off increased transmission bandwidth for improved system performance in the presence of noise.

3. *Modulation permits the use of multiple-access techniques.*
A cellular radio channel represents a major capital investment and must therefore be deployed in a cost-effective manner, permitting mobile users access to the channel. *Multiple access* is a signal-processing operation that makes this possible. In particular, it permits the simultaneous transmission of information-bearing signals from a number of independent users over the channel and on to their respective destinations.

- In wireless communications, the carrier, denoted by $c(t)$, is typically sinusoidal and is written as

$$c(t) = A_c \cos(2\pi f_c t + \theta) \tag{3.1}$$

where A_c is the amplitude, f_c is the frequency, and θ is the phase. With these three carrier parameters individually available for modulation, we have three basic methods of modulation whose specific descriptions depend on whether the information-bearing signal is of an analog or a digital nature. These two families of modulation are discussed in what follows.

3.2.1 Linear and Nonlinear Modulation Processes

Figure 3.3 shows the block diagram of a modulator supplied with a sinusoidal carrier $c(t)$. The modulating signal, acting as input, is denoted by $m(t)$. The modulated signal, acting as output, is denoted by $s(t)$. The input–output relation of the modulator is governed by the manner in which the output $s(t)$ depends on the input $m(t)$. On this basis, we may classify the modulation process as one of two basic types: linear and nonlinear.

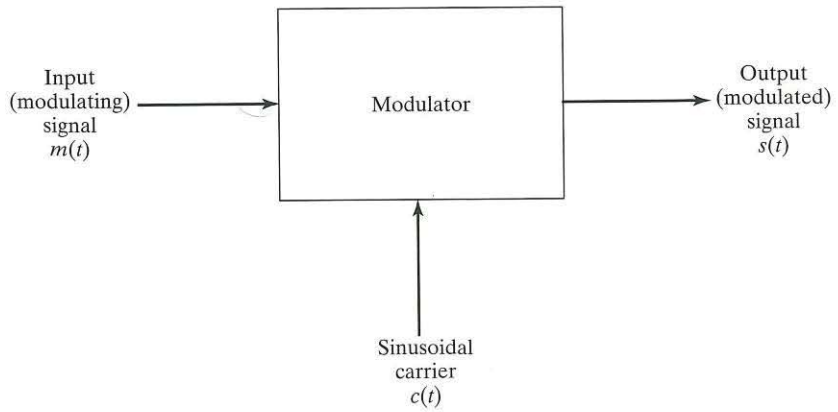

FIGURE 3.3 Block diagram of modulator.

The modulation process is said to be *linear* if the input-output relation of the modulator satisfies the *principle of superposition*. According to this principle, the modulation process, or equivalently, the modulator, satisfies two conditions:

1. The output of the modulator produced by a number of inputs applied simultaneously is equal to the sum of the outputs that result when the inputs are applied one at a time.
2. If the input is scaled by a certain factor, the output of the modulator is scaled by exactly the same factor.

The modulation process, or equivalently, the modulator, is said to be *nonlinear* if the principle of superposition is violated in part or in full.

The linearity or nonlinearity of a modulation process has important consequences, in both theoretical as well as practical terms, as we shall see in the remainder of the chapter.

3.2.2 Analog and Digital Modulation Techniques

Another way of classifying the modulation process is on the basis of whether the message signal $m(t)$ is derived from an analog or a digital source of information. In the analog case, the message signal $m(t)$ is a continuous function of time t. Consequently, the modulated signal $s(t)$ is, likewise, a continuous function of time t. It is for this reason that a modulation process of the analog kind is commonly referred to as *continuous-wave (CW) modulation*.

In the digital case, by contrast, the modulated signal $s(t)$ may exhibit discontinuities at the instants of time at which the message signal $m(t)$ switches from symbol 1 to symbol 0 or vice versa. Note, however (as we will find out later on in the chapter), that under certain conditions it is possible for the modulated signal to maintain continuity even at the instants of switching.

In other words, we may distinguish between analog and digital modulation processes as follows:

- All analog modulated signals are continuous functions of time.
- Digital modulated signals can be continuous or discontinuous functions of time, depending on how the modulation process is performed.

3.2.3 Amplitude and Angle Modulation Processes

Yet another way of classifying modulation processes is on the basis of which parameter of the sinusoidal carrier $c(t)$ is varied in accordance with the message signal $m(t)$. Accordingly, we speak of two kinds of modulation:

1. *Amplitude modulation*, in which the amplitude of the carrier, A_c, is varied linearly with the message signal $m(t)$.

2. *Angle modulation*, in which the angle of the carrier, namely,

$$\psi(t) = 2\pi f_c t + \theta \tag{3.2}$$

is varied linearly with the message signal $m(t)$.

Angle modulation may itself be classified into two kinds:

2.1. *Frequency modulation*, in which the frequency of the carrier, f_c, is varied linearly with the message signal $m(t)$.

2.2. *Phase modulation*, in which the phase of the carrier, θ, is varied linearly with the message signal $m(t)$.

In historical terms, the design of communication systems was dominated by analog modulation techniques. Nowadays, however, we find that the use of digital modulation techniques is the method of choice, due to the pervasive use of silicon chips and digital signal-processing techniques. For this reason, the focus in what follows is on digital modulation techniques.

3.3 LINEAR MODULATION TECHNIQUES

3.3.1 Amplitude Modulation[1]

By definition, amplitude modulation (AM), produced by an analog message signal $m(t)$, is described by

$$s(t) = A_c(1 + k_a m(t))\cos(2\pi f_c t) \tag{3.3}$$

where k_a is the *sensitivity* of the amplitude modulator. For convenience of presentation, we have set the carrier phase θ equal to zero, as it has no bearing whatsoever on the transmission of information.

Appendix A briefly reviews *Fourier theory*, which is basic to the spectral analysis of signals. In light of the theory presented therein, we may portray the spectral characteristics of amplitude modulation as illustrated in Fig. 3.4. The figure clearly shows that the bandwidth of the AM signal $s(t)$ is $2W$, where W is the bandwidth of $m(t)$ itself. Most important, except for the frequency shift in the spectrum of the message signal $m(t)$, denoted by $M(f)$, and the retention of the carrier, exemplified by the impulses at $\pm f_c$, amplitude modulation has no other effect on the spectrum $S(f)$ of $s(t)$.

Problem 3.1 Show that amplitude modulation is a nonlinear process, as it violates the principle of superposition. ■

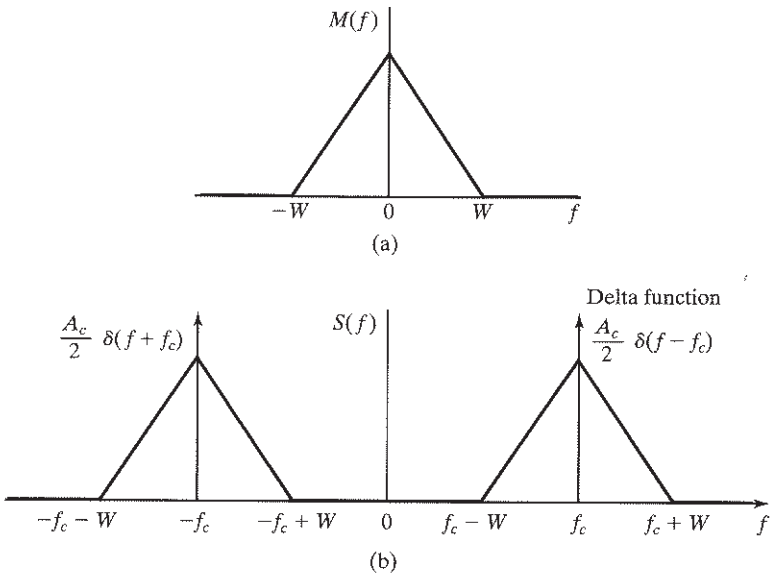

FIGURE 3.4 (a) Message spectrum. (b) Spectrum of corresponding AM signal.

Note, however, that the violation of the principle of superposition described in Problem 3.1 is of a mild sort which permits the application of the Fourier transform to an AM signal, as described in Fig. 3.4.

Another important point to note is that, insofar as information transmission is concerned, retention of the carrier in the composition of the AM signal represents a loss of transmitted signal power. To mitigate this shortcoming of amplitude modulation, the carrier is suppressed, in which case the process is referred to as *double sideband–suppressed carrier (DSB–SC) modulation*. Correspondingly, the DSB–SC modulated signal is defined simply as the product of the message signal $m(t)$ and the carrier $c(t)$; that is,

$$\begin{aligned} s(t) &= c(t)m(t) \\ &= A_c m(t)\cos(2\pi f_c t) \end{aligned} \tag{3.4}$$

Figure 3.5 depicts the spectrum of the new $s(t)$. Comparing this spectrum with that of Fig. 3.4, we see clearly that the absence of the delta functions at $\pm f_c$ is testimony to the suppression of the carrier in Eq. (3.4). Nevertheless, the AM signal of Eq. (3.3) and the DSB–SC modulated signal of Eq. (3.4) do share a common feature: They both require the use of a transmission bandwidth equal to twice the message bandwidth, namely, $2W$.

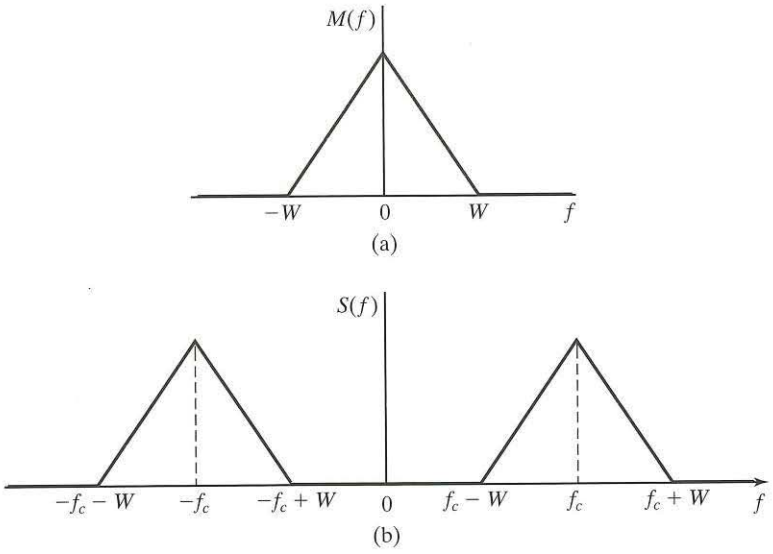

FIGURE 3.5 (a) Message spectrum. (b) Spectrum of corresponding DSB–SC modulated signal.

Problem 3.2 Consider the sinusoidal modulating signal

$$m(t) = A_m \cos(2\pi f_m t)$$

Show that the use of DSB-SC modulation produces a pair of side frequencies, one at $f_c + f_m$ and the other at $f_c - f_m$, where f_c is the carrier frequency. What is the condition that the modulator has to satisfy in order to make sure that the two side-frequencies do not overlap?
Ans. $f_c > f_m$. ■

3.3.2 Binary Phase-Shift Keying

Consider next the case of digital modulation in which the modulating signal is in the form of a binary data stream. Let $p(t)$ denote the *basic pulse* used in the construction of this stream. Let T denote the *bit duration* (i.e., the duration of binary symbol 1 or 0). Then the binary data stream, consisting of a sequence of 1's and 0's, is described by

$$m(t) = \sum_k b_k p(t - kT) \tag{3.5}$$

where

$$b_k = \begin{cases} +1 & \text{for binary symbol 1} \\ -1 & \text{for binary symbol 0} \end{cases} \tag{3.6}$$

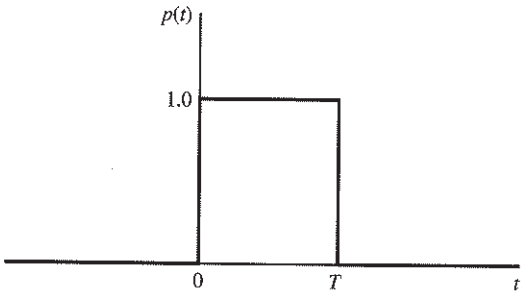

FIGURE 3.6 Rectangular pulse.

For example, in the case of a rectangular pulse we have

$$p(t) = \begin{cases} +1 & \text{for } 0 \le t \le T \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

which is depicted in Fig. 3.6.

In binary phase-shift keying (BPSK), the simplest form of digital phase modulation, the binary symbol 1 is represented by setting the carrier phase $\theta(t) = 0$ radians, and the binary symbol 0 is represented by setting $\theta(t) = \pi$ radians. Correspondingly,

$$s(t) = \begin{cases} A_c \cos(2\pi f_c t) & \text{for binary symbol 1} \\ A_c \cos(2\pi f_c t + \pi) & \text{for binary symbol 0} \end{cases} \tag{3.8}$$

Recognizing that

$$\cos(\theta(t) + \pi) = -\cos(\theta(t)) \text{ for all time } t,$$

we may rewrite Eq. (3.8) as

$$s(t) = \begin{cases} A_c \cos(2\pi f_c t) & \text{for symbol 1} \\ -A_c \cos(2\pi f_c t) & \text{for symbol 0} \end{cases} \tag{3.9}$$

In light of Eqs. (3.6), (3.7), and (3.9), we may express the BPSK signal in the compact form

$$s(t) = c(t)m(t) \tag{3.10}$$

where $m(t)$ is itself defined by Eq. (3.5). Most important, Eq. (3.10) shows that BPSK is another example of linear modulation.

Problem 3.3 Consider a binary data stream $m(t)$ in the form of a square wave with amplitudes ±1, centered on the origin. Determine the spectrum of the BPSK signal obtained by multiplying $m(t)$ by a sinusoidal carrier whose frequency is ten times that of the fundamental frequency of the square wave.
Ans.

$$s(t) = \sum_{k=1,3,5,\ldots} \frac{\sin(k\pi/2)}{(k\pi/2)} \cos\left(2\pi \frac{kf_c}{10} t\right) \cos(2\pi f_c t)$$
$$= \frac{1}{2} \sum_{k=1,3,5,\ldots} \frac{\sin(k\pi/2)}{(k\pi/2)} \left[\cos\left(2\pi t\left(f_c + \frac{kf_c}{10}\right)\right) \cos\left(2\pi t\left(f_c - \frac{kf_c}{10}\right)\right)\right]$$

The spectrum of the BPSK signal consists of side-frequencies at $f_c \pm \frac{kf_c}{10}$ *with decreasing amplitude in accordance with* $\frac{1}{2}\sin(k\pi/2)/(k\pi/2)$, *where* $k = 1, 3, 5\ldots$. ■

3.3.3 Quadriphase-Shift Keying

As with DSB–SC modulation, BPSK requires a transmission bandwidth twice the message bandwidth. Now, channel bandwidth is a primary resource that should be conserved, particularly in wireless communications. How then can we retain the property of linearity that characterizes BPSK, yet accommodate the transmission of a digital phase-modulated signal over a channel whose bandwidth is equal to the bandwidth of the incoming binary data stream? The answer to this fundamental question lies in the use of a digital modulation technique known as *quadriphase-shift keying* (QPSK).

To proceed with a description of QPSK, suppose the incoming binary data stream is first demultiplexed into two substreams, $m_1(t)$ and $m_2(t)$. Next, note that, as the name implies, the phase of the carrier in QPSK assumes one of four equally spaced values, depending on the composition of each *dibit*, or group of two adjacent bits, in the original binary data stream. For example, we may use 0, $\pi/2$, π, and $3\pi/2$ radians as the set of four values available for phase-shift keying the carrier. Specifically, the values 0 and π radians are used to phase-shift key one of the two substreams, $m_1(t)$, and the remaining values, $\pi/2$ and $3\pi/2$ radians, are used to phase-shift key the other substream, $m_2(t)$.

Accordingly, we can formulate the block diagram for the QPSK modulator as shown in Fig. 3.7. On the basis of this structure, we can then describe the QPSK modulator as *the parallel combination of two BPSK modulators that operate in phase quadrature with respect to each other*. By phase quadrature, we mean the arrangement of phases such that the phase of the carrier in the lower path of the modulator is 90° out of phase with respect to the carrier in the upper path.

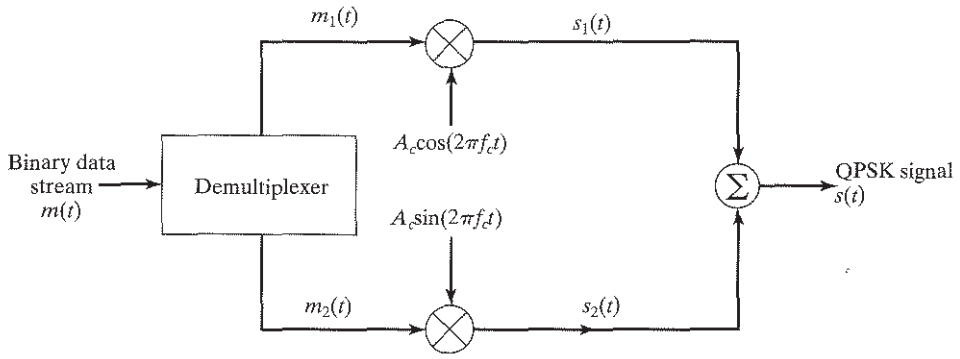

FIGURE 3.7 Block diagram of a QPSK generator, using a phase-quadrature pair of carriers $A_c\cos(2\pi f_c t)$ and $A_c\sin(2\pi f_c t)$.

As remarked previously, $m_1(t)$ and $m_2(t)$ denote the two binary substreams that result from demultiplexing of the binary data stream $m(t)$. Extending the mathematical description of Eq. (3.5) to the situation at hand, we may express the corresponding descriptions of binary substreams $m_1(t)$ and $m_2(t)$ as follows:

$$m_i(t) = \sum_k b_{k,i}\, p(t - kT) \quad \text{for } i = 1, 2 \tag{3.11}$$

For $i = 1,2$ we have

$$b_{k,i} = \begin{cases} +1 & \text{for symbol 1} \\ -1 & \text{for symbol 0} \end{cases} \tag{3.12}$$

and for the case of a rectangular pulse,

$$p(t) = \begin{cases} +1 & \text{for } 0 \le t \le 2T \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

Then the BPSK signal produced in the upper path of Fig. 3.7 is described by

$$s_1(t) = A_c m_1(t)\cos(2\pi f_c t) \tag{3.14}$$

The BPSK signal produced in the lower path of Fig. 3.7 is described by

$$s_2(t) = A_c m_2(t)\sin(2\pi f_c t) \tag{3.15}$$

The QPSK signal is obtained by adding these two BPSK signals:

$$\begin{aligned} s(t) &= s_1(t) + s_2(t) \\ &= A_c m_1(t)\cos(2\pi f_c t) + A_c m_2(t)\sin(2\pi f_c t) \end{aligned} \tag{3.16}$$

Since both BPSK signals $s_1(t)$ and $s_2(t)$ are linear, the QPSK signal $s(t)$ is likewise linear.

The transmission bandwidth requirement of the QPSK signal $s(t)$ is the same as that of the original binary data stream $m(t)$. We justify this important property of QPSK signals as follows:

- The original binary data stream $m(t)$ is based on bits, whereas the substreams $m_1(t)$ and $m_2(t)$ are based on dibits. The symbol duration of both $m_1(t)$ and $m_2(t)$ is therefore twice the symbol duration of $m(t)$.
- The bandwidth of a rectangular pulse is inversely proportional to the duration of the pulse. Hence, the bandwidth of both $m_1(t)$ and $m_2(t)$ is one-half that of $m(t)$.
- The BPSK signals $s_1(t)$ and $s_2(t)$ have a common transmission bandwidth equal to twice that of $m_1(t)$ or $m_2(t)$.
- The QPSK signal $s(t)$ has the same transmission bandwidth as $s_1(t)$ or $s_2(t)$.
- Hence, the transmission bandwidth of the QPSK signal is the same as that of the original binary data stream $m(t)$.

EXAMPLE 3.1 QPSK Waveform

Figure 3.8(a) depicts the waveform of a QPSK signal for which the carrier phase assumes one of the four possible values 0°, 90°, 180°, and 270°. Moreover, the waveform is the result of transmitting a binary data stream with the following composition over the interval $0 \le t \le 10T$:

- The input dibit (i.e., the pair of adjacent bits in $m(t)$) changes in going from the interval $0 \le t \le 2T$ to the next interval $2T \le t \le 4T$.
- In going from the interval $2T \le t \le 4T$ to the next interval $4T \le t \le 6T$, there is no change in the input dibit.
- The input dibit changes again in going from the interval $4T \le t \le 6T$ to $6T \le t \le 8T$.
- The input dibit is unchanged in going from the interval $6T \le t \le 8T$ to the next interval $8T \le t \le 10T$. ■

Examining the waveform of Fig. 3.8(a), we see that QPSK signals exhibit two unique properties:

1. The carrier amplitude is maintained constant.
2. The carrier phase undergoes jumps of 0°, ±90° or ±180° every $2T$ seconds, where T is the bit duration of the incoming binary data stream.

3.3.4 Offset Quadriphase-Shift Keying

Property 2 of the conventional QPSK signal, namely, the fact that the carrier phase may jump by ±90° or ±180° every two bit durations can be of particular concern when the QPSK signal is filtered during the course of transmission over a wireless channel.

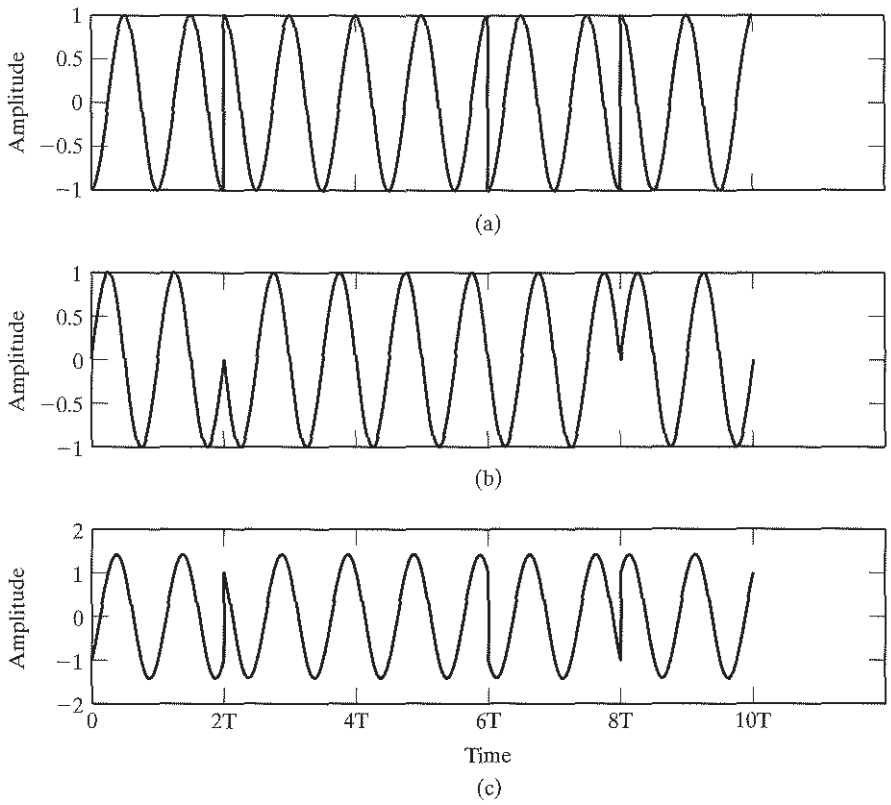

FIGURE 3.8 Waveforms of (a) conventional QPSK (b) offset QPSK, and (c) $\pi/4$-shifted QPSK.

The filtering action can, in turn, cause the carrier amplitude (i.e., the envelope of the QPSK signal) to fluctuate, thereby making the receiver produce additional symbol errors over and above those due to channel noise.

The extent of amplitude fluctuations exhibited by conventional QPSK signals may be reduced by using *offset quadriphase-shift keying (OQPSK)*, which is also referred to as *staggered QPSK*. In this variant of QPSK, the second substream $m_2(t)$, multiplied by the 90° phase-shifted carrier $A_c\sin(2\pi f_c t)$, is delayed (i.e., offset) by a bit duration T with respect to the first substream $m_1(t)$, multiplied by the carrier $A_c\cos(2\pi f_c t)$. Accordingly, unlike the phase transitions in conventional QPSK, the phase transitions likely to occur in offset QPSK are confined to 0°, ±90°, as illustrated in the next example. However, the ±90° phase jumps in OQPSK occur twice as frequently, but with a reduced range of amplitude fluctuations, compared with those of conventional QPSK. Since, in addition to ±90° phase jumps, there are ±180° phase jumps in conventional QPSK, we usually find that amplitude fluctuations in OQPSK due to filtering have a smaller amplitude than in conventional QPSK.

EXAMPLE 3.2 OQPSK Waveform

Part (b) of Fig. 3.8 depicts the waveform of the OQPSK for the same binary data stream responsible for generating the conventional QPSK waveform depicted in part (a) of the figure. Here again, we see that the carrier amplitude of OQPSK is maintained constant. However, unlike the carrier phase in the conventional QPSK of Fig. 3.8(a), the carrier phase of the OQPSK shown in Fig. 3.8(b) has jumps of only $\pm 90°$. ■

3.3.5 $\pi/4$-Shifted Quadriphase-Shift Keying

As mentioned previously, ordinarily the carrier phase of a conventional QPSK signal may reside in one of two possible discrete settings:

1. $0, \pi/2, \pi$, or $3\pi/2$ radians.
2. $\pi/4, 3\pi/4, 5\pi/4$, or $7\pi/4$ radians.

These two phase settings are shifted by $\pi/4$ radians relative to each other. The QPSK waveform depicted in Fig. 3.8(a) follows setting 1. In another variant of QPSK known as *$\pi/4$-shifted QPSK*, the carrier phase used for the transmission of successive dibits is alternatively picked from settings 1 and 2.

An attractive feature of $\pi/4$-shifted QPSK signals is that amplitude fluctuations due to filtering are significantly reduced, compared with their frequency of occurrence in conventional QPSK signals. Thus, the use of $\pi/4$-shifted QPSK provides the bandwidth efficiency of conventional QPSK, but with a reduced range of amplitude fluctuations. The reduced amplitude fluctuations become important when the transmitter includes a slightly nonlinear amplifier, as we shall see in Section 3.9. Indeed, it is for this reason that $\pi/4$-shifted QPSK has been adopted in the North American digital cellular time-division multiple access (TDMA) standard, IS-54 as well as the Japanese digital cellular standard.[2]

EXAMPLE 3.3 $\pi/4$-Shifted QPSK Waveform

Figure 3.8(c) depicts the $\pi/4$-shifted QPSK waveform produced by the same binary data stream used to generate the conventional QPSK waveform of Fig. 3.8(a). Comparing these two waveforms, we see that (1) the phase jumps in the $\pi/4$-shifted QPSK are restricted to $\pm\pi/4$ and $\pm 3\pi/4$ radians and (2) the $\pm\pi$ phase jumps of QPSK are eliminated—hence the advantage of $\pi/4$-shifted QPSK over conventional QPSK. However, this advantage is attained at the expense of increased complexity. ■

3.4 PULSE SHAPING

The pulse defined in Eq. (3.7) for representing binary symbol 1 or 0 is rectangular in shape. From a practical perspective, the use of a rectangular pulse shape is undesirable for two fundamental reasons:

1. The spectrum (i.e., Fourier transform) of a rectangular pulse is *infinite in extent*. Correspondingly, the spectrum of a digitally modulated signal based on the use

of a rectangular pulse is shifted by an amount equal to the carrier frequency but, most important, its frequency content is also infinite in extent. However, a wireless channel is bandlimited, which means that the transmission of such a digitally modulated signal over the channel will introduce *signal distortion* at the receiving end.

2. A wireless channel has *memory* due to the presence of multipath. Consequently, the transmission of a digitally modulated signal over the channel results in a special form of interference called *intersymbol interference* (ISI), which refers to interference between consecutive signaling symbols of the transmitted data sequence.

Now, in a wireless communication system, the goal is to accommodate the largest possible number of users in a prescribed channel bandwidth. To satisfy this important requirement we must use a *premodulation filter*, whose objective is that of *pulse shaping*. Specifically, the shape of the basic pulse used to generate the digitally modulated signal must be designed so as to overcome the signal distortion and ISI problems cited under points 1 and 2.

The design criteria for pulse shaping are covered by the fundamental theoretical work of Nyquist.[3] Let $P(f)$ denote the overall frequency response made up of three components: the transmit filter, the channel, and the receive filter. According to Nyquist, the effect of intersymbol interference can be reduced to zero by shaping the overall frequency response $P(f)$ so as to consist of a *flat* portion and sinusoidal *rolloff* portions, as illustrated in Fig. 3.9(a). Specifically, for a data rate of R bits/second, the channel bandwidth may extend from the minimum value $W = R/2$ to an adjustable value from W to $2W$ by defining $P(f)$ as follows:

$$P(f) = \begin{cases} \dfrac{1}{2W} & 0 \le |f| \le f_1 \\ \dfrac{1}{4W}\left[1 + \cos\left(\dfrac{\pi}{2W\rho}(|f| - W(1-\rho))\right)\right] & f_1 \le |f| < 2W - f_1 \\ 0 & |f| \ge 2W - f_1 \end{cases} \tag{3.17}$$

The frequency parameter f_1 and bandwidth W are related by the parameter

$$\rho = 1 - \frac{f_1}{W} \tag{3.18}$$

Called the *roll-off factor*, ρ indicates the excess bandwidth over the *ideal* solution corresponding to $\rho = 0$. An important characteristic of the frequency response $P(f)$ is that its inverse Fourier transform, denoted by $p(t)$ (i.e., the overall impulse response of the transmit filter, the channel, and the receive filter) has the value of unity at the current signaling instant (i.e., $p(0) = 1$) and zero crossings at all other consecutive signaling instants (i.e., $p(nT) = 0$ for nonzero integer n), as shown in Fig. 3.9(b). The zero crossings of the impulse response $p(t)$ ensure that the ISI problem is reduced to zero. The frequency response of Fig. 3.9(a) is called the *raised-cosine (RC) spectrum*, so called because of its trigonometric form as defined in Eq. (3.17).

The variability of the rolloff factor ρ over the range (0,1) allows the designer to tradeoff transmitted signal bandwidth for robustness of the pulse shape.

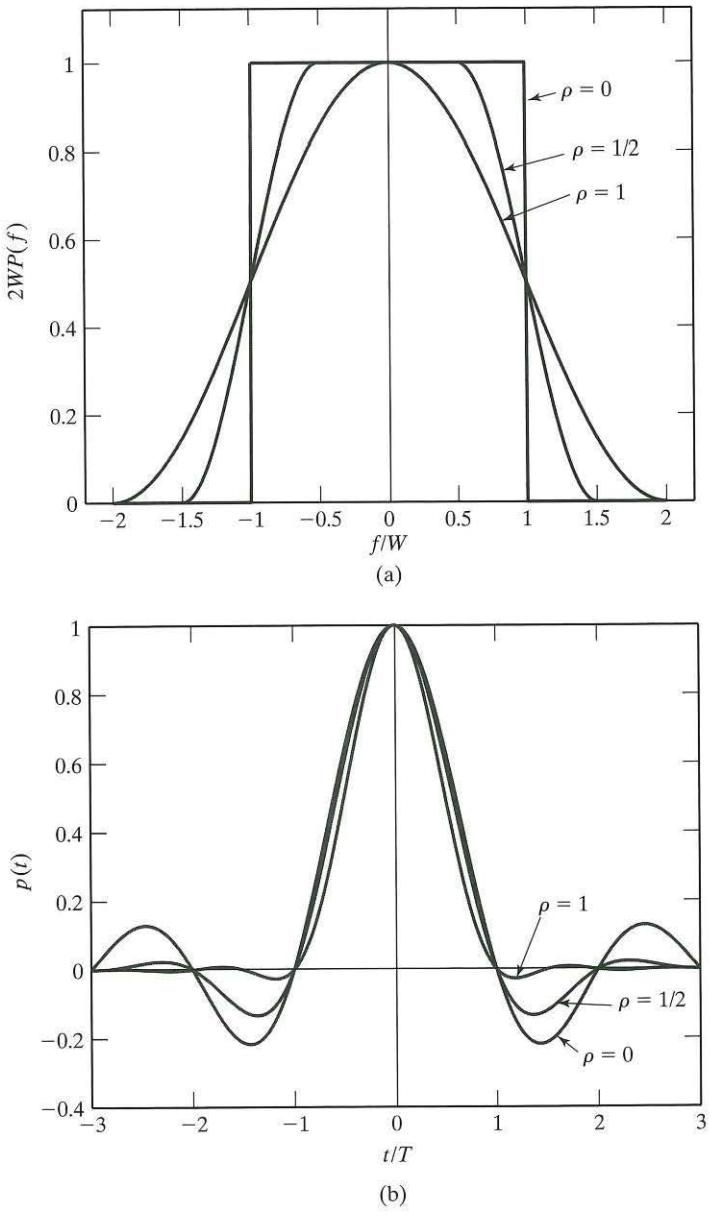

FIGURE 3.9 (a) Frequency response of the raised cosine spectrum for varying roll-off rates. (b) Impulse response of the Nyquist shaping filter (i.e., inverse Fourier transform of the spectrum plotted in part (a) for varying roll-off rates).

Problem 3.4

(a) Starting with the RC spectrum $P(f)$ of Eq. (3.17), evaluate the inverse Fourier transform of $P(f)$ and thus show that.

$$p(t) = \left(\frac{\cos(2\pi\rho Wt)}{1-16\rho^2W^2t^2}\right)\text{sinc}(2Wt) \tag{3.19}$$

(b) Determine $P(f)$ and $p(t)$ for the special case of $\rho = 1$, which is known as the *full-cosine roll-off pluse.*

Ans. (b) $p(t) = \text{sinc}(4Wt)/(1-16W^2t^2)$ ■

3.4.1 Root Raised-Cosine Pulse Shaping[4]

A more sophisticated form of pulse shaping uses the *root raised-cosine (RC) spectrum* rather than the regular RC spectrum of Eq. (3.17). Specifically, the spectrum of the basic pulse is now defined by the square root of the right-hand side of this equation. Thus, using the trigonometric identity

$$\cos^2\theta = \frac{1}{2}(1+\cos 2\theta)$$

where, for the problem at hand,

$$\theta = \frac{\pi}{2W\rho}(|f| - W(1-\rho))$$

and retaining $P(f)$ as the symbol for the root RC spectrum, we may write

$$P(f) = \begin{cases} \dfrac{1}{\sqrt{2W}} & 0 \le |f| \le f_1 \\ \dfrac{1}{\sqrt{2W}}\cos\left(\dfrac{\pi}{4W\rho}(|f| - W(1-\rho))\right) & f_1 \le |f| < 2W - f_1 \\ 0 & |f| \ge 2W - f_1 \end{cases} \tag{3.20}$$

where, as before, the roll-off factor ρ is defined in terms of the frequency parameter f_1 and the bandwidth W as in Eq. (3.18).

If, now, the transmitter includes a *pre-modulation filter* with the transfer function defined in Eq. (3.20) and the receiver includes an identical filter, then the overall pulse waveform will experience the spectrum $P^2(f)$, which is the regular raised cosine spectrum. In effect, by adopting the root RC spectrum $P(f)$ of Eq. (3.20) for pulse shaping, we would be working with $P^2(f)$ in an overall transmitter-receiver sense. On this basis, we find that in the context of wireless communications, if the channel is affected by both flat fading and additive white Gaussian noise, and the pulse-shape filtering is partitioned equally between the transmitter and the receiver in the manner described herein, then effectively the receiver would maximize the output signal-to-noise ratio at the sampling instants.

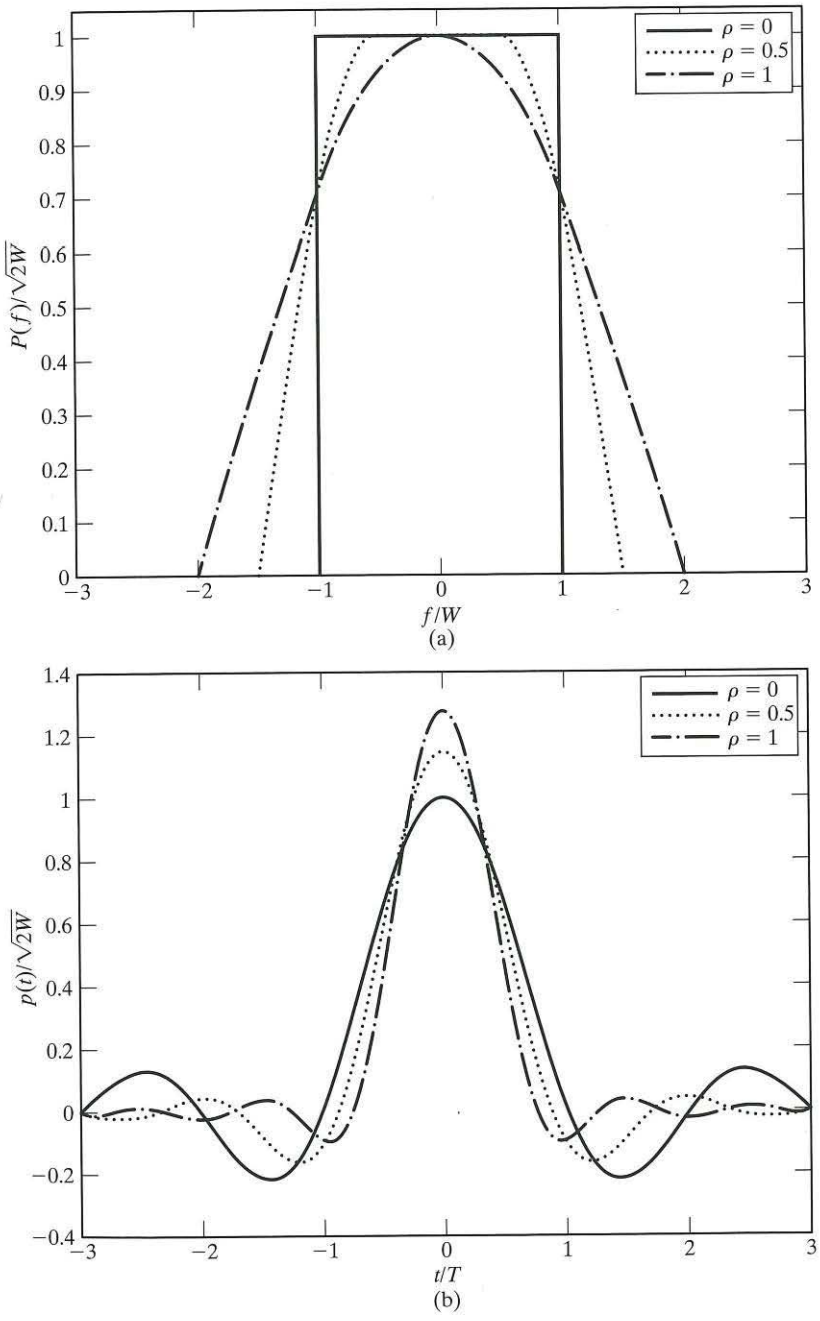

FIGURE 3.10 (a) $P(f)$ for root raised - cosine spectrum. (b) $p(t)$ for root raised - cosine spectrum.

