires 45 to 120 Mbit/s, whereas MPEG-2 requires 1.5 to 15 Mbit/s. On the other hand HDTV would require 800 Mbit/s uncompressed which, under MPEG-2 compression, will transmit at 19.39 Mbit/s.

There are two types of MPEG compression, which eliminate redundancies in the audiovisual signals that are not perceptible by the listener or the viewer:

1. Video
   - Temporal or *interframe* compression by predicting interframe motion and removing interframe redundancy
   - Spatial or intraframe compression, which forms a block identifier for a group of pixels having the same characteristics (color, intensity, etc.) for each frame. Only the block identifier is transmitted.
2. Audio, which uses a psychoacoustic model of masking effects.

The basis for video compression is to remove redundancy in the video signal stream. As an example of interframe redundancy, consider Fig. 6.36a and b. In Fig. 6.36a the runner is in position *A* and in Fig. 6.36b he is in position *B*. Note that the background (cathedral, buildings, and bridge) remains essentially unchanged from frame to frame. Figure 6.36c represents the nonredundant information for transmission; that is, the change between the two frames. The runner image on the left represents the blocks of frame 1 that are replaced by background in frame 2. The runner image on the right represents the blocks of frame 1 that replace the background in frame 2.

Video compression starts with an encoder, which converts the analog video signal from the video camera to a digital format on a pixel-by-pixel basis. Each video frame is divided into 8×8 pixel blocks, which are analyzed by the encoder to determine which blocks must be transmitted, that is, which blocks have significant changes from frame to frame. This process takes place in two stages:

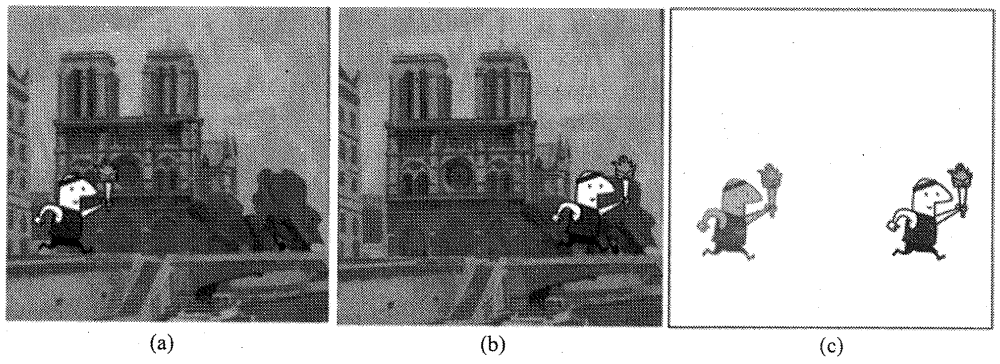


Figure 6.36 (a) Frame 1. (b) Frame 2. (c) Information transferred between frames 1 and 2.

1. Motion estimation and compensation. Here a motion estimator identifies the areas or groups of blocks from a preceding frame that match corresponding areas in the current frame and sends the magnitude and direction of the displacement to a predictor in the decoder. The frame difference information is called the residual.
2. Transforming the residual on a block-by-block basis into more compact form.

The encoded residual signal is transformed into a more compact form by means of a discrete cosine transform (DCT) (see Sec. 6.5.2 in Haskel et al.,[28]), which uses a numerical value to represent each pixel and normalizes that value for more efficient transmission. The DCT is of the form

$$F(j,k) = \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} f(n,m)\cos\left[\frac{(2n+1)j\pi}{2N}\right]\cos\left[\frac{(2m+1)k\pi}{2N}\right]$$

where $f(n, m)$ is the value assigned to the block in the (n, m) position. The inverse transform is

$$f(n,m) = \frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} F(j,k)\cos\left[\frac{(2n+1)j\pi}{2N}\right]\cos\left[\frac{(2m+1)k\pi}{2N}\right]$$

The DCT is typically multiplied, for an 8×8 block, by the expression $C(j)C(k)/4$, where

$$C(x) = \begin{cases} \dfrac{1}{\sqrt{2}} & \text{for } x = 0 \\ 1 & \text{otherwise} \end{cases}$$

Tables 6.2 and 6.3 depict the pixel block values before and after the DCT. One can notice from Table 6.3 that there are relatively few meaningful elements, that is, elements with significant values relative to the values centered about the 0, 0 position. Because of this, most of the matrix values may be assumed to be zero, and, upon inverse transformation, the original values are quite accurately reproduced. This process reduces the amount of data that must be transmitted greatly, perhaps by a factor of 8 to 10 on the average. Note that the size of the transmitted residual may be that of an individual block or, at the other extreme, that of the entire picture.

The transformed matrix values of a block (Table 6.4) are normalized so that most of the values in the block matrix are less than 1. Then the resulting normalized matrix is quantized to

TABLE 6.2
8 × 8 Pixel Block Residual

	n							
	158	158	158	163	161	161	162	162
	157	157	157	162	163	161	162	162
	157	157	157	160	161	161	161	161
	155	155	155	162	162	161	160	159
m	159	159	159	160	160	162	161	159
	156	156	156	158	163	160	155	150
	156	156	156	159	156	153	151	144
	155	155	155	155	153	149	144	139

TABLE 6.3
Transformed 8 × 8 Pixel Block Residual DCT Coefficients

	i							
	1259.6	1.0	−12.1	5.2	2.1	1.7	−2.7	−1.3
	22.6	−17.5	6.2	−3.2	2.9	−0.1	−0.4	−1.2
	−10.9	9.3	−1.6	−1.5	0.2	0.9	−0.6	0.1
	7.1	−1.9	−0.2	1.5	−0.9	−0.1	0.0	0.3
k	−0.6	0.8	1.5	−1.6	−0.1	0.7	0.6	−1.3
	−1.8	−0.2	−1.6	−0.3	0.8	1.5	−1.0	−1.0
	−1.3	0.4	−0.3	1.5	−0.5	−1.7	1.1	0.8
	2.6	1.6	3.8	−1.8	−1.9	1.2	0.6	−0.4

TABLE 6.4
Normalized and Quantized
Residual DCT Coefficients

	jn							
	21	0	−1	0	0	0	0	0
	2	−1	0	0	0	0	0	0
	−1	1	0	0	0	0	0	0
k	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0

obtain Table 6.4. Normalization is accomplished by a dynamic matrix of multiplicative values, which are applied element by element to the transformed matrix. The normalized matrix of Table 6.4 is the block information transmitted to the decoder. The denormalized matrix pictured in Table 6.5 and the reconstructed (inverse-transformed) residual in Table 6.6 are determined by the decoder. The transformation proceeds in a zigzag pattern, as illustrated in Fig. 6.37.

MPEG approaches the motion estimation and compensation to remove temporal (frame-to-frame) redundancy in a unique way. MPEG uses three types of frame, the intraframe or I-frame (sometimes called the independently coded or intracoded frame), the predicted (predictive) or

TABLE 6.5
Denormalized DCT Coefficients

	n							
k	1260	0	−12	0	0	0	0	0
	23	−18	0	0	0	0	0	0
	−11	10	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0

TABLE 6.6
Inverse DCT Coefficients Reconstructed Residual

	n							
m	158	158	158	163	161	161	162	162
	157	157	157	162	163	161	162	162
	157	157	157	160	161	161	161	161
	155	155	155	162	162	161	160	159
	159	159	159	160	160	162	161	159
	156	156	156	158	163	160	155	150
	156	156	156	159	156	153	151	144
	155	155	155	155	153	149	144	139

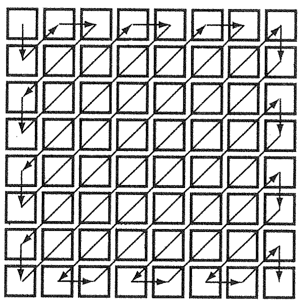


Figure 6.37 Zigzag DCT coefficient scanning pattern.

P-frame, and the bidirectionally predictive frame or B-frame. The P-frames are predicted from the I-frames. The B-frames are bidirectionally predicted from either past or future frames. An I-frame and one or more P-frames and B-frames make up the basic MPEG processing pattern, called a group of pictures (GOP). Most of the frames in an MPEG compressed image are B-frames. The I-frame provides the initial reference for the frame differences to start the MPEG encoding process. Note that the bidirectional aspect of the procedure introduces a delay in the transmission of the frames. This is because the GOP is transmitted as a unit and, hence, transmission cannot start until the GOP is complete (Fig. 6.38). The details of the procedure are beyond the scope of this text. There are many easily accessible books that cover this subject in detail. In addition, one may find numerous references to MPEG compression and HDTV on the internet.