The inverse Fourier transform of Eq. (3.20) defines the root RC shaping pulse

$$p(t) = \frac{\sqrt{2W}}{(1-(8\rho Wt)^2)}\left(\frac{\sin(2\pi W(1-\rho)t)}{2\pi Wt} + \frac{4\rho}{\pi}\cos(2\pi W(1+\rho)t)\right) \tag{3.21}$$

The important point to note here is the fact that the root RC shaping pulse $p(t)$ of Eq. (3.21) is radically different from the standard RC shaping pulse of Eq. (3.19). In particular, the new shaping pulse satisfies an *orthogonality constraint under T-shifts*, as shown by

$$\int_{-\infty}^{\infty} p(t)p(t-nT)dt = 0 \text{ for } n = \pm 1, \quad \pm 2, \ldots \tag{3.22}$$

where T is the symbol duration. Yet, $p(t)$ has exactly the same excess bandwidth as the standard RC pulse.

It is important to note, however, that despite the added property of orthogonality, the root RC shaping pulse of Eq. (3.21) lacks the zero-crossing property of the regular RC shaping pulse defined in Eq. (3.19).

Figure 3.10(a) plots the root RC spectrum $P(f)$ for roll-off factor $\rho = 0, 0.5, 1$; the corresponding time-domain plots are shown in Fig. 3.10(b). These plots are different from those of Fig. 3.9 for nonzero ρ. The following example contrasts the waveform of a specific binary sequence using the root RC shaping pulse with the corresponding waveform using the regular RC shaping pulse.

EXAMPLE 3.4 Pulse Shaping Comparison

Using the root RC shaping pulse $p(t)$ of Eq. (3.21) with roll-off factor $\rho = 0.5$, plot the waveform for the binary sequence 01100, and compare it with the corresponding waveform obtained by using the regular RC shaping pulse $p(t)$ of Eq. (3.19) with the same roll-off factor.

Using the root RC pulse $p(t)$ of Eq. (3.21) with a multiplying plus sign for binary symbol 1 and multiplying minus sign for binary symbol 0, we get the dashed and dotted pulse train shown in Fig. 3.11 for the sequence 01100. The solid pulse train shown in the figure corresponds to the use of the regular RC pulse $p(t)$ of Eq. (3.15). The figure shows the root RC waveform occupies a larger dynamic range than the regular RC waveform. ■

Problem 3.5

(a) Starting with Eq. (3.20), derive the root RC pulse shape $p(t)$ of Eq. (3.21).

(b) Evaluate $p(t)$ at (i) $t = 0$, and (ii) $t = \pm 1/(8\rho W)$.

(c) Show that the $p(t)$ derived in part (a) satisfies the orthogonality constraint described in Eq. (3.22)

Ans. (b) (i) $p(0) = \sqrt{2W}\left(1-\rho+\frac{4\rho}{\pi}\right)$

(ii) $p\left(\frac{\pm 1}{8\rho W}\right) = \sqrt{2W\rho}\left(\left(1+\frac{2}{\pi}\right)\sin\left(\frac{\pi}{4\rho}\right)+\left(1-\frac{2}{\pi}\right)\cos\left(\frac{\pi}{4\rho}\right)\right)$ ■

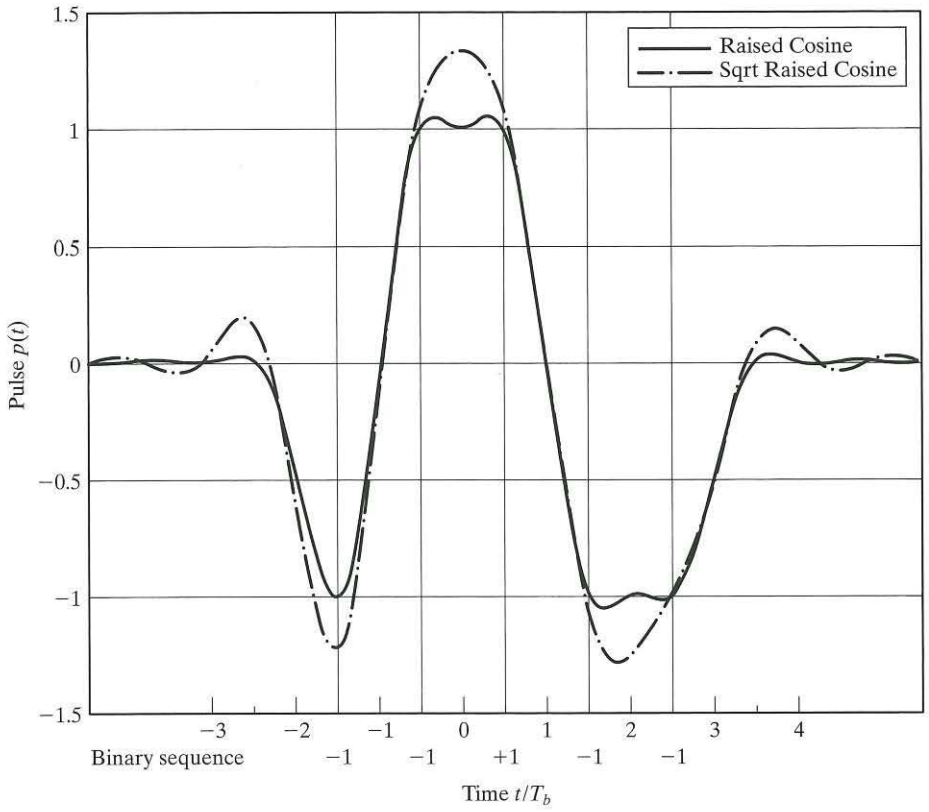

FIGURE 3.11 Two pulse trains for sequence 01100, one using regular RC pulse and the other using root RC pulse.

3.5 COMPLEX REPRESENTATION OF LINEAR MODULATED SIGNALS AND BAND-PASS SYSTEMS[5]

The linear modulation schemes considered in Section 3.3 may be viewed as special cases of the *canonical representation of a band-pass signal*:

$$s(t) = s_I(t)\cos(2\pi f_c t) - s_Q(t)\sin(2\pi f_c t) \tag{3.23}$$

It is customary to refer to $s_I(t)$ as the *in-phase component* of $s(t)$ and to $s_Q(t)$ as the *quadrature component*. This terminology follows from the definition of the sinusoidal carrier $c(t)$ in Eq. (3.1). Table 3.1 summarizes descriptions of AM, DSB–SC, BPSK, and QPSK in terms of the components $s_I(t)$ and $s_Q(t)$.

We may simplify matters further by introducing the complex signal

$$\tilde{s}(t) = s_I(t) + js_Q(t) \tag{3.24}$$

TABLE 3.1 Special Cases of the Canonical Equation (3.23).

	Type of modulation	In-phase component $s_I(t)$	Quadrature component $s_Q(t)$	Defining equation
Analog	Amplitude modulation	$A_c(1 + k_a m(t))$	0	3.3
	Double sideband–suppressed carrier modulation	$A_c m(t)$	0	3.4
Digital	Binary phase-shift keying	$A_c \sum_k b_k p(t - kT)$	0	3.5
	Quadriphase-shift keying	$A_c \sum_k b_{k,1} p(t - 2kT)$	$-A_c \sum_k b_{k,2} p(t - 2kT)$	3.11

where j is the square root of –1. For obvious reasons, the new signal $\tilde{s}(t)$ is referred to as the *complex envelope* of the modulated signal $s(t)$. Next, we invoke *Euler's formula*

$$\exp(j2\pi f_c t) = \cos(2\pi f_c t) + j\sin(2\pi f_c t) \tag{3.25}$$

Hence, in light of Eqs. (3.24) and (3.25), we may considerably simplify the formulation of the modulated signal $s(t)$ of Eq. (3.23) as

$$s(t) = \mathrm{Re}\left\{\tilde{s}(t)\exp(j2\pi f_c t)\right\} \tag{3.26}$$

Equation (3.26) is referred to as a *single-carrier transmission*.

The material presented in this section is of profound theoretical importance in the study of linear modulation theory, be it in the context of analog or digital modulation techniques. Specifically, we may make the following four statements:

1. The in-phase component $s_I(t)$ and the quadrature component $s_Q(t)$ are both real-valued functions of time that are uniquely defined in terms of the baseband (i.e., message) signal $m(t)$. Given the two components $s_I(t)$ and $s_Q(t)$, we may thus use the scheme shown in Fig. 3.12(a) to *synthesize* the modulated signal $s(t)$.
2. Given the modulated signal $s(t)$, we may use the scheme shown in Fig. 3.12(b) to *analyze* the modulated signal $s(t)$ and thereby construct the in-phase component $s_I(t)$ and the quadrature component $s_Q(t)$.
3. The in-phase and quadrature components are *orthogonal* to each other, occupying exactly the same bandwidth as the message signal $m(t)$.
4. The complex envelope $\tilde{s}(t)$ given in Eq. (3.24) completely preserves the information content of the modulated signal $s(t)$ except for the carrier frequency f_c.

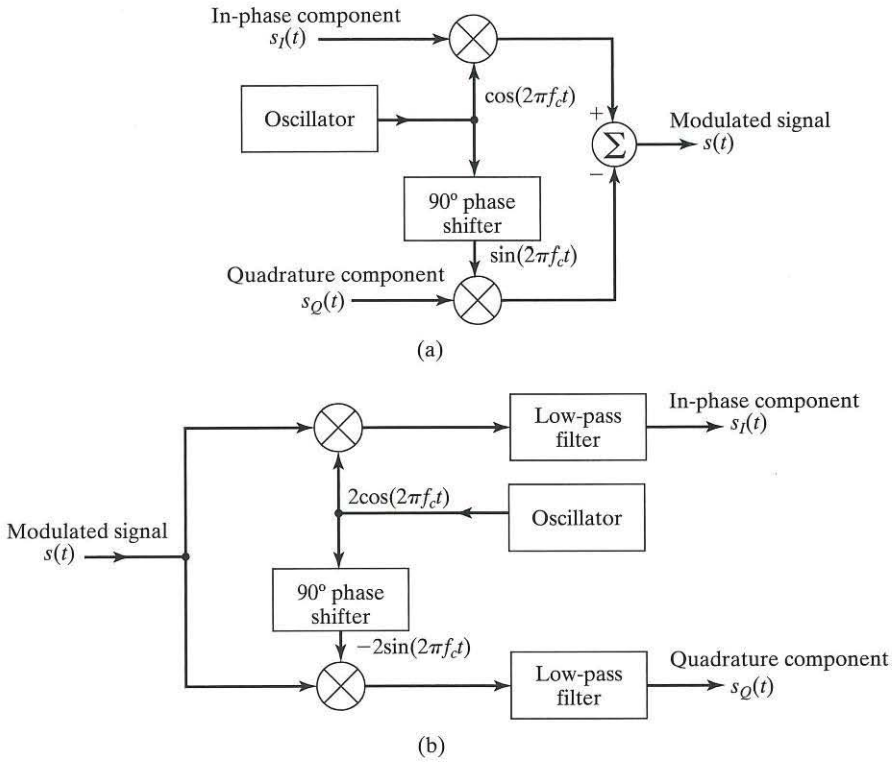

FIGURE 3.12 (a) Synthesizer for constructing a modulated signal from its in-phase and quadrature components. (b) Analyzer for deriving the in-phase and quadrature components of the modulated band-pass signal.

3.5.1 Complex Representation of Linear Band-Pass Systems

In a communication system, modulators exist alongside linear band-pass systems represented by band-pass filters and narrowband communication channels. As with any other linear system, these band-pass systems are uniquely characterized by an impulse response in the time domain and the corresponding transfer function in the frequency domain. From an analytic viewpoint, we find it highly instructive to develop complex representations of linear band-pass systems in a manner analogous to that followed for linear modulated (i.e., band-pass) signals. This form of representation not only simplifies the mathematical analysis of communication systems, but also provides the basis for their simulation on a digital computer.

Consider, then, a linear band-pass system with impulse response $h(t)$ and fed by an input signal $x(t)$ to produce an output signal $y(t)$, as depicted in Fig. 3.13. Two assumptions are made:

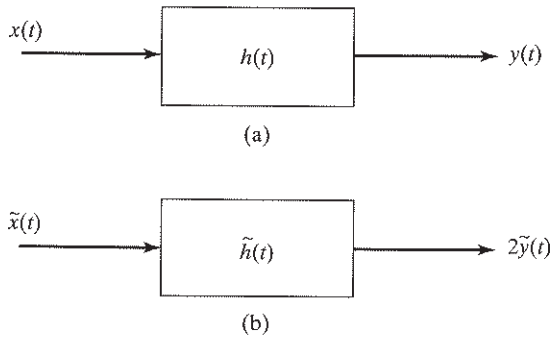

FIGURE 3.13 (a) Block diagram of linear band-pass system driven by a modulated signal $x(t)$ to produce the band-pass signal $y(t)$ as output. (b) Equivalent complex baseband model of the system, where the input signal, the impulse response, and output signal are all complex and in their baseband form.

1. The system is *narrowband*, which means that the system bandwidth is small compared with the midband frequency.
2. The input signal is a modulated signal whose carrier frequency, f_c, is the same as the midband frequency of the system.

The analysis of the system depicted in Fig. 3.13(a) is considerably simplified by replacing it with the *equivalent complex baseband model* of Fig. 3.13(b). This equivalent model is said to be complex in that its impulse response may be expressed as

$$\tilde{h}(t) = h_I(t) + jh_Q(t) \tag{3.27}$$

where $h_I(t)$is the *in-phase component* of $\tilde{h}(t)$ and $h_Q(t)$ is the *quadrature component*. In a manner similar to the complex representation of modulated signals, the impulse response of the original band-pass system, $h(t)$, is related to the complex impulse response $\tilde{h}(t)$ as

$$h(t) = \mathrm{Re}\left\{\tilde{h}(t)\exp(j2\pi f_c t)\right\} \tag{3.28}$$

The input signal supplied to the equivalent model, namely, $\tilde{x}(t)$, is the complex envelope of the original input signal $x(t)$. Likewise, the output signal of the equivalent model, namely, $\tilde{y}(t)$, is the complex envelope of the original output signal $y(t)$. These two complex envelopes are related by the complex convolutional integral

$$\begin{aligned}\tilde{y}(t) &= \frac{1}{2}\int_{-\infty}^{\infty}\tilde{x}(\lambda)\tilde{h}(t-\lambda)d\lambda \\ &= \frac{1}{2}\int_{-\infty}^{\infty}\tilde{h}(\lambda)\tilde{x}(t-\lambda)d\lambda\end{aligned} \tag{3.29}$$

Using shorthand notation, we may simply write

$$\begin{aligned}\tilde{y}(t) &= \frac{1}{2}\tilde{x}(t) \otimes \tilde{h}(t) \\ &= \frac{1}{2}(\tilde{h}(t) \otimes \tilde{x}(t))\end{aligned} \tag{3.30}$$

where the symbol $\otimes$ stands for convolution. The two lines of Eq. (3.29) and those of Eq. (3.30) simply emphasize the *commutative property* of convolution. Note the need for adding the multiplying factor 1/2 in these two equations, which is required to maintain the exact equivalence between the real system of Fig. 3.13(a) and its complex equivalent baseband model of Fig. 3.13(b). Note also that this equivalence assumes that the midband frequency of the system and the carrier frequency of the modulated input $x(t)$ are coincident.

The important point to take from the equivalence between the two systems, one real and the other complex, is that the carrier frequency f_c has been eliminated from the equivalent model. In effect, we have *traded complex analysis for the elimination of the carrier frequency*, thereby simplifying the analysis considerably without any loss of information. In particular, having determined the complex envelope $\tilde{y}(t)$ of the output signal by convolving the complex envelope $\tilde{x}(t)$ of the input signal with the complex impulse response $\tilde{h}(t)$, we may readily determine the actual output signal $y(t)$ in Fig. 3.13(a) by using the formula

$$y(t) = \text{Re}\left\{\tilde{y}(t)\exp(j2\pi f_c t)\right\} \tag{3.31}$$

A similar formula holds for the input signal $x(t)$ in terms of its complex envelope $\tilde{x}(t)$.

3.6 SIGNAL-SPACE REPRESENTATION OF DIGITALLY MODULATED SIGNALS[6]

Digital modulation distinguishes itself from analog modulation in that the transmitted signal $s(t)$ assumes one of a discrete set of possible forms. The corresponding mapping of the complex envelope $\tilde{s}(t)$ onto a signal space is referred to as a *signal constellation* or *signal pattern*. Much of the literature on digital modulation techniques focuses on *two-dimensional* signal constellations whose structure naturally depends on the specific form of modulation under study. The traditional approach is to use energy-normalized versions of the in-phase component $s_I(t)$ and quadrature component $s_Q(t)$ of the modulated signal $s(t)$ as the horizontal axis and vertical axis of the two-dimensional signal space, respectively. In what follows, these *normalized coordinates of unit energy* are denoted by $\phi_1(t)$ and $\phi_2(t)$, respectively, defined in turn by

$$\phi_1(t) = \sqrt{\frac{2}{T}}\cos(2\pi f_c t) \qquad 0 \le t \le T \tag{3.32}$$

and

$$\phi_2(t) = \sqrt{\frac{2}{T}}\sin(2\pi f_c t) \qquad 0 \le t \le T \tag{3.33}$$

where T is the symbol duration and the carrier frequency is an integer multiple of $1/T$. From these two equations we readily see the following two properties:

1. $$\int_0^T \phi_1(t)\phi_2(t)dt = 0 \tag{3.34}$$

which confirms the orthogonality of $\phi_1(t)$ and $\phi_2(t)$ over the interval $0 \le t \le T$.

2. $$\int_0^T \phi_1^2(t)dt = \int_0^T \phi_2^2(t)dt = 1 \tag{3.35}$$

which shows that both $\phi_1(t)$ and $\phi_2(t)$ have unit energy.

Thus, $\phi_1(t)$ and $\phi_2(t)$ are said to constitute an *orthonormal set*.

Figure 3.14(a) shows the constellation of binary phase-shift keying (BPSK). The constellation is one dimensional as the coordinate $\phi_1(t)$ is sufficient for the description of BPSK. The constellation has two signal points along the ϕ_1 - axis at $\pm\sqrt{E_b}$, which correspond to the transmission of symbol 1 (the plus sign) and the symbol 0 (the minus sign).

Figure 3.14(b) shows the two-dimensional signal constellation of *quadriphase-shift keying* (QPSK), in which the phase of the transmitted signal takes on one of four possible values: 45°, 135°, 225°, and 315°. The radius of the circle on which the four points of the constellation lie is equal to $\sqrt{2E_b}$, denoting the square root of the transmitted signal energy per symbol. Table 3.2 summarizes the coordinates of the four possible message points characterizing the QPSK signal constellation of Fig. 3.14(b). In terms of the message points (s_{i1}, s_{i2}) defined in the table, we may now express the QPSK signal as

$$s_i(t) = s_{i1}\phi_1(t) + s_{i2}\phi_2(t), \qquad \begin{matrix} i = 1, 2 \\ 0 \le t \le T \end{matrix} \tag{3.36}$$

Each point in the constellation of Fig. 3.14(b) corresponds to a specific dibit made up of a group of two binary symbols; hence, the symbol period T for QPSK is twice the bit duration of the incoming binary stream. Figure 3.14(b) includes the four dibits corresponding to the *message points* of the constellation. Note that as we go around the constellation from one message point to the adjacent one, only one of the two binary symbols in a dibit experiences a change. This form of message representation is referred to as a *Gray code*. Table 3.2 also includes the Gray encoding descriptions of the four input dibits.

As remarked previously, the QPSK signal may also be constructed from the discrete set of phase values: 0°, 90°, 180°, and 270°. Figure 3.14(c) shows the signal constellation for this second construction of QPSK.

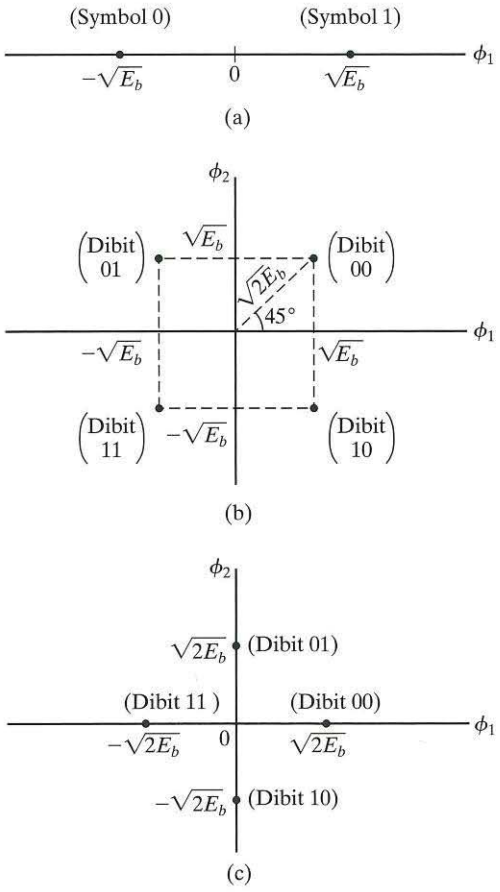

FIGURE 3.14 Signal constellations for (a) BPSK, (b) one version of QPSK, and (c) another version of QPSK.

TABLE 3.2 Signal-space characterization of the QPSK signal constellation described in Fig. 3.14(b).

Input dibit	Gray-encoded phase of QPSK signal (radians)	Coordinates of message points	
		s_{i1}	s_{i2}
10	$7\pi/4$	$+\sqrt{E_b}$	$-\sqrt{E_b}$
11	$5\pi/4$	$-\sqrt{E_b}$	$-\sqrt{E_b}$
01	$3\pi/4$	$-\sqrt{E_b}$	$+\sqrt{E_b}$
00	$\pi/4$	$+\sqrt{E_b}$	$+\sqrt{E_b}$

Problem 3.6 Construct a table summarizing the signal-space characterization of the QPSK signal constellation described in Fig. 3.14(c).

Ans.

Input dibit	Coordinates of message points
00	$(\sqrt{2E_b}, 0)$
01	$(0, \sqrt{2E_b})$
11	$(-\sqrt{2E_b}, 0)$
10	$(0, -\sqrt{2E_b})$

■

Figure 3.15 shows the signal constellation of another commonly used linear form of modulation: *16-quadrature amplitude modulation* (16-QAM). The modulated signal illustrated is hybrid in that it combines amplitude and phase modulations. Each message point of the constellation corresponds to a specific *quadbit*, which is made up of a group of four adjacent bits. Here again, we have used Gray coding in that, as we go from one message point to the adjacent one, only one of the binary symbols in a quadbit experiences a change. However, there is a major difference between the signal constellations of Figs. 3.14 and 3.15: In the QPSK signal represented by the constellation of

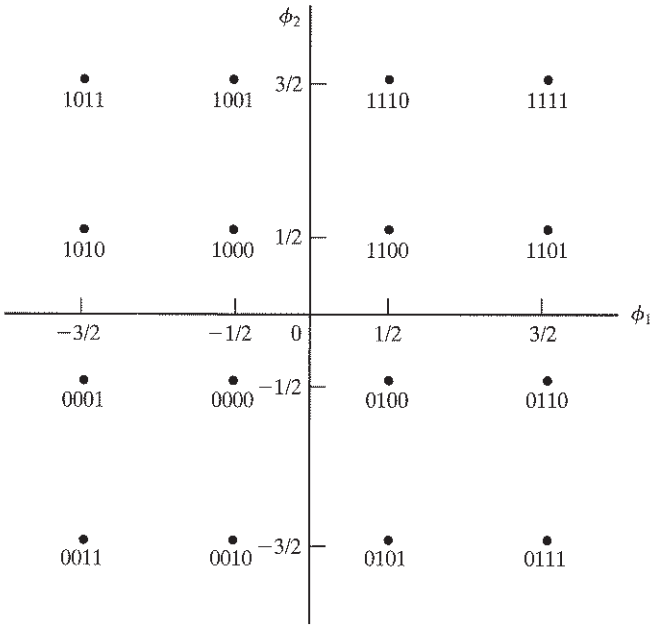

FIGURE 3.15 Signal-space diagram of M-ary QAM for $M = 16$; the message points in each quadrant are identified with Gray-encoded quadbits.

Fig. 3.14(a) and (b), the energy transmitted remains fixed; in the 16-QAM signal represented by the constellation of Figs. 3.15, the energy transmitted is variable, depending on the particular quadbit (i.e., 4-bit symbol) chosen for transmission. Note that QPSK signal can be transmitted over a nonlinear channel, whereas the transmission of the 16-QAM signal requires the use of a linear channel.

3.7 NONLINEAR MODULATION TECHNIQUES

The canonical representation of Eq. (3.23) naturally provides the Cartesian basis for the construction of signal constellations of linear modulated signals. Equivalently, we may recast that equation in the *polar* form

$$s(t) = a(t)\cos[2\pi f_c t + \theta(t)] \tag{3.37}$$

where

$$a(t) = \sqrt{s_I^2(t) + s_Q^2(t)} \tag{3.38}$$

is called the *envelope* of the modulated signal $s(t)$ and

$$\theta(t) = \tan^{-1}\left(\frac{s_Q(t)}{s_I(t)}\right) \tag{3.39}$$

is the *phase* of $s(t)$. Equation (3.37) provides the natural basis for the study of *nonlinear* modulation techniques, as discussed next.

3.7.1 Frequency Modulation

In *frequency modulation* (FM), a form of angle modulation, the instantaneous frequency of the sinusoidal carrier, denoted by $\psi(t)$, is varied in accordance with the information-bearing signal $m(t)$. A complete oscillation occurs whenever the phase $\psi_i(t)$ changes by 2π radians. If $\psi_i(t)$ increases monotonically with time, the average frequency in hertz, over an incremental interval from t to $t + \Delta t$, is given by

$$f_{\Delta t}(t) = \frac{\psi(t + \Delta t) - \psi(t)}{2\pi\Delta t} \tag{3.40}$$

Letting Δt approach zero leads to the following definition for the instantaneous frequency of the FM signal:

$$\begin{aligned} f(t) &= \lim_{\Delta t \to 0} f_{\Delta t}(t) \\ &= \lim_{\Delta t \to 0}\left[\frac{\psi(t + \Delta t) - \psi(t)}{2\pi\Delta t}\right] \\ &= \frac{1}{2\pi}\frac{d\psi(t)}{dt} \end{aligned} \tag{3.41}$$

Equation (3.41) states that, except for a scaling factor, the instantaneous frequency $f(t)$ is the derivative of the instantaneous phase $\psi(t)$ with respect to time t. Conversely, we may define the instantaneous phase as the integral of the instantaneous frequency with respect to time; that is,

$$\psi(t) = \int_0^t f(\tau)d\tau \tag{3.42}$$

where, for convenience of presentation, it is assumed that the initial phase

$$\psi(0) = \int_{-\infty}^0 f(t)dt$$

is zero.

To formally describe frequency modulation in mathematical terms, we write

$$f(t) = f_c + k_f m(t) \tag{3.43}$$

where k_f is the sensitivity of the frequency modulator and f_c is the frequency of the unmodulated carrier. Substituting Eq. (3.43) into Eq. (3.39) and multiplying the result by 2π, we get

$$\psi(t) = 2\pi f_c t + 2\pi k_f \int_0^t m(\tau)d\tau \tag{3.44}$$

The frequency-modulated signal is thus described in the time domain by

$$\begin{aligned} s(t) &= A_c \cos(\psi(t)) \\ &= A_c \cos\left(2\pi f_c t + 2\pi k_f \int_0^t m(\tau)d\tau\right) \end{aligned} \tag{3.45}$$

From Eq. (3.45), we see that frequency modulation is a *nonlinear* process, in that the dependence of the modulated signal on the information-bearing signal violates the *principle of superposition*. Consequently, unlike the spectrum of the AM signal of Eq. (3.2), the spectrum of the FM signal of Eq. (3.45) is *not* related to the spectrum of the information-bearing signal $m(t)$ in a simple manner. This means that, in general, the spectral analysis of an FM signal is a more difficult task than that of an AM signal. Elsewhere,[7] it is shown that the transmission bandwidth of the FM signal $s(t)$ is approximately given by *Carson's rule*, which states that

$$B_T \approx 2\Delta f\left(1 + \frac{1}{D}\right) \tag{3.46}$$

Carson's rule involves two parameters:

1. *The frequency deviation*, Δf, which is defined as the maximum deviation in the instantaneous frequency of the FM signal away from the carrier frequency f_c.
2. *The deviation ratio*, D, which is defined as the ratio of the frequency deviation to the highest frequency component contained in the modulating signal $m(t)$.

Unfortunately, Carson's rule does not always provide a good estimate of the bandwidth requirement of a wireless communication system using frequency modulation. For a more accurate estimate of the bandwidth requirement, we may have to resort to the use of computer simulations or actual measurements.

Unlike amplitude modulation, frequency modulation offers the means for exchanging the increased transmission bandwidth for an improved noise performance. Specifically, under the simplifying assumption of a high carrier-to-noise ratio, we may make the following statement:

> *An increase in the FM transmission bandwidth B_T produces a quadratic increase in the signal-to-noise ratio at the output of the receiver.*

Indeed, it is because of the *bandwidth–noise trade-off* capability that frequency modulation was adopted in the *first generation* of wireless communication systems, based on frequency-division multiple access (FDMA).

However, with the ever-increasing emphasis on the use of digital signal-processing techniques, analog frequency modulation is being superseded by its digital counterparts, discussed in the rest of the section.

3.7.2 Binary Frequency-Shift Keying

In binary frequency-shift keying (BFSK), the symbols 1 and 0 are distinguished from each other by the transmission of one of two sinusoidal waves that differ in frequency by a fixed amount. A typical pair of sinusoidal waves is described by

$$s_i(t) = \begin{cases} \sqrt{\frac{2E_b}{T}}\cos(2\pi f_i t) & 0 \le t \le T_b \\ 0 & \text{otherwise} \end{cases} \tag{3.47}$$

where $i = 1,2$, T is the symbol (bit) duration, and E_b is the energy transmitted per bit; the frequency transmitted is

$$f_i = \frac{n_c + i}{T} \quad \text{for some fixed integer } n_c \text{ and } i = 1,2 \tag{3.48}$$

The symbol 1 is represented by $s_1(t)$ and the symbol 0 by $s_2(t)$. The BFSK signal described here is known as *Sunde's FSK*, in recognition of its inventor. It is a *continuous-phase signal* in the sense that phase continuity is maintained everywhere, including the interbit switching times. This form of digital modulation is an example of *continuous-phase frequency-shift keying* (CPFSK), on which we have more to say in the next subsection.

Figure 3.16 shows the signal constellation of BFSK, with the two coordinates of the constellation defined by

$$\phi_i(t) = \begin{cases} \sqrt{\frac{2}{T}}\cos(2\pi f_i t) & 0 \le t \le T \\ 0 & \text{otherwise} \end{cases} \tag{3.49}$$

where $i = 1,2$, and f_i is itself defined by Eq. (3.48).

Problem 3.7 What conclusion can you draw from the signal-space diagram of Fig. 3.16 for BFSK, compared with that of Fig. 3.14(a) for BPSK?

Ans. The signal-space diagram of BPSK in Fig. 3.14(a) is one-dimensional, whereas that of BFSK in Fig. 3.16 has two dimensions. Dimensions as two coordinates, $\phi_1(t)$ and $\phi_2(t)$, are required for the description of BFSK. ■

Problem 3.8 How is BFSK extended to M-ary FSK?

Ans. The dimensionality of the BFSK signal is two, exactly the same as the number of symbols embodied in the generation of the signal. In the M-ary FSK signal, there are M distinct signals to be accounted for; hence, the dimensionality of the M-ary FSK signal is M. M-ary FSK includes BFSK as a special case. ■

3.7.3 Continuous-Phase Modulation: Minimum Shift Keying[8]

In constructing the BFSK signal constellation of Fig. 3.16, the only design constraint we imposed on the construction is that the two frequencies f_1 and f_2 (representing symbols 1 and 0, respectively) satisfy the requirement of Eq. (3.48). However, through a more constrained choice of these two frequencies, we can make a significant difference to two important characteristics of this nonlinear method of digital signaling, namely, improved spectral efficiency and improved noise performance.

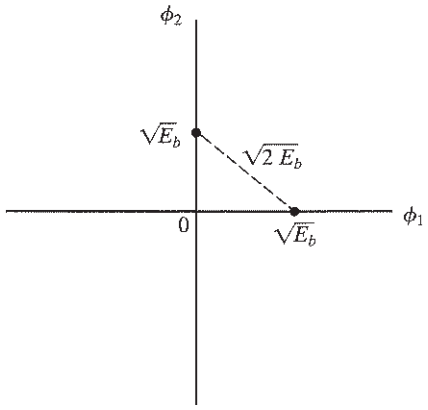

FIGURE 3.16 Signal constellation of binary FSK signal.

Consider, then, a *CPFSK signal*, which, for the interval $0 \le t \le T$, is defined as

$$s(t) = \begin{cases} \sqrt{\frac{2E_b}{T}}\cos[2\pi f_1 t + \theta(0)] & \text{for symbol 1} \\ \sqrt{\frac{2E_b}{T}}\cos[2\pi f_2 t + \theta(0)] & \text{for symbol 0} \end{cases} \tag{3.50}$$

where E_b is the energy transmitted per bit and T is the symbol (bit) duration. The phase $\theta(0)$, denoting the value of the phase at time $t = 0$, sums up the past history of the modulation process up to time $t = 0$. The frequencies f_1 and f_2 are sent in response to binary symbols 1 and 0 respectively appearing at the modulator input.

Another useful way of representing the CPFSK signal $s(t)$ is to express it in the conventional form of an angle-modulated signal as

$$s(t) = \sqrt{\frac{2E_b}{T}}\cos[2\pi f_c t + \theta(t)] \tag{3.51}$$

where $\theta(t)$ is the phase of $s(t)$; Eq. (3.51) is a special form of Eq. (3.37) with $a(t) = \sqrt{2E_b/T}$. When the phase $\theta(t)$ is a continuous function of time, we find that the modulated signal $s(t)$ itself is also continuous at all times, including the interbit switching times. The phase $\theta(t)$ of a CPFSK signal increases or decreases linearly with time during each bit duration of T seconds, as shown by the relationship

$$\theta(t) = \theta(0) \pm \frac{\pi h}{T} t \qquad 0 \le t \le T \tag{3.52}$$

where the plus sign corresponds to sending symbol 1 and the minus sign corresponds to sending symbol 0; the parameter h is to be defined. Substituting Eq. (3.52) into Eq. (3.51), and then comparing the angle of the cosine function with that of Eq. (3.50), we deduce the following pair of relations:

$$f_c + \frac{h}{2T} = f_1 \tag{3.53}$$

$$f_c - \frac{h}{2T} = f_2 \tag{3.54}$$

Solving Eqs. (3.53) and (3.54) for f_c and h, we thus get

$$f_c = \frac{1}{2}(f_1 + f_2) \tag{3.55}$$

and

$$h = T(f_1 - f_2) \tag{3.56}$$

The nominal carrier frequency f_c is therefore the arithmetic mean of the transmit frequencies f_1 and f_2. The difference between the frequencies f_1 and f_2, normalized with respect to the bit rate $1/T$, defines the dimensionless parameter h, which is referred to as the *deviation ratio* of the CPFSK signal. We have purposely used a different symbol for this deviation ratio to distinguish it from the deviation ratio of an analog FM signal.

From Eq. (3.52) we find that at time $t = T$,

$$\theta(T) - \theta(0) = \begin{cases} \pi h & \text{for symbol 1} \\ -\pi h & \text{for symbol 0} \end{cases} \tag{3.57}$$

That is to say, sending of the symbol 1 increases the phase of a CPFSK signal $s(t)$ by πh radians, whereas sending of the symbol 0 reduces the phase by an equal amount.

The variation of phase $\theta(t)$ with time t follows a path consisting of a sequence of straight lines, the slopes of which represent changes in frequency. Figure 3.17 depicts possible paths starting from time $t = 0$. A plot like that shown is called a *phase tree*. The tree makes clear the transitions of phase across interval boundaries of the incoming sequence of data bits. Moreover, it is evident from the figure that the phase of a CPFSK signal is an odd or even multiple of πh radians at odd or even multiples, respectively, of the bit duration T.

The phase tree described in Fig. 3.17 is a manifestation of phase continuity, which is an inherent characteristic of a CPFSK signal. To appreciate the notion of phase continuity, let us go back for a moment to Sunde's FSK, which is the simplest form of a CPFSK scheme. In this case, the deviation ratio h is exactly unity, in accordance with Eq. (3.56). Hence, according to Fig. 3.17 the phase change over one bit interval is $\pm\pi$ radians. But a change of $+\pi$ radians is exactly the same as a change of $-\pi$ radians, modulo 2π. It follows, therefore, that in the case of Sunde's FSK, there is no memory; that is, knowing which particular change occurred in the *previous* bit interval provides no help in the *current* bit interval.