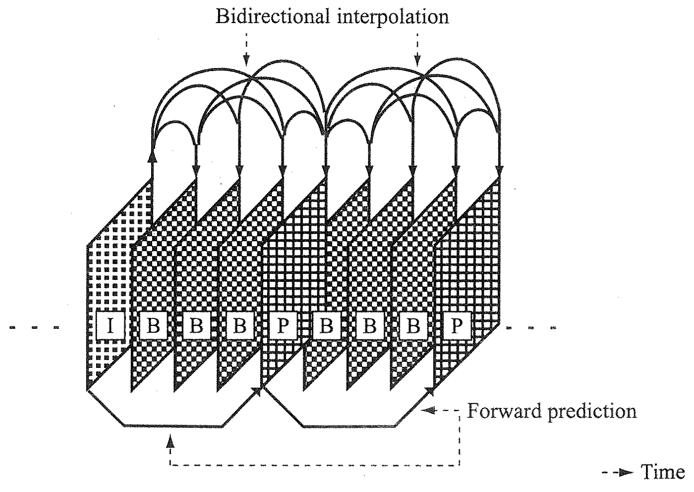


Figure 6.38 MPEG temporal frame structure.

Other Video Compression Standards

We should mention that in addition to MPEG, there is a parallel attempt by ITU-T to standardize video coding. These standards apply similar concepts for video compression. Today, the well-known ITU-T video compression standards are the H.26x series, including H.261, H.263, and H.264. H.261 was developed for transmission of video at a rate of multiples of 64 kbit/s in applications such as videophone and videoconferencing. Similar to MPEG compression, H.261 uses motion-compensated temporal prediction.

H.263 was designed for very low bit rate coding applications, such as videoconferencing. It uses block motion-compensated DCT structure for encoding.[30] Based on H.261, H.263 is better optimized for coding at low bit rates and achieves much higher efficiency than H.261 encoding. Flash Video, a highly popular format for video sharing on many web engines such as YouTube and MySpace, uses a close variant of the H.263 codec called the Sorenson Spark codec.

In fact, H.264 represents a recent convergence between ITU-T and MPEG and is a joint effort of the two groups. Also known as MPEG-4 Part 10, H.264 typically outperforms MPEG-2 by cutting the data rate nearly in half. This versatile standard supports video applications over multiple levels of bandwidth and quality, including, mobile phone service at 50 to 60 kbit/s, Internet/standard definition video at 1 to 2 Mbit/s, and high-definition video at 5 to 8 Mbit/s. H.264 is also supported in many other products and applications including iPod, direct broadcasting satellite TV, some regional terrestrial digital TV, Mac OS X (Tiger), and Sony's Playstation Portable.

A Note on High-Definition Television (HDTV)

Utilizing MPEG-2 for video compression, high-definition television (HDTV) is one of the advanced television (ATV) functions along with 525-line compressed video for direct broadcast satellite (DBS) or cable. The concept of HDTV appeared in the late 1970s. Early development work was performed primarily in Japan based on an analog system. In the mid-1980s it became apparent that the bandwidth requirements of an analog system would be excessive, and work began on a digital system that could utilize the 6 MHz bandwidth of NTSC television. In the early 1990s seven digital systems were proposed, but testing indicated that none would be highly satisfactory. Therefore, in 1993 the FCC suggested the formation of an industrial "Grand Alliance" (GA) to develop a common HDTV standard. In December 1997, Standard

A/53 for broadcast transmission, proposed by the Advanced Television Systems Committee (ATSC), was finalized by the FCC in the United States.

The GA HDTV standard is based on a 16:9 aspect ratio (motion picture aspect ratio) rather than the 4:3 aspect ratio of NTSC television. HDTV uses MPEG-2 compression at 19.39 Mbit/s and a digital modulation format called 8-VSB (vestigial sideband), which uses an eight-amplitude-level symbol to represent 3 bits of information. Transmission is in 207-byte blocks, which include 20 parity bytes for Reed-Solomon forward error correction. The remaining 187-byte packet format is a subset of the MPEG-2 protocol and includes headers for timing, switching, and other transmission control.

The Advanced Television Systems Group, the successor to the Grand Alliance, has been developing standards and recommended practices for HDTV. These are found, along with a great deal of other information, on their website: http://www.atsc.org/.

6.9 MATLAB EXERCISES

In the MATLAB exercises of this section, we provide examples of signal sampling, signal reconstruction from samples, uniform quantization, pulse-coded modulation (PCM), and delta modulation (DM).

Sampling and Reconstruction of Lowpass Signals

In the sampling example, we first construct a signal $g(t)$ with two sinusoidal components of 1-second duration; their frequencies are 1 and 3 Hz. Note, however, that when the signal duration is infinite, the bandwidth of $g(t)$ would be 3 Hz. However, the finite duration of the signal implies that the actual signal is not band-limited, although most of the signal content stays within a bandwidth of 5 Hz. For this reason, we select a sampling frequency of 50 Hz, much higher than the minimum Nyquist frequency of 6 Hz. The MATLAB program, `Exsample.m`, implements sampling and signal reconstruction. Figure 6.39 illustrates the original signal, its uniform samples at the 50 Hz sampling rate, and the frequency response of the sampled signal. In accordance with our analysis of Section 6.1, the spectrum of the sampled signal $g_T(t)$ consists of the original signal spectrum periodically repeated every 50 Hz.

```
% (Exsample.m)
% Example of sampling, quantization, and zero-order hold
clear;clf;
td=0.002;        %original sampling rate 500 Hz
t=[0:td:1.];     %time interval of 1 second
xsig=sin(2*pi*t)-sin(6*pi*t);    % 1Hz+3Hz sinusoids
Lsig=length(xsig);

ts=0.02;         %new sampling rate = 50Hz.
Nfactor=ts/td;
% send the signal through a 16-level uniform quantizer
[s_out,sq_out,sqh_out,Delta,SQNR]=sampandquant(xsig,16,td,ts);
%   receive 3 signals:
%       1. sampled signal s_out
%       2. sampled and quantized signal sq_out
%       3. sampled, quantized, and zero-order hold signal sqh_out
%
```