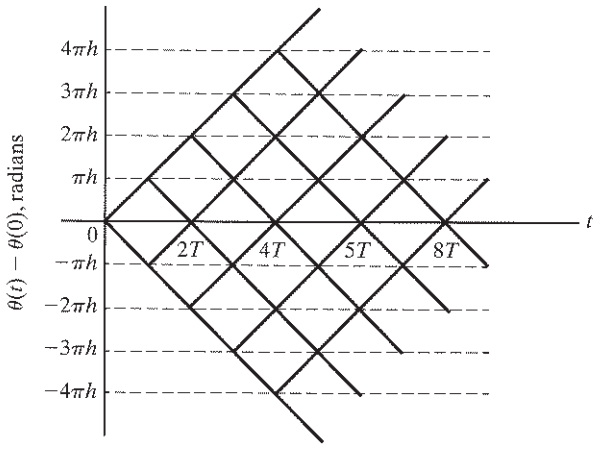

FIGURE 3.17 Phase tree of a CPFSK signal.

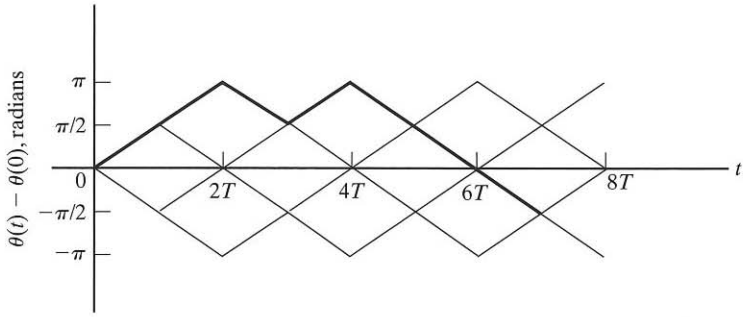

FIGURE 3.18 Phase trellis; boldfaced path represents the sequence 1101000.

In contrast, we have a completely different situation when the deviation ratio h is assigned the special value 1/2. We now find that the phase can take on only the two values $\pm\pi/2$ at odd multiples of T and only the two values 0 and π at even multiples of T, as shown in Fig. 3.18. This second graph is called a *phase trellis*, since a trellis is a treelike structure with reemerging branches. Each path from left to right through the trellis of the figure corresponds to a specific binary sequence input. For example, the path shown in boldface in Fig. 3.18 corresponds to the binary sequence 1101000 with $\theta(t) = 0$. Henceforth, we assume that $h = 1/2$.

With $h = 1/2$, we find from Eq. (3.56) that the frequency deviation (i.e., the difference between the two signaling frequencies f_1 and f_2) equals half the bit rate. This is the minimum frequency spacing that allows the two FSK signals representing the symbols 1 and 0, as in Eq. (3.51), to be coherently orthogonal, in the sense that they do not interfere with one another in the process of detection. It is for this reason that a CPFSK signal with a deviation ratio of one-half is commonly referred to as *minimum shift keying* (MSK).

Table 3.3 delineates how the memory of the MSK generator manifests itself. For example, if $\theta(0) = 0$, then the transmission of the symbol 0 results in $\theta(T) = -\pi/2$. If, however, $\theta(0) = \pi$, then the transmission of the symbol 0 results in $\theta(T) = \pi/2$.

Problem 3.9 Assume that the frequencies f_1 and f_2 are both odd integer multiples of $1/4T$. Then show that the MSK signal may be expressed in terms of the orthonormal pair of coordinates

$$\phi_1(t) = \sqrt{\frac{2}{T}}\cos\left(\frac{\pi}{2T}t\right)\cos(2\pi f_c t) \qquad 0 \le t \le T \tag{3.58}$$

and

$$\phi_2(t) = \sqrt{\frac{2}{T}}\sin\left(\frac{\pi}{2T}t\right)\sin(2\pi f_c t) \qquad 0 \le t \le T \tag{3.59}$$

In particular, show that

$$s(t) = s_1\phi_1(t) + s_2\phi_2(t) \tag{3.60}$$

TABLE 3.3 Transition characterization of MSK.

Transmitted Binary Symbol, $0 \le t \le T$	Phase States (radians)		Coordinates of Message Points	
	$\theta(0)$	$\theta(T)$	s_1	s_2
0	0	$-\pi/2$	$+\sqrt{E_b}$	$+\sqrt{E_b}$
1	π	$-\pi/2$	$-\sqrt{E_b}$	$+\sqrt{E_b}$
0	π	$+\pi/2$	$-\sqrt{E_b}$	$-\sqrt{E_b}$
1	0	$+\pi/2$	$+\sqrt{E_b}$	$-\sqrt{E_b}$

where

$$s_1 = \int_0^{2T} s(t)\phi_1(t)dt = \sqrt{E}\cos(\theta(0)) \qquad -T \le t \le T \tag{3.61}$$

and

$$s_2 = \int_{-T}^{T} s(t)\phi_2(t)dt = \sqrt{E}\sin(\theta(T)) \qquad 0 \le t \le 2T \tag{3.62}$$

Hence, using the formulas of Eqs. (3.61) and (3.62), verify the entries for s_1 and s_2 in Table 3.3. ■

Problem 3.10 Using the results of Problem 3.9, construct the waveform of the MSK signal for the binary sequence 1101000, assuming that $f_1 = 5/(4T)$, and $f_2 = 3/(4T)$ by plotting the components $s_1\phi_1(t)$ and $s_2\phi_2(t)$.

Ans. See Fig. 3.19, where the input sequence occupies the time interval $0 \le t/T \le 7$ ■

3.7.4 Power Spectra of MSK Signal

It is informative to formulate the power spectrum of an MSK signal. To that end, we assume that the input binary sequence is random, with the symbols 1 and 0 equally likely and the symbols transmitted during different time slots being statistically independent. We may thus make the following observations:

1. Depending on the value of the phase state $\theta(0)$, the in-phase component equals $+g_I(t)$ or $-g_I(t)$, where

$$g_I(t) = \begin{cases} \sqrt{\dfrac{2E_b}{T}}\cos\left(\dfrac{\pi t}{2T}\right) & -T \le t \le T \\ 0 & \text{otherwise} \end{cases} \tag{3.63}$$

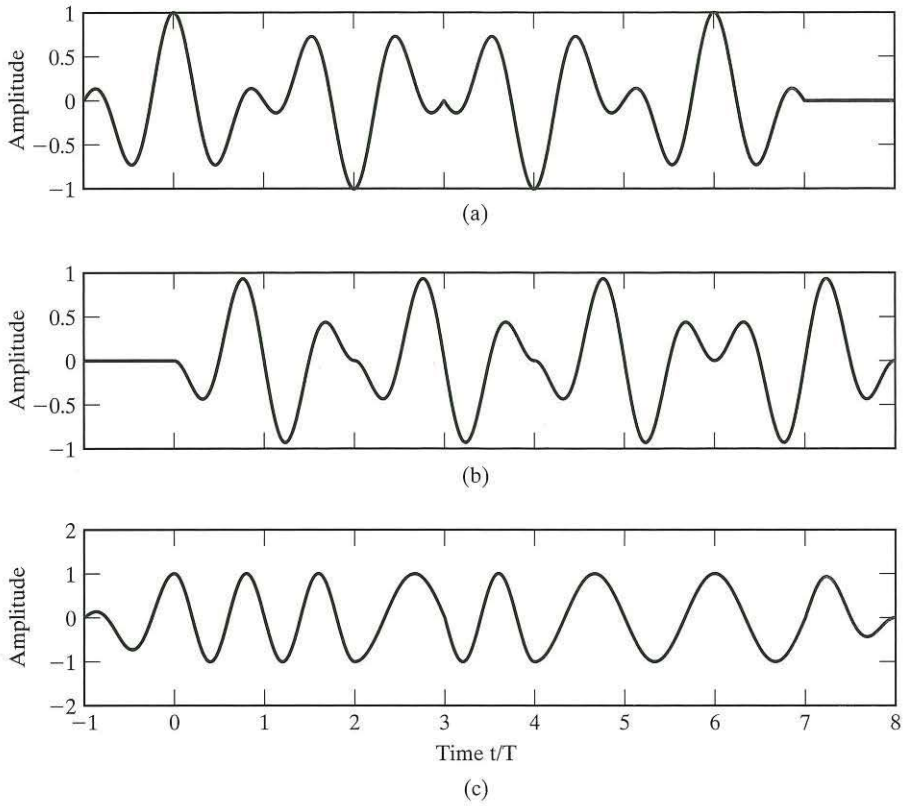

FIGURE 3.19 Waveforms for Problem 3.10: (a) $s_1\phi_1(t)$, (b) $s_2\phi_2(t)$, and (c) MSK signal $s(t)$.

2. Depending on the value of the phase state $\theta(T)$, the quadrature component equals $+g_Q(t)$ or $-g_Q(t)$, where

$$g_Q(t) = \begin{cases} \sqrt{\frac{2E_b}{T}}\sin\left(\frac{\pi t}{2T}\right) & -T \le t \le T \\ 0 & \text{otherwise} \end{cases} \tag{3.64}$$

3. The in-phase and quadrature components of the MSK signal are statistically independent, yielding the baseband power spectral density of the MSK signal:

$$\begin{aligned} S_B(f) &= 2\left[\frac{\Psi_g(f)}{2T}\right] \\ &= \frac{32E_b}{\pi^2}\left[\frac{\cos(2\pi Tf)}{16T^2f^2-1}\right]^2 \end{aligned} \tag{3.65}$$

where $\psi_g(f)$ is the common power spectral density of $g_I(t)$ and $g_Q(t)$. According to Eq. (3.65), the baseband power spectral density of the MSK signal falls off as the inverse fourth power of frequency, compared with the inverse square power of frequency for QPSK. Accordingly, MSK does not produce as much interference outside the signal band of interest as does QPSK. This is a desirable characteristic of MSK, especially when the digital communication system operates with a bandwidth limitation.

Problem 3.11 Find the frequencies at which the baseband power spectrum of the MSK signal attains its minimum value of zero.
Ans. $f = 1/4T$. ■

3.7.5 Gaussian-Filtered MSK

From the study of minimum shift keying (MSK) just presented, we may summarize the desirable properties of the MSK signal. The MSK signal should have

1. a constant envelope,
2. a relatively narrow bandwidth, and
3. a coherent detection performance equivalent to that of QPSK.

However, the out-of-band spectral characteristics of MSK signals, as good as they are, still do not satisfy the stringent requirements of wireless communications. To illustrate this limitation, we find from Eq. (3.65) that at $fT = 0.5$, the baseband power spectral density of the MSK signal drops by only $10 \log_{10} 9 = 9.54$ dB below its midband value. Hence, when the MSK signal is assigned a transmission bandwidth of $1/T$, the adjacent channel interference of a wireless communication system using MSK is not low enough to satisfy the practical requirements of such a multiuser communications environment.

Recognizing that the MSK signal can be generated by direct frequency modulation of a voltage-controlled oscillator, we may overcome this serious limitation of MSK by modifying the power spectrum of the signal into a *compact* form, while maintaining the constant-envelope property of the signal. This modification can be achieved through the use of a premodulation low-pass filter, hereafter referred to as a baseband *pulse-shaping filter*. Such a filter should have

1. a frequency response with a narrow bandwidth and sharp cutoff characteristics,
2. an impulse response with relatively low overshoot, and
3. a phase trellis which evolves in such a manner that the carrier phase of the modulated signal assumes the two values $\pm\pi/2$ at odd multiples of T and the two values 0 and π at even multiples of T, as in MSK.

Condition 1 is needed to suppress the high-frequency components of the transmitted signal. Condition 2 avoids excessive deviations in the instantaneous frequency of the FM signal. Finally, condition 3 ensures that the modified FM signal can be coherently detected in the same way as the MSK signal is, or else it can be noncoherently detected as a simple BFSK signal.

These desirable properties can be achieved by passing a polar *nonreturn-to-zero (NRZ)* binary data stream through a baseband pulse-shaping filter whose impulse response (and, likewise, frequency response) is defined by a *Gaussian* function. In this line code, the binary symbols 0 and 1 are represented by levels –1 and +1, respectively. The resulting method of binary frequency modulation is naturally referred to as *Gaussian-filtered MSK* or just *GMSK*. The use of filtering has the effect of introducing additional memory into the generation of GMSK.

Let W denote the *3-dB baseband bandwidth* of the pulse-shaping filter. We may then respectively define the transfer function $H(f)$ and impulse response $h(t)$ of the pulse-shaping filter as

$$H(f) = \exp\left(-\frac{\log_e 2}{2}\left(\frac{f}{W}\right)^2\right) \tag{3.66}$$

and

$$h(t) = \sqrt{\frac{2\pi}{\log_e 2}}\, W \exp\left(-\frac{2\pi^2}{\log_e 2} W^2 t^2\right) \tag{3.67}$$

The response of this Gaussian filter to a rectangular pulse of unit amplitude and duration T (centered on the origin) is given by

$$\begin{aligned} g(t) &= \int_{-T/2}^{T/2} h(t-\tau)\, d\tau \\ &= \sqrt{\frac{2\pi}{\log_e 2}}\, W \int_{-T/2}^{T/2} \exp\left(-\frac{2\pi^2}{\log_e 2} W^2 (t-\tau)^2\right) d\tau \end{aligned} \tag{3.68}$$

which may be expressed as the difference between two complementary error functions:

$$g(t) = \frac{1}{2}\left[\operatorname{erfc}\left(\pi\sqrt{\frac{2\pi}{\log_e 2}}\, WT\left(\frac{t}{T} - \frac{1}{2}\right)\right) - \operatorname{erfc}\left(\pi\sqrt{\frac{2\pi}{\log_e 2}}\, WT\left(\frac{t}{T} + \frac{1}{2}\right)\right)\right] \tag{3.69}$$

(For a formal definition of the *complementary error function* and its properties, see Appendix E.) The pulse response, $g(t)$ constitutes the *frequency-shaping pulse* of the GMSK modulator, with the dimensionless *time–bandwidth product* WT playing the role of a design parameter.

The frequency-shaping pulse $g(t)$, as defined in Eq. (3.69), is noncausal, in that it is nonzero for $t < -T/2$, where $t = -T/2$ is the time at which the input rectangular pulse (symmetrically positioned around the origin) is applied to the Gaussian filter. For a causal response, $g(t)$ must be truncated and shifted in time. Figure 3.20 presents plots of $g(t)$, which have been truncated at $t = \pm 2.5T$ and then shifted in time by $2.5T$, for $WT = 0.2$, 0.25, and 0.3. Note that as WT is reduced, the time spread of the frequency-shaping pulse is correspondingly increased.

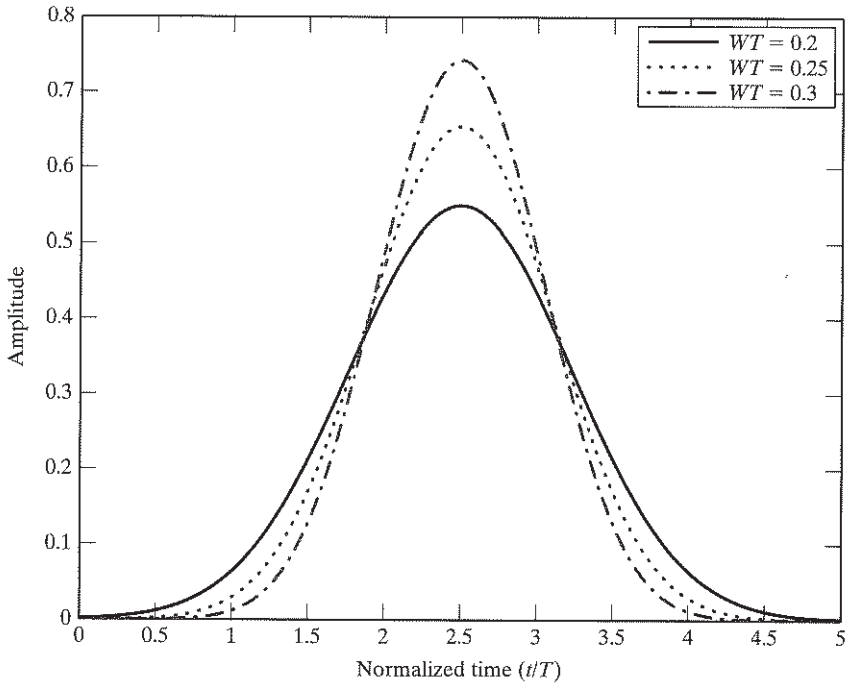

FIGURE 3.20 Frequency-shaping pulse $g(t)$ of Eq. (3.63), shifted in time by $2.5T$ and truncated at $\pm 2.5T$ for varying time–bandwidth product WT.

Figure 3.21 shows some machine-computed power spectra of MSK signals (expressed in decibels) versus the normalized frequency difference $(f - f_c)T$, where f_c is the midband frequency and T is the bit duration. The plots are for varying values of the time–bandwidth product WT. From the figure, we may make the following observations:

- The curve for the limiting condition $WT = \infty$ corresponds to the ordinary MSK.
- When WT is less than unity, increasingly more of the transmit power is concentrated inside the passband of the GMSK signal.

An undesirable feature of GMSK, readily apparent from Fig. 3.20, is that the processing of NRZ binary data by a Gaussian filter generates a modulating signal that is no longer confined to a single bit interval as in ordinary MSK. Stated in another way, the tails of the Gaussian impulse response of the pulse-shaping filter cause the modulating signal to spread out to adjacent symbol intervals. The net result is the generation of a *controlled form of intersymbol interference*, the extent of which increases with decreasing WT. In light of this observation and the observation we made on the basis of Fig. 3.21 about the power spectra of GMSK signals, we may say that the choice of the time–bandwidth product WT offers a trade-off between spectral compactness and a reduction in receiver performance. The compromise value $WT = 0.3$ ensures that the sidelobes of the power spectrum of the GMSK signal drop by an amount larger than 40 dB relative to the midband frequency, which means that the effect of adjacent channel interference is practically negligible. (The issue of adjacent channel interference is discussed in Section 3.9.)

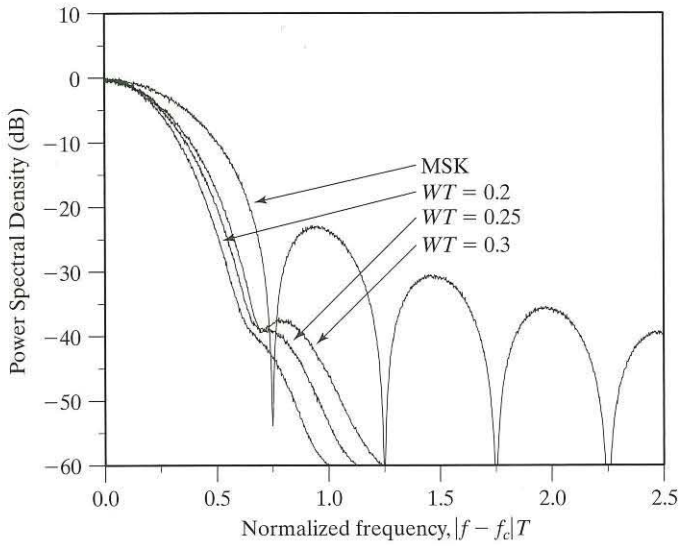

FIGURE 3.21 Power spectra of MSK and GMSK signals for varying time–bandwidth product.

The corresponding degradation in noise performance incurred by the choice of $WT = 0.3$ is about 0.46 dB, compared with the standard MSK, which is a small price to pay for the highly desirable spectral compactness of the GMSK signal; this degradation in performance is justified in Section 3.12.

3.8 FREQUENCY-DIVISION MULTIPLE ACCESS[7]

In *frequency-division multiple access* (FDMA), the available spectrum is divided into a set of continuous frequency bands labeled 1 through N, and the bands are assigned to individual mobile users for the purpose of communications on a continuous-time basis for the duration of a telephone call. An issue of particular concern in the design of FDMA systems is that of adjacent channel interference, which is discussed in Section 3.9 in the context of different modulation techniques. In any event, to reduce this form of interference, two precautionary measures are usually taken:

1. The power spectral density of the modulated signal is carefully *controlled* so that the power radiated into the adjacent band is 60 to 80 dBs below that in the desired band; this system requirement, in turn, requires the use of highly selective filters.
2. *Guard bands* are inserted as buffer zones in the channel assignment, as illustrated in Fig. 3.22. The guard bands provide additional protection because of the impossibility of achieving the ideal (i.e., brick wall) filtering characteristic to separate the different users.

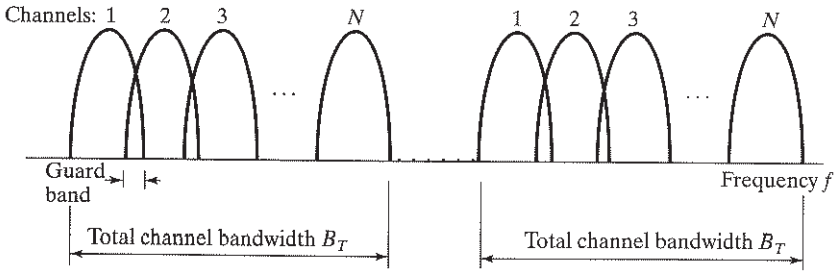

FIGURE 3.22 Channel allocations in FDD/FDMA system.

In a wireless communication system, it is highly desirable for a mobile user to send information-bearing signals to the base station while receiving information-bearing signals from the base station itself on the same antenna. To provide for such a capability, some form of *duplexing* is used, either in the frequency domain or in the time domain. Figure 3.23 shows the block diagram of a *frequency-division diplexer* (FDD), which incorporates a modulator and demodulator, with the modulator acting on input *control* data and the demodulator producing corresponding output data.

For *frequency-division diplex* (FDD) transmissions in an FDMA system accommodating, N users, as illustrated in Fig. 3.22, there are two groups of subbands or channels:

- One group of N contiguous subbands, occupying a total bandwidth of B_T hertz, is used for *forward-link radio transmissions* from a base station to its mobile users; a forward-link is also referred to as a *downlink*.
- A similar group of N subbands also occupying a total bandwidth of B_T hertz. These subbands are used for reverse-link radio transmissions from the mobile users to their base station; a reverse-link is also referred to as an *up-link*.

Separation of these two groups of subbands is facilitated by the FDD, as illustrated in Fig. 3.23. Each mobile user is allocated a subband in both groups of subbands, as well as the FDD band.

The first generation of analog wireless communication systems uses FDMA/FDD, with speech signals being transmitted over the forward or reverse links through frequency modulation (FM). The data control functions are performed digitally by means of frequency-shift keying (FSK) for data transmission, which is the simplest digital form of frequency modulation. Specifically, one frequency is used for the transmission

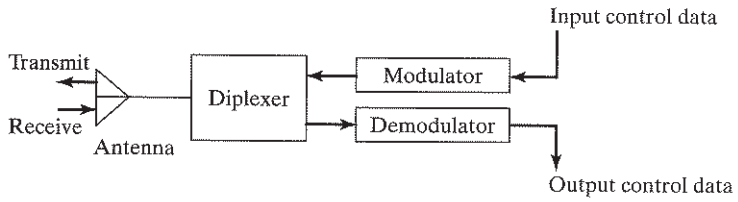

FIGURE 3.23 Frequency-division diplexer.

of binary symbol 0 and a different frequency is used for the transmission of binary symbol 1. (See Subsection 3.7.2.)

A useful feature of FDMA systems is that, for each mobile user, the radio transmission takes place over a narrow channel of bandwidth (B_T/N) hertz, in which case any fading encountered in the course of transmission tends to be flat. From Chapter 2 we recall that flat fading is relatively easy to handle, hence the adoption of FDMA for the first generation of analog wireless communications. However, FDMA systems suffer from a serious practical limitation: A separate transmitter-receiver, termed *transceiver*, is needed at the base station for each of the mobile users actively operating in its coverage area. This limitation is mitigated by using TDMA, which is discussed in Chapter 4.

3.9 TWO PRACTICAL ISSUES OF CONCERN

As already stated, modulation lies at the heart of an FDMA system. Moreover, modulation, in one form or another, features prominently in the other types of multiple-access techniques. It is therefore important that we provide a comparative assessment of the various modulation techniques in the context of wireless communications, keeping in mind two pertinent practical issues:

1. Adjacent channel interference
2. Power amplifier nonlinearity

These two issues are discussed in this section. The comparison of modulation strategies in the context of these issues is deferred to the next section.

3.9.1 Adjacent Channel Interference[10]

A performance measure of profound practical importance in the design of wireless communication systems is that of spectral efficiency. Formally, *spectral efficiency* is defined as the ratio of the permissible source rate (in bits per second) to the available channel bandwidth (in hertz); hence, spectral efficiency is measured in bits/s/Hz. The more spectrally efficient a multiple-access system is, the greater will be the number of mobile users that can operate satisfactorily inside the prescribed channel bandwidth. In the FDMA system discussed in Section 3.8, the spectral efficiency depends on how closely the individual channels (frequency bands) can be spaced. There are several factors that limit this spacing, the most important of which is *adjacent channel interference* (ACI), referred to in the preceding discussions on GMSK and FDMA.

To illustrate the ACI problem, consider a rectangular pulse $p(t)$ of duration T and whose energy spectral density is defined by

$$\begin{aligned} |P(f)|^2 &= \left|\frac{\sin(\pi fT)}{(\pi fT)}\right|^2 \\ &= \text{sinc}^2(fT) \end{aligned} \tag{3.70}$$

Equation (3.70) also defines the power spectrum of a random binary data stream in which the symbols 0 and 1, represented by the amplitudes ±1, occur with equal

probability. Suppose the pulse $p(t)$ is transmitted simultaneously over two channels labeled 0 and 1. Figure 3.24 presents two plots of Eq. (3.70). One plot, shown as a solid line, pertains to channel 0; the other plot, shown as a dashed line, pertains to channel 1. Pulse transmission over these two channels is accounted for by the main lobes of the respective spectra. The sidelobes account for the ACI problem. Specifically, the first two sidelobes to the left of the main lobe of channel 1 give rise to ACI in channel 0. By the same token, the first two sidelobes to the right of the main lobe of channel 0 give rise to ACI in channel 1. In both cases, however, the levels of the sidelobes are suppressed by only 13 and 17 dB relative to the main lobes (i.e., desired signals). If we combine the interference generated in channel 0 by signal transmission in channel 1 with the interference generated by the next adjacent channels +2 and −1, we see that the transmission of the desired signal in channel 0 is significantly affected by the presence of ACI.

In Fig. 3.24, the received signals are all assumed to be at the same power level. In wireless communications, signals may be transmitted at the same power level. Because of the effects of propagation over different distances to the receivers, the transmitted signals are then received at widely differing power levels with the result that in some systems, the range of received signal levels can vary by as much as 80 dB or more. If, for example, the signal on channel 1 is received 30 dB stronger than the signal on channel 0, it is possible for the ACI produced by channel 1 to completely overwhelm the desired signal in channel 0. This phenomenon is called the *near-far problem.*

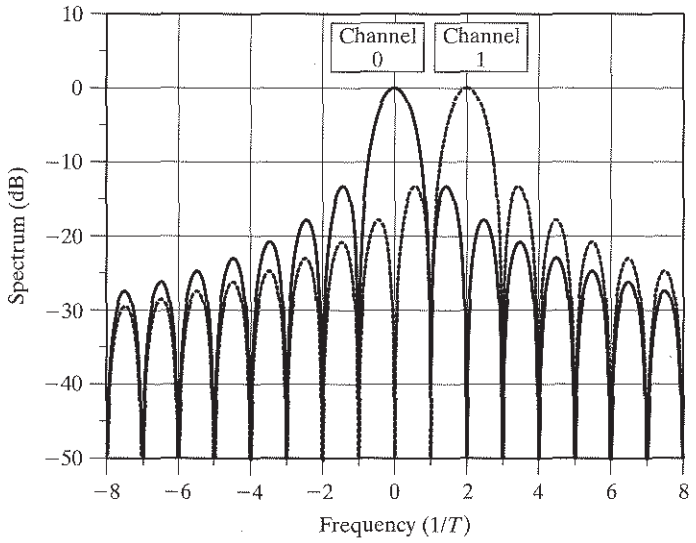

FIGURE 3.24 Adjacent-channel interference problem, illustrated with the spectrum of a rectangular pulse of duration T.

3.9.2 Power Amplifier Nonlinearity

One of the most critical constraints imposed on mobile radio terminals is their *limited battery power*. Terminals are designed for a certain battery life or time between recharges, and the corresponding circuitry must respect the underlying power budget. A significant consumer of power in mobile radios is the *transmit power amplifier*. For this reason, considerable attention is always paid to this component of the wireless system.

There are many amplifier designs, and they have been traditionally categorized in the electronics literature as Class A, Class B, Class AB, Class C, Class D, and so on, where Class A represents a linear amplifier and the amplifier classes that follow are typically increasingly nonlinear. Although Class A is considered to be a linear amplifier, no amplifier is truly linear; what linearity means in this context is that the operating point is chosen such that the amplifier behaves linearly over the signal range. The drawback of a Class A amplifier is that it is power inefficient. Typically, 25 percent or less of the input power is actually converted to radio frequency (RF) power; the power that is left is converted to heat and, therefore, wasted. The remaining amplifier classes are designed to provide increasingly improved power efficiency, but at the expense of making the amplifier increasingly more nonlinear.

In Fig. 3.25, we show the measured gain characteristic of a solid-state power amplifier at two different frequencies: 1626 MHz and 1643 MHz. The curves show that the amplifier gain is approximately constant, that is, the amplifier is linear over a wide range of inputs. However, as the input level increases, the gain decreases, indicating that the amplifier is saturating. It can also be seen that there is a significant difference in

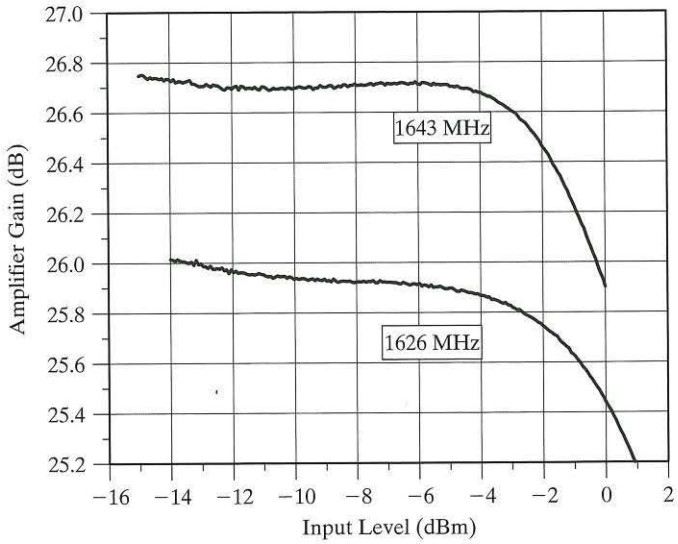

FIGURE 3.25 Gain characteristic of a solid-state amplifier at two different operating frequencies: 1626 MHz and 1643 MHz.

amplifier performance at different frequencies. If this amplifier is operated at an average input level of –10 dBm with an amplitude swing of ±2 dB, then the amplifier would be considered linear. If, however, the signal has an amplitude swing of ±10 dB, the amplifier would be considered nonlinear. The fact that the gain is not constant over all input levels means that the amplifier introduces *amplitude distortion* in the form of amplitude modulation (AM). Since the amplitude distortion depends upon the input level, it is referred to as *AM-to-AM distortion.*

An ideal amplifier does not affect the phase of an input signal, except possibly for a constant phase rotation. Unfortunately, a practical amplifier behaves quite differently, as illustrated in Fig. 3.26, which shows the phase characteristic of the same power amplifier considered in Fig. 3.25. The fact that the phase characteristic is not constant over all input levels means that the amplifier introduces *phase distortion* in the form of phase modulation (PM). Since the phase distortion depends upon the input level, this second form of distortion is called *AM-to-PM distortion.*

An "ideal" amplifier nonlinearity has the AM-to-AM characteristic illustrated in Fig. 3.27. That is, the amplifier acts linearly up to a given point, whereafter it sets a hard limit on the input signal. This can sometimes be achieved by placing appropriate compensation around a nonideal amplifier. With this ideal nonlinearity, the phase distortion is assumed to be zero. The operating point of the amplifier is often specified as the *input back-off*, defined as the root-mean-square (rms) input signal level $V_{\text{in, rms}}$ relative to the saturation input level $V_{\text{in, sat}}$ in dB. That is,

$$\text{Input back-off} = 10\log_{10}\left(\frac{V_{\text{in, rms}}}{V_{\text{out, sat}}}\right)^2 \tag{3.71}$$

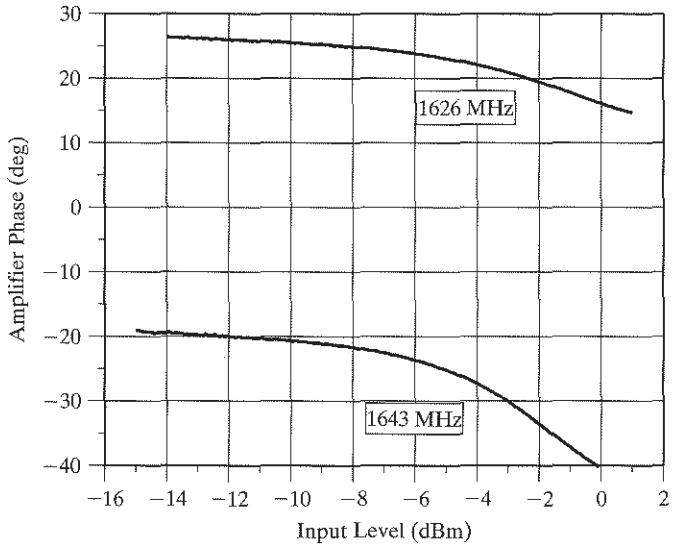

FIGURE 3.26 Phase characteristic of a nonlinear amplifier at two different operating frequencies: 1626 MHz and 1643 MHz.

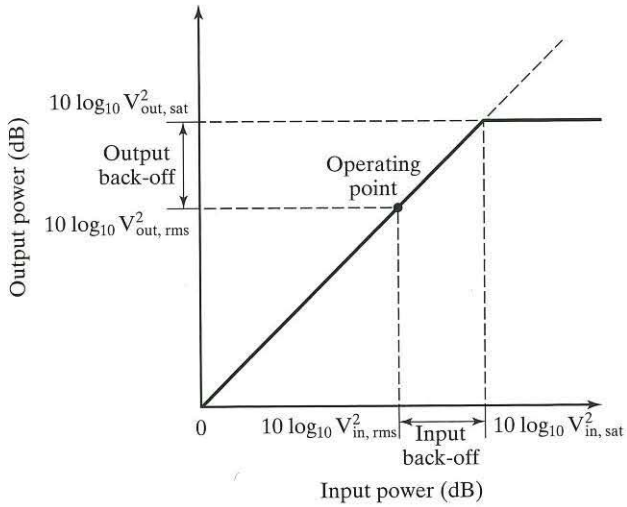

FIGURE 3.27 AM/AM characteristic of an "ideal" form of amplifier nonlinearity.

Alternatively, the operating point can be expressed in terms of the *output back-off*, defined as follows:

$$\text{Output back-off} = 10\log_{10}\left(\frac{V_{\text{out, rms}}}{V_{\text{out, sat}}}\right)^2 \tag{3.72}$$

where $V_{\text{out, rms}}$ is the rms output signal level and $V_{\text{out, sat}}$ is the saturation output level. In both Eqs. (3.71) and (3.72), the closeness to saturation determines the amount of distortion introduced by the amplifier.

3.10 COMPARISON OF MODULATION STRATEGIES FOR WIRELESS COMMUNICATIONS

With the material presented in Section 3.9 at hand, we are now ready to address the issue of comparing the modulation strategies described in Sections 3.3 and 3.7, linear and nonlinear modulation strategies for wireless. The comparison will be made under two headings: linear and nonlinear channels, which is determined by whether the transmit power amplifier is operated in its linear or nonlinear region. In both cases, the discussion will focus on digital modulation techniques because of the ever-increasing pervasiveness of digital wireless communications.

3.10.1 Linear Channels

Consider first the effect of adjacent channel interference, in which case the *transmit spectrum* is clearly an important criterion for selecting between modulation strategies in a wireless system. The transmit spectrum is determined by three factors:

1. *Pulse shaping* that is performed as part of the modulation process, for example, rectangular pulse shaping, Gaussian filtering, and so on.
2. *Other filtering* that may be necessary in the transmitter RF chain, to remove the presence of images that may occur during the up-conversion to an RF frequency.
3. *Presence of nonlinearities* in the transmitter RF chain, for example, power amplifiers that may be operated close to saturation.

In general, the "other filtering" is often the least significant of the three and we shall ignore it in this presentation. This subsection is devoted to linear channels; nonlinear effects will be covered in the next subsection.

In Fig. 3.28, we compare the baseband transmit spectra of the three modulation strategies discussed in this chapter.[11] The baseband spectra shown in the figure are normalized to the same peak spectral density. The three spectra correspond to

1. QPSK with rectangular pulse shaping; this spectrum is characterized as the $\text{sinc}^2(fT)$ function where T is the symbol duration.
2. MSK with its pulse shape corresponding to half a sine wave.
3. QPSK with root raised-cosine (RC) pulse shaping with 50% rolloff as defined in Section 3.4.

Comparing QPSK with rectangular pulse shaping to MSK, we find that QPSK has a narrower mainlobe but the sidelobes decrease much more slowly than MSK. For $f >> 1/T$, the baseband power spectrum of MSK signal falls off as the inverse fourth power of frequency, whereas in the case of QPSK, it falls off as the inverse square of frequency. Consequently, the ACI produced by MSK is less troublesome than that produced by QPSK.

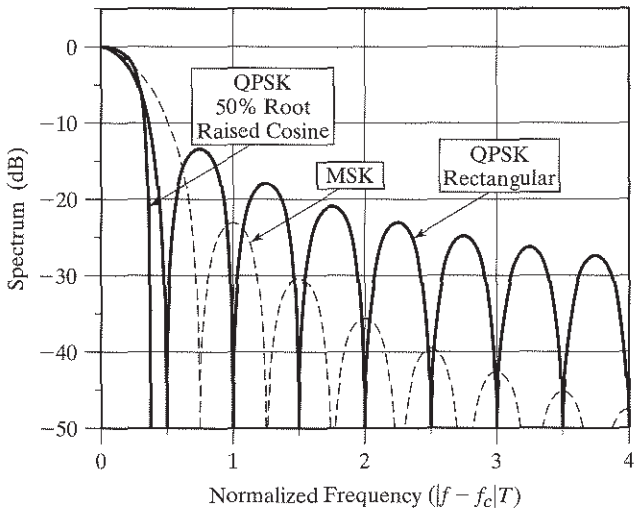

FIGURE 3.28 Comparison of the spectra of QPSK with rectangular pulse shaping, MSK, and QPSK with 50% root-raised-cosine pulse shaping.

Comparing QPSK with root pulse shaping to the other two modulation strategies, the advantages of the former linear method of modulation are clear:

- The main lobe of QPSK with root RC pulse shaping is the narrowest among the three modulation strategies considered herein.
- The QPSK with root RC pulse shaping has negligibly small sidelobes.

In a linear channel, the QPSK with root RC pulse shaping is clearly the superior strategy in terms of minimizing adjacent channel interference. In practical implementations of root RC filters, there are sidelobes; however, by careful filter design, these sidelobes can be easily kept 40 to 50 dB below the level of the main lobe. Note that in a linear channel, the spectra of the three modulations—QPSK, OQPSK, and $\pi/4$-QPSK—are identical when they use the same pulse shaping.

3.10.2 Nonlinear Channels

The use of a nonlinear transmit power amplifier may have a significant effect on system performance, depending upon on how close to saturation the amplifier is operated. The effect of the amplifier also depends on the type of modulation employed. The two examples of Subsection 3.10.1, namely, rectangular QPSK and MSK, are constant-envelope modulations. Since the nonlinear effects depend upon envelope variation, the spectra of these modulations are unaffected by a nonlinear amplifier. The same can be said of the GMSK, the power spectrum, which is illustrated in Fig. 3.21.

However, the same cannot be said of modulations such as QPSK with root RC filtering, which rely on its envelope variations to produce a compact spectrum. In fact, because the envelope characteristics of the three modulation schemes QPSK, OQPSK, and $\pi/4$ -shifted QPSK are different, the effect of the nonlinear amplifier on the respective modulated signals will differ. In Fig. 3.23, we compare the spectra of these three modulations after passing through an "ideal" nonlinear amplifier. This ideal nonlinearity has a linear response up to the saturation point, at which point it clips the complex envelope of the signal to a constant level in accordance with Fig. 3.27. For the results shown in Fig. 3.29, the average input power was 1 dB below saturation.

For this ideal nonlinearity, the different responses of the three modulation types are evident but the variation between them is small. Of the three, OQPSK has the better spectral response. Compared to the spectrum experienced in a linear channel, the nonlinear channel has raised the sidelobes up to 30 dB with a slow rolloff. This is still superior to MSK, but not as good as various forms of GMSK, whose power spectra are shown in Fig. 3.21.

In practice, the choice of modulation is a tradeoff between various characteristics. The transmit spectrum is an important consideration in the tradeoff. Other considerations include the simplicity of detection and error-rate performance; the latter issue is discussed in Section 3.12. For many new wireless communication systems, it appears that linear modulation strategies, such as QPSK with root raised-cosine filtering, have emerged as the methods of choice.

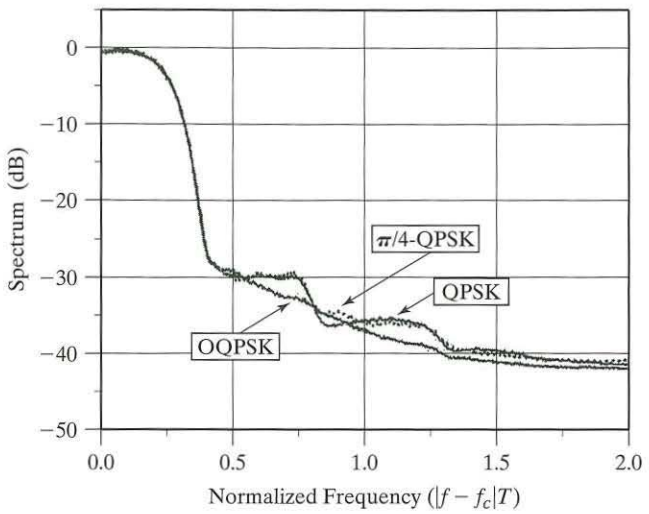

FIGURE 3.29 Comparison of different QPSK spectrum when passed through an ideal nonlinear amplifier with a 1 dB input backoff. All three modulation schemes use root-raised-cosine pulse shaping with 50% rolloff.

One last comment on phase distortion is in order. Just as with AM-to-AM distortion, AM-to-PM distortion varies with the signal level at the power amplifier input. With small signal levels, there is little phase distortion to be concerned with. As the signal level approaches the point of saturating the amplifier, the phase distortion increases. For modulation schemes with a constant envelope, AM-to-PM distortion amounts to a simple phase rotation of the signal and is not really a distortion at all. For nonconstant envelope modulations, the amount of distortion that can be tolerated depends upon the constellation. With BPSK modulation—where there is a shift of 180 degrees between constellation points—significant phase distortion can be tolerated. However, with more complicated modulation schemes such as 64-QAM, where the phase shift between constellation points can be as low as 10 degrees or less, the tolerance to phase distortion is much lower.

3.11 CHANNEL ESTIMATION AND TRACKING

From Chapter 2 we recall that fading is a major source of channel impairment in wireless communications. By definition, however, with slowly fading channels, the phase change over one symbol interval due to fading is *small* compared with the phase change in the transmitted signal. Accordingly, we can exploit this property and use differential detection to track phase variations in the transmitted signal. Differential detection is simple to implement—hence its wide application in practice. But the use of differential detection is usually limited to phase modulation systems, although there are some extensions to systems that combine amplitude and phase modulation.

Another method for tracking a slowly fading channel is based on pilot symbol transmission, which operates by sending known symbols at regular intervals (bursts) throughout the course of data transmission. Pilot symbol transmission schemes offer a potential performance advantage over differential detection in that, in addition to tracking, they can be used to estimate the impulse response of the channel; hence, they can improve the noise performance of the receiver. This performance advantage, however, is attained at the cost of a somewhat reduced spectral efficiency and increased system complexity.

3.11.1 Differential Detection

In pass-band transmission systems, the primary concern is the recovery of the desired signal from its assigned frequency channel. If we assume linear modulation, then in light of the material presented in Section 3.3, the transmitted band-pass signal can be represented as

$$s(t) = A\mathrm{Re}\left\{m(t)e^{j2\pi f_c t}\right\} \tag{3.73}$$

where $m(t)$ is to be defined, A is the transmit amplitude and f_c is the transmit carrier frequency. In previous sections of this chapter, we have assumed perfect recovery of the carrier frequency and phase in converting to complex baseband. Although the nominal frequency of the channel may be known, there will be some frequency errors due to inaccuracies of the transmit and receive oscillators. In practice, the received complex baseband signal is accurately represented as

$$\tilde{x}(t) = A'm(t)e^{j(2\pi\Delta f t + \phi)} + w(t) \tag{3.74}$$

where A' is the received amplitude, Δf is the residual frequency error after downconversion, ϕ is the residual phase error, and $w(t)$ is the additive channel noise. Equation (3.74) assumes an AWGN channel. It is left to the baseband circuitry to acquire and track this residual frequency and phase error.

In wireless channels, the coherent recovery and tracking of the residual frequency error Δf can be difficult. A simple, robust technique for recovering the carrier frequency is *differential detection*.[12] In this technique, it is assumed that the receiver "knows" the transmit frequency to within a certain accuracy; typically, this error is less than 5% of the data rate.

Differential detection can be illustrated with linear modulation, with the complex baseband signal written as

$$m(t) = \sum_k b_k p(t - kT) \tag{3.75}$$

where $p(t)$ is the pulse shape used to modulate each data symbol b_k. If the residual frequency error is small compared with the signal bandwidth, it is practically transparent

to the matched-filtering process, in which case the sampled value of the matched filter output is given by the convolution integral

$$y_k = \int_{nT}^{(n+1)T} p^*(t - kT)\tilde{x}(t)dt$$

which approximates to

$$y_k = A' b_k e^{j(2\pi\Delta f kT + \phi)} + w_k \tag{3.76}$$

where it is assumed that the pulse shape is normalized to unit energy. (The characterization of *matched filters* is discussed in Appendix D.) To simplify the presentation, the received amplitude A' is hereafter ignored without loss of generality.

If the data stream is *differentially encoded* at the transmitter such that $b_k = a_k b_{k-1}$, where a_k is the original data symbol, then the original data can be recovered with the "delay-and-multiply" circuit shown in Fig. 3.30. Mathematically, this circuit performs the following operation:

$$\begin{aligned}
\hat{a}_k &= y_k y_{k-1}^* \\
&= [b_k e^{(j2\pi\Delta f kT + \phi)} + w_k][b_{k-1} e^{(j2\pi\Delta f(k-1)T + \phi)} + w_{k-1}]^* \\
&= b_k b_{k-1}^* e^{j2\pi\Delta f T} + \eta_k \\
&\approx a_k + \eta_k
\end{aligned} \tag{3.77}$$

In the last line of Eq. (3.77), it is assumed that Δf is small enough for $e^{j2\pi\Delta f T} \approx 1$. The noise term η_k represents the sum of three noise-related terms resulting from the multiplication of terms in the second line of the equation. The approach described herein is well suited for both BPSK and QPSK forms of modulation.

To sum up, the practical significance of differential detection is twofold:

1. Differential detection is based on phase differences. It therefore lends itself naturally to linear phase modulation schemes, but it can also be applied to nonlinear phase modulation schemes,[12] so long as the phase is differentially encoded.

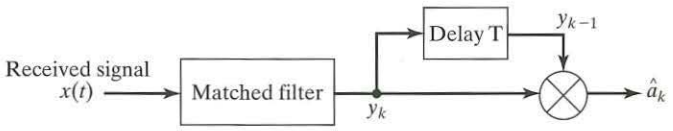

FIGURE 3.30 Delay-and-multiply circuit for differential detection.

2. If amplitude is not important (i.e., the data stream is phase modulated), then the effects of fading are similar to a small time-varying frequency error on the received signal. Provided that the fading is slow relative to the data rate, differential detection may be applied to such wireless systems.

For these reasons, differential detection has established itself as a robust strategy for the detection of phase modulation over flat-fading channels.

3.11.2 Pilot Symbol Transmission[13]

In some applications, it may be necessary to perform *coherent detection* so as to avoid the degradation in receiver performance associated with differential detection; or it could be that some of the information about the data is encoded in amplitude variations, as in QAM for example. In these cases, the insertion of *known pilot symbols* in the transmit data stream is the preferred approach.

If the channel exhibits flat fading, then the complex baseband received signal of Eq. (3.74) may be modeled as

$$\tilde{x}(t) = \tilde{\alpha}(t)m(t) + \tilde{w}(t) \tag{3.78}$$

where $\tilde{\alpha}(t)$ is a complex Rayleigh fading process. In Eq. (3.78), we have included the received amplitude A' as part of the fading process $\tilde{\alpha}(t)$. Also, to simplify matters, we have neglected the small frequency error Δf. In a manner analogous to the development of Eq. (3.76), with linear modulation and Nyquist pulse shaping, the samples at the output of the matched filter can be represented in discrete time as

$$y_k = \alpha_k b_k + w_k \tag{3.79}$$

where b_k is the data symbol in symbol period k, w_k is the corresponding white Gaussian noise sample, and α_k is the complex gain due to the fading process. The model of Eq. (3.79) is based on the assumption that the fading is approximately constant over the significant portion of the pulse length. If we use a Rayleigh-fading channel model, then $\{\alpha_k\}$ are samples of a Rayleigh fading process.

If the pilot symbols are known at regular times—say, at $k = Ki$, where K is the pilot-symbol spacing and i is an integer—(i.e., the pilot symbols are known at the receiver), then we can use Eq. (3.79) to form the equation

$$\begin{aligned} h_{Ki} &= b_{Ki}^* y_{Ki} \\ &= \alpha_{Ki}|b_{Ki}|^2 + b_{Ki}^* w_{Ki} \\ &= \alpha_{Ki} + w_{Ki} \end{aligned} \tag{3.80}$$

The development from the second to the third line of Eq. (3.80) assumes that the pilot symbols are BPSK modulated—that is, $b_{Ki} = \pm 1$. (The random nature of the noise term w_{Ki} is unaffected by whether b_{ki} equals -1 or $+1$). It is straightforward to show that, under these assumptions, the statistical properties of the noise samples are unchanged.

The samples $\{h_{Ki}\}$ can be used as estimates of the fading process, since the known modulation has been removed. However, they are corrupted by noise.

In practice, we usually wish to improve upon this estimate of the fading process by smoothing adjacent estimates. In particular, we make a *linear estimator* of the fading, based on the $2L + 1$ surrounding samples, obtaining

$$\hat{\alpha}_{Ki} = \sum_{m=-L}^{L} a_m h_{K(i+m)} \tag{3.81}$$

where the a_m are complex weighting parameters for smoothing the estimate. These parameters are often chosen to minimize the expected error in the estimate. That is, the $\{a_m\}$ are chosen to minimize the *cost function*

$$\begin{aligned} J &= \mathbf{E}[|\hat{\alpha}_{Ki} - \alpha_{Ki}|^2] \\ &= \mathbf{E}\left[\left| \sum_{m=-L}^{L} a_m h_{K(i+m)} - \alpha_{Ki} \right|^2\right] \end{aligned} \tag{3.82}$$

where $\mathbf{E}$ is the statistical expectation operator. Equation (3.82) can be expanded as follows:

$$\begin{aligned} J = \mathbf{E}\Bigg[|\alpha_{Ki}|^2 &+ \sum_{m=-L}^{L} \sum_{n=-L}^{L} a_m h_{K(i+m)} a_n^* h_{K(i+n)}^* \\ &- \sum_{m=-L}^{L} a_m h_{K(i+m)} \alpha_{Ki}^* - \sum_{n=-L}^{L} a_n^* h_{K(i+n)}^* \alpha_{Ki} \Bigg] \end{aligned} \tag{3.83}$$

Next, we define the average power

$$P = \mathbf{E}|\alpha_{Ki}|^2 \tag{3.84}$$

and the autocorrelation of the fading-plus-noise samples as

$$\begin{aligned} \mathbf{E}[h_{K(i+m)} h_{K(i+n)}^*] &= \mathbf{E}[(\alpha_{K(i+m)} + w_{K(i+m)})(\alpha_{K(i+n)}^* + w_{K(i+n)}^*)] \\ &= \mathbf{E}[\alpha_{K(i+m)} \alpha_{K(i+n)}^*] + \mathbf{E}[w_{K(i+m)} w_{K(i+n)}^*] \\ &= \begin{cases} r_{K(m-n)} & m \neq n \\ r_0 + \sigma^2 & m = n \end{cases} \end{aligned} \tag{3.85}$$

where $r_m = R(mT)$ is the sampled autocorrelation function of the fading process and T is the symbol period. The second line of Eq.(3.85) follows from the first line due to the statistical independence of the noise and fading processes. The third line follows from the second line because the noise samples are drawn from a white-noise process and are therefore uncorrelated. Similarly, we find that

$$\begin{aligned} \mathbf{E}[h_{K(i+m)}\alpha^*_{Ki}] &= \mathbf{E}[(\alpha_{K(i+m)} + w_{K(i+m)})\alpha^*_{Ki}] \\ &= \mathbf{E}[\alpha_{K(i+m)}\alpha^*_{Ki}] + \mathbf{E}[w_{K(i+m)}w^*_{K(i+n)}] \\ &= r_{Km} \end{aligned} \tag{3.86}$$

Now we define the matrix $\mathbf{R}$ with elements $R_{ij} = r_{K(i-j)}$ and the vector $\mathbf{r}$ with elements r_{Ki}. Then, by using Eqs. (3.84) through (3.86), we may rewrite Eq. (3.83) in the compact form

$$J = P + \mathbf{a}^\dagger \mathbf{R}\mathbf{a} - \mathbf{a}^\dagger \mathbf{r} - \mathbf{r}^\dagger \mathbf{a} \tag{3.87}$$

where $\mathbf{a} = [a_{-L},...,a_L]^T$, and the superscript denotes Hermitian transposition. By completing the square, Eq.(3.87) can be rewritten as

$$J = P - \mathbf{r}^\dagger \mathbf{R}^{-1}\mathbf{r} + (\mathbf{a} - \mathbf{R}^{-1}\mathbf{r})^\dagger \mathbf{R}(\mathbf{a} - \mathbf{R}^{-1}\mathbf{r}) \tag{3.88}$$

Since the vector $\mathbf{a}$ is the only free parameter in this equation, the cost function J is clearly minimized if

$$\mathbf{a} = \mathbf{R}^{-1}\mathbf{r} \tag{3.89}$$

Equation (3.89) is a form of what is known as the *Wiener–Hopf equation* in the signal-processing literature. The weighting parameter vector $\mathbf{a}$ depends only on the correlation properties of the fading process and the noise. It provides a solution for the weighting that is an optimal estimate of the fading in the mean-square error sense. (Appendix I also includes a discussion of adaptive filters in the context of adaptive antennas.)

Problem 3.12 Alternatively, we can derive the Wiener–Hopf equation (3.89) by differentiating the cost function J with respect to the weighting parameter vector $\mathbf{a}$, setting the result equal to zero, and then solving for $\mathbf{a}$. Show that this procedure also leads to Eq. (3.89). ■

The combination of data estimation and channel tracking may be implemented as shown in Fig. 3.31. The signal is first matched-filtered, as described by Eq. (3.76), to produce, at its output, $\{y_k\}$. The pilot samples are then demultiplexed from the data,

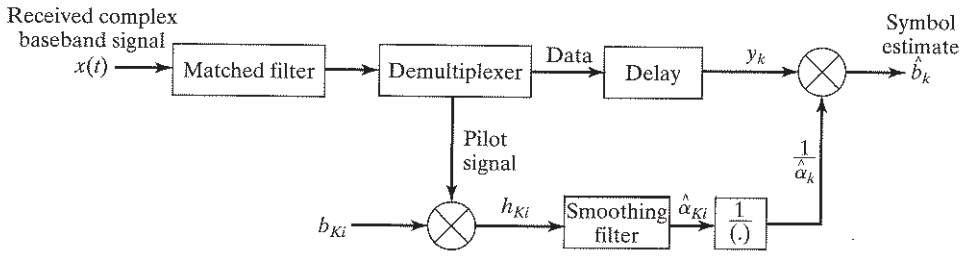

FIGURE 3.31 Pilot scheme for tracking and compensating for channel variations. Note: The b_{Ki} are pilot symbols that, by definition, are known at the receiver.

and the data modulation is removed from the pilot samples according to Eq. (3.80). Then the pilot samples are processed by a smoothing filter, in accordance with Eq. (3.81). The parameter vector of the smoothing filter is given by Eq. (3.89). Since the smoothing filter (i.e., linear estimator) introduces a delay, the data in the upper branch of Fig. 3.31 must be delayed to match the delay introduced by the filter. After this delay, we compensate for the channel by multiplying the received sample stream by the inverse of the estimated channel, $\hat{b}_k = y_k/\hat{\alpha}_k$ to obtain estimates of the data. We note that, since the pilot samples are spaced K symbols apart, the smoothing filter shown in Fig. 3.31 will also need to perform some interpolation to provide estimates of the fading channel on the intervening symbols.

If we plot the frequency response of the finite-duration impulse-response (FIR) filter defined by the parameter vector **a**, we find that this filter is approximately matched to the fading process. In practice, if the fading bandwidth is unknown, we would typically consider the worst-case fading in designing this filter. The pilot symbols will not only track the fading but will also track any residual frequency and phase errors between the transmitter and receiver, as long as such errors fall within the bandwidth of the tracking filter. Thus, the pilot symbols permit channel estimation, tracking, and coherent detection in a single operation.

Pilot-symbol techniques do introduce some losses. Energy that could otherwise be devoted to the information symbols must be devoted to the pilot symbols. In addition, the bandwidth of the transmitted signal must be expanded to accommodate the inclusion of the pilot symbols as well as the data. What is the minimum number of pilot symbols required? From Eq. (3.80), we see that the pilot symbols are effectively sampling the fading process. Consequently, if the pilot symbol rate is at least twice the bandwidth of the fading process, then, by Nyquist's theorem, we should be able to estimate the channel reliably. In practice, the pilot symbol rate is typically several times the fading process bandwidth, compensating for the effects of noise on the samples and simplifying the interpolation process for estimating the fading on data symbols between the pilot symbols.

Problem 3.13 If the transmission frequency is 1.9 GHz and the receiver uses a simple quartz crystal for regenerating the channel frequency, what is the range of residual frequency error on the baseband signal? A quartz crystal typically has an accuracy of 10 parts per million (ppm). If the data rate is 9.6 kilobits/s, how much phase rotation will occur during one symbol period with this error?
Ans. ±19 kHz; $\pm 2\pi$ *radians (approximately).* ■

Problem 3.14 Show that frequency shifting and phase rotation do not affect the statistical properties of zero-mean white Gaussian noise. ■

3.12 RECEIVER PERFORMANCE: BIT ERROR RATE

A study of digitally modulated signals would be incomplete without discussing how the presence of channel noise and channel fading would affect the performance of a wireless receiver. Recognizing the underlying input-output binary operation of a digital wireless communication system varies randomly with time, the performance of the system is measured in terms of the *average probability of symbol error*, P_e. An error is incurred when the symbol 1 in transmitted but the receiver decides in favor of the symbol 0, or vice versa. The P_e is obtained by averaging over these two conditional error probabilities with respect to the prior probabilities of symbols 0 and 1. In such a situation, P_e is often referred to as the *bit error rate* (BER).

In what follows, the receiver performance is considered under the following two conditions:

- The presence of additive channel noise
- The presence of multiplicative noise exemplified by a frequency-flat, slowly fading channel

For coherent receivers operating on simple modulation schemes, such as BPSK, QPSK, BFSK, and MSK, exact formulas are derivable for P_e. On the other hand, in the case of coherent receivers operating on more elaborate modulation schemes, such as Gaussian-filtered MSK, M-ary PSK, M-ary QAM, and M-ary FSK, the analytic approach becomes formidable and we therefore resort to the use of error bounds or computer simulations to derive approximate formulas for P_e.

3.12.1 Channel Noise[14]

In a *coherent receiver*, the locally generated carrier is synchronized with the carrier in the transmitter in both phase and frequency. To achieve synchronization, additional circuitry is required in designing the receiver. The receiver design can be simplified by ignoring phase information, in which case the receiver is said to be *noncoherent*. However, the design simplification is accomplished at the expense of degraded noise performance.

First of all, reference to Table 3.4 presents a summary of the bit error rates for the following receivers:

- Coherent binary phase-shift keying (BPSK)
- Coherent quadriphase-shift keying (QPSK)
- Coherent binary frequency-shift keying (BFSK)

TABLE 3.4 Summary of Formulas for the bit error rate (BER) of coherent and noncoherent digital communication receivers.

Signaling Scheme	BER (Additive white Gaussian noise channel)	BER (Slow Rayleigh fading channel)
(a) Coherent BPSK Coherent QPSK Coherent MSK	$\frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_b}{N_0}}\right)$	$\frac{1}{2}\left(1-\sqrt{\frac{\gamma_0}{1+\gamma_0}}\right)$
(b) Coherent BFSK	$\frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_b}{2N_0}}\right)$	$\frac{1}{2}\left(1-\sqrt{\frac{\gamma_0}{2+\gamma_0}}\right)$
(c) Binary DPSK	$\frac{1}{2}\exp\left(-\frac{E_b}{N_0}\right)$	$\frac{1}{2(1+\gamma_0)}$
(d) Noncoherent BFSK	$\frac{1}{2}\exp\left(-\frac{E_b}{2N_0}\right)$	$\frac{1}{2+\gamma_0}$

Definitions:
E_b = transmitted energy per bit
N_0 = one-sided power spectral density of channel noise
γ_0 = mean value of the received energy per bit-to-noise spectral density ratio

- Coherent minimum shift keying (MSK)
- Differential phase-shift keying (DPSK)
- Noncoherent binary frequency-shift keying (BFSK)

In noncoherent BFSK, the receiver consists basically of a frequency discriminator that responds only to shifts imposed on the carrier frequency by the incoming binary data. DPSK combines differential detection with binary phase-shift keying; it may, therefore, be viewed as the noncoherent version of BPSK; differential detection was discussed in subsection 3.11.1.

In Fig. 3.32, we have used the formulas summarized in Table 3.4 for an additive white Gaussian noise channel to plot the BER as a function of the signal energy per bit-to-noise spectral density ratio, E_b/N_0, which is also referred to as the signal-to-noise ratio (SNR). On the basis of the noise performance curves shown in the figure, we can make the following observations:

- The BERs for all the receivers listed in the previous paragraph decrease monotonically with increasing E_b/N_0; the curves have a similar shape in the form of a *waterfall*.
- For any prescribed E_b/N_0, coherent versions of BPSK, QPSK, and MSK produce a smaller BER than does any of the other modulation schemes.
- For a prescribed BER, coherent BPSK and DPSK require an E_b/N_0 that is 3 dB less than the corresponding values for coherent BFSK and noncoherent BFSK, respectively.

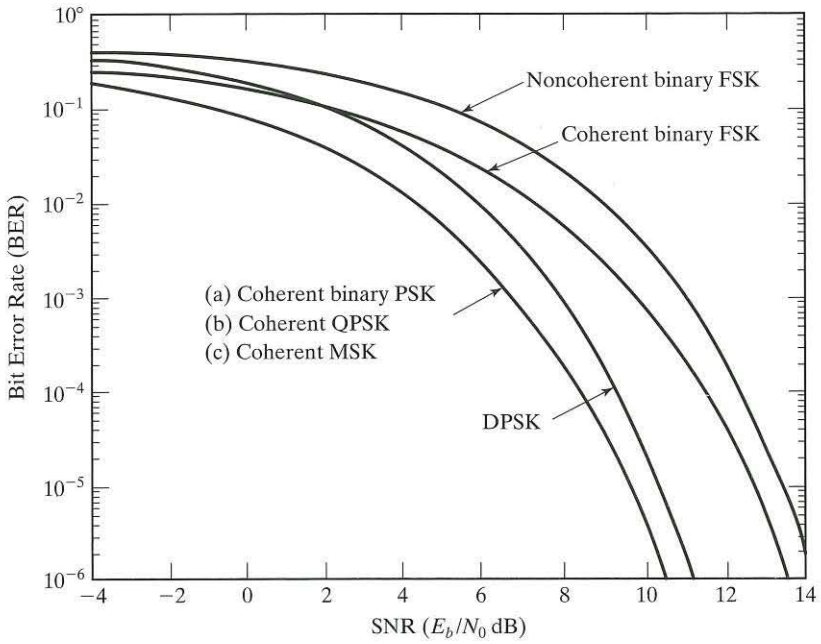

FIGURE 3.32 Comparison of the noise performance of different PSK and FSK schemes.

- At high values of E_b/N_0, DPSK and noncoherent BFSK perform almost as well (to within 1 dB) as coherent BPSK and coherent BFSK, respectively, for the same BER and signal energy per bit.

The noise performance of Gaussian-filtered minimum-shift keying (GMSK) does not lend itself to exact analysis in the way that conventional MSK does. This is rather unfortunate, since GMSK is widely used in digital wireless communications. Nevertheless, the BER of GMSK is reasonably well described by the empirical formula

$$P_e \approx \frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{\beta E_b}{2N_0}}\right) \tag{3.90}$$

where β is a constant whose value depends on the time–bandwidth product of the MSK. For $\beta = 2$, Eq. (3.90) reduces to the exact formula for the BER of conventional MSK. Thus, we may view $10\log_{10}(\beta/2)$, expressed in decibels, as a measure of the degradation in performance of GMSK compared with that of conventional MSK. For GMSK with a time–bandwidth product $WT = 0.3$, the degradation in noise performance of the receiver is about 0.46, which corresponds to $(\beta/2) = 0.9$. This degradation is a small price to pay for the highly desirable spectral compactness of the GMSK signal.

Frequently Flat, Slowly Fading Channel

Table 3.4 also includes the exact formulas for the bit error rates for a slow Rayleigh fading channel, where

$$\gamma_0 = \frac{E_b}{N_0}\mathbf{E}[\alpha^2] \tag{3.91}$$

is the *mean value of the received signal energy per bit-to-noise spectral density ratio.* In Eq. (3.91), the expectation $\mathbf{E}[\alpha^2]$ is the mean value of the Rayleigh-distributed random variable α characterizing the channel. For the derivations of the fading-channel formulas listed in the last column of Table 3.4, the reader is referred to Problem 3.35. Comparing these formulas with the formulas for their additive white Gaussian noise (i.e., nonfading) channel counterparts, we find that the Rayleigh fading process results in a severe degradation in the noise performance of a digital communication receiver, with the degradation measured in terms of decibels of additional mean signal-to-noise spectral density ratio. In particular, the asymptotic decrease in the bit error rate with γ_0 follows an *inverse law.* This form of asymptotic behavior is dramatically different from the case of a nonfading channel, for which the asymptotic decrease in the bit error rate with γ_0 follows an *exponential law.*

In graphical terms, Fig. 3.33 plots the formulas under part (a) of Table 3.4 to compare the bit error rates of BPSK over the AWGN and Rayleigh fading channels. The

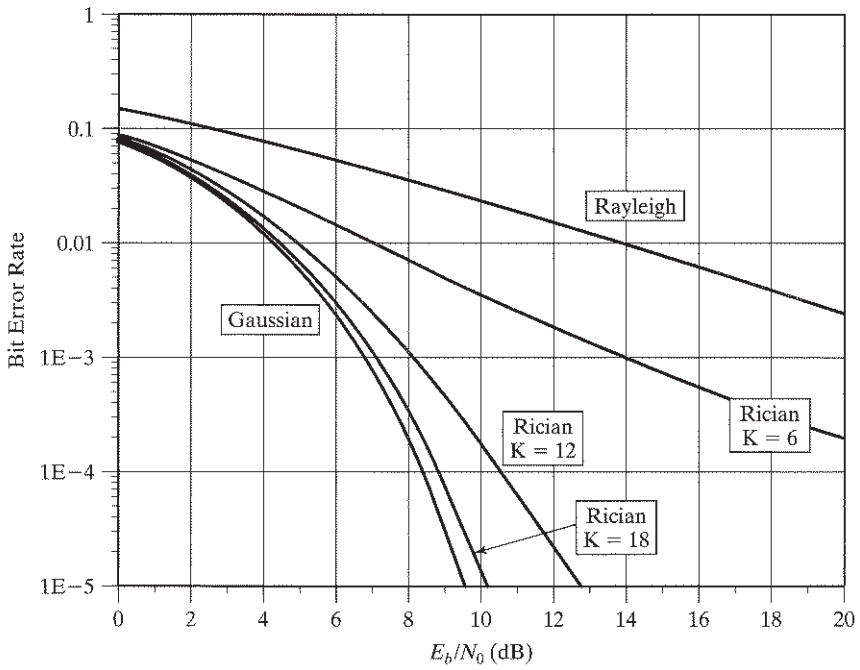

FIGURE 3.33 Comparison of performance of coherently-detected BPSK over different fading channels.

figure also includes corresponding plots for the Rice fading channel with different values of the Rice factor K. We see that as K increases from zero to infinity, the behavior of the receiver varies all the way from the Rayleigh channel to the AWGN channel. Note that the results plotted in Fig. 3.33 for the Rice channel were obtained using a procedure based on computer simulation.[15]

From Fig. 3.33 we see that as matters stand, we have a serious problem caused by channel fading. For example, at a signal-to-noise ratio of 10 dB and the presence of Rayleigh fading, the use of BPSK results in a BER of about 3×10^{-2}, which may not be good enough for the transmission of speech or digital data over the wireless channel. To overcome this problem, the traditional approach is to use diversity-on-receive, which is discussed in Chapter 6.

3.13 THEME EXAMPLE 1: ORTHOGONAL FREQUENCY-DIVISION MULTIPLEXING[16]

In the first theme example of this chapter, we discuss parts of a multicarrier system used for wireless local area networks (LANs) that fall under IEEE802.11a; this standard was discussed in Section 1.22. Recall that the object of the service is to provide wireless data links supporting rates up to 54 megabits/s to link workstations, laptops, printers, and personal digital assistants to a network access node without the expense of cabling. The service illustrates a number of the topics discussed in this chapter.

The transmission bandwidth for the service is constrained to be less than 20 MHz. One of the permitted transmission rates is 36 megabits/s of user information. If we add the overhead due to forward error-correction coding (discussed in Chapter 4), the required throughput is 48 megabits/s. To transmit 48 megabits/s through a channel bandwidth less than 20 MHz means that we need to transmit over 2 bits/Hz. This need demands that an M-ary digital modulation technique be used. The particular technique used for the service is the *16-QAM*, discussed in Section 3.6. With that technique, the in-phase and quadrature channels are independently modulated with a four-level signal derived from the incoming binary data stream. Nominally, the levels are ±1 and ±3. Figure 3.15 plots the pattern constellation of 16-QAM. At each symbol time, one of the 16 points of the constellation is selected for transmission. Since the points are equally likely to be selected, $\log_2 M = 4$ bits are transmitted at each symbol time.

At this point in the discussion, it is instructive to use the complex representation of band-pass signals presented in Section 3.5. Specifically, the complex envelope of the 16-QAM signal is defined by

$$\tilde{s}(t) = b_k\, p(t - kT_d) \qquad (k-1)T_d \le t < kT_d \tag{3.92}$$

where $p(t)$ is a rectangular pulse, the complex coefficient b_k is selected in accordance with the 16-QAM constellation, and T_d is the symbol period of the 16-QAM signal (i.e., four times the incoming bit duration). At each symbol time kT_d, a different symbol is selected to be transmitted. In a conventional system, the band-pass

signal to be transmitted is defined by Eq. (3.26), reproduced here for convenience of presentation:

$$s(t) = \mathrm{Re}\{\tilde{s}(t)\exp(j2\pi f_c t)\} \tag{3.93}$$

For reasons discussed in Chapter 2, it is preferred to use *multicarrier transmission* for this application. In particular, instead of sending 36 megabits/s (effectively 48 megabits/s) over one carrier, it is preferred to send 750 kilobits/s over each of 48 distinct *subcarriers*. The term subcarrier is used to distinguish this set of carriers from the RF carrier f_c defined by Eq.(3.93). We denote the subcarrier frequencies as $f_1, f_2, \ldots, f_{48}$ and consider all of the subcarriers in their complex form

$$\tilde{c}_i(t) = \exp(j2\pi f_i t) \qquad i = 1,2,\ldots,48 \tag{3.94}$$

If the data set $\{b_k\}$ is demultiplexed into 48 parallel streams represented by $\{b_{k,i}\}_{i=1}^{48}$, running at 1/48 of the incoming data rate, then the modulation of each subcarrier can be represented in the complex low-pass form as follows:

$$\left.\begin{aligned} &\vdots \\ \tilde{s}_i(t) &= b_{k,i}\, g(t-kT)\exp(j2\pi f_i t) \\ \tilde{s}_{i+1}(t) &= b_{k+1,i}\, g(t-kT)\exp(j2\pi f_{i+1} t) \\ &\vdots \end{aligned}\right\} \quad \begin{aligned} &\text{for } (k-1)T \le t < kT \\ &i = 1, 2, \ldots, 48 \end{aligned} \tag{3.95}$$

where the new symbol duration $T = 48T_d$ and the new pulse $g(t)$ is 48 times as long as $p(t)$.

Each of the terms in Eq. (3.95) is equivalent to a band-pass signal, and the aggregate of all the band-pass signals is the overall signal to be transmitted over the wireless channel. The complex low-pass representation of the combined modulation for one symbol period is given by

$$\tilde{s}(t) = \sum_{i=1}^{48} \tilde{s}_i(t) \tag{3.96}$$

At first sight, the combination of Eqs. (3.95) and (3.96) appears to be a complicated modulation strategy to implement, even though we can represent it rather simply in its complex low-pass equivalent form. However, analogous to the Fourier transform, briefly reviewed in Appendix A, is a discrete equivalent called the *discrete Fourier*

transform (DFT). The DFT transforms a set of samples in the time domain into an equivalent set of samples in the frequency domain. The *inverse discrete Fourier transform* (IDFT)[17] performs the reverse operation. This transform-pair is described mathematically by

$$\left.\begin{aligned} \text{DFT:} \quad b_n &= \sum_{m=0}^{M-1} B_m \exp(-j2\pi mn/M) \quad n = 0, 1, \ldots, M-1 \\ \text{IDFT:} \quad B_m &= \frac{1}{M}\sum_{n=0}^{M-1} b_n \exp(j2\pi mn/M) \quad m = 0, 1, \ldots, M-1 \end{aligned}\right\} \tag{3.97}$$

where the sequences $\{b_n\}$ and $\{B_m\}$ are the frequency-domain and time-domain samples, respectively. Considering Eq. (3.95) in the context of the IDFT and noting that $g(t)$ has a rectangular pulse shape, we observe that, for a fixed k, we may represent samples at each subcarrier frequency f_i, i = 1, 2, ..., 48. We now make two assumptions:

1. The subcarrier frequencies are selected such that

$$f_i = \frac{i}{T} \qquad i = 1, 2, \ldots, 48 \tag{3.98}$$

2. Each subcarrier is sampled M times per symbol interval T.

If we also let the output be sampled at M times per symbol interval T, then the sampled low-pass equivalent of Eq. (3.96) is given by

$$\tilde{s}(m) = \sum_{n=0}^{M-1} b_n \exp(j2\pi mn/M) \qquad m = 0,1,\ldots,M-1 \tag{3.99}$$

That is, the samples of the complex low-pass equivalent of the transmitted signal are given by the IDFT of the subcarriers.