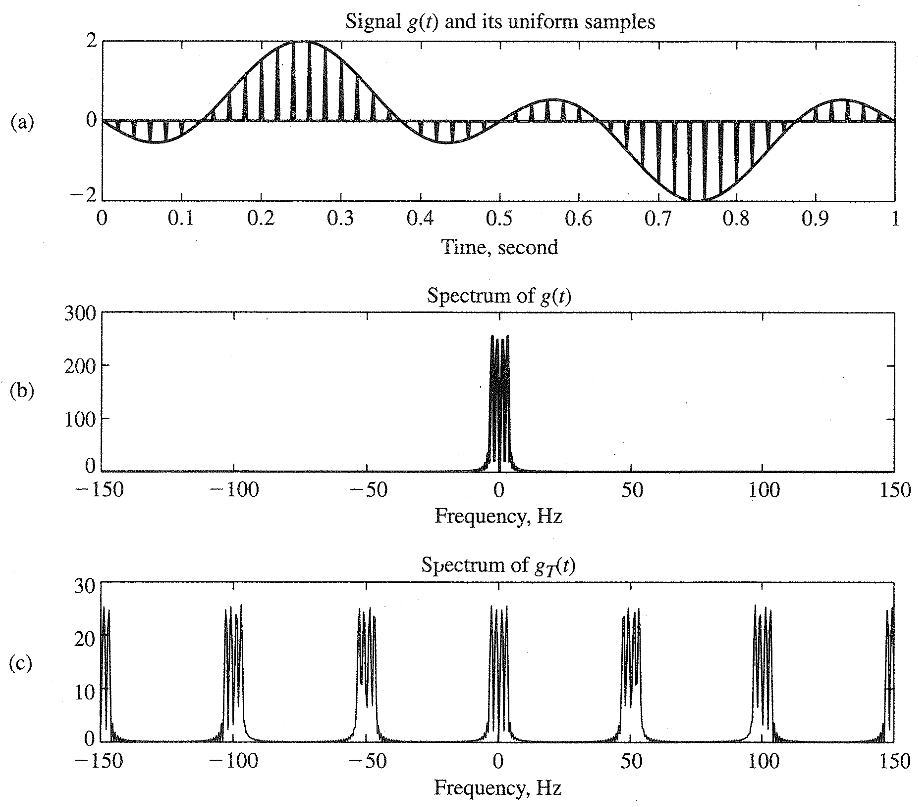


Figure 6.39 The relationship between the original signal and the ideal uniformly sampled signal in the time (a) and frequency (b, c) domains.

```
%   calculate the Fourier transforms
 Lfft=2^ceil(log2(Lsig)+1);
 Fmax=1/(2*td);
 Faxis=linspace(-Fmax,Fmax,Lfft);
 Xsig=fftshift(fft(xsig,Lfft));
 S_out=fftshift(fft(s_out,Lfft));
%   Examples of sampling and reconstruction using
%      a) ideal impulse train through LPF
%      b) flat top pulse reconstruction through LPF
%   plot the original signal and the sample signals in time
%   and frequency domain
figure(1);
subplot(311); sfig1a=plot(t,xsig,'k');
hold on; sfig1b=plot(t,s_out(1:Lsig),'b'); hold off;
set(sfig1a,'Linewidth',2); set(sfig1b,'Linewidth',2.);
xlabel('time (sec)');
title('Signal {\it g}({\it t}) and its uniform samples');
subplot(312); sfig1c=plot(Faxis,abs(Xsig));
xlabel('frequency (Hz)');
axis([-150 150 0 300])
set(sfig1c,'Linewidth',1);  title('Spectrum of {\it g}({\it t})');
subplot(313); sfig1d=plot(Faxis,abs(S_out));
xlabel('frequency (Hz)');
axis([-150 150 0 300/Nfactor])
```

```
set(sfig1c,'Linewidth',1);  title('Spectrum of {\it g}_T({\it t})');
%   calculate the reconstructed signal from ideal sampling and
%   ideal LPF
%   Maximum LPF bandwidth equals to BW=floor((Lfft/Nfactor)/2);
 BW=10;              %Bandwidth is no larger than 10Hz.
 H_lpf=zeros(1,Lfft);H_lpf(Lfft/2-BW:Lfft/2+BW-1)=1;  %ideal LPF
 S_recv=Nfactor*S_out.*H_lpf;              % ideal filtering
 s_recv=real(ifft(fftshift(S_recv)));      % reconstructed f-domain
 s_recv=s_recv(1:Lsig);                    % reconstructed t-domain
%   plot the ideally reconstructed signal in time
%   and frequency domain
figure(2)
subplot(211); sfig2a=plot(Faxis,abs(S_recv));
xlabel('frequency (Hz)');
axis([-150 150 0 300]);
title('Spectrum of ideal filtering (reconstruction)');
subplot(212); sfig2b=plot(t,xsig,'k-.',t,s_recv(1:Lsig),'b');
legend('original signal','reconstructed signal');
xlabel('time (sec)');
title('original signal versus ideally reconstructed signal');
set(sfig2b,'Linewidth',2);
%   non-ideal reconstruction
 ZOH=ones(1,Nfactor);
 s_ni=kron(downsample(s_out,Nfactor),ZOH);
 S_ni=fftshift(fft(s_ni,Lfft));
 S_recv2=S_ni.*H_lpf;             % ideal filtering
 s_recv2=real(ifft(fftshift(S_recv2)));     % reconstructed f-domain
 s_recv2=s_recv2(1:Lsig);                   % reconstructed t-domain
%   plot the ideally reconstructed signal in time
%   and frequency domain
figure(3)
subplot(211); sfig3a=plot(t,xsig,'b',t,s_ni(1:Lsig),'b');
xlabel('time (sec)');
title('original signal versus flat-top reconstruction');
subplot(212); sfig3b=plot(t,xsig,'b',t,s_recv2(1:Lsig),'b--');
legend('original signal','LPF reconstruction');
xlabel('time (sec)');
set(sfig3a,'Linewidth',2); set(sfig3b,'Linewidth',2);
title('original and flat-top reconstruction after LPF');
```

To construct the original signal $g(t)$ from the impulse sampling train $g_T(t)$, we applied an ideal low-pass filter with bandwidth 10 Hz in the frequency domain. This corresponds to the interpolation using the ideal sinc function as shown in Sec. 6.1.1. The resulting spectrum, as shown in Fig. 6.40, is nearly identical to the original message spectrum of $g(t)$. Moreover, the time domain signal waveforms are also compared in Fig. 6.40 and show near perfect match.

In our last exercise in sampling and reconstruction, given in the same program, we use a simple rectangular pulse of width T_s (sampling period) to reconstruct the original signal from the samples (Fig. 6.41). A low-pass filter is applied on the rectangular reconstruction and also shown in Fig. 6.41. It is clear from comparison to the original source signal that the

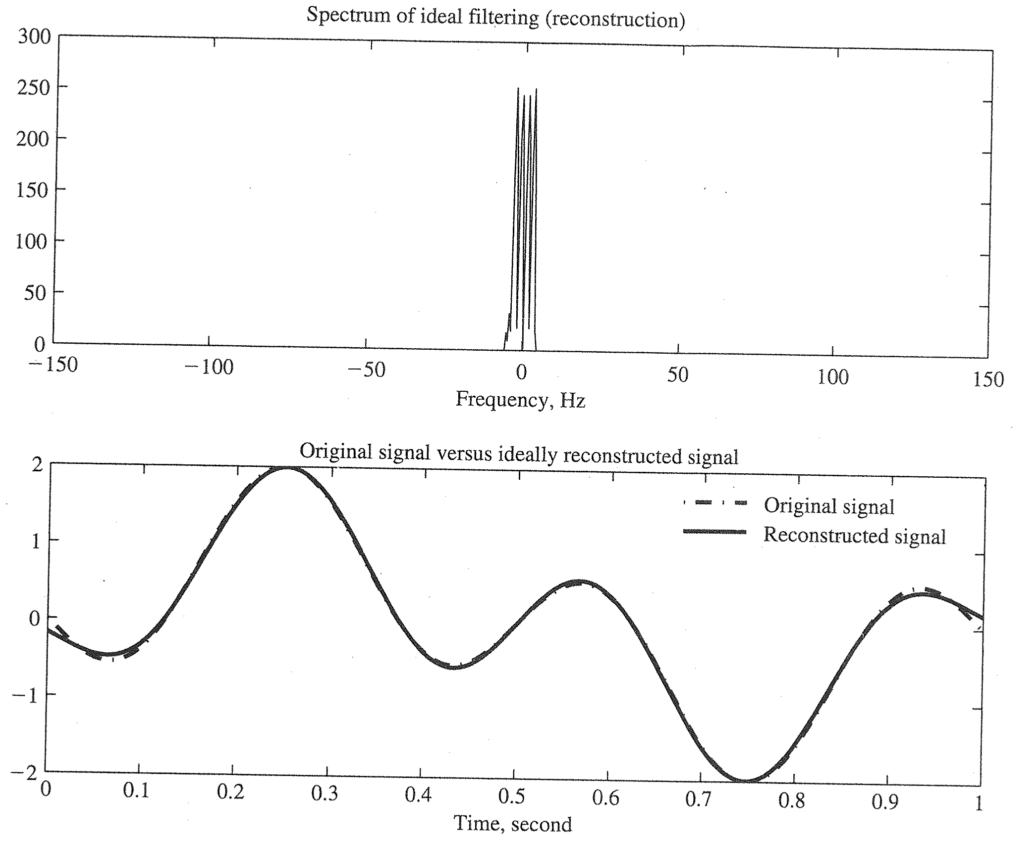


Figure 6.40 Reconstructed signal spectrum and waveform from applying the ideal impulse sampling and ideal low-pass filter reconstruction.

recovered signal is still very close to the original signal $g(t)$. This is because we have chosen a high sampling rate such that $T_p = T_s$ is so small that the approximation of Eq. (6.17) holds. Certainly, based on our analysis, by applying the low-pass equalization filter of Eq. (6.16), the reconstruction error can be greatly reduced.