The typical implementation of the transmitter is shown in Fig. 3.34(a). First, the incoming binary data stream is forward error-correction encoded, followed by 16-QAM modulation. (Error-correction encoding is discussed in Chapter 4.) Then it goes through a serial-to-parallel conversion device to create 48 independent data streams. Next, these independent data streams are combined by a computationally efficient implementation of the IDFT known as the *inverse fast Fourier transform (IFFT) algorithm*. The output of the IFFT consists of the time-domain samples to be transmitted over the channel. Besides displaying the 48 data-bearing subcarriers, Fig. 3.34(a) shows additional subcarriers, used by the receiver for synchronization and tracking purposes. Furthermore, the figure indicates the use of a 64-point IFFT. This is because the computationally efficient implementation of the FFT and IFFT algorithms requires that the number of samples be an integer power of 2. With this representation, any unused

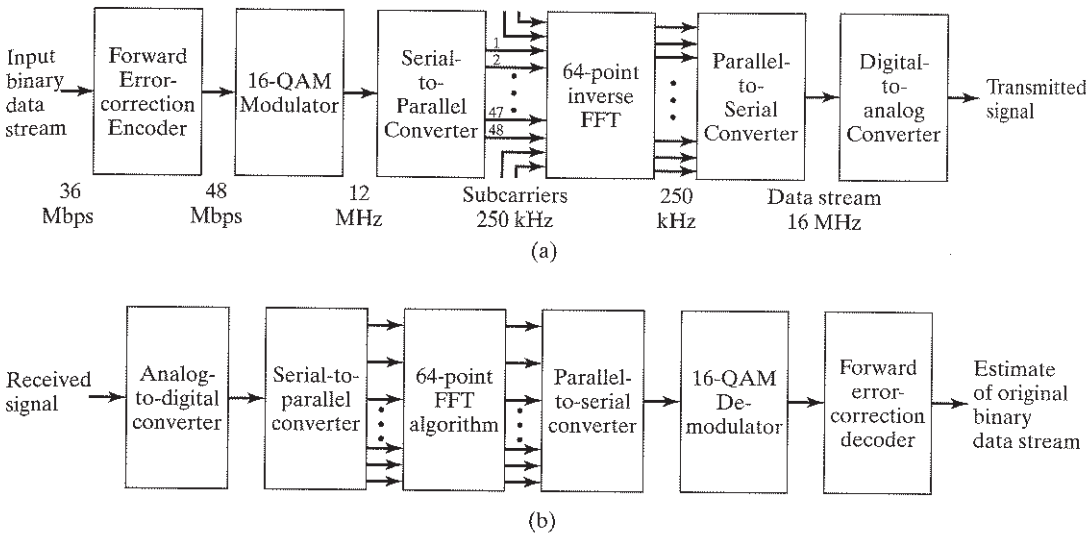

FIGURE 3.34 Block diagram of (a) OFDM transmitter, and (b) OFDM receiver.

subcarriers on the input are set equal to zero. The output of the IFFT consists of 64 of time samples of a complex envelope for each period T, which are parallel-to-serial converted and then finally digital-to-analog converted to facilitate transmission of the incoming data stream over the wireless channel.

Figure 3.34(b) presents the corresponding implementation of the receiver, which follows a sequence of operations in the reverse order to those performed in the transmitter of Fig. 3.34(a). Specifically, to recover the original input binary data stream, the received signal is passed through the following processors:

- Analog-to-digital converter
- Serial-to-parallel converter
- 64-point FFT algorithm
- Parallel-to-serial converter
- 16-QAM demodulator
- Forward error-correction decoder

The two parts of the system in Fig. 3.34 are instructive from a computationally efficient implementation point of view. To develop insight into the underlying communication-theoretic operations carried out in the system, consider Fig. 3.35 that depicts another viewpoint of what goes on basically in the system. Parts (a) and (b) of the figure pertain to transmission and reception aspects of the system, respectively. In particular, focusing on the transmission depicted in Fig. 3.35(a), we may make two statements:

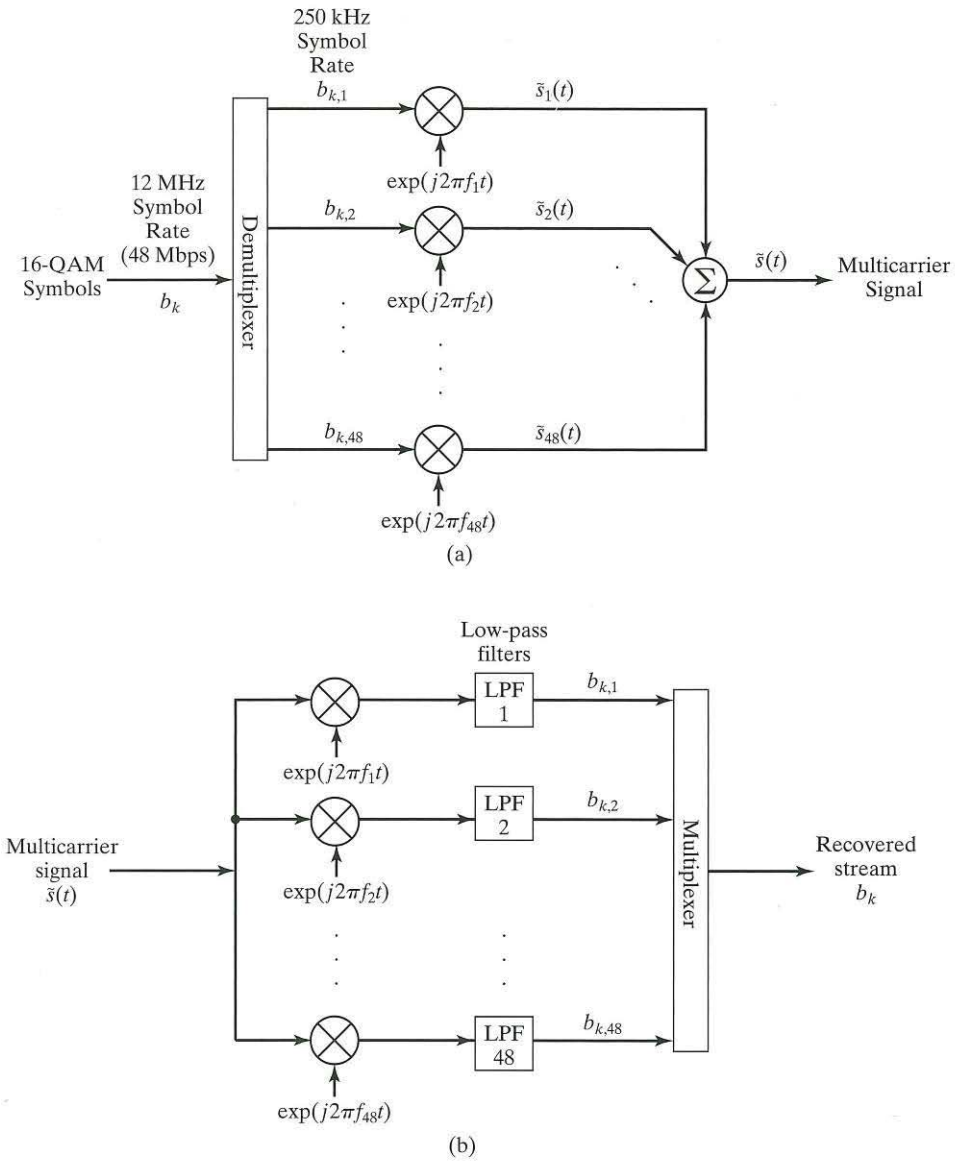

FIGURE 3.35 Communication - theoretic interpretation of the OFDM system: (a) Transmission (b) Reception.

1. *The subcarriers $\tilde{c}_i(t)$ constitute an orthogonal (orthonormal, to be more precise) set.* Given that the frequencies of the subcarriers satisfy Eq. (3.98), it follows that the subcarriers themselves satisfy the conditions of orthonormality

over the symbol period T, as the following series of manipulations of the cross-correlation of the complex exponential subcarriers shows:

$$\begin{aligned}\frac{1}{T}\int_0^T \tilde{c}_i(t)\tilde{c}_k^*(t)dt &= \frac{1}{T}\int_0^T \exp(j(2\pi i)t/T)\exp((-j))(2\pi k)t/T)dt \\ &= \frac{1}{T}\int_0^T \exp(j2\pi(i-k)t/T)dt \\ &= \frac{1}{T}\int_0^T \cos(2\pi(i-k)t/T)dt + \frac{j}{T}\int_0^T \sin(2\pi(i-k)t/T)dt \\ &= \frac{\sin(2\pi(i-k))}{2\pi(i-k)} \\ &= \begin{cases} 1 & \text{for all integer } i = k \\ 0 & \text{for all integer } i \neq k \end{cases}\end{aligned} \tag{3.100}$$

2. *The complex modulated (heterodyned) signals are multiplexed in the frequency domain.*

Put together, these two statements therefore justify referring to the communication system of Fig. 3.34 as an *orthogonal frequency-division multiplexing (OFDM) system.*

3.13.1 Cyclic Prefix

Guard intervals are included in the serial data stream of the OFDM transmitter so as to overcome the effect of intersymbol interference (ISI) produced by signal transmission over the wireless channel. To take care of this matter, the pertinent OFDM symbol is cyclically extended in each guard interval. Specifically, the *cyclic extension* of an OFDM symbol is the periodic extension of the DFT output, as shown by

$$s(-k) = s(N-k) \text{ for } k = 1, 2, \ldots, v \tag{3.101}$$

where N is the number of subchannels in the OFDM system and v is the duration of the baseband impulse response of the wireless channel. In effect, v is the memory length of the channel. The condition described in Eq. (3.101) is called a *cyclic prefix*. Clearly, inclusion of cyclic prefixes results in an increase in the OFDM transmission bandwidth.

Having stuffed the guard intervals in the parallel-to-serial converter in the transmitter with cyclic prefixes, the guard intervals (and with them, the cyclic prefixes) are removed in the serial-to-parallel converter in the receiver *before* the conversion takes place. Then the output of the serial-to-parallel converter is in the correct form for discrete Fourier transformation.

To summarize, the theme example on OFDM has demonstrated the following desirable features:

1. Spectrally efficient digital modulation schemes, such as 16-QAM can be represented simply by their complex low-pass equivalents.
2. Modulations of even greater complexity such as OFDM, can be presented and understood easily in terms of their complex low-pass equivalents.
3. The clear understanding of these modulation schemes allows us to take advantage of digital signal-processing techniques such as the fast Fourier transform algorithm so as to simplify the implementation of some quite complicated modulation schemes with the use of multiple carriers.

3.14 THEME EXAMPLE 2: CORDLESS TELECOMMUNICATIONS

Cordless telephones are almost a ubiquitous consumer item. Although commonplace, they include some of the more advanced concepts, on a very small scale, that we will discuss later in this book. In particular, they are an example of an FDMA system operating with a very small cell size. Accordingly, they usually operate with near *line-of-sight* conditions, with a minor amount of the propagation effects considered in Chapter 2. In this second theme example of the chapter, we will describe a European standard for cordless telephony known as CT-2.

The CT-2 cordless telephones operate in the 4-MHz band of frequencies extending from 864.15 to 868.15 MHz. This band is divided into 40 channels, each with a bandwidth of 100 kHz. Why so many channels? The system is designed to allow operation of many cordless phones in close proximity. This could be for multiple phones in an office environment, or it could be just for phones in separate residences that are closely spaced. The access strategy for using these 40 channels is known as *dynamic channel allocation* (DCA) to assign frequencies. With this strategy, each base and portable unit selects an appropriate channel on the basis of their measurements of traffic conditions and channel quality. This is an example of FDMA in which multiple users share the same 4 MHz of bandwidth by dividing it up into 100-kHz channels and accessing the free channels when needed.

Transmissions from the base to a portable and from the portable to the base use the same frequency channel with a technique known as *time-division duplex* (TDD). The frame structure for this duplex transmission is shown in Fig. 3.36. In a 2-millisecond frame, the base transmits a burst of 66 bits and the portable follows with a burst of 66 bits; the sequence repeats every two milliseconds. The two bursts are separated by *guard intervals* that have a duration equivalent to 5.5 and 6.5 bits, respectively. TDD is a simple form of *time-division multiple access* (TDMA) in which the base station transmitter and mobile transmitter share the same channel by using different time slots. A more sophisticated form of TDMA will be considered in Chapter 4.

With CT-2, speech is encoded digitally, using a method known as *adaptive pulse-coded modulation* (ADPCM).[18] With this technique, the analog voice signal is converted into a bit stream that can be transmitted with digital modulation techniques.

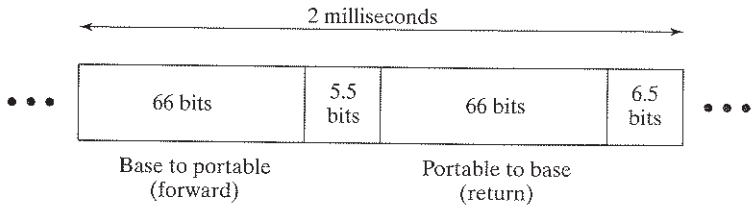

FIGURE 3.36 Frame structure for CT-2 for cordless telecommunication.

The ADPCM standard uses a data rate of 32 kbps to provide a high-quality voice signal. From the frame structure of Fig. 3.36, it is clear that the overall bit rate is 72 kilobits/s; this allows for 32 kilobits/s transmission in both directions, plus the overhead for the guard intervals.

The phones use digital modulation with a variant of FSK signalling that includes Gaussian filtering. This form of digital modulation is closely related to Gaussian minimum-shift keying (GMSK), discussed in Section 3.7. Prior to filtering, the modulated signal at complex baseband is given by

$$\tilde{s}(t) = \sqrt{\frac{2E_b}{T}}\cos(2\pi f_i t + \phi) \qquad 0 \le t < T \tag{3.102}$$

where the f_i, $i = 1,...,M$, are the tones in the M-ary FSK modulation strategy. The transmitted signal is given by (see Eq. (3.26))

$$s(t) = \mathrm{Re}\{\tilde{s}(t)\exp(j2\pi f_c t)\} \tag{3.103}$$

where f_c is the center frequency of one of the 40 RF channels. The separation of the two transmitter operations of modulation in Eq. (3.102) and upconversion in Eq. (3.103) not only is convenient for analysis, but also has its implementation advantages. Many advanced signal-processing techniques can be applied at complex baseband with the use of a microprocessor that would be very difficult to carry out if the modulation process was to be performed at higher frequencies.

The advantage of M-ary FSK for wireless channels is that it is a robust modulation strategy. The demodulator is noncoherent, being implemented as a simple *energy detector* situated on each of the frequency bins with center frequency f_c. The receiver is noncoherent in that it does not require knowledge of the phase of the signal—a property that follows from the discussion of fading channels presented in Chapter 2.

Power control is an effective way of minimizing interference and extending the life of the battery. The CT-2 system uses a simple two-level power control scheme. Normal transmissions occur with a nominal transmit power of 5 milliwatts of power. However, if the received signal strength exceeds a certain level, then in-band signalling is used to request the transmitter to reduce its power level by approximately 15 dB.

Problem 3.15 What is the round-trip path delay for a 60-m path with CT-2? Does this amount of delay explain the allocated guard intervals?
Ans. The round trip is 0.4 microseconds. The guard intervals include (1) processing time for the terminals, and (2) allowances for other timing errors. ■

Problem 3.16 The maximum frequency error allowed in the CT-2 standard is 10 kHz or less. What is the relative frequency error, expressed in parts per million (ppm)?
Ans. The clock error is 22 ppm. ■

3.15 SUMMARY AND DISCUSSION

In this chapter, we discussed the modulation process, focusing on those techniques which are of particular interest to wireless communications. Among the analog techniques, frequency modulation (FM) stands out by virtue of its inherent capability to trade off channel bandwidth for improved noise performance at the expense of increased system complexity.

Much of the discussion, however, was devoted to digital modulation techniques, which distinguish themselves from their analog modulation counterparts by their signal-space representation, in which the transmitted message points show up as discrete points in the signal space. The discussion also highlighted detailed treatments of quadriphase-shift keying (QPSK), minimum-shift keying (MSK), and its Gaussian-filtered version (GMSK). A variant of QPSK known as the $\pi/4$-shifted QPSK is used in the TDMA standard IS-54. On the other hand, GMSK features prominently in the design of a TDMA system, commonly referred to as GSM, which will be discussed in Chapter 4.

The analysis of modulation systems, be they of an analog or digital nature, is simplified considerably by using the complex baseband representation of band-pass signals and systems with no loss of information. The simplification results from exchanging the use of complex baseband signals and systems for their real counterparts, complicated by the presence of a carrier frequency. Simply put, elimination of the sinusoidal carrier permits us to get rid of tedious trigonometry at the expense of complex signal analysis.

Frequency-division multiple access (FDMA), rooted in frequency-domain concepts, relies on the use of modulation and band-pass filtering to place a user's information-bearing signal inside a permissible subband. It is largely because of its conceptual simplicity and the status of technology that frequency modulation became the technique upon which the early generation of wireless communication systems were based. This analog method of modulation provides protection against channel noise by virtue of its ability to exchange increased transmission bandwidth for improved receiver noise performance in accordance with a square power law. For data control, FDMA uses binary frequency-shift keying, a digital form of frequency modulation.

However, FDMA suffers from the limitation that the system needs a separate transceiver (i.e., transmitter-receiver) for each active user at the base station under its coverage. It is this limitation that motivated the development of time-division multiple access (TDMA), the deployment of which has been made possible by virtue of the availability of digital signal-processing techniques. (TDMA is discussed in Chapter 4.)

NOTES AND REFERENCES

[1] For a detailed treatment of analog modulation techniques including both amplitude modulation and frequency modulation, time-domain and frequency-domain representations, methods of generation and detection, and the evaluation of noise performances, see Chapter 2 of Haykin (2001).

[2] For details of the IS-54 standard, see the Telecommunication Industry Association Report 1992. For details of the Japanese digital cellular standard using the $\pi/4$-shifted QPSK, see Nakahima *et al.* (1990). See also Pahlavan and Levesque (1995), pp. 272–273.

[3] The fundamental work of Nyquist on pulse shaping is described in the classic 1928 paper.

[4] The root raised-cosine pulse shaping is discussed in Chennakesku and Saulnier (1993) in the context of $\pi/4$-shifted differential QPSK for digital cellular radio. It is also discussed in Anderson (1999), pp. 26–27, and Stüber (1996), pp. 169–172.

[5] The complex representation of band-pass signals and systems is discussed in Haykin (2001). The material presented therein involves the use of the *Hilbert transform*, which may be viewed as a device whose frequency response satisfies two conditions:

- The amplitude response is constant at all frequencies.
- The phase response is equal to +90° for negative frequencies and –90° for positive frequencies.

[6] For a detailed exposition of the signal-space analysis of digitally modulated signals, see Chapter 5 of Haykin (2001). The classic book on this topic is that of Wozencroft and Jacobs (1965).

[7] For a discussion of Carson's rule on the bandwidth of FM signals, see Chapter 2 of Haykin (2001).

[8] The invention of minimum shift keying (MSK) may be traced to Doelz and Heald (1961). Independently of this early work, deBuda (1972) derived the fast FSK, another way of viewing MSK. For discussion of GMSK, see Stüber (1996) and Steele and Hanzo (1999). The noise performances of MSK and GMSK are discussed in Chapter 6 of Haykin (2001).

[9] Frequency-division multiple access is discussed in Steele and Hanzo (1999), and Rappaport (2002).

[10] The adjacent channel interference problem is discussed in Shankar (2002).

[11] Spectral analysis of QPSK and MSK signals is presented in Haykin (2001), pp. 360–361, and pp. 394–396, respectively.

[12] Early digital radios were made by modifying FM voice radios. An FM modulator integrates the baseband signal to determine the transmitted phase, analogous to the way a differential encoder works. Since FM is a constant-envelope modulation scheme, FM receivers typically place a hard limit on the received signal to remove the effect of channel gain variations. In addition, the received signal is passed through a discriminator, which, in mathematical terms, is equivalent to a differentiator. The discriminator performs an operation quite similar to differential detection. For that reason, differential detection of digital FM was an important early transmission scheme for mobile radio. Generalizations of this approach are called continuous phase modulation (CPM) and the reader can find more information in Anderson *et al.* (1986).

[13] Moher and Lodge (1989) and earlier papers were the first to propose the use of pilot symbols for estimating and compensating the variations of flat-fading in mobile radio. Pilot symbol techniques are used in a variety of forms, not necessarily just for uniform spacing, in many current radio systems.

[14] For a detailed treatment of the noise performance of modulation schemes for passband data transmission, see Chapter 6 of Haykin (2001).

[15] Benedetto and Biglieri (1999) describe a general technique for computing bit error rates over a fading channel. The technique proceeds from a bound known as the union bound. Specifically, it is shown that the calculation of the bit error rate can be reduced to the calculation of the probability that a random variable takes on a negative value. This probability is in the form of an integral that lends itself to numerical computation, which can be made arbitrarily accurate.

[16] OFDM theory is closely related to the theory of discrete multitone (DMT) modulation, which is discussed in Starr *et al.* (1999). Bahai and Saltzberg (1999) is devoted to OFDM theory and applications; this book also discusses wireless local area networks and future trends using OFDM.

[17] The Discrete Fourier Transform and its inverse are discussed in Oppenheim *et al.* (1999), which also describes the Fast Fourier Transform (FFT) algorithm.

[18] Pulse-code modulation (PCM) provides a technique for the conversion of an analog signal into a binary stream. The conversion relies on three basic operations:

- Sampling
- Quantization
- Coding

PCM provides a significant improvement over FM in terms of the bandwidth-noise trade-off issue. Whereas this trade-off follows a square-power law in FM, the law in PCM is of an exponential form. Adaptive pulse-code modulation provides an improvement over the standard PCM by making the quantizer, the encoder, or both adaptive. For a discussion of ordinary PCM and adaptive PCM, see Chapter 3 of Haykin (2001).

ADDITIONAL PROBLEMS

Modulation Techniques

Problem 3.17 Figure 3.37(a) presents the block diagram of a passband digital phase modulator, which lends itself to VLSI implementation (Steele and Hanzo, 1999). The premodulation filter of impulse response $h(t)$, as configured in Fig. 3.37(b), is designed to produce a data-dependent phase signal $\theta(t)$, which addresses two read-only-memory (ROM) units to yield values of the trigonometric terms $\cos(\theta(t))$ and $\sin(\theta(t))$. The resulting digital signals are converted into analog form.

(a) In effect, the baseband model of Fig. 3.12(a) is being extended to deal with nonlinear phase modulation. This extension is however subject to the assumption that the essentially highest frequency component of both $\cos(\theta(t))$ and $\sin(\theta(t))$ is less than the carrier frequency f_c. Justify the need for this assumption.

(b) Under this assumption, show that the radiated output of Fig. 3.37(a) can be formulated as the phase-modulated signal

$$s(t) = A\cos(2\pi f_c t + \theta(t))$$

where A is a constant amplitude.

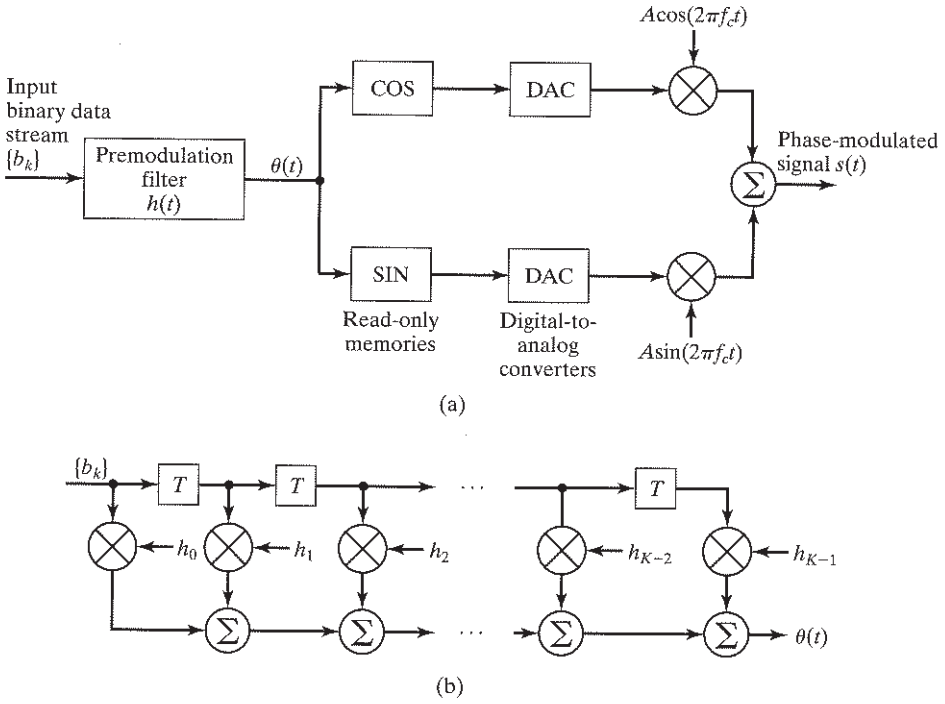

FIGURE 3.37 Block diagrams for Problem 3.17.

(c) The premodulation filter can be implemented in the form of a tapped-delay-line filter, as in Fig. 3.37(b), where T is the symbol duration. Justify this method of implementation.

Problem 3.18

(a) Construct and label the constellations of M-ary PSK for (i) $M = 8$, and (ii) $M = 16$.

(b) Discuss the differences that distinguish 16-PSK considered in part (a) and 16-QAM described in Fig. 3.15, doing so in the context of information transmission over a wireless channel.

Problem 3.19 Quadriphase-shift keying and minimum-shift keying provide two spectrally efficient methods for the digital transmission of binary data over a wireless channel. List the advantages and disadvantages of these two methods of digital modulation.

Problem 3.20 The *$\pi/4$-shifted DQPSK* is characterized by two combined features:

- the use of 8 carrier-phase states, and
- the transmission of an information-bearing signal in the differential carrier phase.

Specifically, the differential carrier phase $\Delta\theta_k$ is governed by the mapping

$$\Delta\theta_k = \begin{cases} -3\pi/4 & \text{for } b_k = -3 \\ -\pi/4 & \text{for } b_k = -1 \\ +\pi/4 & \text{for } b_k = +1 \\ +3\pi/4 & \text{for } b_k = +3 \end{cases}$$

where $\{b_k\}$ denotes the incoming data stream.

Formulate the expression or the complex envelope of the $\pi/4$-shifted DQPSK signal.

Problem 3.21 Continuing with the $\pi/4$-shifted DQPSK described in Problem 3.20, suppose that the incoming pulse amplitude is shaped in accordance with the *square root raised cosine spectrum* discussed in Section 3.4.1.

(a) Using simulation, compute the phase trajectory of the $\pi/4$-shifted DQPSK signal by plotting its quadrature component versus the in-phase component for an incoming random quaternary sequence.

(b) Demonstrate that the phase trajectory does not pass through the origin. What are the practical implications of this property in the context of information transmission over a wireless channel?

Problem 3.22 In this problem, we explore the effect of a multipath channel on the waveform of three digitally modulated signals. The multipath channel is represented by the simple tapped-delay model of Fig. 3.38, where T denotes the symbol duration. Values of the tap-weights are

$$w_0 = 0.25$$
$$w_1 = 0.25$$
$$w_2 = 0.25$$

The incoming binary sequence is ...0000110101110101...

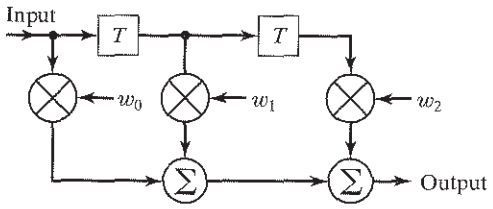

FIGURE 3.38 Tapped-delay-line model of multipath channel.

(a) Plot the waveforms of the modulated signals at the model input, using the following three methods:

- **(i)** QPSK
- **(ii)** OQPSK
- **(iii)** $\pi/4$ -shifted QPSK

(b) For each of these methods, plot the waveform produced at the model output.

(c) What conclusions can you draw from the results of parts (a) and (b)?

Assume the use of non-return-to-zero signaling.

Problem 3.23 Repeat Problem 3.22, this time using the more difficult set of tap-weights:

$$w_0 = 0.5$$
$$w_1 = \sqrt{0.5}$$
$$w_2 = 0.5$$

Why is this set of tap-weights more difficult to deal with than those of Problem 3.22?

Problem 3.24
Equations (3.58) and (3.59) define the two coordinates for signal-space analysis of MSK. Given these two orthonormal coordinates, depict the signal-space representation of the MSK signal.

Problem 3.25 Equation (3.67) defines the impulse response of the pulse-shaping filter $h(t)$ used to generate a GMSK signal.

(a) Show that the time function $h(t)$ satisfies all the properties of a probability density function.

(b) Expanding on the interpretation of $h(t)$ as a probability density function, determine the variance of the distribution. What is the significance of this interpretation?

Problem 3.26 The *tamed frequency modulation (TFM)*, due to deJager and Dekker (1978), is designed to provide a frequency-modulated signal whose power spectrum is compact without sidelobes. This desirable spectral characteristic is achieved by careful control of the phase transitions of the frequency-modulated signal. Figure 3.39 depicts the TFM modulator that consists of a premodulation filter feeding a voltage-controlled oscillator. The premodulation filter itself

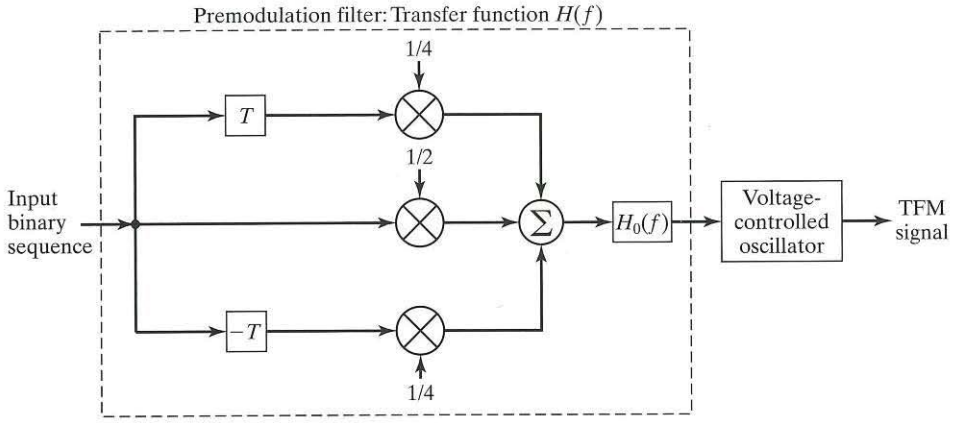

FIGURE 3.39 Block diagram of TFM modulator.

consists of the tapped-delay-line filter cascaded with a low-pass filter whose impulse response is the inverse Fourier transform of the example transfer function

$$H_0(f) = \begin{cases} (\pi fT)/\sin(\pi fT) & \text{for } 0 \le |f| \le 1/2T \\ 0 & \text{otherwise} \end{cases}$$

where T is the symbol duration.

(a) Show that the overall transfer function of the corresponding premodulation filter is given by

$$H(f) = \begin{cases} \dfrac{(\pi fT)}{\sin(\pi fT)}\cos^2(\pi fT) & \text{for } 0 \le |f| \le 1/2T \\ 0 & \text{otherwise} \end{cases}$$

(b) Using computer simulation, plot the overall impulse response of the premodulation filter, denoted by $h(t)$.

(c) The filter defined in part (b) is noncausal. Propose the use of a delay that would make the filter causal for all practical purposes.

(d) How does the impulse response of the premodulation filter computed in part (b) compare with that of the premodulation filter used in the GMSK modulator?

(e) Discuss the practical benefit that could be gained by using TFM for a FDMA system.

Frequency-Division Multiple Access

Problem 3.27 An FDMA system using frequency modulation accommodates a total of $N = 100$ mobile users assigned to a particular cell. The largest frequency component of the speech signal is $W = 3.4$ kHz. Using Carson's rule, determine the bandwidth of the uplink and downlink of the system for each of the following frequency deviations:

(a) $D = 1$
(b) $D = 2$
(c) $D = 3$

Problem 3.28 The use of frequency hopping makes it possible for the carrier frequency to hop randomly from one frequency to another. Discuss how the use of frequency hopping can improve the performance of an FDMA system operating in a wireless communication environment.

Adjacent Channel Interference

Problem 3.29 As a measure of the adjacent channel interference problem illustrated in Fig. 3.24, consider the following index of performance:

$$\text{ACI} = \frac{P_{SL}}{P_{ML}}$$

where P_{SL} is the power spilling over into a channel of interest due to the sidelobes of an adjacent channel, and P_{ML} is the power produced in that channel due to its own main lobe.

(a) Using Eq. (3.70), derive a formula for ACI.
(b) Calculate the index of performance for the example illustrated in Fig. 3.24.

Problem 3.30 A general formula for assessing the adjacent-channel interference problem is

$$\text{ACI}(\delta f) = \frac{\int_{-\infty}^{\infty} G(f)|H(f-\delta f)|^2 df}{\int_{-\infty}^{\infty} G(f)|H(f)|^2 df}$$

where $G(f)$ is the power spectral density of the input signal, $H(f)$ is the frequency response of the band-pass filter used to separate adjacent channels, and δf is the frequency separation between the two channels. Justify the validity of this formula.

Amplifier Nonlinearities

Problem 3.31 One way of linearizing a nonlinear power amplifier is to predistort the input signal. In effect, the cascade connection of two nonlinear components behaves like a linear memoryless system. Discuss the rationale of how such a scheme can be implemented.

Channel Estimation and Tracking

Problem 3.32 Plot the loss in dB as a function of residual phase error using BPSK modulation. For typical modem implementations, there is an allocated implementation margin that may range from 0.5 dB to 2 dB, depending upon the application. This implementation margin includes all losses due to nonideal implementation of the modem. If the portion of the implementation margin allocated for phase errors is 0.25dB, what is the maximum phase error allowed if the target BER is 10^{-5}?

Problem 3.33 The statement "The residual frequency error is small compared to the signal bandwidth" implies that there is minimal phase rotation over between two successive symbols. If the maximum phase rotation permitted is 10°, what is the maximum frequency error that would be permitted as a fraction of the symbol rate? What does this imply about the accuracy the local oscillator for down-converting the received signal?

Problem 3.34 Show that pilot symbols can be used to track both fading and small residual frequency errors in the receiver. What are the constraints on this residual frequency error?

Receiver Noise Performance

Problem 3.35 The last column of Table 3.4 lists the exact formulas for the bit error rates of different digital modulation schemes operating over a slow Rayleigh fading channel. The parameter γ_0 denotes the mean value of the received signal-energy-to-noise spectral density ratio.

(a) Derive these exact formulas.

(b) Assuming that γ_0 is large compared to unity, find the approximate forms of these formulas.

Problem 3.36 A digital communication system uses MSK for information transmission. The requirement is to do the transmission with a bit error rate that must not exceed 10^{-4}. Calculate the minimum signal-to-noise ratio needed to meet this requirement for the following two scenarios:

(a) Additive white Gaussian noise channel

(b) Rayleigh fading channel

Orthogonal Frequency-Division Multiplexing

Problem 3.37 In this problem, we address the issue of evaluating the power spectrum of an OFDM signal. From the discussion presented in Section 3.13, we may treat the OFDM signal as a modulated set of orthogonal subcarriers whose frequencies are separated by the reciprocal of the symbol duration T. Consider an incoming signal constellation characterized by two features:

- zero mean, and
- amplitude-shaping pulse $P(f)$.

(a) Derive the expression for the power spectrum of the complex envelope of the OFDM signal.

(b) Plot the power spectrum derived in part (a) for the following specifications:

  (i) Number of subcarriers, $N = 16$.

  (ii) Pulse-amplitude shaping pulse in the form of a rectangular function of time t.

(c) Repeat the power spectrum computation of part (b) for $N = 48$.

CHAPTER 4

Coding and Time-Division Multiple Access

4.1 INTRODUCTION

In conceptual terms, FDMA operates by partitioning the prescribed radio spectrum (i.e., prescribed for wireless communications by regulatory agencies) among potential users. Hence, FDMA belongs to the *analog world*. In contrast, TDMA operates by partitioning prescribed *time intervals* among potential users. This alternative to multiple access on a frequency-division basis belongs to the *ever-expanding digital world* that continues to improve over time in terms of both computing power and the cost of fabricating equipment. Thus, although the many-faceted implementation of TDMA requires the use of sophisticated digital signal-processing techniques, thanks to breakthroughs in digital signal-processing theory, groundbreaking discoveries in solid-state physics, and major industrial refinements of microfabrication machinery, we find the following

- For the same level of performance, TDMA systems are cheaper to build than FDMA systems.
- For the same cost, TDMA systems deliver a superior performance compared with FDMA systems.

Later in the chapter, we present a detailed account of the advantages of TDMA over FDMA. For the present, it suffices to say that the increase in system complexity of TDMA systems (compared with FDMA systems) comes about from a sequence of operations, each of which is designed for a specific purpose:

Coding. For digital signal processing, the speech signal is sampled at a uniform rate, and the speech samples are subsequently encoded into a digital sequence through two operations:

1. Source encoding for effective utilization of channel bandwidth
2. Channel encoding for protection against channel noise

An attractive feature of TDMA is that it does accommodate the transmission of source-channel encoded digital data alongside digitized speech in a straightforward manner.

Equalization. With TDMA, the channel bandwidths are wider than with FDMA, and the fading is no longer frequency flat, but rather frequency selective. This form of

modification introduces a type of distortion known as *intersymbol interference* (ISI). Channel equalization provides a practical means to mitigate the ISI problem.

Modulation. For the transmission of digitized speech and data over a wireless channel, we naturally require the use of passband modulation techniques, which, in turn, mandates the use of synchronization so as to establish a strict one-to-one correspondence between the locally generated carrier frequency, carrier phase, and symbol timing at the receiver, on the one hand, and their corresponding counterparts at the transmitter, on the other. Modulation was discussed in Chapter 3.

Figure 4.1 shows the block diagram of a *basic TDMA link*. The source signal (e.g., speech signal) is known to contain redundant information. With the efficient utilization of channel bandwidth as a primary objective of wireless communications, the transmitter begins by sampling the incoming speech signal, followed by encoding the signal through a process designed to remove as much of the *natural redundancy* in the speech signal as possible without compromising the ability of the receiver to provide a high-quality reproduction of the original signal. The next functional block in the transmitter is a *channel encoder*, whose function is to introduce *controlled redundancy* into the speech-encoded signal to provide protection against channel noise. The channel encoder fulfills this function by making the controlled redundancy known to the receiver. One other important point to take into account is that a wireless channel distinguishes itself from other communication channels (e.g., an ordinary telephone channel) by its tendency to produce errors in the form of *bursts*, attributed to deep fades, which degrade receiver performance. To mitigate this particular channel impairment, an *interleaver* is included in the transmitter for the purpose of pseudorandomizing the order of the binary symbols in the channel-encoded signal in a deterministic manner. As with the controlled redundancy introduced by the channel encoder, the pseudorandomization introduced by the interleaver is also known to the receiver. The next functional block in the transmitter is a *packetizer*, whose function is to convert the encoded and interleaved sequence of digitized speech data into successive *packets*, each of which occupies a significant portion of a *frame*. Each frame also includes a *synchronization information-bearing signal*, the purpose of which is to synchronize the timing operations in the receiver with the corresponding ones in the transmitter. The remainder of the frame is occupied by a *probing signal*, whose function, as the name implies, is to probe the channel and thereby make it possible to estimate the unknown impulse response of the channel on a frame-by-frame basis. With the estimate of the channel impulse response at hand, equalization of the channel at the receiving end of the link is made possible. The final functional block of the transmitter is used to *modulate* the packetized speech data onto a sinusoidal carrier for transmission over the channel.

As indicated in the figure, the receiver consists of a cascade of the following blocks: a quadrature demodulator, a baseband processor for channel estimation and equalization, a deinterleaver, a channel decoder, a speech decoder, and a reconstruction (low-pass) filter. The individual functions of these blocks are to reverse the corresponding operations performed by the channel and the transmitter. The *quadrature demodulator* converts the received RF signal into its baseband form without any loss of information; this conversion permits the use of digital signal-processing tools.

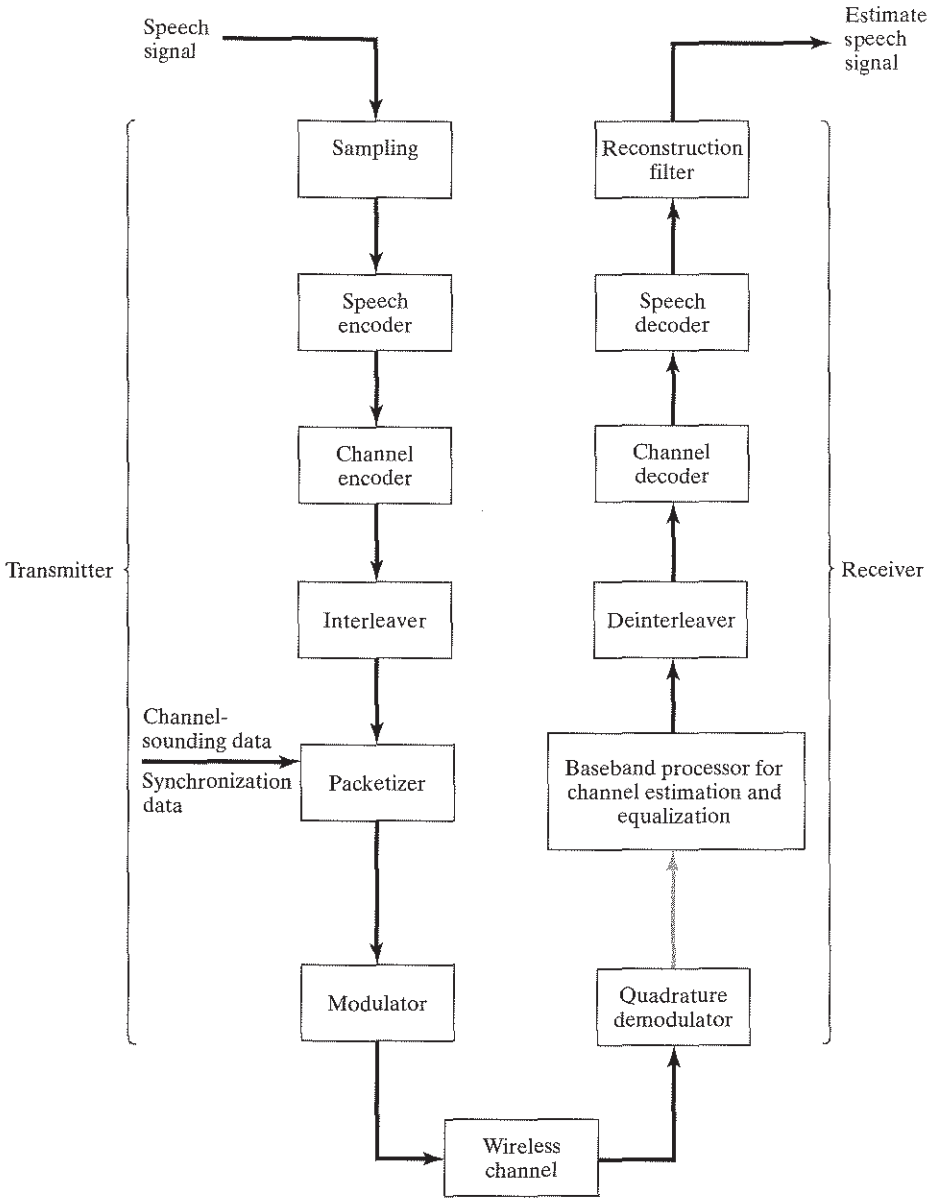

FIGURE 4.1 Block diagram of a basic TDMA link; the shaded arrow indicates a complex signal with real and imaginary parts.

The baseband processor operates on the resulting *complex baseband signal* to perform two functions: estimating the unknown channel impulse response, and using the estimate obtained to reverse the convolution performed on the transmitted signal by the channel (i.e., channel equalization). The resulting output is then *deinterleaved*, *channel*

decoded, *source decoded*, and, finally, low-pass filtered. In this way, an *estimate* of the original speech signal is delivered to a mobile user at the receiver output. Note that the shaded arrow connecting the baseband processor and the quadrature demodulator indicates the transmission of complex baseband signals (i.e., signals with real and imaginary parts).

From the block diagram of Fig. 4.1 for a basic TDMA link, it is apparent that the use of digital wireless communications requires a considerable amount of electronic circuitry. Fortunately, nowadays electronics are inexpensive, due to the ever-increasing availability of very large-scale integrated (VLSI) circuits in the form of silicon chips. Thus, although cost considerations used to be a factor in selecting analog wireless communications over digital wireless communications, that is no longer the case today.

The chapter is organized as follows. Section 4.2 reviews the sampling process, which is the first step in the digitization of an analog signal (e.g., speech signal), followed by rationale for the use of source and channel coding techniques in Section 4.3. The stage is then set for a review of Shannon's information theory, which provides the mathematical foundation for the various signal-processing operations embodied in the TDMA link of Fig. 4.1. Section 4.5 discusses speech-coding techniques that are of particular interest in wireless communications. Sections 4.6 through 4.13 examine channel-coding and Turbo-coding techniques, including several related issues that are pertinent to wireless communications. Section 4.14 discusses partial-response modulation. This is followed by the treatment of channel estimation/equalization in Section 4.15. At that point in the chapter, the stage is set for a description of TDMA in Section 4.16. Finally, Sections 4.17 through 4.19 discuss three theme examples: the global system of mobile (GSM) communications; joint equalization-decoding in the receiver; and random access techniques, in that order. The chapter draws to a conclusion in Section 4.17.

4.2 SAMPLING

We started our introductory discussion of the basic TDMA link of Fig. 4.1 by considering the sampling process, which is basic to the digital representation of all analog signals, including speech. The purpose of sampling is to convert an analog information-bearing signal, without significant loss of information, into a corresponding sequence of samples that are usually spaced uniformly in time. For such a process to have practical utility, it is necessary that we choose the sampling rate properly, so that the sequence of samples uniquely defines the original signal, which, in turn, enables the original signal to be reconstructed from the sequence of samples. This is the essence of the sampling theorem.

A derivation of the sampling theorem follows from the Fourier transform that provides a mathematical link between the time- and frequency-domain descriptions of an analog signal. In light of the derivation, presented in Appendix A, we may state the *sampling theorem* for strictly band-limited signals of finite energy in two equivalent parts:

1. *A band-limited signal of finite energy that has no frequency components greater than W hertz is completely described by specifying the values of the signal at instants of time separated by 1/2W seconds.*

2. *A band-limited signal of finite energy that has no frequency components higher than W hertz may be completely recovered from a knowledge of its samples taken at the rate of 1/2W samples per second.*

Part 1 of the sampling theorem applies to the transmitter, while Part 2 applies to the receiver. The sampling rate of $2W$ samples per second for a signal bandwidth of W hertz is called the *Nyquist rate*; its reciprocal, $1/2W$ (measured in seconds), is called the *Nyquist interval*.

The statement of the sampling theorem, as presented herein, is based on the assumption that the information-bearing signal $m(t)$ is strictly band limited. In practice, however, $m(t)$ is *not* strictly band limited, with the result that some degree of undersampling is encountered. Consequently, some *aliasing* is produced by the sampling process, resulting in (hopefully) only a minor loss of information. *Aliasing* refers to the phenomenon of a high-frequency component in the spectrum of the signal $m(t)$ seemingly taking on the identity of a lower frequency in the spectrum of the sampled version of the signal, as illustrated in Fig. 4.2. The aliased spectrum, shown by the solid curve in Fig. 4.2(b), pertains to an undersampled version of the signal represented by the spectrum of Fig. 4.2(a).

To combat the effects of aliasing in practice, we use two corrective measures:

1. Prior to sampling, a low-pass *antialiasing filter* is used to attenuate high-frequency components of the signal $m(t)$ that are not essential to the information being conveyed.
2. The output of the low-pass filter is sampled at a rate slightly higher than the Nyquist rate, which has the beneficial effect of easing the design of the reconstruction filter used to recover the original signal in the receiver.

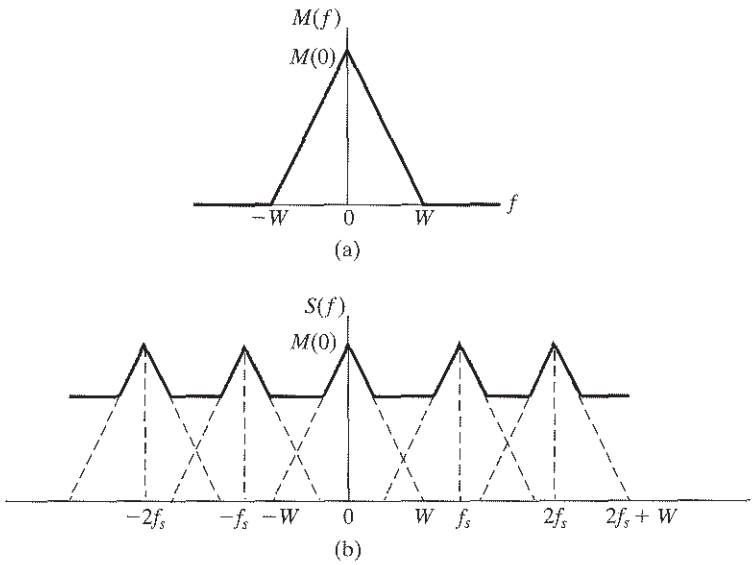

FIGURE 4.2 (a) Spectrum of a message signal band limited to $-W \le f \le W$.
(b) Spectrum of the corresponding sampled version of the signal for a sampling rate $f_s < 2W$.

For example, the spectrum of a speech signal extends well into the kilohertz region. However, insofar as telephonic communication is concerned, limiting the spectrum of the speech signal to about 3.1 kHz is adequate for such an application. Thus, to cater to corrective measure 1, the customary practice is to pass the speech signal through a low-pass filter with a cutoff frequency equal to 3.1 kHz. To cater to corrective measure 2, the filtered speech signal is sampled at a rate equal to 8 kHz, which is slightly larger than the Nyquist rate of $2 \times 3.1 = 6.2$ kHz. Indeed, the sampling rate of 8 kHz is the international standard for the sampling of speech signals.

4.3 WHY FOLLOW SAMPLING WITH CODING?

An analog signal (e.g., a speech signal) has a continuous range of amplitudes; therefore, its samples will have a continuous amplitude range, too. In other words, within the amplitude range of the signal, we find an infinite number of possible amplitude levels. However, recognizing that any human sense (e.g., the ear) can detect only finite intensity differences, we may *approximate* the analog signal by its quantized version, which is constructed of discrete amplitudes selected from an available set on a minimum-error basis. This approximation process is called *quantization*, and, unlike the sampling process, it is not reversible. In any event, with the combined use of sampling and quantization at our disposal, the specification of the analog signal becomes limited to a *discrete set of values*—discrete in both time and amplitude. Unfortunately, such a set is not in the form best suited for transmission over a wireless channel. Rather, to facilitate the transmission in an efficient manner, we require the use of an *encoding process* that translates the discrete set of sample values to a more appropriate form. In this context, the use of a *binary code* (with two symbols, namely, 0 and 1) offers the maximum benefit over the effects of channel noise on the signal. Thus, by sampling, quantizing, and encoding the analog signal, in that order, we end up with a *digital* representation of the signal in binary form.

However, with a code word of n binary symbols used to represent each sample of the analog signal, the digital representation of the analog signal requires an expansion of the channel bandwidth by a factor of n. This requirement imposes a new burden on the design of the wireless communication system. To lessen the burden, we recognize that when a speech signal, for example, is sampled at a rate slightly higher than the Nyquist rate, the resulting sampled signal is found to exhibit a high degree of *correlation* between adjacent samples. The meaning of this correlation is that the speech signal does not change rapidly from one sample to the next; the result is that the encoded version of the signal contains *redundant information*. Accordingly, not all of the symbols resulting from the encoding process are essential to the transmission of information over the wireless channel. By removing the redundant elements before encoding, we obtain a more efficient coded signal. The operation of redundancy removal, called *source coding*, has the net effect of reducing the channel bandwidth required to transmit the speech signal over the wireless channel.

The use of an encoded version of the speech signal has another beneficial effect: It offers the potential for mitigating the effects of channel noise on the signal. In particular,

through the use of *channel coding* as a follow-up to source coding, a wireless communication system is capable of correcting transmission errors incurred in transporting the information-bearing signal from the source at the transmitter input to the sink (user) at the receiver output.

The efficient implementation of coding systems relies on digital signal-processing technology in the form of silicon chips. Interestingly enough, it is this same enabling technology that has made it possible to build time-division multiple-access systems and incorporate source-coding and channel-coding techniques so as to realize the following two practical objectives of wireless communications in a cost-effective manner:

- Efficient transmission of an information-bearing signal across a wireless channel
- Reliable delivery of the signal to its destination

4.4 SHANNON'S INFORMATION THEORY

In a classic paper published in 1948, Claude Shannon laid down the mathematical foundation of communication, which has survived the test of time. In basic terms, Shannon's information theory addresses two issues of practical importance: the efficient encoding of a source signal and its reliable transmission over a noisy channel. Both issues are of fundamental importance to the study of wireless communications. In what follows, we briefly review the underpinnings of information theory.

4.4.1 Source-Coding Theorem[1]

The *source-coding theorem* is motivated by two facts:

- A common characteristic of information-bearing signals generated by physical sources (e.g., speech signals) is that, in their natural form, they contain a certain amount of information that is *redundant*, the transmission of which is wasteful of primary communication resources, namely, transmit power and channel bandwidth.
- For *efficient* signal transmission, the redundant information should be removed from the information-bearing signal prior to transmission.

In its simplest form, the source-coding theorem may be stated as follows:

> *Given a discrete memoryless source characterized by a certain amount of entropy, the average code-word length for a distortionless source-encoding scheme is upper bounded by the entropy.*

In information theory, *entropy* is a measure of the average information content per symbol emitted by the source. According to the source-coding theorem, entropy represents a fundamental limit on the average number of bits per source symbol necessary to represent a discrete memoryless source, in that that number can be made as small as, but no smaller than, the entropy of the source. We may thus express the *efficiency* of a

source encoder as

$$\eta = \frac{H(S)}{\bar{L}} \tag{4.1}$$

where $H(S)$ is the entropy of the source with source alphabet S and $\bar{L}$ is the average number of bits per symbol used in the source-encoding process. Entropy is itself defined by

$$H(S) = \sum_{k=0}^{K-1} p_k \log_2\left(\frac{1}{p_k}\right) \tag{4.2}$$

where p_k is the probability that a certain symbol s_k is emitted by the source. With the base of the logarithm in this definition equal to 2, the entropy is measured in *bits*, a basic unit of information.

4.4.2 Channel-Coding Theorem

Another practical reality is the inevitable presence of channel noise, which produces errors between the output and input data sequences of a digital communication system. For a wireless communication channel, the probability of error may exceed 10^{-1}, which means that (on the average) only 9 out of 10 transmitted symbols are received correctly. For many applications, this level of reliability is unacceptable. To achieve *reliable* communication over a noisy wireless channel, we resort to the use of *channel coding*, which consists of mapping an incoming data sequence into an output data sequence in such a way that the overall effect of channel noise on the system is minimized. The mapping operation performed in the transmitter is accomplished by a *channel encoder*, and the inverse mapping operation performed in the receiver is accomplished by a *channel decoder*. The channel encoder and channel decoder are both under the designer's control, with the objective of optimizing the overall reliability of the wireless communication system. The approach taken to realize this objective is to purposely introduce *redundancy* in the channel encoder so as to reconstruct the original data sequence as accurately as possible. Thus, in a rather loose sense, we may view channel coding as the *dual* of source coding, in that the former introduces controlled redundancy to improve data transmission reliability, whereas the latter reduces natural redundancy to improve data transmission efficiency.

In a theoretical context, a concept basic to channel coding is that of *channel capacity*, which, for a discrete memoryless channel, represents the maximum amount of information that can be transmitted per channel use. The *channel-coding theorem* may be stated as follows:

> *If a discrete memoryless channel has capacity C and a source generates information at a rate less than C, then there exists a coding technique such that the output of the source may be transmitted over the channel with an arbitrarily low probability of symbol error.*

The channel-coding theorem thus specifies the channel capacity C as a *fundamental limit* on the rate at which the transmission of reliable (error-free) messages can take place over a discrete, memoryless noisy channel.

Consider the case of a block code, in which an incoming message sequence is subdivided into sequential blocks k bits long and each k-bit block is mapped into a new n-bit block with $n > k$. The number of redundant bits added by the channel encoder to each transmitted block is $n - k$ bits. The ratio k/n is called the *code rate*. Using r to denote the code rate, we may thus write

$$r = \frac{k}{n} \tag{4.3}$$

where, of course, r is always less than unity. For a prescribed block length k, the code rate r (and therefore the system's coding efficiency) approaches zero as the encoded block length n approaches infinity.

The most unsatisfactory feature of the channel-coding theorem, however, is its nonconstructive nature. The theorem asserts the existence of good codes, but does not tell us how to find them. By *good codes*, we mean families of channel codes that are capable of providing reliable transmission of information (i.e., with an arbitrarily small probability of symbol error) over a noisy channel of interest at bit rates up to a maximum value less than the capacity of that channel. The *construction* of good codes is taken up in Sections 4.6 through 4.9.

4.4.3 Information Capacity Theorem

The source-coding theorem focuses on the efficient mapping of a source sequence. The channel-coding theorem focuses on the reliable transmission of a sequence over a noisy channel. Shannon's third theorem, the information capacity theorem, brings out the trade-off between channel bandwidth and signal-to-noise ratio at the channel output in the most insightful way.

Consider, then, the idealized setting of a discrete-time, memoryless, Gaussian channel, the output of which is described by the signal-to-noise ratio P/σ^2, where P is the average transmitted power and σ^2 is the variance of the zero-mean Gaussian-distributed channel noise. Let B denote the channel bandwidth, measured in hertz, and C denote the corresponding channel capacity, measured in bits per second. Then, according to the celebrated information capacity theorem, we may express the information capacity as

$$C = B\log_2\left(1 + \frac{P}{\sigma^2}\right) \text{ bits/s} \tag{4.4}$$

The information capacity theorem is one of the most remarkable results of information theory, for, in a single formula, it highlights most vividly the interplay among three key system parameters: channel bandwidth, average transmitted power, and channel noise variance $\sigma^2 = N_0 B$, where $N_0/2$ is the (two-sided) power spectral density of the additive channel noise. More specifically, the dependence of information capacity C on the channel bandwidth is *linear*, whereas its dependence on the signal-to-noise ratio P/σ^2 is *logarithmic*. Accordingly, it is easier to increase the information capacity of a wireless

channel by expanding its bandwidth than increasing the average transmitted power, for a prescribed noise spectral density.

Equation (4.4) applies to a *single-input, single-output channel*. Extension of the information capacity theorem to *multiple-input, multiple-output channels* is discussed in Chapter 6.

4.4.4 Rate Distortion Theory

According to the source-coding theorem, for a discrete memoryless source, the average code-word length must be as large as the source entropy in order to represent the source perfectly. However, in many practical situations, there are constraints that force the source coding to be imperfect, thereby giving rise to unavoidable *distortion*. In situations of this kind, we speak of *source coding with a fidelity criterion*. The branch of information theory that deals with this criterion is referred to as *rate distortion theory*, which finds applications in two types of situations:

1. *Source coding*, wherein the permitted alphabet of the source code cannot represent the source output exactly, thereby forcing us to put up with *lossy data compression*.
2. *Information transmission*, which is required at a rate greater than the permissible channel capacity.

We may therefore view rate distortion theory as a natural extension of the source-coding and channel-coding theorems.

In the context of lossy data compression, we may think of a device that supplies a code with the least number of symbols for the representation of the source output, subject to an acceptable distortion. The data compressor thus retains the essential information content of the source output by blurring fine details in a deliberate, but controlled, manner. The data compression is *lossy*, in the sense that the source entropy is reduced. In an analog system, an example of lossy compression would be removing the frequency content of a voice signal above 3.1 kHz for a telephone call.

The entropy of an analog source is infinite by virtue of the fact that, in theory, its amplitude may occupy an infinitely large range. Hence, encoding of the source output at a *finite rate* will necessitate the use of a signal compression code. Stated another way, regardless of the sampling rate, it is impossible to digitally encode an analog signal with a finite number of bits without producing some distortion (i.e., loss of information). In addition to sampling, the digital representation of the analog signal involves the process of *quantization*, whereby the amplitude of each sample is rounded off to the nearest level selected from a finite set of levels.

Various types of quantizers have been proposed in the literature. However, insofar as speech is concerned—a matter that is of primary concern in wireless communications—*vector quantizers* are of special interest because of their improved signal-to-quantization noise ratio. A distinctive feature of this class of quantizers is that each block of consecutive samples of the speech signal is treated as a single entity, namely, a *vector*. The essential operation in a vector quantizer is the quantization of a random vector of samples by encoding it as a binary code word. The encoding is

accomplished by comparing the vector with a *codebook* consisting of a set of stored reference vectors known as *code vectors* or *patterns*. Each pattern in the codebook is used to represent input vectors that are identified by the encoder as similar to the particular pattern, subject to the maximization of an appropriate fidelity criterion. The encoding process in a vector quantizer may thus be viewed as a *pattern-matching operation*. The code-excited linear predictive encoder, considered in the next section, is an example of a vector quantizer.

4.5 SPEECH CODING[2]

A recurrent theme in the design of digital wireless communication systems is the *efficient utilization of the allotted spectrum*. With this important objective in mind, digital wireless communication systems rely on the use of *speech coding* to remove almost all of the natural redundancy inherent in a speech signal, while maintaining a high-quality reproduction of the original signal on decoding. The most common approach to speech coding is to use a *linear predictive coding* (LPC) approach in one form or another. As the name implies, *linear prediction* is basic to this approach—hence the devotion of Subsection 4.5.1 to the essence of this operation. Subsections 4.5.2 and 4.5.3 discuss two extensions of linear prediction, namely, multipulse excited LPC and code-excited LPC.

4.5.1 Linear Prediction

One of the most celebrated problems in signal processing is that of *predicting* the present (or future) value of a discrete-time signal, given a set of past samples of the signal. In so doing, we have a powerful approach for building a *predictive model* for the underlying dynamics responsible for the generation of the signal. The smaller the prediction error in a statistical sense, the more reliable the model will be. The *prediction error* is defined as the difference between the actual future value of the signal and the predicted value produced by the model.

To be specific, consider a discrete-time signal represented by the set of samples $x(t), x(t-T_s),...,x(t-NT_s)$, where T_s denotes the sampling period. The sampling rate f_s is related to the sampling period as $f_s = 1/T_s$. In the *one-step* form of linear prediction, the requirement is to estimate the present value of the signal $x(t)$, given the N past samples $x(t-T_s), x(t-2T_s),...,x(t-NT_s)$. Let $\hat{x}(t)$ denote the output of the predictive model. In linear prediction, $\hat{x}(t)$ is expressed as a linear combination of the N past samples via the formula

$$\begin{aligned}\hat{x}(t) &= \sum_{n=1}^{N} a_n x(t-nT_s) \\ &= \mathbf{a}^T\mathbf{x}\end{aligned} \tag{4.5}$$

where the superscript T denotes matrix transposition and the N-by-1 signal vector $\mathbf{x}$ and parameter vector $\mathbf{a}$ are, respectively,

$$\mathbf{x} = [x(t-T_s), x(t-2T_s), \ldots, x(t-NT_s)]^T \tag{4.6}$$

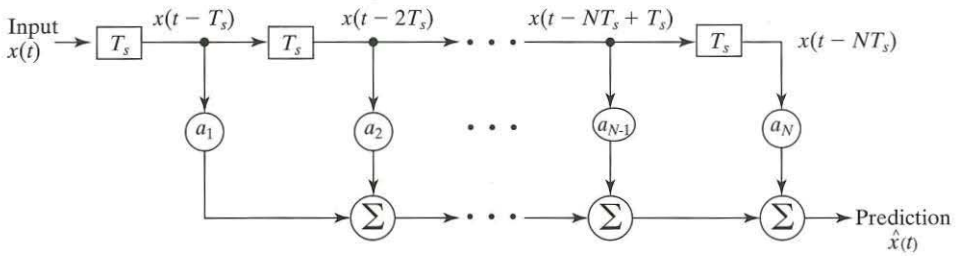

FIGURE 4.3 Structure of the FIR (i.e., tapped-delay-line) predictor.

and

$$\mathbf{a} = [a_1, a_2, \ldots, a_N]^T \tag{4.7}$$

The parameters $a_1, a_2, \ldots, a_N$ in effect define the N degrees of freedom available to us in designing the predictive model. Equation (4.5) readily suggests the *tapped-delay-line (TDL)* or *finite-duration impulse (FIR) filter*, shown in Fig. 4.3 as the structure for the predictive model.

The key question is: how do we determine the filter coefficients? To answer this question, we need a statistical criterion for optimizing the design of the filter. A criterion widely used in practice is the *mean-square-error (MSE) criterion*, defined by

$$\text{MSE} = \mathbf{E}[(x(t) - \hat{x}(t))^2] \tag{4.8}$$

where $\mathbf{E}$ denotes the statistical expectation operator and the difference $x(t) - \hat{x}(t)$ stands for the prediction error. It turns out that if $x(t)$ is the sample value of a stationary random process, then the optimum value of the parameter vector $\mathbf{a}$ is given by

$$\mathbf{a} = \mathbf{R}^{-1}\mathbf{r} \tag{4.9}$$

where the N-by-N matrix $\mathbf{R}$ is the correlation matrix of the tap inputs of the predictive model and the N-by-1 vector $\mathbf{r}$ has as its elements the autocorrelation of the input signal $x(t)$ for lags $T_s, 2T_s, \ldots, NT_s$. The symbol $\mathbf{R}^{-1}$ in Eq. (4.9) stands for the *inverse* of the correlation matrix. (Note the similarity between a linear predictor exemplified by Eq. (4.9) and a smoothing filter defined by Eq. (3.89).)

If, however, the physical process responsible for the generation of the signal $x(t)$ is *nonstationary* (i.e., its statistics vary with time), then we require the use of an adaptive procedure whereby the model parameters are allowed to vary with time. A brief discussion of adaptive filtering algorithms is presented in Appendix E.

4.5.2 Multipulse Excited LPC

This form of speech coding exploits the *principle of analysis by synthesis*, which means that the encoder includes a replica of the decoder in its design. Specifically, the encoder consists of three main parts, as indicated in Fig. 4.4(a):

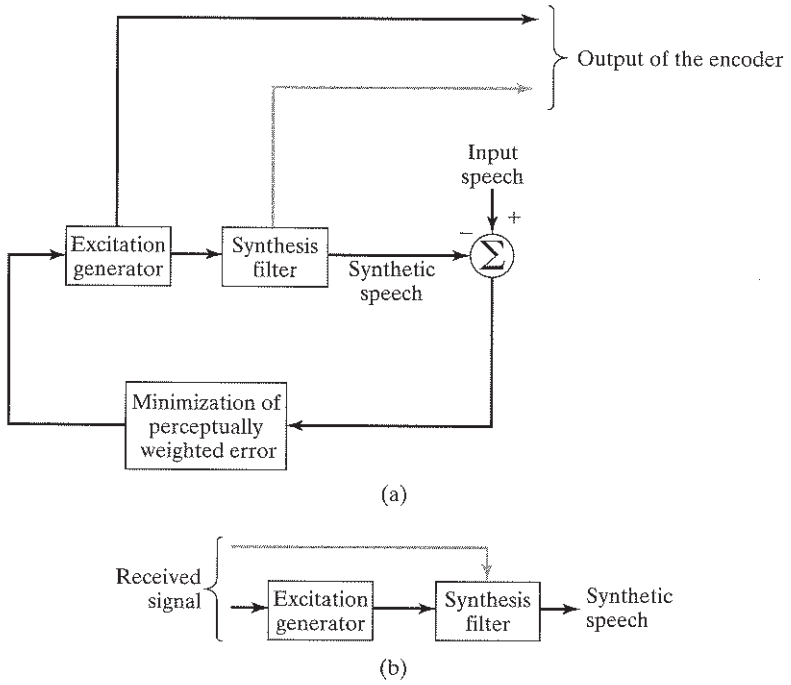

FIGURE 4.4 Multipulse excited linear predictive codec. (a) Encoder. (b) Decoder whose input (the received signal) consists of quantized filter parameters (signified by lighter arrows) and quantized excitation, as produced by the generator.

1. A *synthesis filter*, for the predictive modeling of speech. This may consist of an all-pole filter (i.e., a linear filter whose transfer function has poles only), which is designed to model the *short-term* spectral envelope of speech. The term "short-term" refers to the fact that the filter parameters are computed on the basis of predicting the present sample of the speech signal from 8 to 16 previous samples. The synthesis filter may also include a *long-term* predictor for modeling the fine structure of the speech spectrum. In such a case, the long-term predictor is connected in cascade with the short-term predictor. In any event, the function of the synthesis filter is to produce a synthetic version of the original speech that is configured to be of high quality.
2. An *excitation generator*, for producing the excitation applied to the synthesis filter. The excitation consists of a definite number of pulses every 5 to 15 ms. The amplitudes and positions of the individual pulses are adjustable.
3. *Error minimization*, for optimizing the perceptually weighted error between the original speech and the synthesized speech. The aim of this minimization is to optimize the amplitudes and positions of the pulses used in the excitation. Typically, a mean-square-error criterion (i.e., the mean-square value of the difference between an actual sample of the speech signal and its predicted value) is used for the minimization.

Thus, as shown in Fig. 4.4(a), the three parts of the encoder form a *closed-loop* optimization procedure, which permits the encoder to operate at a bit rate below 16 kb/s while maintaining high-quality speech on reconstruction.

The encoding procedure itself has two main steps:

1. The free parameters of the synthesis filter are computed with the use of the actual speech samples as input. This computation is performed outside the optimization loop over a period of 10 to 30 ms, during which the speech signal is treated as pseudostationary.
2. The optimum excitation for the synthesis filter is computed by minimizing the perceptually weighted error with the loop closed, as in Fig. 4.4(a).

Thus, the speech samples are divided into *frames* (10 to 30 ms long) for computing the filter parameters, and each frame is divided further into *subframes* (5 to 15 ms long) for optimizing the excitation. The quantized filter parameters and quantized excitation constitute the transmitted signal.

Note that by first permitting the filter parameters to vary from one frame to the next and then permitting the excitation to vary from one subframe to the next, the encoder is able to track the nonstationary behavior of speech, albeit on a batch-by-batch basis.

The decoder, located in the receiver, consists simply of two parts: the excitation generator and the synthesis filter, as shown in Fig. 4.4(b). These two parts are identical to the corresponding ones in the encoder. The function of the decoder is to use the received signal to produce a synthetic version of the original speech signal. This is achieved by passing the decoded excitation through the synthesis filter, whose parameters are set equal to those in the encoder.

To reduce the computational complexity of the *codec* (a contraction of coder/decoder), the intervals between the individual pulses in the excitation are constrained to assume a common value. The resulting analysis-by-synthesis codec is said to have a *regular-pulse excitation*.

4.5.3 Code-Excited LPC

Figure 4.5 shows the block diagram of the *code-excited LPC*, commonly referred to as CELP, which provides another method for speech coding. The distinguishing feature of CELP is the use of a predetermined *codebook* of stochastic (zero-mean white Gaussian) vectors as the source of excitation for the synthesis filter. The synthesis filter itself consists of two all-pole filters connected in cascade. One filter performs short-term prediction and the other performs long-term prediction.

As with the multipulse excited LPC, the free parameters of the synthesis filter are computed first, using the actual speech samples as input. Next, the choice of a particular vector (code) stored in the excitation codebook and the gain factor G in Fig. 4.5 is optimized by minimizing the average power of the perceptually weighted error between the original speech and the synthesized speech (i.e., the output of the synthesis filter). The address of the random vector selected from the codebook and the corresponding quantized gain factor, together with the quantized filter parameters, constitute the transmitted signal.

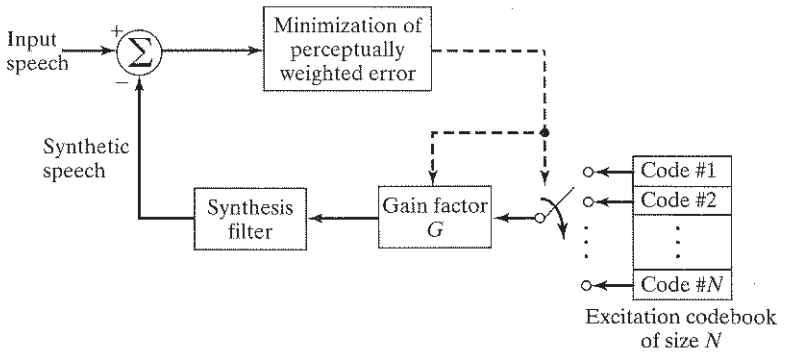

FIGURE 4.5 Encoder of the code-excited linear predictive codec (CELP). The transmitted signal consists of the address of the code selected from the codebook, the quantized gain factor G, and quantized filter parameters.

An identical copy of the codebook is made available to the decoder, and likewise for the synthesis filter. Hence, given the received signal, the decoder is able to parameterize its own synthesis filter and determine the appropriate excitation for the synthesis filter, thereby producing a synthetic version of the original speech signal.

CELP is capable of producing good-quality speech at bit rates below 8 kb/s. However, its computational complexity is intensive, because of the exhaustive search it makes of the excitation codebook. In particular, the weighted synthesized speech in the encoder has to be computed for all the entries in the codebook and then compared with the weighted original speech. Nevertheless, real-time implementation of CELP codecs has been made possible by virtue of advances in digital signal processing and VLSI technology.

4.6 ERROR-CONTROL CODING[3]

The next function to be addressed in the basic TDMA link of Fig. 4.1 is that of channel coding. There are two broadly defined categories of error-control coding techniques to be considered for this function:

1. *Forward error-correction (FEC) codes*, which rely on the controlled use of redundancy in the transmitted code word for both the *detection and correction* of errors incurred during the course of transmission over a noisy channel. Irrespective of whether the decoding of the received (noisy) code word is successful, no further processing is performed at the receiver. Accordingly, channel-coding techniques suitable for FEC require only a one-way link between the transmitter and the receiver.
2. *Automatic-repeat request (ARQ) schemes*, which use redundancy merely for the purpose of *error detection*. Upon the detection of an error in a transmitted code word, the receiver requests a repeat transmission of the word in question, thereby necessitating the use of a *return path* (i.e., feedback channel).

Unfortunately, the need to retransmit a code word introduces *latency* into the operation of the system, making the use of ARQ unsuitable for speech communications by wireless, which requires information transmission in real time. As for data, the ARQ technique is closely related to link-layer functions to be described in Chapter 7 and will therefore not be discussed here further.

Historically, FEC codes have been classified into *block codes* and *convolutional codes*. The distinguishing feature of this particular classification is the presence or absence of *memory* in the encoders for the two codes.

As mentioned in Section 4.1, to generate an (n,k) block code, the channel encoder accepts information in successive k-bit *blocks* and, to each block, adds $n - k$ redundant bits that are algebraically related to the k message bits, thereby producing an overall encoded block of n bits, where $n > k$. The channel encoder produces bits at the rate $R_0 = (n/k)R_s$, where R_s is the bit rate of the information source and $r = k/n$ is the *code rate*. The bit rate R_0, coming out of the encoder, is called the *channel data rate*. The code rate is a dimensionless ratio, whereas the data rate produced by the source and the channel data rate are both measured in bits per second.

In a convolutional code, the encoding operation may be viewed as the *discrete-time convolution* of the input sequence with the impulse response of the encoder. The duration of the impulse response equals the *memory* of the encoder. Accordingly, the encoder for a convolutional code operates on the incoming message sequence, using a "sliding window" equal in duration to its own memory. This, in turn, means that in a convolutional code, as opposed to what happens in a block code, the channel encoder accepts message bits as a continuous sequence and generates a continuous sequence of encoded bits at a higher rate.