PCM Illustration

The uniform quantization of an analog signal using L quantization levels can be implemented by the MATLAB function `uniquan.m`.

```
% (uniquan.m)
function [q_out,Delta,SQNR]=uniquan(sig_in,L)
%   Usage
%   [q_out,Delta,SQNR]=uniquan(sig_in,L)
%   L   -     number of uniform quantization levels
%   sig_in -    input signal vector
%   Function outputs:
%             q_out - quantized output
%             Delta - quantization interval
%             SQNR  - actual signal to quantization noise ratio
sig_pmax=max(sig_in);   % finding the positive peak
sig_nmax=min(sig_in);   % finding the negative peak
Delta=(sig_pmax-sig_nmax)/L;    % quantization interval
```

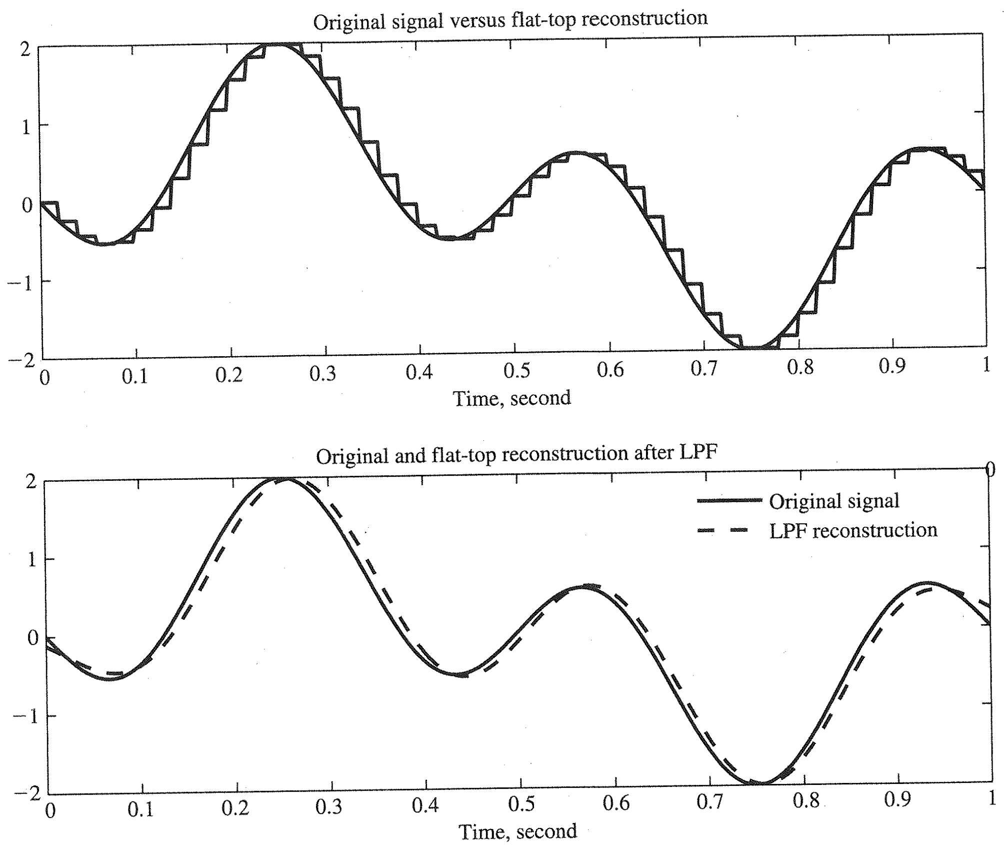


Figure 6.41 Reconstructed signal spectrum and waveform from applying the simple rectangular reconstruction pulse (Fig. 6.6) followed by LPF without equalization.

```
q_level=sig_nmax+Delta/2:Delta:sig_pmax-Delta/2;    % define Q-levels
L_sig=length(sig_in);    % find signal length
sigp=(sig_in-sig_nmax)/Delta+1/2;    % convert into 1/2 to L+1/2 range
qindex=round(sigp);      % round to 1, 2, ... L levels
qindex=min(qindex,L);    % eleminate L+1 as a rare possibility
q_out=q_level(qindex);  % use index vector to generate output
SQNR=20*log10(norm(sig_in)/norm(sig_in-q_out)); %actual SQNR value
end
```

The function sampandquant.m executes both sampling and uniform quantization simultaneously. The sampling period ts is needed, along with the number L of quantization levels, to generate the sampled output s_out, the sampled and quantized output sq_out, and the signal after sampling, quantizing, and zero-order-hold sqh_out.

```
% (sampandquant.m)
function [s_out,sq_out,sqh_out,Delta,SQNR]=sampandquant(sig_in,L,td,ts)
%   Usage
%        [s_out,sq_out,sqh_out,Delta,SQNR]=sampandquant(sig_in,L,td,fs)
%   L   -    number of uniform quantization levels
%   sig_in -    input signal vector
%   td  -    original signal sampling period of sig_in
%   ts  -    new sampling period
% NOTE:  td*fs must be a positive integer;
%   Function outputs:
%            s_out  - sampled output
%            sq_out - sample-and-quantized output
%            sqh_out- sample,quantize,and hold output
%             Delta - quantization interval
%             SQNR  - actual signal to quantization noise ratio

if (rem(ts/td,1)==0)
nfac=round(ts/td);
p_zoh=ones(1,nfac);
s_out=downsample(sig_in,nfac);
[sq_out,Delta,SQNR]=uniquan(s_out,L);
s_out=upsample(s_out,nfac);
sqh_out=kron(sq_out,p_zoh);
sq_out=upsample(sq_out,nfac);
else
    warning('Error! ts/td is not an integer!');
    s_out=[];sq_out=[];sqh_out=[];Delta=[];SQNR=[];
end
end
```

The MATLAB program ExPCM.m provides a numerical example that uses these two MATLAB functions to generate PCM signals.

```
% (ExPCM.m)
% Example of sampling, quantization, and zero-order hold
clear;clf;
td=0.002;        %original sampling rate 500 Hz
t=[0:td:1.];     %time interval of 1 second
xsig=sin(2*pi*t)-sin(6*pi*t);   % 1Hz+3Hz sinusoids
Lsig=length(xsig);
Lfft=2^ceil(log2(Lsig)+1);
Xsig=fftshift(fft(xsig,Lfft));
Fmax=1/(2*td);
Faxis=linspace(-Fmax,Fmax,Lfft);
ts=0.02;         %new sampling rate = 50Hz.
Nfact=ts/td;
% send the signal through a 16-level uniform quantizer
[s_out,sq_out,sqh_out1,Delta,SQNR]=sampandquant(xsig,16,td,ts);
%   obtained the PCM signal which is
%       - sampled, quantized, and zero-order hold signal sqh_out
%   plot the original signal and the PCM signal in time domain
figure(1);
subplot(211);sfig1=plot(t,xsig,'k',t,sqh_out1(1:Lsig),'b');
set(sfig1,'Linewidth',2);
title('Signal {\it g}({\it t}) and its 16 level PCM signal')
```

```
xlabel('time (sec.)');
% send the signal through a 16-level uniform quantizer
[s_out,sq_out,sqh_out2,Delta,SQNR]=sampandquant(xsig,4,td,ts);
%   obtained the PCM signal which is
%       - sampled, quantized, and zero-order hold signal sqh_out
%   plot the original signal and the PCM signal in time domain
subplot(212);sfig2=plot(t,xsig,'k',t,sqh_out2(1:Lsig),'b');
set(sfig2,'Linewidth',2);
title('Signal {\it g}({\it t}) and its 4 level PCM signal')
xlabel('time (sec.)');

 Lfft=2^ceil(log2(Lsig)+1);
 Fmax=1/(2*td);
 Faxis=linspace(-Fmax,Fmax,Lfft);
 SQH1=fftshift(fft(sqh_out1,Lfft));
 SQH2=fftshift(fft(sqh_out2,Lfft));
% Now use LPF to filter the two PCM signals
 BW=10;              %Bandwidth is no larger than 10Hz.
 H_lpf=zeros(1,Lfft);H_lpf(Lfft/2-BW:Lfft/2+BW-1)=1;  %ideal LPF
 S1_recv=SQH1.*H_lpf;              % ideal filtering
 s_recv1=real(ifft(fftshift(S1_recv)));      % reconstructed f-domain
 s_recv1=s_recv1(1:Lsig);                    % reconstructed t-domain
  S2_recv=SQH2.*H_lpf;             % ideal filtering
 s_recv2=real(ifft(fftshift(S2_recv)));      % reconstructed f-domain
 s_recv2=s_recv2(1:Lsig);                    % reconstructed t-domain
 % Plot the filtered signals against the original signal
figure(2)
 subplot(211);sfig3=plot(t,xsig,'b-',t,s_recv1,'b-.');
 legend('original','recovered')
set(sfig3,'Linewidth',2);
title('Signal {\it g}({\it t}) and filtered 16-level PCM signal')
xlabel('time (sec.)');
subplot(212);sfig4=plot(t,xsig,'b-',t,s_recv2(1:Lsig),'b-.');
 legend('original','recovered')
set(sfig4,'Linewidth',2);
title('Signal {\it g}({\it t}) and filtered 4-level PCM signal')
xlabel('time (sec.)');
```

In the first example, we maintain the 50 Hz sampling frequency and utilize $L = 16$ uniform quantization levels. The resulting PCM signal is shown in Fig. 6.42. This PCM signal can be low-pass-filtered at the receiver and compared against the original message signal, as shown in Fig. 6.43. The recovered signal is seen to be very close to the original signal $g(t)$.