Both linear block codes and convolutional codes find applications in wireless communication systems. In the next section, we discuss convolutional codes for which there exist powerful decoding schemes that are, in theory, capable of achieving optimal performance. In Section 4.12, we discuss an equally powerful class of linear block codes known as turbo codes.

4.6.1 Cyclic Redundancy Check Codes

An an example of a linear block code, we consider *cyclic codes*, which have the property that any cyclic shift of a code word in the code is also a code word. Cyclic codes are extremely well suited for *error detection*. We make this statement for two reasons: first, cyclic codes can be designed to detect many combinations of likely errors; second, the implementation of both encoding and error-detecting circuits is very simple with cyclic codes. It is for these reasons that many of the error-detecting codes used in practice are of the cyclic-code variety. A cyclic code used for error detection is referred to as a *cyclic redundancy check (CRC) code*.

We define an *error burst* of length B in an n-bit received word as a contiguous sequence of B bits in which the first and last bits or any number of intermediate bits are received in error. Binary (n,k) CRC codes are capable of detecting the following error patterns:

TABLE 4.1 CRC Codes.

Code	Generator Polynomial, $g(X)$	$n-k$
CRC-12 code	$1+D+D^2+D^3+D^{11}+D^{12}$	12
CRC-16 code (USA)	$1+D^2+D^{15}+D^{16}$	16
CRC-ITU code	$1+D^5+D^{12}+D^{16}$	16

1. All error bursts of length $n-k$ or less.
2. A fraction of error bursts of length equal to $n-k+1$; the fraction equals $1-2^{-(n-k-1)}$.
3. A fraction of error bursts of length greater than $n-k+1$; the fraction equals $1-2^{-(n-k-1)}$.
4. All combinations of $d_{\min}-1$ (or fewer) errors, where $d_{\min}$ is the minimum distance that defines the error-correcting capability of the code.
5. All error patterns with an odd number of errors if the generator polynomial $g(D)$ for the code has an even number of nonzero coefficients, where D denotes a unit-delay operator; the generator polynomial of a code is responsible for generating all the code words in the code in response to the incoming information bits.

The issues of generator polynomial and minimum distance are discussed in greater detail later in the chapter.

Table 4.1 presents the generator polynomials of three CRC codes that have become international standards. All three codes contain $1+D$ as a prime factor. The CRC-12 code is used for 6-bit characters, and the other two codes are used for 8-bit characters. CRC codes provide a powerful method of error detection for use in automatic-repeat request (ARQ) strategies.

4.7 CONVOLUTIONAL CODES

As already mentioned, the encoder in block coding accepts a k-bit message block and generates an n-bit code word. Thus, code words are produced on a block-by-block basis. Clearly, a provision must be made in the encoder to *buffer* an entire message block before generating the associated code word. There are applications, however, in which the message bits come in *serially* rather than in large blocks, in which case the use of a buffer may be undesirable. In such situations, the use of *convolutional coding* may be the preferred method. A convolutional coder generates redundant bits by using *modulo-2 convolutions*—hence the name of the coder.

The encoder of a binary convolutional code with rate $1/n$, measured in bits per symbol, may be viewed as a *finite-state machine (FSM)* that consists of an M-stage shift register with prescribed connections to n modulo-2 adders and a multiplexer that serializes the outputs of the adders. An L-bit message sequence produces a coded output sequence of length $n(L+M)$ bits. The code rate of the convolutional code is therefore given by

$$r = \frac{L}{n(L+M)} \text{ bits/symbol} \tag{4.10}$$

Typically, $L >> M$. Hence, the code rate reduces to

$$r \approx \frac{1}{n} \text{ bits/symbol} \tag{4.11}$$

The *constraint length* of a convolutional code, expressed in terms of message bits, is defined as the number of shifts over which a single message bit can influence the encoder output. In an encoder with an M-stage shift register, the *memory* of the encoder equals M message bits, and $K = M + 1$ shifts are required for a message bit to enter the shift register and finally come out. Hence, the constraint length of the encoder is K.

Figure 4.6 shows a convolutional encoder with $n = 2$ and $K = 3$. Hence, the code rate of this encoder is 1/2. The encoder operates on the incoming message sequence, one bit at a time. Note that the convolutional code generated by the encoder is *nonsystematic*, in that the message bits are transmitted in altered form.

Each path connecting the output to the input of a convolutional encoder may be characterized in terms of its *impulse response*, defined as the response of that path to a symbol 1 applied to its input, with each flip-flop in the encoder set initially in the zero state. Equivalently, we may characterize each path in terms of a *generator polynomial*, defined as the *unit-delay transform* of the impulse response. To be specific, let the *generator sequence* $(g_0^{(i)}, g_1^{(i)}, g_2^{(i)}, \ldots, g_M^{(i)})$ denote the impulse response of the ith path, where the coefficients $(g_0^{(i)}, g_1^{(i)}, g_2^{(i)}, \ldots, g_M^{(i)})$ equal 0 or 1, depending on the connections in the encoder. Correspondingly, the *generator polynomial* of the ith path is defined by

$$g^{(i)}(D) = g_0^{(i)} + g_1^{(i)} D + g_2^{(i)} D^2 + \ldots + g_M^{(i)} D^M \qquad i = 1,2,\ldots,n \tag{4.12}$$

where D denotes the unit-delay variable. The complete convolutional encoder is described by the set of generator polynomials $\{g^{(1)}(D), g^{(2)}(D), \ldots, g^{(n)}(D)\}$.

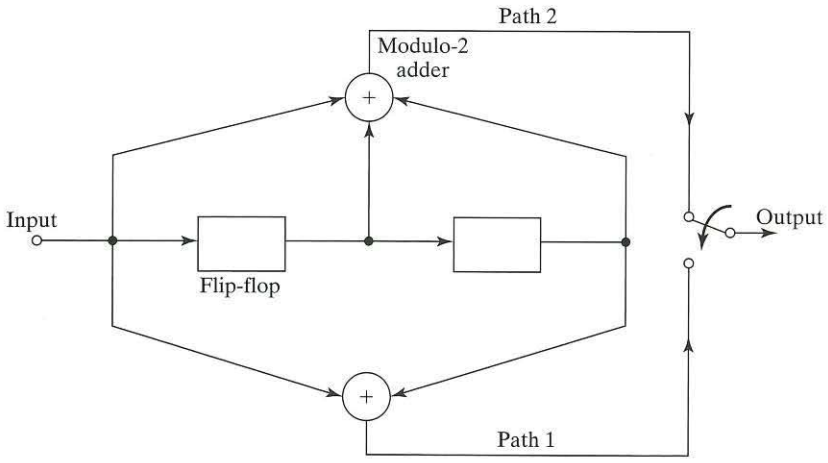

FIGURE 4.6 Convolutional encoder with constraint length 3 and rate 1/2.

EXAMPLE 4.1 Convolutional Code of Constraint Length $K = 3$

Consider the convolutional encoder of Fig. 4.6, which has two paths, numbered 1 and 2 for convenience of reference. The impulse response of path 1 (the lower path) is (1,0,1). Hence, the corresponding generator polynomial is given by

$$g^{(1)}(D) = 1 + D^2$$

The impulse response of path 2 (the upper path) is (1,1,1). Hence, the corresponding generator polynomial is given by

$$g^{(2)}(D) = 1 + D + D^2$$

For, say, the message sequence (10011), we have the polynomial representation

$$m(D) = 1 + D^3 + D^4$$

As with Fourier transformation, convolution in the time domain is transformed into multiplication in the D-domain, which is, of course, performed in a modulo-2 fashion.

The output polynomial of path 1 is given by

$$\begin{aligned} c^{(1)}(D) &= g^{(1)}(D)m(D) \\ &= (1 + D^2)(1 + D^3 + D^4) \\ &= 1 + D^2 + D^3 + D^4 + D^5 + D^6 \end{aligned}$$

From this result, we immediately deduce that the output sequence of path 1 is (1011111). Similarly, the output polynomial of path 2 is given by

$$\begin{aligned} c^{(2)}(D) &= g^{(2)}(D)m(D) \\ &= (1 + D + D^2)(1 + D^3 + D^4) \\ &= 1 + D + D^2 + D^3 + D^6 \end{aligned}$$

The output sequence of path 2 is therefore (1111001). Finally, multiplexing the two output sequences, we get the encoded sequence

$$\mathbf{c} = (11, 10, 11, 11, 01, 01, 11)$$

Note that the message sequence of length $L = 5$ bits produces an encoded sequence of length $n(L + K - 1) = 14$ bits. Note also that, for the shift register to be restored to its zero initial state, a terminating sequence of $K - 1 = 2$ zeros is appended to the last input bit of the message sequence. The terminating sequence of $K - 1$ zeros is called the *tail of the message*, or the *flush bits*. ■

Problem 4.1 In Example 4.1, we determined the encoded sequence $\mathbf{c}$ by operating in the transformed-D domain. Repeat the evaluation of $\mathbf{c}$ by operating exclusively in the time domain. ■

To simplify the presentation, we find it convenient to use an *octal* representation to describe the generator polynomial of a convolutional code. Specifically, *octal* refers to the value assumed by the generator polynomial $g^{(i)}(D)$ of Eq. (4.12) when

the unit-delay variable D is set equal to the number 2 and the result is then converted to base 8. For example, the generator polynomial of the convolutional encoder of Fig. 4.6 is referred to simply as (5,7). For path 1, putting $D = 2$ in $g^{(1)}(D)$ yields $1 + 2^2 = 5$, and putting $D = 2$ in $g^{(2)}(D)$ for path 2 yields $1 + 2 + 2^2 = 7$—hence reference to the convolutional code as (5,7).

4.7.1 Trellis and State Diagrams of Convolutional Codes

To develop an insight into the operation of a convolutional encoder, we can use a graphic representation known as a *trellis*. This diagram is so called because a trellis is a treelike structure with reemerging branches. Figure 4.7 shows the trellis diagram for the convolutional encoder of Fig. 4.6. The convention used in Fig. 4.7 to distinguish between input symbols 0 and 1 is as follows. A code branch produced by an input 0 is drawn as a *solid* line, whereas a code branch produced by an input 1 is drawn as a *dashed* line. Each input (message) sequence corresponds to a specific path through the trellis. For example, we readily see from the figure that the message sequence (10011) produces the encoded output sequence (11, 10, 11, 11, 01), which agrees with the result of Example 4.1.

A trellis is insightful in that it brings out explicitly the fact that the associated convolutional encoder is a *finite-state machine*. We define the *state* of a convolutional encoder of rate $1/n$ in terms of the $(K-1)$ message bits stored in the encoder's shift register. At time j, the portion of the message sequence containing the most recent K bits is written as $m_{j-K+1}, \ldots, m_{j-1}, m_j$, where m_j is the *current* bit. The $(K-1)$-bit state of the encoder at time j is therefore written simply as $m_{j-1}, \ldots, m_{j-K+2}, m_{j-K+1}$. In the case of the simple convolutional encoder of Fig. 4.6, we have $(K-1) = 2$. Hence, the state of this encoder can assume any one of four possible values, as described in Table 4.2(a). The trellis contains $L + K$ levels, where L is the length of the incoming message sequence and K is the constraint length of the code. The *levels* of the trellis are labeled $j = 0,1,\ldots,$ $L + K - 1$ in Fig. 4.7 for $K = 3$. Level j is also referred to as the *depth* j; both terms are used interchangeably. The first $K - 1$ levels correspond to the encoder's departure from

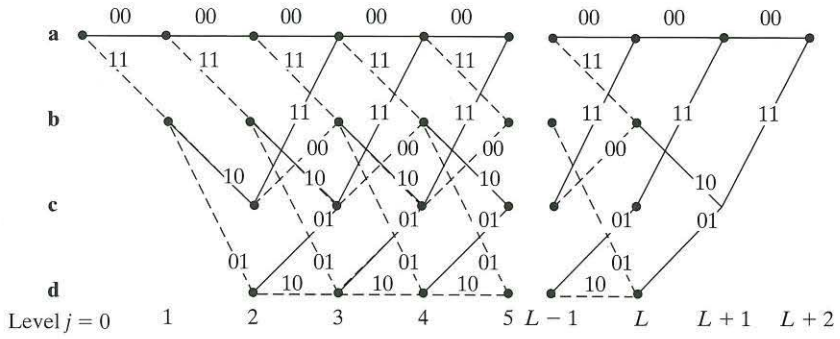

FIGURE 4.7 Trellis for the convolutional encoder of Figure 4.6.

an initial state, and the last $K-1$ levels correspond to its return to the final state. Clearly, not all the states can be reached in these two portions of the trellis. However, in the central portion, for which the level j lies in the range $K-1 \leq j \leq L$, all the states of the encoder are reachable. Note that the central portion of the trellis exhibits a fixed periodic structure.

Problem 4.2 The *state diagram* provides another graphical representation of a convolutional encoder. This second representation can be derived from the trellis diagram by focusing on a portion of the trellis from level j to level $j+1$, chosen in such a way that all possible states of the encoder are fully displayed. In the case of the encoder of Fig. 4.6, that requirement is attained for $j \geq 2$. The state diagram is obtained by coalescing the left and right nodes of the portion of the trellis diagram so chosen. The following convention is used in the construction of the state diagram: A transition from one state to another in response to input 0 is represented by a *solid* branch, whereas a transition in response to input 1 is represented by a *dashed* branch.

(1) Starting with the trellis diagram of Fig. 4.7, justify the construction of the state diagram shown in Fig. 4.8.

(2) Derive Table 4.2 for the encoder, with part (i) of the table showing the actual states of the encoder and part (ii) showing the pertinent state transitions.

(3) Starting at state ***a***, corresponding to the *all-zero state*, walk through the state diagram of Fig. 4.8 in response to the message sequence (10011). Hence, verify that this sequence produces the encoded output sequence (11, 10, 11, 11, 01) in agreement with Example 4.1. ■

TABLE 4.2 The Convolutional Encoder of Figure 4.6.

(i) State table

State	Binary Description
a	00
b	10
c	01
d	11

(ii) State-transition table

Input binary symbol	Old state	New state	Emitted symbol
0	***a***	***a***	00
	b	***c***	10
	c	***a***	11
	d	***c***	01
1	***a***	***b***	11
	b	***d***	01
	c	***b***	00
	d	***d***	10

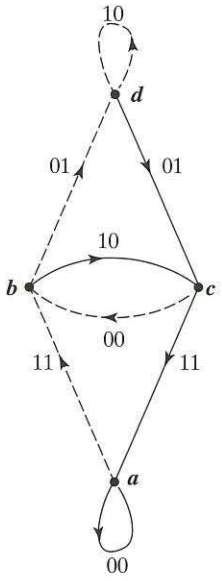

FIGURE 4.8 State diagram of the convolutional encoder of Figure 4.6.

4.7.2 Free Distance of a Convolutional Code

The performance of a convolutional code depends not only on the decoding algorithm used, but also on the error-correction properties of the code. In this context, the most important single measure of a convolutional code's ability to combat channel noise is the *free distance*, denoted by d_{free}. To pave the way for the definition of this important parameter, we need to introduce a couple of related definitions:

- *The Hamming weight* of a linear code is the number of nonzero elements in the code vector of the code.
- *The Hamming distance* between a pair of code vectors is the number of locations where their respective elements are different.

On this basis, the free distance of a convolutional code is formally defined as the *minimum (i.e., smallest) Hamming distance* between any two code vectors in the code. The important point to note here is that a convolutional code with free distance d_{free} can correct t errors only if d_{free} is greater than $2t$. Stated another way, a convolutional decoding algorithm will fail if the number of closely spaced errors in the received sequence exceeds $d_{\text{free}}/2$.

The free distance d_{free} can be determined from the state diagram of the convolutional encoder. Given the introductory nature of this book, the reader is referred elsewhere[4] for this determination. Suffice it to say, however, that the free distance of a convolutional code depends on the constraint length K of the code, as indicated in Table 4.3. In this table, the free distances of two types of convolutional codes—systematic

TABLE 4.3 Maximum Free Distances Attainable with Systematic and Nonsystematic Convolutional Codes of Rate 1/2.

Constraint Length K	Systematic	Nonsystematic
2	3	3
3	4	5
4	4	6
5	5	7
6	6	8
7	6	10
8	7†	10
9	‡	12

† The free distance given here is the best that can be achieved by a systematic nonrecursive code with rate 1/2, $K = 8$, and generator polynomials 400, 671 (Claude Berrou, private communication).
‡ The free distance for a systematic convolutional code for $K = 9$ is not known to the authors.

and nonsystematic—are listed for varying K. In a *systematic convolutional code*, the incoming message bits are transmitted in unaltered form; this constraint is removed from the generation of a *nonsystematic convolutional code*. The convolutional code generated by the encoder of Fig. 4.6 is of the nonsystematic kind. An example of a systematic convolutional code is presented in Section 4.12.

According to Table 4.3, the nonsystematic convolutional code of constraint length $K = 3$, generated by the encoder of Fig. 4.6, has a free distance $d_{\text{free}} = 5$. Hence, this convolutional code is capable of correcting up to two errors in the sequence it receives over a binary symmetric channel, as will be demonstrated later in Example 4.2.

4.8 MAXIMUM-LIKELIHOOD DECODING OF CONVOLUTIONAL CODES

Now that we understand the operation of a convolutional encoder, the next issue to be considered is the decoding of a convolutional code. To that end, let $\mathbf{m}$ denote a *message vector* and $\mathbf{c}$ denote the corresponding *code vector* applied by the encoder to the input of a discrete memoryless channel. Let $\mathbf{r}$ denote the *received vector*, which may differ from the transmitted code vector due to channel noise. Given the received vector $\mathbf{r}$, the decoder is required to make an *estimate* $\hat{\mathbf{m}}$ of the message vector. Since there is a one-to-one correspondence between the message vector $\mathbf{m}$ and the code vector $\mathbf{c}$, the decoder may equivalently produce an estimate $\hat{\mathbf{c}}$ of the code vector. We may then put $\hat{\mathbf{m}} = \mathbf{m}$ if and only if $\hat{\mathbf{c}} = \mathbf{c}$; otherwise, a *decoding error* is committed in the receiver. The *decoding rule* for choosing the estimate $\hat{\mathbf{c}}$, given the received vector $\mathbf{r}$, is said to be optimum when the *probability of decoding error* is minimized. For equiprobable messages, we may state that the probability of decoding error is minimized if, in accordance with the maximum likelihood principle, the estimate $\hat{\mathbf{c}}$ is chosen to maximize the *log-likelihood function* for the receiver. Let $p(\mathbf{r}|\mathbf{c})$ denote the conditional probability of receiving $\mathbf{r}$, given that $\mathbf{c}$ was sent. Then, by definition, the log-likelihood

function equals $\log p(\mathbf{r}|\mathbf{c})$. The *maximum-likelihood decoder* or decision rule may now be stated as follows:

> *Choose the estimate* $\hat{\mathbf{c}}$ *for which the log-likelihood function* $\log p(\mathbf{r}|\mathbf{c})$ *is maximum* (4.13)

To illustrate, consider the special case of a *memoryless binary symmetric channel* in which both the transmitted code vector $\mathbf{c}$ and the received vector $\mathbf{r}$ represent binary sequences, say, of length N. The channel is also characterized by equality of the conditional probabilities p_{01} and p_{10}, where p_{01} refers to the receiver making a decision in favor of symbol 0 when symbol 1 was sent, and p_{10} refers to the situation in which the reverse is true. Naturally, the two vectors $\mathbf{c}$ and $\mathbf{r}$ may differ from each other in some locations because of errors due to channel noise. Let $\mathbf{r}$ be the received code vector, and $\hat{\mathbf{c}}$ be a candidate code vector. Then, for each $\hat{\mathbf{c}}$, we compute the conditional probability

$$p(\mathbf{r}|\hat{\mathbf{c}}) = \prod_{i=1}^{N} p(r_i|\hat{c}_i) \tag{4.14}$$

which follows from the memoryless property. Correspondingly, the log-likelihood function for the convolutional decoder is

$$\log p(\mathbf{r}|\hat{\mathbf{c}}) = \sum_{i=1}^{N} \log p(r_i|\hat{c}_i) \tag{4.15}$$

Let the transition probability be defined as

$$p(r_i|\hat{c}_i) = \begin{cases} p & \text{if } r_i \neq \hat{c}_i \\ 1-p & \text{if } r_i = \hat{c}_i \end{cases} \tag{4.16}$$

Suppose also that the received vector $\mathbf{r}$ differs from the candidate code vector $\hat{\mathbf{c}}$ in exactly d positions. The number d defines the *Hamming distance* between vectors r and $\hat{\mathbf{c}}$. Then we may rewrite the log-likelihood function in Eq. (4.15) as

$$\begin{aligned} \log p(\mathbf{r}|\hat{\mathbf{c}}) &= d\log p + (N-d)\log(1-p) \\ &= d\log\left(\frac{p}{1-p}\right) + N\log(1-p) \end{aligned} \tag{4.17}$$

In general, the probability of an error occurring is low enough for us to assume that $p < 1/2$. We also recognize that $N\log(1-p)$ is a constant for all $\hat{\mathbf{c}}$. Accordingly, we may restate the maximum-likelihood decoding rule for the binary symmetric channel as follows:

> *Choose the estimate* $\hat{\mathbf{c}}$ *that miminizes the Hamming distance* d *between the candidate code vector* $\hat{\mathbf{c}}$ *and the received vector* $\mathbf{r}$. (4.18)

That is, for a binary symmetric channel, the maximum-likelihood decoder reduces to a *minimum-distance decoder*. In such a decoder, the received vector $\mathbf{r}$ is compared with each possible candidate code vector $\hat{\mathbf{c}}$, and the particular one closest to $\mathbf{r}$ is chosen as an estimate of the transmitted code vector. The term "closest" is used in the sense of minimum number of differing binary symbols between the code vectors $\hat{\mathbf{c}}$ and $\mathbf{r}$—that is, the Hamming distance between them.

4.9 THE VITERBI ALGORITHM[5]

The equivalence between maximum-likelihood decoding and minimum-distance decoding for a binary symmetric channel implies that we may use a trellis to decode the sequence received over a binary symmetric channel in response to the transmission of a convolutionally encoded sequence characterized by that trellis. For a given convolutional code, the decoding is performed by choosing the particular path in the trellis whose coded sequence differs from the received sequence in the fewest number of places. The search procedure through the trellis of a convolutional code for maximum-likelihood decoding of the received sequence is a dynamic programming problem. The standard approach to solving this problem in coding applications is known as the *Viterbi algorithm*.

The Viterbi algorithm operates by computing a *metric* (i.e., a measure) for discrepancy for every possible path in the trellis. The metric for a particular path is defined as the Hamming distance between the coded sequence represented by that path and the received sequence. Thus, for each node (state) in the trellis of the encoder, the algorithm compares the two paths entering that particular node. The path with the lower metric is retained, and the other path is discarded. This computation is repeated for every level j of the trellis in the range $M \leq j \leq L$, where $M = K - 1$ is the encoder's memory and L is the length of the incoming message sequence. The paths that are retained by the algorithm are called *survivor paths* or *active paths*. For a convolutional code of constraint length K, no more than 2^{K-1} survivor paths and their metrics will be stored. This list of 2^{K-1} paths is always guaranteed to contain the maximum-likelihood choice—that is, a binary sequence closest to the transmitted sequence in Euclidean distance.

A summary of the Viterbi algorithm is presented in Table 4.4; its application is best illustrated by way of an example that follows the table.

TABLE 4.4 Summary of the Viterbi Algorithm.

Initialization

Label the states of the trellis from top to bottom as shown in Fig. T.1. Label the leftmost column of the trellis as time $j = 0$. Initialize the *cumulative path metric* to state $s = 0, 1, ..., S - 1$, at time step 0 to

$$J_0(s) = \begin{cases} 0 & \text{if } s = 0 \\ \infty & \text{if } s \neq 0 \end{cases}$$

If the encoder produces L bits per transition, then we index the received vector $\mathbf{r}$ in groups of L bits; that is, $\mathbf{r} = (r_{1,1},...,r_{1,L},r_{2,1},...,r_{2,L},r_{3,1},...)$. We also label the L bits that are output from the encoder on a transition

TABLE 4.4 Summary of the Viterbi Algorithm.

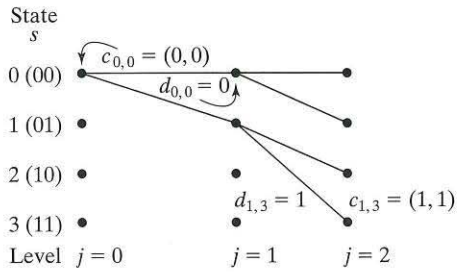

FIGURE T.1 Labeling of trellis states and transitions.

from state q to s as $(c_{q,s,1},c_{q,s,2},...,c_{q,s,L})$ and let $d_{q,s}$ be the corresponding bits at the input to the encoder for this transition, as shown in figure. Note that $d_{q,s}$ and $c_{q,s,1}$ are predetermined by the definition of the code. Define the survivor path to state s at time 0 to be the empty set, $\varnothing$.

Computation step j + 1

Let $j = 0, 1, 2,...$, and suppose that at the previous step j we have done only two things:

- *Identify all survivor paths.* The survivor path to state s has the smallest cumulative path metric to state s; that is,

$$J_{j+1}(s) = \min_q \{J_j(q) + H_{j+1}(q, s)\}$$

The smallest cumulative path metric is determined from the cumulative path metrics at the previous step and the branch metric for the current step. The branch metric at step $j + 1$ from state q to state s is given by

$$H_{j+1}(q, s) = \text{Hamming distance}\{(r_{j+1,1}, \cdots, r_{j+1,L}), (c_{q,s,1}, \cdots q, s, L)\}$$

which is the Hamming distance between the received vector in symbol period $j + 1$ and the symbols on the trellis branch from state q to s. If there is no trellis branch from state q to s, then $H_{j+1}(q, s) = \infty$.

- *For each state, the survivor path and its metric are stored.* If $q_0(s)$ is the optimal solution to the minimization step, then the surviving path is given by the ordered set

$$\hat{m}_{j+1}(s) = \left(\hat{m}_j(q_0(s)), d_{q_0(s),s}\right)$$

and its metric is $J_{j+1}(s)$.

Final Step

Continue the computation until the algorithm completes its forward search through the trellis and therefore reaches the termination node (i.e., an all-zero state), at which time it makes a decision on the maximum-likelihood path. Then, as with a block decoder, the sequence of symbols associated with that path is released to the destination as the decoded version of the received sequence. In this sense, it is therefore more correct to refer to the Viterbi algorithm as a *maximum-likelihood sequence estimator*.

EXAMPLE 4.2 Decoding of a Received Binary Sequence with Two Errors

Suppose that the encoder of Fig. 4.6 generates an all-zero sequence that is sent over a binary symmetric channel, and that the received sequence is (0100010000 ...). There are two errors in the received sequence due to noise in the channel: one in the second bit and the other in the sixth bit. We wish to show that this double-error pattern is correctable through the application of the Viterbi decoding algorithm.

In Fig. 4.9, we show the results of applying the algorithm for level (i.e., time step) $j = 1,2,3,4,5$. We see that for $j = 2$ there are (for the first time) four paths, one for each of the four states of the encoder. The figure also includes the metric of each path for each level in the computation.

In the left side of Fig. 4.9, for $j = 3$ we show the paths entering each of the states, together with their individual metrics. In the right side of the figure, we show the four survivors that result from application of the algorithm for level $j = 3,4,5$.

Examining the four survivors in Fig. 4.9 for $j = 5$, we see that the all-zero path has the smallest metric and will remain the path of smallest metric from this point forward. This clearly shows that the all-zero sequence is the maximum likelihood choice of the Viterbi decoding algorithm, which agrees exactly with the transmitted sequence. ■

Problem 4.3 Suppose that in Example 4.2 the received sequence is (1100010000 ...), which contains three errors compared to the transmitted all-zero sequence. This time, show that the Viterbi algorithm fails to decode the transmitted sequence correctly. That is, a triple-error pattern is uncorrectable by the Viterbi algorithm when applied to a convolutional code of constraint length $K = 3$. (The exception to this rule is a triple-error pattern spread over a time span longer than one constraint length, in which case it is likely to be correctable.) ■

4.9.1 Modifications of the Viterbi Algorithm

In reality, the Viterbi algorithm is a maximum likelihood sequence estimator, which means that the algorithm waits until the entire sequence at the channel output has been received before making a decision. However, when the received sequence is very long (near infinite), the storage requirement of the Viterbi algorithm becomes too high, and some compromises must be made. The approach usually taken is to "truncate" the path memory of the decoder as described here. A *decoding window* of acceptable length l is specified, and the Viterbi algorithm operates on a corresponding frame of the received sequence, always stopping after l steps. A decision is then made on the "best" path and the symbol associated with the first branch on that path is released to the user. Next, the decoding window is moved forward one time interval, and a decision on the next code frame is made, and so on. The decoding decisions made in this way are no longer truly maximum likelihood, but they can be made almost as good provided that the decoding window is long enough. Experience and analysis have shown that satisfactory results are obtained if the decoding window length l is on the order of 5 times the constraint length K of the convolutional code or more.

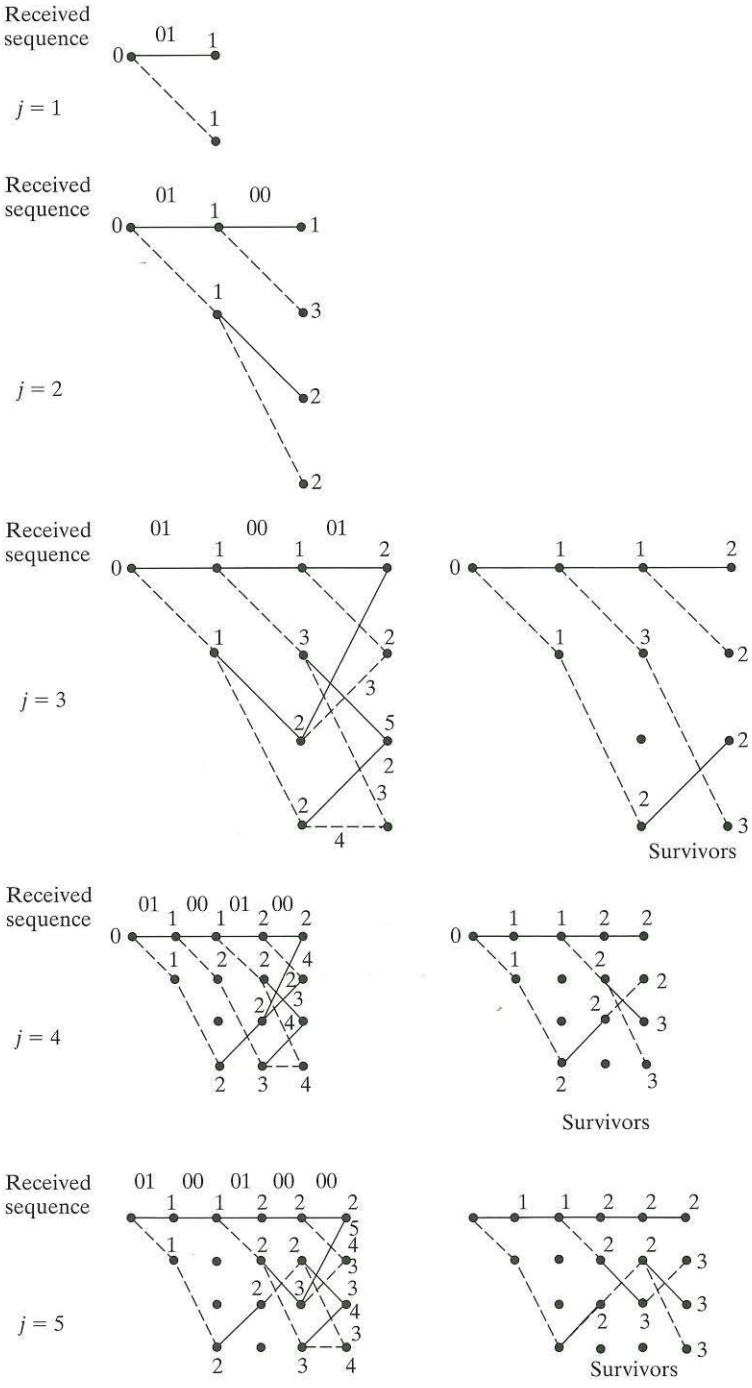

FIGURE 4.9 Steps in the Viterbi algorithm for Example 4.2.

4.10 INTERLEAVING[6]

Previous sections of this chapter have shown us how a digital wireless communication system can be separated by function into source-coding and channel-coding applications on the transmitting side and the corresponding inverse functions on the receiving side. We have also learned how analog signals can be captured in a digital format. The motivation behind these techniques is to minimize the amount of information that has to be transmitted over the wireless channel. Such minimization has at least two potential benefits in the two primary resources—transmit power and channel bandwidth—available to wireless communications:

1. *Reducing the amount of data that must be transmitted, which usually means that less power has to be transmitted.* Power consumption is always a serious concern for mobile terminals, which are typically battery operated.
2. *Reducing the spectral (or radio frequency) resources that are required for satisfactory performance.* This reduction enables us to increase the number of users who can share the same, but limited, channel bandwidth.

Moreover, insofar as channel coding is concerned, forward error-correction (FEC) coding provides a powerful technique for transmitting information-bearing data reliably from a source to a sink across the wireless channel.

However, to obtain the maximum benefit from FEC coding in many wireless channels, we require an additional technique known as *interleaving*. The need for this new technique is justified on the grounds that, in light of the material presented in Chapter 2, we know that wireless channels have *memory* due to multipath fading—the arrival of signals at the receiver via multiple propagation paths of different lengths. Of particular concern is *fast* fading, which arises out of reflections from objects in the local vicinity of the transmitter, the receiver, or both. The term *fast* refers to the speed of fluctuations in the received signal due to these reflections, relative to the speeds of other propagation phenomena. Compared with transmit data rates, even fast fading can be relatively slow. That is, fast fading can be approximately constant over a number of transmission symbols, depending upon the data transmission speed and the terminal velocity. Consequently, fast fading may be viewed as a time-correlated form of channel impairment, the presence of which results in statistical dependence among continguous (sets of) symbol transmissions. That is, instead of being isolated events, transmission errors due to fast fading tend to occur in *bursts*.

Now, most FEC channel codes are designed to deal with a limited number of bit errors, assumed to be randomly distributed and statistically independent from one bit to the next. To be specific, in Section 4.9 on convolutional decoding, we indicated that the Viterbi algorithm, as powerful as it is, will fail if there are $d_{\text{free}}/2$ closely spaced bit errors in the received signal, where d_{free} is the free distance of the convolutional code. Accordingly, in the design of a *reliable* wireless communication system, we are confronted with *two conflicting phenomena: a wireless channel that produces bursts of correlated bit errors and a convolutional decoder that cannot handle error bursts.* Interleaving is an indispensable technique for resolving this conflict. First and foremost, however, it is important to note that, for interleaving, we do *not* need an exact statistical characterization of the

wireless channel. Rather, we require only a knowledge of the *coherence time* for fast fading. Using the Clarke model for fast fading, considered in Example 2.11 of Chapter 2, we found that the *coherence time* for fast fading is approximately

$$T_{\text{coherence}} \approx \frac{0.3}{2f_D} \tag{4.19}$$

where f_D is the maximum Doppler shift. The Doppler shift depends upon the relative velocity between the transmitter and the receiver. Consequently, we would expect a "bad" signal period to have approximately the length of the Doppler shift. Equivalently, an error burst would typically have a duration $T_{\text{coherence}}$.

EXAMPLE 4.3 Interleaver Design Considerations

A mobile radio transmits data at 19.2 kbits/s in the 400-MHz band and must operate at vehicle speeds up to 100 km/hour. The radio design includes an off-the-shelf FEC chip with a rate-1/2, constraint-length-7 convolutional code. What is the expected duration of an error burst at top speed? How many closely spaced errors can this codec correct?

The maximum Doppler shift is given by

$$f_D = \frac{v}{c}(f_c) = \frac{100 \text{ km/hr}}{3 \times 10^8 \text{ m/s}}(400 \text{ MHz}) = 37 \text{ Hz}$$

The expected length of an error burst is therefore

$$R_b T_{\text{coherence}} = 19.2 \text{ kbits/s} \times \frac{0.3}{2 \times 37} = 78 \text{ bits}$$

On average, we would expect only half of these bits to be in error. (Why?) From Table 4.3, the free distance of a nonsystematic constraint-length $K = 7$ code is 10, so it can be expected to handle a maximum of only five closely spaced errors. ■

To reconcile the two conflicting phenomena illustrated in Example 4.3, we do two things:

- use an *interleaver* (i.e., a device that performs interleaving), which randomizes the order of encoded bits *after* the channel encoder in the transmitter, and
- use a *deinterleaver* (i.e., a device that performs deinterleaving), which undoes the randomization *before* the data reach the channel decoder in the receiver.

Interleaving has the net effect of breaking up any error bursts that occur during the course of data transmission over the wireless channel and spreading them over the duration of operation of the interleaver. In so doing, the likelihood of a correctable received sequence is significantly improved. The positions of the interleaver in the transmitter and the deinterleaver in the receiver are shown in Fig. 4.1.

Three types of interleaving are commonly used in practice, as discussed next.