To illustrate the effect of quantization, we next apply $L = 4$ PCM quantization levels. The resulting PCM signal is again shown in Fig. 6.42. The corresponding signal recovery is given in Fig. 6.43. It is very clear that smaller number of quantization levels ($L = 4$) leads to much larger approximation error.

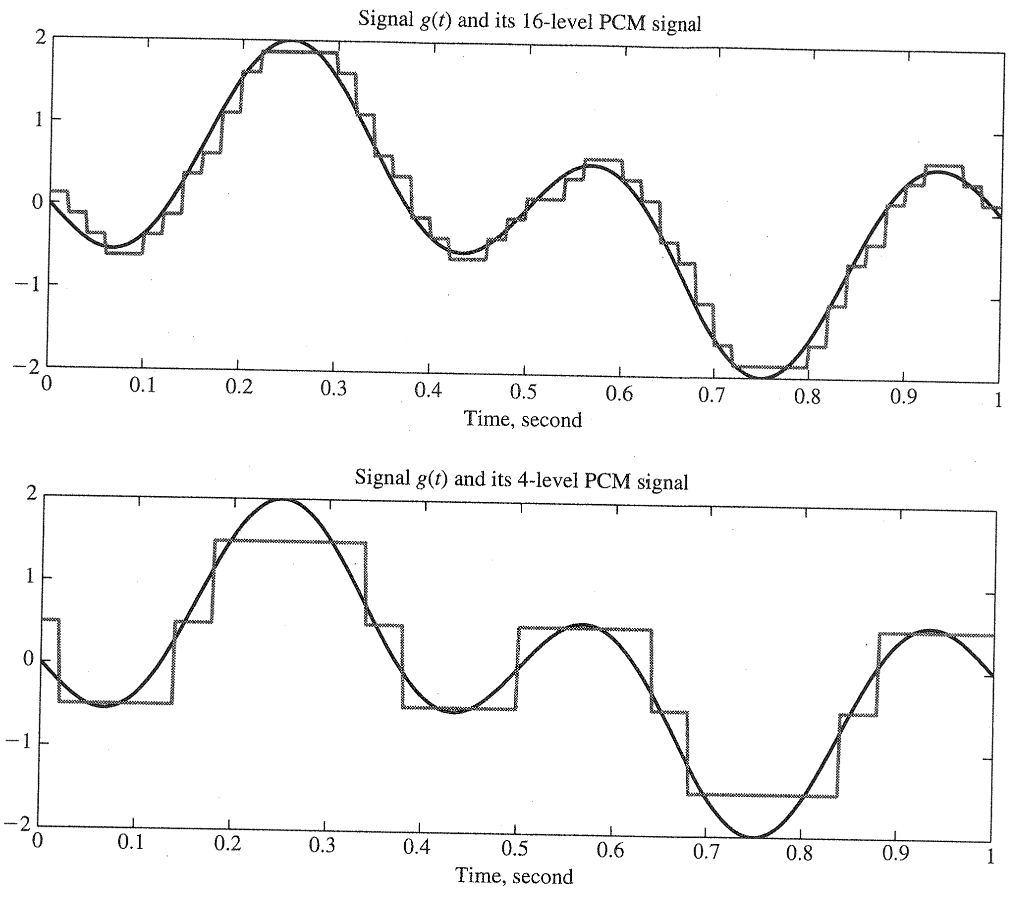


Figure 6.42 Original signal and the PCM signals with different numbers of quantization levels.

Delta Modulation

Instead of applying PCM, we illustrate the practical effect of step size selection Δ in the design of DM encoder. The basic function to implement DM is given in `deltamod.m`.

```
% (deltamod.m)
function s_DMout= deltamod(sig_in,Delta,td,ts)
%   Usage
%        s_DMout = deltamod(xsig,Delta,td,ts))
%   Delta  -    DM stepsize
%   sig_in -    input signal vector
%   td  -    original signal sampling period of sig_in
%   ts  -    new sampling period
% NOTE:  td*fs must be a positive integer;
%   Function outputs:
%              s_DMout  - DM sampled output
if (rem(ts/td,1)==0)
nfac=round(ts/td);
p_zoh=ones(1,nfac);
s_down=downsample(sig_in,nfac);
Num_c=length(s_down);
```

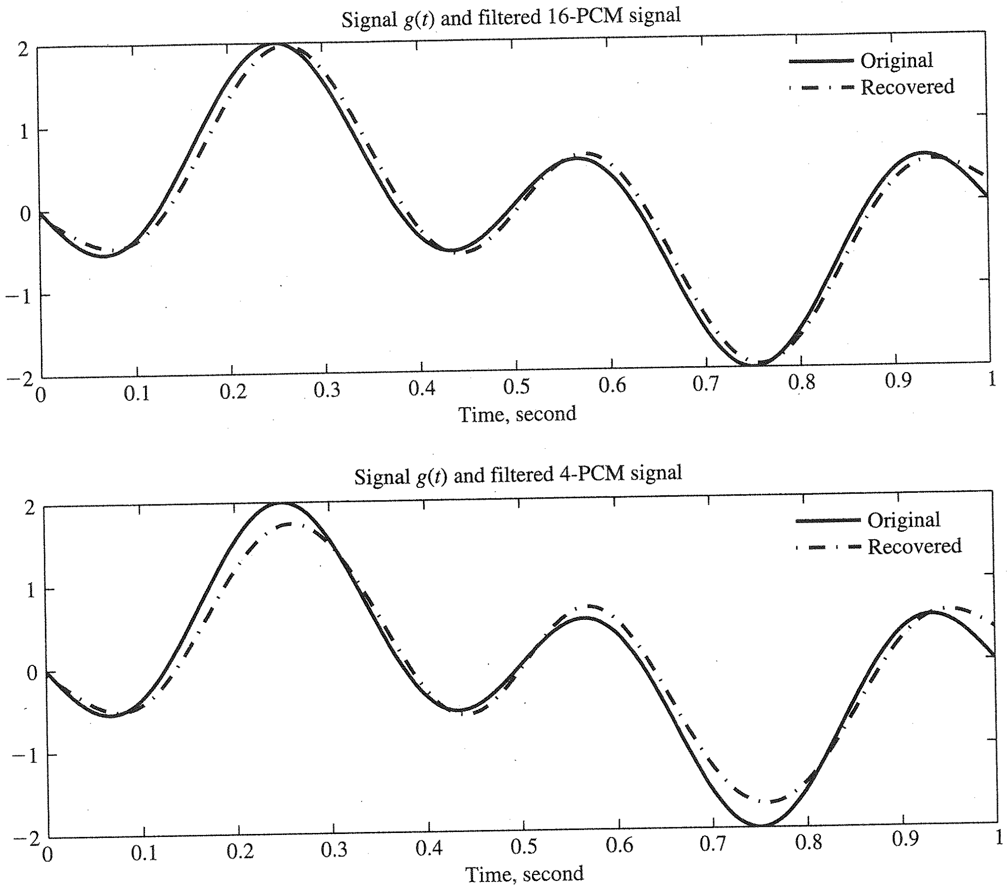


Figure 6.43 Comparison between the original signal and the PCM signals after low-pass filtering to recover the original message.

```
s_DMout(1)=-Delta/2;
for k=2:Num_it
    xvar=s_DMout(k-1);
    s_DMout(k)=xvar+Delta*sign(s_down(k-1)-xvar);
end
s_DMout=kron(s_DMout,p_zoh);
else
    warning('Error! ts/td is not an integer!');
    s_DMout=[];
end
end
```

To generate DM signals with different step sizes, we apply the same signal $g(t)$ as used in the PCM example. The MATLAB program ExDM.m applies three step sizes: $\Delta_1 = 0.2$, $\Delta_2 = 2\Delta_1$, and $\Delta_3 = 4\Delta_1$.

```
%  (ExDM.m)
% Example of sampling, quantization, and zero-order hold
clear;clf;
td=0.002;          %original sampling rate 500 Hz
```

```
t=[0:td:1.];     %time interval of 1 second
xsig=sin(2*pi*t)-sin(6*pi*t);   % 1Hz+3Hz sinusoids
Lsig=length(xsig);
ts=0.02;         %new sampling rate = 50Hz.
Nfact=ts/td;
% send the signal through a 16-level uniform quantizer
Delta1=0.2;      % First select a small Delta=0.2 in DM
s_DMout1=deltamod(xsig,Delta1,td,ts);
%   obtained the DM signal
%   plot the original signal and the DM signal in time domain
figure(1);
subplot(311);sfig1=plot(t,xsig,'k',t,s_DMout1(1:Lsig),'b');
set(sfig1,'Linewidth',2);
title('Signal {\it g}({\it t}) and DM signal')
xlabel('time (sec.)'); axis([0 1 -2.2 2.2]);
%
%  Apply DM again by doubling the Delta
Delta2=2*Delta1;           %
s_DMout2=deltamod(xsig,Delta2,td,ts);
%   obtained the DM signal
%   plot the original signal and the DM signal in time domain
subplot(312);sfig2=plot(t,xsig,'k',t,s_DMout2(1:Lsig),'b');
set(sfig2,'Linewidth',2);
title('Signal {\it g}({\it t}) and DM signal with doubled stepsize')
xlabel('time (sec.)'); axis([0 1 -2.2 2.2]);
%
Delta3=2*Delta2;           % Double the DM Delta again.
s_DMout3=deltamod(xsig,Delta3,td,ts);
%   plot the original signal and the DM signal in time domain
subplot(313);sfig3=plot(t,xsig,'k',t,s_DMout3(1:Lsig),'b');
set(sfig3,'Linewidth',2);
title('Signal {\it g}({\it t}) and DM signal with quadrupled
    stepsize')
xlabel('time (sec.)'); axis([0 1 -2.2 2.2]);
```

To illustrate the effect of DM, the resulting signals from the DM encoder are shown in Fig. 6.44. This example clearly shows that when the step size is too small (Δ_1), there is a severe overloading effect as the original signal varies so fast that the small step size is unable to catch up. Doubling the DM step size clearly solves the overloading problem in this example. However, quadrupling the step size (Δ_3) would lead to unnecessarily large quantization error. This example thus confirms our earlier analysis that a careful selection of the DM step size is critical.

REFERENCES

1. D. A. Linden, "A discussion of sampling theorem," *Proc. IRE*, vol. 47, no.7, pp. 1219–1226, July 1959.
2. H. P. Kramer, "A Generalized Sampling Theorem," *J. Math. Phys.*, vol. 38, pp. 68–72, 1959.

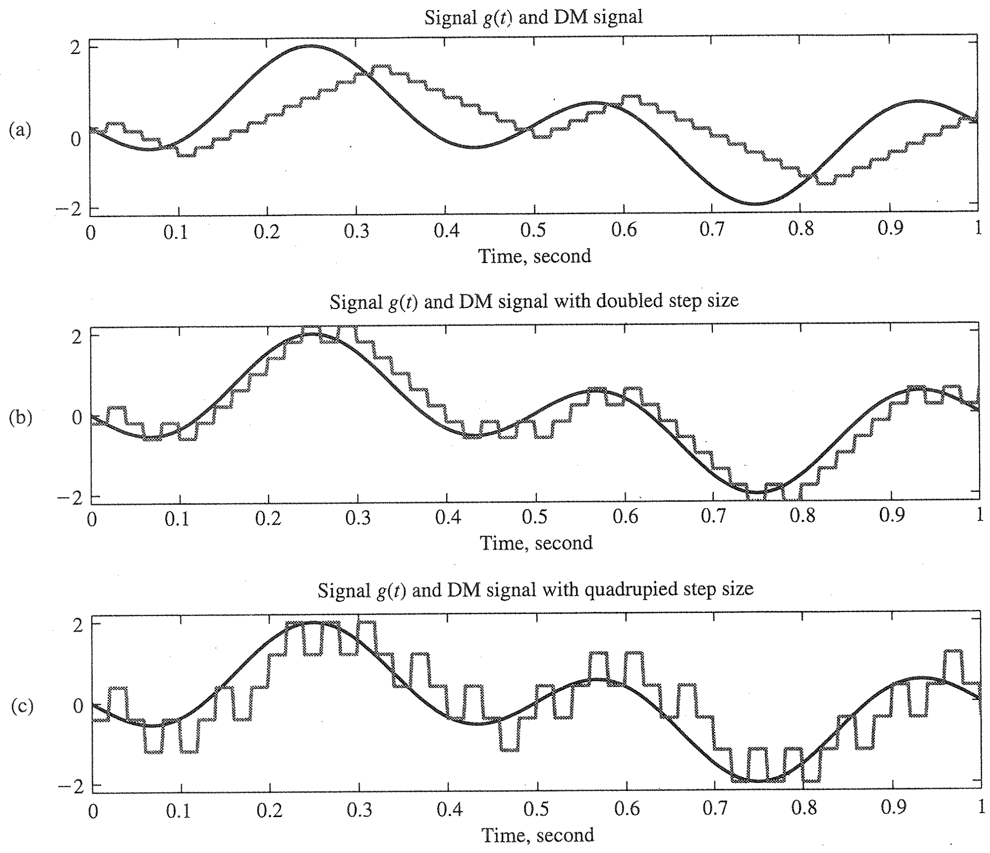


Figure 6.44 Examples of delta modulation output with three different step sizes: (a) small step size leads to overloading; (b) reasonable step size; (c) large step size causes large quantization errors.

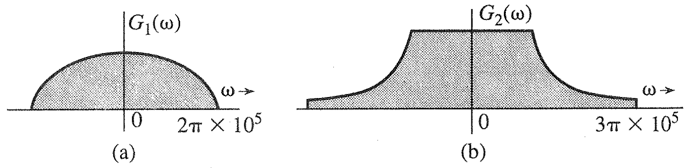
3. W. R. Bennett, *Introduction to Signal Transmission*, McGraw-Hill, New York, 1970.
4. W. S. Anglin and J. Lambek, *The Heritage of Thales*, Springer, Berlin, 1995.
5. B. Smith, "Instantaneous Companding of Quantized Signals," *Bell Syst. Tech. J.*, vol. 36, pp. 653–709, May 1957.
6. ITU-T Standard Recommendation G.711, English, 1989.
7. ITU-T Standard Recommendation G.726, English, 1990.
8. C. L. Dammann, L. D. McDaniel, and C. L. Maddox, "D-2 Channel Bank Multiplexing and Coding," *Bell Syst. Tech. J.*, vol. 51, pp. 1675–1700, Oct. 1972.
9. K. W. Cattermole, *Principles of Pulse-Code Modulation*, Ilife, England, 1969.
10. Bell Telephone Laboratories, *Transmission Systems for Communication*, 4th ed., Bell, Murray Hill, NJ, 1970.
11. E. L. Gruenberg, *Handbook of Telemetry and Remote Control*, McGraw-Hill, New York, 1967.
12. J. B. O'Neal, Jr., " Delta Modulation Quantizing Noise: Analytical and Computer Simulation Results for Gaussian and Television Input Signals," *Bell Syst. Tech. J.*, pp. 117–141, Jan. 1966.
13. F. de Jager, "Delta Modulation, a Method of PCM Transmission Using the 1-Unit Code," *Philips Res. Rep.*, no. 7, pp. 442–466, 1952.
14. A. Tomozawa and H. Kaneko, "Companded Delta Modulation for Telephone Transmission," *IEEE Trans. Commun. Technol.*, vol. CT-16, pp. 149–157, Feb. 1968.
15. B. S. Atal, "Predictive Coding of Speech Signals at Low Bit Rates," *IEEE Trans. Commun.*, vol. COMM-30, pp. 600–614, 1982.
16. J. P. Campbell and T. E. Tremain, "Voiced/Unvoiced Classification of Speech with Applications to the U.S. Government LPC-10E Algorithm," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Tokyo, pp. 473–476, 1986.
17. A. Gersho, "Advances in Speech and Audio Compression," *Proc. IEEE*, vol. 82, pp. 900–918, 1994.

18. L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, NJ, 1978.
19. Lajos Hanzo, Jason Woodward, and Clare Sommerville, *Voice Compression and Communications*, Wiley, Hoboken; NJ, 2001.
20. N. Levinson, "The Wiener rms Error Criterion in Filter Design and Prediction," *J. Math. Phys.*, vol. 25, pp. 261–278, 1947.
21. A. H. Sayed, *Fundamentals of Adaptive Filtering*, Wiley-IEEE Press, Hoboken, NJ, 2003.
22. J. Y. Stein, *Digital Signal Processing: A Computer Science Perspective*, Wiley, Hoboken, NJ, 2000.
23. K. K. Paliwal and B. W. Kleijn, "Quantization of LPC Parameters," in *Speech Coding and Synthesis,* W. B. Kleijn and K. K. Paliwal, Eds. Elsevier Science, Amsterdam, 1995.
24. T. E. Tremain, "The Government Standard Linear Predictive Coding Algorithm LPC-10," *Speech Technol.*, 40–49, 1982.
25. M. R. Schroeder and B. S. Atal, "Code-Excited Linear Prediction (CELP): High-Quality Speech at Very Low Bit Rates," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process. (ICASSP)*, vol. 10, pp. 937–940, 1985.
26. S. Mallat, "A Theory of Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. Pattern Anal. Machine Intel.*, vol. 11, pp. 674–693, 1989.
27. M. J. Smith and T. P. Barnwell, "Exact Reconstruction for Tree Structured Sub-Band Coders," *IEEE Trans. Acoustics, Speech, Signal Process.*, vol. 34, no. 3, pp. 431–441, 1986.
28. B. G. Haskel, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, Chapman & Hall, New York, 1996.
29. J. L. Mitchell, W. B. Pennebaker, C.E Fogg, and D. J. LeGall, *MPEG Video Compression Standard*, Chapman & Hall, New York, 1996.
30. ITU-T Recommendation H.263, Video Coding for Low Bit Rate Communication.

PROBLEMS

6.1-1 Figure P6.1-1 shows Fourier spectra of signals $g_1(t)$ and $g_2(t)$. Determine the Nyquist interval and the sampling rate for signals $g_1(t)$, $g_2(t)$, $g_1^2(t)$, $g_2^m(t)$, and $g_1(t)g_2(t)$.
Hint: Use the frequency convolution and the width property of the convolution.

Figure P.6.1-1



6.1-2 Determine the Nyquist sampling rate and the Nyquist sampling interval for the signals:

(a) sinc $(100\pi t)$

(b) sinc^2 $(100\pi t)$

(c) sinc $(100\pi t)$ + sinc $(50\pi t)$

(d) sinc $(100\pi t)$ + 3 sinc^2 $(60\pi t)$

(e) sinc $(50\pi t)$ sinc $(100\pi t)$

6.1-3 A signal $g(t)$ band-limited to B Hz is sampled by a periodic pulse train $p_{T_s}(t)$ made up of a rectangular pulse of width $1/8B$ second (centered at the origin) repeating at the Nyquist rate ($2B$ pulses per second). Show that the sampled signal $\overline{g}(t)$ is given by

$$\overline{g}(t) = \frac{1}{4}g(t) + \sum_{n=1}^{\infty} \frac{2}{n\pi} \sin\left(\frac{n\pi}{4}\right) g(t) \cos 4n\pi Bt$$

Show that the signal $g(t)$ can be recovered by passing $\overline{g}(t)$ through an ideal low-pass filter of bandwidth B Hz and a gain of 4.

6.1-4 A signal $g(t) = \text{sinc}^2\,(5\pi t)$ is sampled (using uniformly spaced impulses) at a rate of **(i)** 5 Hz; **(ii)** 10 Hz; **(iii)** 20 Hz. For each of the three cases:

(a) Sketch the sampled signal.

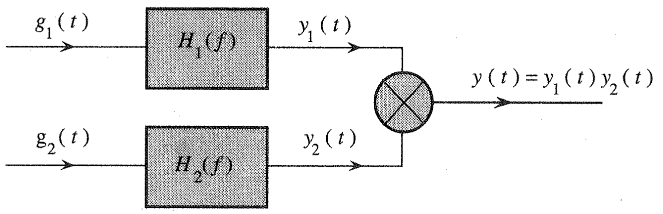
(b) Sketch the spectrum of the sampled signal.

(c) Explain whether you can recover the signal $g(t)$ from the sampled signal.

(d) If the sampled signal is passed through an ideal low-pass filter of bandwidth 5 Hz, sketch the spectrum of the output signal.

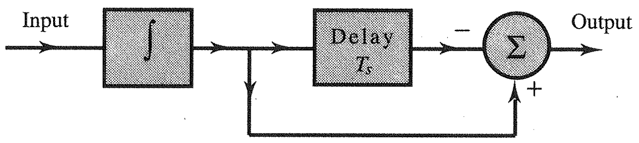
6.1-5 Signals $g_1(t) = 10^4\,\Pi(10^4 t)$ and $g_2(t) = \delta(t)$ are applied at the inputs of ideal low-pass filters $H_1(f) = \Pi(f/20{,}000)$ and $H_2(f) = \Pi(f/10{,}000)$ (Fig. P6.1-5). The outputs $y_1(t)$ and $y_2(t)$ of these filters are multiplied to obtain the signal $y(t) = y_1(t)y_2(t)$. Find the Nyquist rate of $y_1(t), y_2(t)$, and $y(t)$. Use the convolution property and the width property of convolution to determine the bandwidth of $y_1(t)y_2(t)$. See also Prob. 6.1-1.

Figure P.6.1-5



6.1-6 A zero-order hold circuit (Fig. P6.1-6) is often used to reconstruct a signal $g(t)$ from its samples.

Figure P.6.1-6



(a) Find the unit impulse response of this circuit.

(b) Find the transfer function $H(f)$ and sketch $|H(f)|$.

(c) Show that when a sampled signal $\overline{g}(t)$ is applied at the input of this circuit, the output is a staircase approximation of $g(t)$. The sampling interval is T_s.

6.1-7 **(a)** A first-order hold circuit can also be used to reconstruct a signal $g(t)$ from its samples. The impulse response of this circuit is

$$h(t) = \Delta\left(\frac{t}{2T_s}\right)$$

where T_s is the sampling interval. Consider a typical sampled signal $\overline{g}(t)$ and show that this circuit performs the linear interpolation. In other words, the filter output consists of sample tops connected by straight-line segments. Follow the procedure discussed in Sec. 6.1.1 (Fig. 6.2b).

(b) Determine the transfer function of this filter and its amplitude response, and compare it with the ideal filter required for signal reconstruction.

(c) This filter, being noncausal, is unrealizable. Suggest a modification that will make this filter realizable. How would such a modification affect the reconstruction of $g(t)$ from its samples? How would it affect the frequency response of the filter?

6.1-8 Prove that a signal cannot be simultaneously time-limited and band-limited.

Hint: Show that the contrary assumption leads to contradiction. Assume a signal simultaneously time-limited and band-limited so that $G(f) = 0$ for $|f| > B$. In this case, $G(f) = G(f)\,\Pi(f/2B')$ for $B' > B$. This means that $g(t)$ is equal to $g(t) * 2B'\text{sinc}\,(2\pi B' t)$. Show that the latter cannot be time-limited.

6.2-1 The American Standard Code for Information Interchange (ASCII) has 128 characters, which are binary-coded. If a certain computer generates 100,000 characters per second, determine the following:

(a) The number of bits (binary digits) required per character.

(b) The number of bits per second required to transmit the computer output, and the minimum bandwidth required to transmit this signal.

(c) For single error detection capability, an additional bit (parity bit) is added to the code of each character. Modify your answers in parts **(a)** and **(b)** in view of this information.

6.2-2 A compact disc (CD) records audio signals digitally by using PCM. Assume that the audio signal bandwidth equals 15 kHz.

(a) If the Nyquist samples are uniformly quantized into $L = 65,536$ levels and then binary-coded, determine the number of binary digits required to encode a sample.

(b) If the audio signal has average power of 0.1 watt and peak voltage of 1 volt. Find the resulting signal-to-quantization-noise ratio (SQNR) of the uniform quantizer output in part **(a)**.

(c) Determine the number of binary digits per second (bit/s) required to encode the audio signal.

(d) For practical reasons discussed in the text, signals are sampled at a rate well above the Nyquist rate. Practical CDs use 44,100 samples per second. If $L = 65,536$, determine the number of bits per second required to encode the signal, and the minimum bandwidth required to transmit the encoded signal.

6.2-3 A television signal (video and audio) has a bandwidth of 4.5 MHz. This signal is sampled, quantized, and binary coded to obtain a PCM signal.

(a) Determine the sampling rate if the signal is to be sampled at a rate 20% above the Nyquist rate.

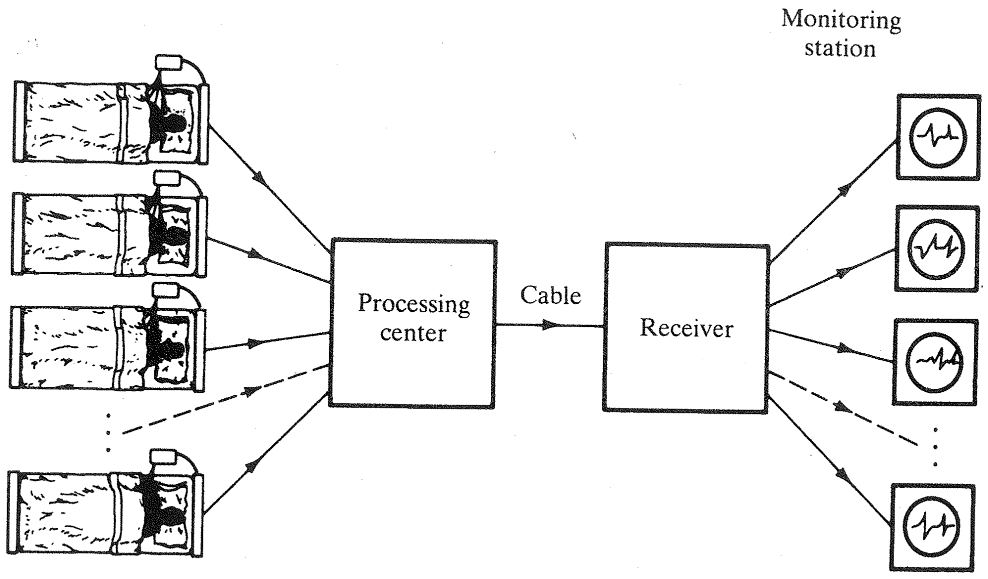
(b) If the samples are quantized into 1024 levels, determine the number of binary pulses required to encode each sample.

(c) Determine the binary pulse rate (bits per second) of the binary-coded signal, and the minimum bandwidth required to transmit this signal.

6.2-4 Five telemetry signals, each of bandwidth 240 Hz, are to be transmitted simultaneously by binary PCM. The signals must be sampled at least 20% above the Nyquist rate. Framing and synchronizing requires an additional 0.5% extra bits. A PCM encoder is used to convert these signals before they are time-multiplexed into a single data stream. Determine the minimum possible data rate (bits per second) that must be transmitted, and the minimum bandwidth required to transmit the multiplex signal.

6.2-5 It is desired to set up a central station for simultaneous monitoring of the electrocardiograms (ECGs) of 10 hospital patients. The data from the 10 patients are brought to a processing center over wires and are sampled, quantized, binary-coded, and time-division-multiplexed. The multiplexed data are now transmitted to the monitoring station (Fig. P6.2-5). The ECG signal bandwidth is 100 Hz. The maximum acceptable error in sample amplitudes is 0.25% of the peak signal amplitude. The sampling rate must be at least twice the Nyquist rate. Determine the minimum cable bandwidth needed to transmit these data.

Figure P.6.2-5

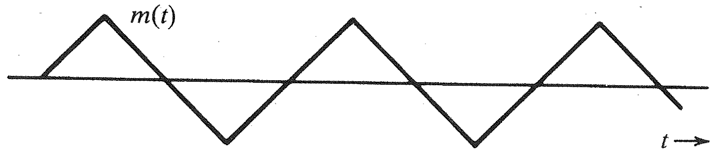


6.2-6 A message signal $m(t)$ is transmitted by binary PCM without compression. If the SQNR is required to be at least 47 dB, determine the minimum value of $L = 2^n$ required, assuming that $m(t)$ is sinusoidal. Determine the actual SQNR obtained with this minimum L.

6.2-7 Repeat Prob. 6.2-6 for $m(t)$ shown in Fig. P6.2-7.

Hint: The power of a periodic signal is its energy averaged over one cycle. In this case, however, because the signal amplitude takes on the same values every quarter cycle, the power can also be found by averaging the signal energy over a quarter cycle.

Figure P.6.2-7



6.2-8 For a PCM signal, determine L if the compression parameter $\mu = 100$ and the minimum SNR required is 45 dB. Determine the output SQNR with this value of L. Remember that L must be a power of 2, that is, $L = 2^n$ for a binary PCM.

6.2-9 A signal band-limited to 1 MHz is sampled at a rate 50% higher than the Nyquist rate and quantized into 256 levels by using a μ-law quantizer with $\mu = 255$.

(a) Determine the signal-to-quantization-noise ratio.

(b) The SQNR (the received signal quality) found in part **(a)** was unsatisfactory. It must be increased at least by 10 dB. Would you be able to obtain the desired SQNR without increasing the transmission bandwidth if it was found that a sampling rate 20% above the Nyquist rate is adequate? If so, explain how. What is the maximum SQNR that can be realized in this way?

6.2-10 The output SQNR of a 10-bit PCM was found to be insufficient at 30 dB. To achieve the desired SNR of 42 dB, it was decided to increase the number of quantization levels L. Find the fractional increase in the transmission bandwidth required for this increase in L.

6.4-1 In a certain telemetry system, there are four analog signals $m_1(t)$, $m_2(t)$, $m_3(t)$, and $m_4(t)$. The bandwidth of $m_1(t)$ is 3.6 kHz, but for each of the remaining signals it is 1.4 kHz. These signals are to be sampled at rates no less than their respective Nyquist rates and are to be word-by-word multiplexed. This can be achieved by multiplexing the PAM samples of the four signals and then binary coding the multiplexed samples (as in the case of the PCM T1 carrier in Fig. 6.20a). Suggest a suitable multiplexing scheme for this purpose. What is the commutator frequency (in rotations per second)? *Note:* In this case you may have to sample some signal(s) at rates higher than their Nyquist rate(s).

6.4-2 Repeat Prob. 6.4-1 if there are four signals $m_1(t)$, $m_2(t)$, $m_3(t)$, and $m_4(t)$ with bandwidths 1200, 700, 300, and 200 Hz, respectively.

Hint: First multiplex m_2, m_3, and m_4 and then multiplex this composite signal with $m_1(t)$.

6.4-3 A signal $m_1(t)$ is band-limited to 3.6 kHz, and the three other signals $m_2(t)$, $m_3(t)$, and $m_4(t)$ are band-limited to 1.2 kHz each. These signals are sampled at the Nyquist rate and binary coded using 512 levels ($L = 512$). Suggest a suitable bit-by-bit multiplexing arrangement (as in Fig. 6.12). What is the commutator frequency (in rotations per second), and what is the output bit rate?

6.7-1 In a single-integration DM system, the voice signal is sampled at a rate of 64 kHz, similar to PCM. The maximum signal amplitude is normalized as $A_{\max} = 1$.

(a) Determine the minimum value of the step size σ to avoid slope overload.

(b) Determine the granular noise power N_o if the voice signal bandwidth is 3.4 kHz.

(c) Assuming that the voice signal is sinusoidal, determine S_o and the SNR.

(d) Assuming that the voice signal amplitude is uniformly distributed in the range $(-1, 1)$, determine S_o and the SNR.

(e) Determine the minimum transmission bandwidth.

7 PRINCIPLES OF DIGITAL DATA TRANSMISSION

Throughout most of the twentieth century, a significant percentage of communication systems was in analog form. However, by the end of the 1990s, the digital format began to dominate most applications. One does not need to look hard to witness the continuous migration from analog to digital communications: from audiocassette tape to MP3 and CD, from NTSC analog TV to digital HDTV, from traditional telephone to VoIP, and from VHS videotape to DVD. In fact, even the last analog refuge of broadcast radio is facing a strong digital competitor in the form of satellite radio. Given the dominating importance of digital communication systems in our lives today, it is never too early to study the basic principles and various aspects of digital data transmission, as we will do in this chapter.

This chapter deals with the problems of transmitting digital data over a channel. Hence, the starting messages are assumed to be digital. We shall begin by considering the binary case, where the data consist of only two symbols: **1** and **0**. We assign a distinct waveform (pulse) to each of these two symbols. The resulting sequence of these pulses is transmitted over a channel. At the receiver, these pulses are detected and are converted back to binary data (**1**s and **0**s).

7.1 DIGITAL COMMUNICATION SYSTEMS

A digital communication system consists of several components, as shown in Fig. 7.1. In this section, we conceptually outline their functionalities in the communication systems. The details of their analysis and design will be given in dedicated sections later in this chapter.

7.1.1 Source

The input to a digital system takes the form of a sequence of digits. The input could be the output from a data set, a computer, or a digitized audio signal (PCM, DM, or LPC), digital facsimile or HDTV, or telemetry data, and so on. Although most of the discussion in this chapter is confined to the binary case (communication schemes using only two symbols), the more general case of M-ary communication, which uses M symbols, will also be discussed in Secs. 7.7 and 7.9.