4.10.1 Block Interleaving

In basic terms, a *classical block interleaver* acts as a memory buffer, as shown in Fig. 4.10. Data are written into this $N \times L$ rectangular array from the channel encoder in column fashion. Once the array is filled, it is read out in row fashion and its contents

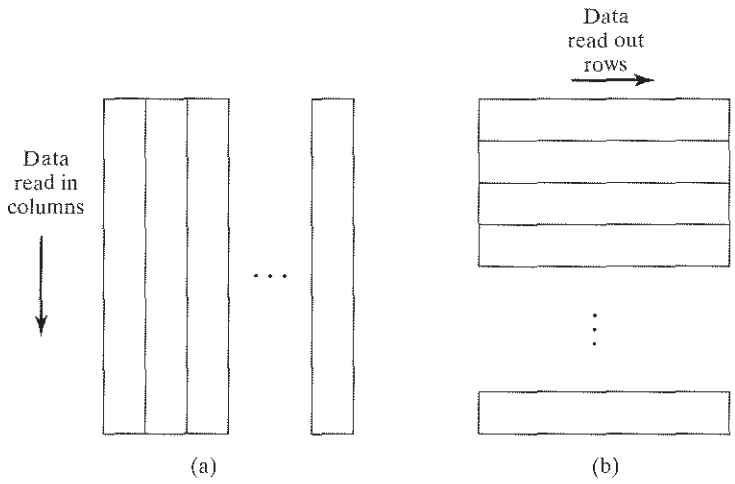

FIGURE 4.10 Block interleaver structure.
(a) Data "read in."
(b) Data "read out."

are sent to the transmitter. At the receiver, the inverse operation is performed: The contents of the array in the receiver are written rowwise with data, and once the array is filled, it is read out columnwise into the Viterbi decoder. Note that the (N,L) interleaver and deinterleaver described herein are both *periodic* with fundamental period $T = NL$.

Suppose the correlation time, or error-burst-length time, corresponds to L received bits. Then, at the receiver, we expect that the effect of an error burst would corrupt the equivalent of one row of the deinterleaver block. However, since the deinterleaver block is read out columnwise, all of these "bad" bits would be separated by $N - 1$ "good"' bits when the burst is read into the Viterbi decoder. If N is greater than the constraint length of the convolutional code being employed, then the Viterbi decoder will correct all of the errors in the error burst.

In practice, due to the frequency of error bursts and the presence of other errors caused by channel noise, the interleaver should ideally be made as large as possible. However, *an interleaver introduces delay* into the transmission of the message signal, in that we must fill the $N \times L$ array before it can be transmitted. This is an issue of particular concern in real-time applications such as voice, because it limits the usable block size of the interleaver and necessitates a compromise solution.

EXAMPLE 4.4 Interleaving Example

Figure 4.11(a) depicts an original sequence of encoded words, with each word consisting of five symbols. Figure 4.11(b) depicts the interleaved version of the encoded sequence, with the symbols shown in reordered positions. An error burst occupying four symbols, caused by channel impairment, is shown alongside Fig. 4.11(b). Note that the manner in which the encoded symbols are reordered by the interleaver is the same from one word to the next.

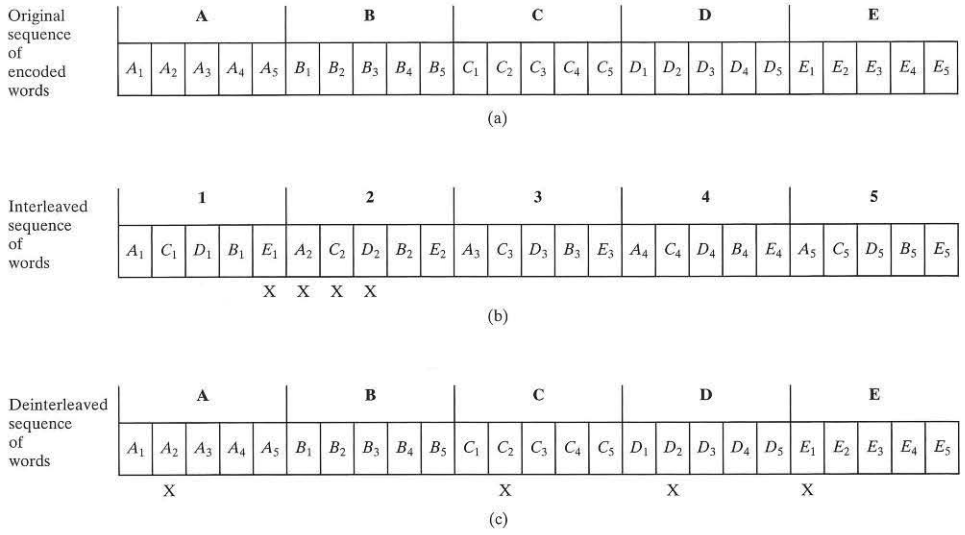

FIGURE 4.11 Interleaving example. (a) Original sequence. (b) Interleaved sequence. (c) Deinterleaved sequence.

On deinterleaving in the receiver, the shuffling of symbols is undone, yielding a sequence that resembles the original sequence of encoded symbols, as shown in Fig. 4.11(c). This figure also includes the new positions of the transmission errors. The important thing to note here is that the error burst is dispersed as a result of deinterleaving.

This example teaches us that (1) the burst of transmission errors is acted upon only by the deinterleaver and (2) insofar as the encoded symbols are concerned, the deinterleaver cancels the shuffling action of the interleaver. ■

Problem 4.4 Consider a mobile terminal moving at 30 km/hr and transmitting at 1.9 GHz. What is the expected coherence time of fading in the channel? The modem employed has a constraint-length-7 convolutional code. What is the minimum block size of the interleaver that you would recommend? If the data rate is 9.6 kbps, what is the delay introduced by this interleaver?

Ans. Coherence time is approximately 9.6 milliseconds. The $N \times L$ interleaver should have $N = 12$ and L corresponding to 9.6 milliseconds. The interleaver size does not depend on the data rate (Why?) but the delay would be 115.2 milliseconds. ■

4.10.2 Convolutional Interleaving

The block diagram of a *convolutional interleaver/deinterleaver* is shown in Fig. 4.12. Defining the period

$$T = LN,$$

we refer to the interleaver as an $(L \times N)$ convolutional interleaver, which has properties similar to those of the $(L \times N)$ block interleaver.

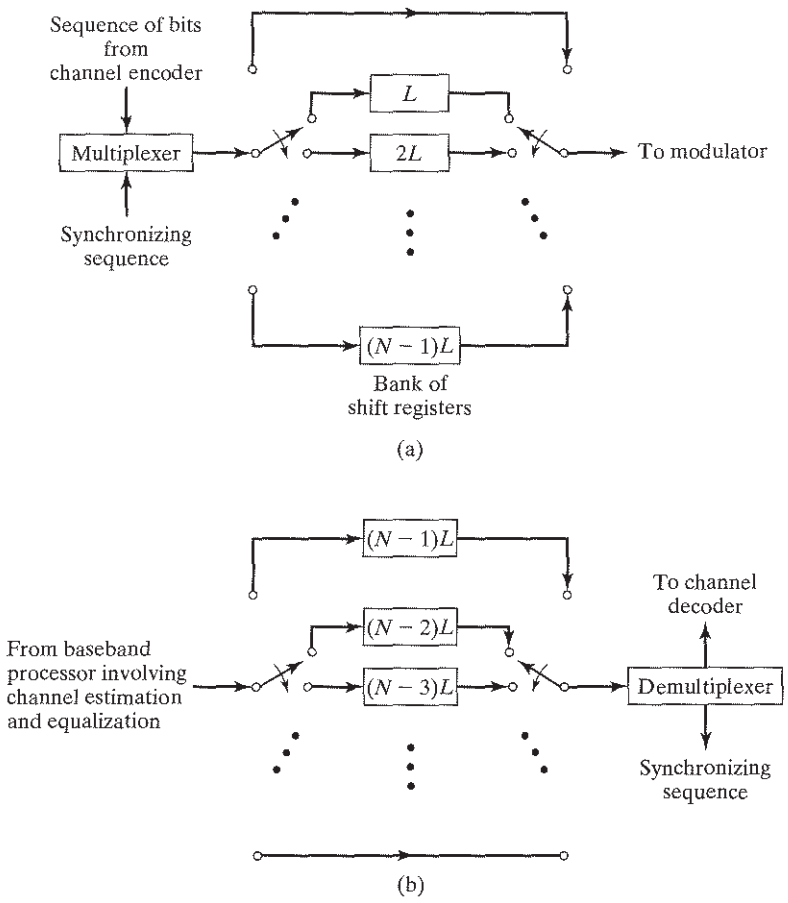

FIGURE 4.12 (a) Convolutional interleaver. (b) Convolutional deinterleaver.

The sequence of encoded bits to be interleaved in the transmitter is arranged in blocks of L bits. For each block, the encoded bits are sequentially shifted into and out of a bank of N registers by means of two synchronized input and output *commutators*. The interleaver, depicted in Fig. 4.12(a), is structured as follows:

1. The zeroth shift register provides no storage; that is, the incoming encoded symbol is transmitted immediately.
2. Each successive shift register provides a storage capacity of L symbols more than the preceding shift register.
3. Each shift register is visited regularly on a periodic basis.

With each new encoded symbol, the commutators switch to a new shift register. The new symbol is shifted into the register, and the oldest symbol stored in that register is shifted out. After finishing with the $(N-1)$th shift register (i.e., the last register), the

commutators return to the zeroth shift register. Thus the switching–shifting procedure is repeated periodically on a regular basis.

The deinterleaver in the receiver also uses N shift registers and a pair of input/output commutators that are synchronized with those in the interleaver. Note, however, that the shift registers are stacked in the *reverse* order of those in the interleaver, as shown in Fig. 4.12(b). The net result is that the deinterleaver performs the inverse operation of that in the transmitter.

An advantage of convolutional over block interleaving is that in convolutional interleaving, the total *end-to-end delay* is $L(N-1)$ symbols and the memory requirement is $L(N-1)/2$ in both the interleaver and deinterleaver, which are one-half of the corresponding values in a block interleaver/deinterleaver for a similar level of interleaving.

The description of the convolutional interleaver/deinterleaver in Fig. 4.12 is presented in terms of shift registers. The actual implementation of the system can also be accomplished with *random access memory* (RAM) units in place of shift registers. This alternative implementation simply requires that access to the memory units be appropriately controlled.

4.10.3 Random Interleaving

In a *random interleaver*, a block of N input bits is written into the interleaver in the order in which the bits are received, but they are read out in a random manner. Typically, the permutation of the input bits is defined by a *uniform distribution*. Let $\pi(i)$ denote the permuted location of the ith input bit, where $i = 1,2\ldots,N$. The set of integers denoted by $\{\pi(i)\}_{i=1}^{N}$, defining the order in which the stored input bits are read out of the interleaver, is generated according to the following two-step algorithm:

1. Choose an integer i_1 from the uniformly distributed set $\mathcal{A} = \{1,2,\ldots,N\}$, with the probability of choosing i_1 being $P(i_1) = 1/N$. The chosen integer i_1 is set to $\pi(1)$.
2. For $k > 1$, choose an integer i_k from the uniformly distributed set $\mathcal{A}_k = \{i \in \mathcal{A}, i \neq i_1, i_2,\ldots, i_{k-1}\}$, with the probability of choosing i_k being $P(i_k) = 1/(N-k+1)$. The chosen integer i_k is set to $\pi(k)$. Note that the size of the set $\mathcal{A}_k$ is progressively reduced for $k > 1$. When $k = N$, we are left with a single integer, i_N, that is set to $\pi(N)$.

To be of practical use in communications, random interleavers are configured to be *pseudorandom*, meaning that, within a block of N input bits, the permutation is random as just described, but the permutation order is exactly the same from one block to the next. Accordingly, pseudorandom interleavers are designed *off-line*. Pseudorandom interleavers are of interest in the construction of turbo codes,[7] which are discussed in Section 4.12.

4.11 NOISE PERFORMANCE OF CONVOLUTIONAL CODES

In Fig. 4.13, the simulated performance of several convolutional codes is compared with uncoded performance in an AWGN channel. The codes all have a rate of 1/2 and constraint lengths (K) of 3, 5, 7, and 9. The corresponding generator polynomials of the

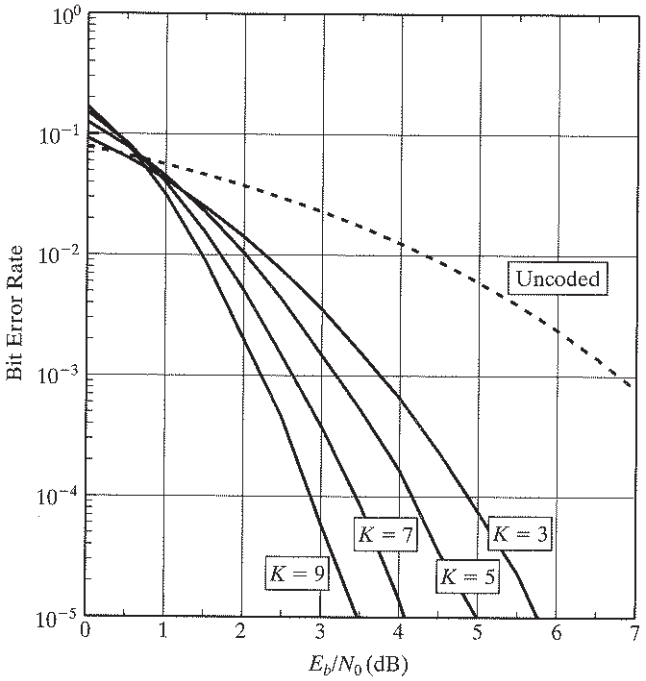

FIGURE 4.13 Comparison of uncoded and coded performance in AWGN channel.

encoders are given in Table 4.5. (The term "octal" used in the table was explained in Section 4.6, and the (5,7) convolution code of constraint length $K = 3$ was discussed in Example 4.1.) The horizontal axis in the figure represents the ratio of energy per information bit, E_b, to one-sided noise spectral density N_0. For the uncoded case, an information bit is the same as a channel bit. For a rate-1/2 code, there are two channel bits for each information bit. Binary PSK modulation with coherent detection is assumed in both the coded and uncoded cases.

In Fig. 4.14, the corresponding results are shown for a Rayleigh-fading channel. The coding also includes the equivalent of infinite interleaving; that is, there is no correlation of the fading process from one channel bit to the next. Here again, binary PSK modulation with coherent detection is assumed in both the coded and uncoded cases. Coherent detection is usually difficult to perform in fading channels, so the curves presented in the figure should be viewed as a limit on noise performance. Note also that the codes used in the simulations are all nonsystemic.

TABLE 4.5 Generators for Convolutional Codes Used in Figs. 4.13 and 4.14.

Constraint Length K	Generator in Octal
3	5,7
5	23, 35
7	133, 171
9	561, 753

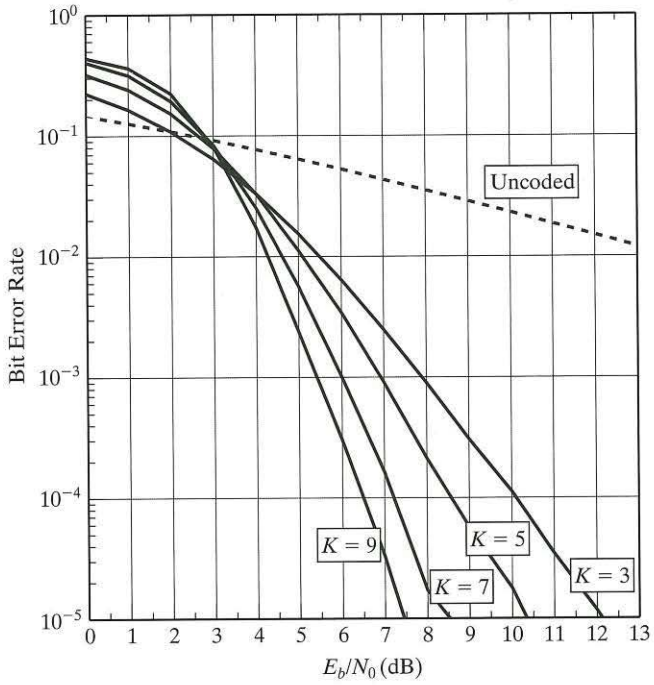

FIGURE 4.14 Comparison of uncoded and coded performance in Rayleigh-fading channel (with infinite interleaving).

Examining Figs. 4.13 and 4.14, we may make the following observations:

1. For low values of E_b/N_0, the uncoded performance is better than the coded performance, irrespective of whether the channel is an ANGN channel or a fading channel. This statement presumes the use of a maximum-likelihood decoder (e.g., Viterbi decoder). The coded performance improves on the uncoded performance only when the E_b/N_0 is large enough for the pairwise error probability of the code to assume exponentially decreasing values. The *pairwise error probability* of a code is defined as the probability that the received code vector $\mathbf{r}$ is closer (in the sense of Euclidean distance) to a candidate code vector $\hat{\mathbf{c}}$ in the code than the true code vector for $\hat{\mathbf{c}} \neq \mathbf{c}$.
2. For a prescribed E_b/N_0, the noise performance (measured in terms of the bit error rate) improves with increasing constraint length K for both AWGN and fading channels, which is intuitively satisfying.
3. For a prescribed constraint length K, the E_b/N_0 must be increased for the fading channel to exhibit a noise performance comparable to that attainable with the corresponding AWGN channel.
4. For constraint length $K = 9$ in the Rayleigh-fading channel, we can realize a bit error rate of 2×10^{-4} by using an $E_b/N_0 = 6$, which is manageable, thanks to the use of forward error-correction coding.

4.12 TURBO CODES[8]

As mentioned in Note 3, convolutional codes, discussed in Section 4.7, were first described by Elias in 1955. In this section, we discuss a new class of block codes known as *turbo codes*, which were discovered by Berrou *et al.* in 1993. Turbo codes are powerful codes by virtue of combining two important ideas, one built into the design of the transmitter and the other into the design of the receiver:

1. *Pair of encoders*, separated by an interleaver
2. *Iterative detection*, involving the use of feedback around a pair of decoders separated by a deinterleaver and an interleaver

The combination of these two ideas in a novel way has made it possible for turbo codes to provide significant improvements in the quality of data transmission over a noisy channel, yet the computational cost is modest compared with traditional encoding–decoding procedures.

4.12.1 Turbo Encoding

Turbo coding, illustrated in Fig. 4.15, uses a parallel FEC encoding scheme. With this scheme, the information is *systematically* encoded by two separate encoders. Often, the two encoders are identical, and to ensure that they do not produce the same output, the information bits are reordered through the use of a pseudorandom interleaver before the second encoding stage. This *pseudorandom interleaver* is often referred to as the *turbo interleaver* to distinguish it from *channel interleavers*, discussed in Section 4.10. The information bits and the parity bits generated by the two encoders are then transmitted over the channel. (In a typical turbo encoding scheme, the parity bits are punctured in a repeating pattern to increase the code rate, but this is not a requirement; *puncturing* refers to the omission of certain parity-check bits in the code so as to increase the data rate.)

Although any valid code is permissible, the component codes that are recommended for turbo encoding are short-constraint-length, systematic, recursive convolutional codes. With a systematic code, the original information bits are part of the

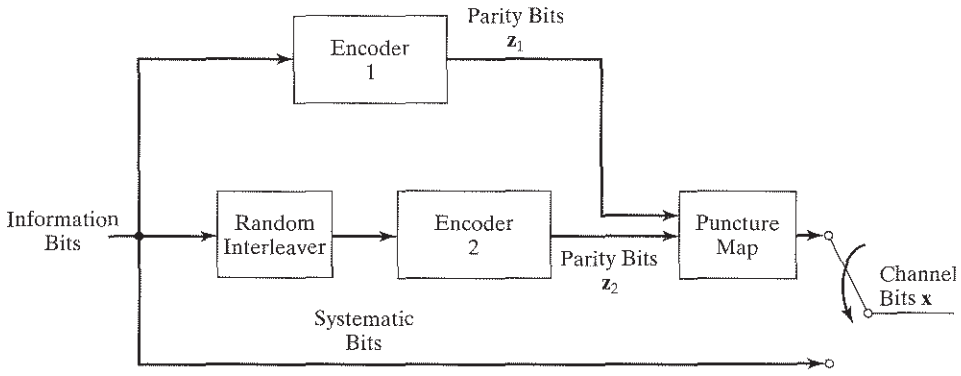

FIGURE 4.15 Illustration of turbo encoding strategy.

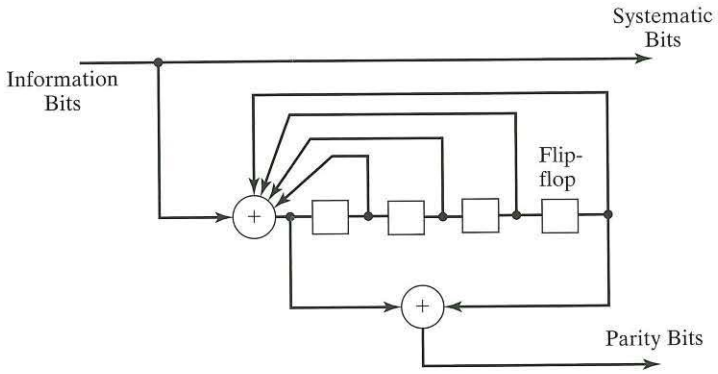

FIGURE 4.16 A systematic recursive convolutional code structure; delay T is equal to the symbol duration.

transmitted sequence; this is what is assumed in Fig. 4.15. *A recursive convolutional code means that the code has a feedback structure*, as opposed to the feedforward structure that characterizes the convolutional codes discussed in Section 4.7. The original turbo component code described by Berrou et al. is illustrated in Fig. 4.16. This is a simple 16-state code. The feedback nature of these component codes means that a single bit error corresponds to an infinite sequence of channel errors. To see this, consider an information sequence consisting of the binary digit 1, followed by an infinite number of zeros, as the input to the encoder shown in Fig. 4.16. If the encoder starts in the all-zero state, then it is straightforward to show that it never returns to that state with that kind of input.

Although the component codes are convolutional, turbo codes are inherently block codes, with the block size being determined by the size of the turbo interleaver. The block nature of the code also raises a practical problem of how to terminate the trellises of both encoders correctly. Often, the trellis of the second encoder is left unterminated.

4.12.2 Turbo Decoding

In addition to turbo codes having a novel parallel encoding structure, the decoding strategy applied to turbo codes is highly innovative. The scheme draws its name from the analogy of the structure of the decoding algorithm to the turbo engine principle. A block diagram of the decoder structure is shown in Fig. 4.17.

To illustrate the processing of the turbo decoder, let $(\mathbf{x}, \mathbf{z}_1, \mathbf{z}_2)$ be the vector of outputs from the turbo encoder of Fig. 4.15, where, for a particular turbo block of data, we have

- $\mathbf{x}$ as the vector of information (systematic) bits,
- $\mathbf{z}_1$ as the vector of parity bits from the first encoder, and
- $\mathbf{z}_2$ as the vector of parity bits from the second encoder.

Corresponding to these inputs applied to the channel, let $(\mathbf{u}, \boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2)$ be the vector of noisy outputs from the *demodulator after matched filtering*.

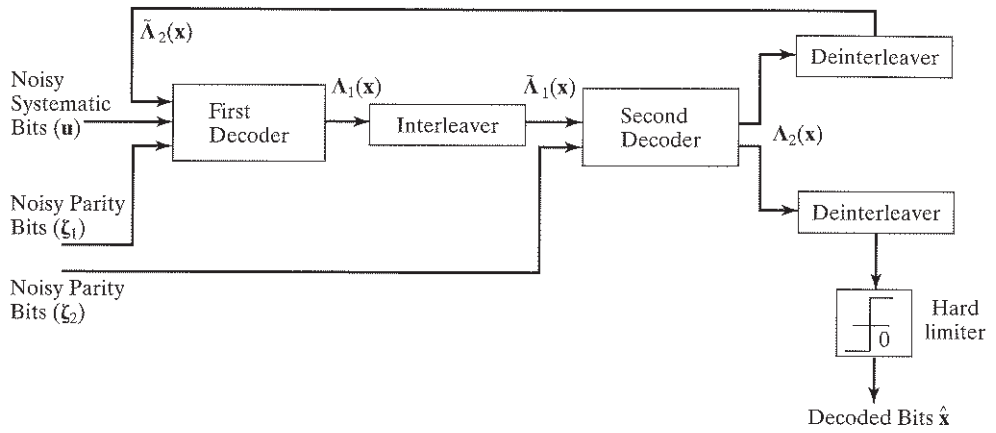

FIGURE 4.17 Block diagram of Turbo decoder.

In Fig. 4.17, the inputs to the first decoding stage are the channel samples $(\mathbf{u}, \boldsymbol{\zeta}_1)$ corresponding, respectively, to the systematic (information) bits $(\mathbf{x})$, the channel samples corresponding to the parity bits of the first encoder $(\mathbf{z}_1)$, and the *extrinsic information* about the systematic bits that were determined from previous decoder stages $(\tilde{\mathbf{\Lambda}}_2(\mathbf{x}))$. We will define extrinsic information later, but suffice it to say that, on the first decoding iteration, the extrinsic information is zero. (If puncturing was applied at the transmitter, then the parity bits must be stuffed with zeros in the appropriate positions.)

Crucial to the performance of turbo decoding is the use of a *soft-input, soft-output (SISO) decoding algorithm.* That is, not only does the algorithm accept a soft input—a real value whose magnitude indicates the reliability of the input—but it also produces a soft output, generally the probability that a particular bit is a 0 or a 1. In the next subsection, we will describe a *maximum a posteriori probability decoding algorithm* (hereafter referred to as the *MAP algorithm*) that meets this requirement.

The first decoding stage uses the MAP algorithm to produce a refined (soft) estimate, $\text{Prob}(x(j) \,|\, \mathbf{u}, \boldsymbol{\zeta}_1, \tilde{\mathbf{\Lambda}}_2(\mathbf{x}))$, of the systematic bits. However, for the implementation, it is more convenient to express this soft estimate as the equivalent log-likelihood ratio

$$\Lambda_1(x(j)) = \log\left(\frac{\text{Prob}(x(j)=1 \,|\, \mathbf{u}, \boldsymbol{\zeta}_1, \tilde{\mathbf{\Lambda}}_2(\mathbf{x}))}{\text{Prob}(x(j)=0 \,|\, \mathbf{u}, \boldsymbol{\zeta}_1, \tilde{\mathbf{\Lambda}}_2(\mathbf{x}))}\right) \tag{4.20}$$

In practice, calculation of this log-likelihood ratio is much simpler than Eq. (4.20) would suggest. Before entering the second stage of processing on the first iteration, the vector of refined estimates, $\mathbf{\Lambda}_1$, is reordered to compensate for the turbo interleaving at the transmitter, whereafter it is combined with the second set of parity samples from the channel as the input to the second decoding stage.

The second decoding stage uses the MAP algorithm and the second set of parity bits to produce a further refined estimate, $\text{Prob}(x(j) \,|\, \tilde{\mathbf{\Lambda}}_1((\mathbf{x}), \boldsymbol{\zeta}_2))$, of the systematic bits. This second refined estimate is also expressed as a log-likelihood ratio, namely, $\mathbf{\Lambda}_2(x(j))$. The estimate so produced can be hard-detected, if so desired, to provide bit

estimates at this point. Alternatively, the output of the second stage can be used to provide extrinsic information to the first stage. In either case, the estimates must be reordered to compensate for the turbo interleaving.

As can be seen from Fig. 4.17, the first decoding stage, the interleaver, the second decoding stage, and the deinterleaver constitute a single-loop *feedback system*, thereby making it possible to iterate the decoding process in the receiver as many times as is deemed necessary for a satisfactory performance. Indeed, this iterative process constitutes the *turbo coding principle* at the receiver. Note, however, that during each set of iterations, the noisy input vector $\mathbf{u}$ is unaltered.

The decoding scheme of Fig. 4.17 relies on the implicit assumption that the bit probabilities remain independent from one iteration to another. To increase the independence of the inputs from one processing stage to the next, the turbo algorithm makes use of the concepts of intrinsic and extrinsic information. *Intrinsic information refers to the information inherent in a sample prior to a decoding operation. On the other hand, extrinsic information refers to the incremental information obtained through decoding.* To maintain as much independence as is practically possible from one iteration to the next, *only extrinsic information is fed from one stage to the next.*

An important property of the MAP algorithm is that it includes both intrinsic and extrinsic information in a product relationship (including a scaling factor). However, from a computational perspective, it is more convenient to express this relationship in terms of log-likelihood ratios, because then the product becomes a sum and the scaling term assumes a nonsignificant role. In particular, the extrinsic information at the output of the second stage is

$$\tilde{\Lambda}_2(\mathbf{x}) = \Lambda_2(\mathbf{x}) - \tilde{\Lambda}_1(\mathbf{x}) \tag{4.21}$$

That is, the extrinsic information (expressed in logarithmic form) is the difference between the input and output log-likelihood ratios for the systematic bits. (On the first pass, $\tilde{\Lambda}_1(\mathbf{x}) = \Lambda_1(\mathbf{x})$.) Similarly, at the output of the first stage, the extrinsic information supplied to the second stage by the first stage is given by

$$\tilde{\Lambda}_1(\mathbf{x}) = \Lambda_1(\mathbf{x}) - \tilde{\Lambda}_2(\mathbf{x}) \tag{4.22}$$

The basic MAP decoding algorithm is not changed in the turbo decoder. Details of the actual computation of Λ_1 and Λ_2 are omitted in Fig. 4.17 to simplify the exposition. The differences occur only in the processing, which is performed on the inputs to, and outputs from, the MAP algorithm.

On the last iteration of the decoding process, a hard decision is applied to the output of the second decoder (after deinterleaving) to produce an estimate of the jth information bits:

$$\hat{x}(j) = \text{sign}(\Lambda_2(x(j))) \tag{4.23}$$

4.12.3 Noise Performance

The impressive aspect of turbo codes is their noise performance. In Fig. 4.18, an example of that performance is presented. The results correspond to the encoding strategy described in Section 4.12.1, for an interleaver size of 64 kilobits. The performance is

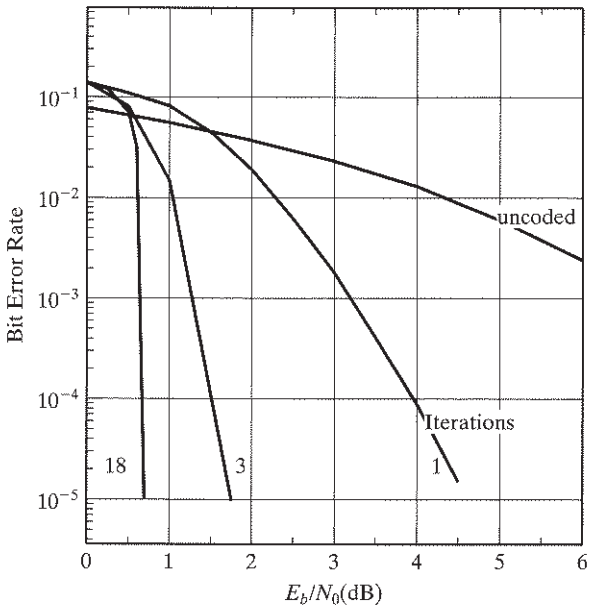

FIGURE 4.18 Performance of turbo codes as a function of number of iterations. (Reproduced from Berrou *et al.*, 1993, with permission of the IEEE.)

quantified as a function of the number of iterations. It is clear from the figure that the performance with a small number of iterations of the decoding process is not particularly impressive, but as the number of iterations is increased, the performance continually improves. With 18 iterations, the code achieves a bit error rate (BER) of 10^{-5} at an E_b/N_0 of only 0.7 dB. This is only about 0.7 dB away from the Shannon limit for this code-particular rate.

The performance of turbo codes at low E_b/N_0 is a function of the interleaver size. With smaller interleavers, the performance tends to degrade. However, even with an interleaver size of 300 bits, a BER of 10^{-4} can be obtained at an E_b/N_0 of 2.3 dB, which is indeed remarkable.

4.12.4 Maximum a Posteriori Probability Decoding

The *maximum a posteriori probability (MAP) decoder* considered herein makes certain assumptions about the transmission system. In particular, we assume $\mathbf{x} = \mathbf{x}_{[1,J]}$ is the information (data) to be transmitted and $\hat{\mathbf{x}}$ is the vector of corresponding data estimates. The transmitted code word is represented by $\mathbf{c} = \mathbf{c}_{[1,N]}$.

The N transmitted bits will be grouped together into T symbols, $\mathbf{v}_{[1,T]}$. This grouping will arise naturally from the structure of the code. For example, with a rate-1/2 code, T is $N/2$. The noisy received bits will be denoted by $\mathbf{r} = \mathbf{r}_{[1,N]}$ and the corresponding received symbols by $\mathbf{w}_{[1,T]}$. The detection (decoding) algorithm provides an estimate $\hat{\mathbf{x}}$ of the data, based on the sequence $\mathbf{r}$ (or $\mathbf{w}$) and prior knowledge of the

FEC code. The channel is assumed to be an AWGN channel. The channel encoder is assumed to have a trellis structure similar to that illustrated in Fig. 4.7 for four states. It is assumed that *the trellis starts and ends in the zero state* and that the appropriate null bits are added to the information stream to ensure proper termination of the trellis. Since the trellis is terminated, we are effectively assuming only block codes, even if a convolutional code is used.

The number of trellis transitions T, corresponding to the N code bits, is a function of the encoding strategy. Its relationship to the index of the channel bits or the information bits depends on the code rate and how the trellis is truncated. At time j, each channel symbol $v(j)$ corresponds to a state transition from $s(j-1)$ to $s(j)$. For example, with a rate-2/3 convolutional code, each channel symbol will have three bits and each pair of information bits will correspond to a state transition until the decoder reaches the termination stage. In the termination stage, information bits that are zeros are appended to the input, and the corresponding channel symbols are generated until the trellis returns to the all-zero state.

For the turbo decoder application, the objective of the MAP algorithm is to determine the conditional probability Prob $(x(j)|\,\mathbf{r})$ where $\mathbf{r} = (\mathbf{u}, \boldsymbol{\zeta}_1, \tilde{\boldsymbol{\Lambda}}_2)$ in one decoding instance and $\mathbf{r} = (\tilde{\boldsymbol{\Lambda}}_1, \boldsymbol{\zeta}_2)$ in the second decoding instance.

The *MAP algorithm includes a backward and forward recursion*, unlike the Viterbi algorithm, which contains only a forward recursion. The fundamental assumption of the MAP algorithm is that the channel encoding can be represented as a trellis, as illustrated in Fig. 4.7,where the next state depends only on the current state and the new input bit. The states of the trellis are indexed by the integer $s = 0,1,...,S-1$. The state of the trellis at time j is denoted by $s(j)$, and the output symbol due to the transition from $s(j-1)$ to $s(j)$ is denoted by $v(j)$. A state sequence from time j to i' is denoted by $\mathbf{s}_{[j,\,i']}$, and the corresponding output sequence is $\mathbf{v}_{[j,\,i']}$.

Although the ultimate goal of the MAP algorithm is to estimate the probability that a given symbol or information bit was transmitted, it is simpler to first derive the *a posteriori* probabilities of the states and transitions of the trellis, based on the observation $\mathbf{r}$. From these results, most of the probabilities of interest can be obtained by performing summations over selected subsets of states or transitions. To proceed, we denote the S-vector of state probabilities at time j based on the set of observations by

$$\boldsymbol{\lambda}(j) = \text{Prob}(\mathbf{s}(j)|\mathbf{r}) \qquad \boldsymbol{\lambda}(j) \in R^S \tag{4.24}$$

where the jth element is

$$\lambda_{s(j)} = \text{Prob}(s(j) = s|\mathbf{r}) \tag{4.25}$$

Then, for a rate $(1/n)$ convolutional code without feedback, assuming that the transmitted information bit is the least significant bit (LSB) of the state, the probability that a 1 was the information bit is given by

$$\text{Prob}(x(j) = 1|\mathbf{r}) = \sum_{s:\text{LSB}(s)=1} \lambda_s(j) \tag{4.26}$$

where LSB(s) is the least significant bit of the state s. For a recursive code, the summation in Eq. (4.26) is over the set of transitions that correspond to a 1 at the input. (Transition probabilities will be explained shortly.)

Next, we define the *forward estimator* of state probabilities as the S-vector

$$\boldsymbol{\alpha}(j) = \text{Prob}[\mathbf{s}(j)|\mathbf{r}_{[1,j]}] \qquad \boldsymbol{\alpha}(j) \in R^M \tag{4.27}$$

Similarly, we define the *backward estimator* of state probabilities as

$$\boldsymbol{\beta}(j) = \text{Prob}[\mathbf{s}(j)|\mathbf{v}_{[j,1]}] \qquad \beta(j) \in R^M \tag{4.28}$$

Equations (4.27) and (4.28) are the estimates of the state probabilities at time j based on past and future observations, respectively. The important result related to these quantities is their relationship to $\lambda(j)$. First, however, let us define the vector product $\mathbf{c} = \mathbf{a} \cdot \mathbf{b}$ as $c(s) = a(s)b(s)$ for all s and define the L^1 norm for probability vectors as

$$\|\boldsymbol{a}\| = \sum_s a(s) \tag{4.29}$$

Then we may write

$$\boldsymbol{\lambda}(j) = \frac{\boldsymbol{\alpha}(j) \cdot \boldsymbol{\beta}(j)}{\|\boldsymbol{\alpha}(j) \cdot \boldsymbol{\beta}(j)\|_1} \tag{4.30}$$

That is, the state probabilities at time j are completely defined by the forward and backward estimators at that time.

A proof of the *separation theorem* embodied in Eq. (4.30) is presented in Appendix F. One aspect of this theorem is intuitively obvious: By definition, for a *Markov process*, the state distribution at time j, given the past, is independent of the state distribution at time j, given the future. However, the theorem says more: It says that there is a simple way of combining the forward and backward estimates to obtain a complete estimate; specifically, that estimate is just the normalized product of the state distributions based on the past and the future.

Thus, the objective is to find simple algorithms for obtaining the forward and backward estimates, which would be a solution to the MAP detection problem. If we represent the state transition probability at time j by

$$\gamma_{s',s}(j) = \text{Prob}(s(j) = s, \boldsymbol{w}(j)|s(j-1) = s') \tag{4.31}$$

and denote the matrix of these probabilities as

$$\boldsymbol{\Gamma}(j) = [\gamma_{s',s}(j)] \qquad \boldsymbol{\Gamma}(j) \in R^{S \times S}, \tag{4.32}$$

then we have the following recursion equations for calculating the forward and backward state estimates:

$$\boldsymbol{\alpha}^T(j) = \frac{\boldsymbol{\alpha}(j-1)^T \boldsymbol{\Gamma}(j)}{\|\boldsymbol{\alpha}(j-1)^T \boldsymbol{\Gamma}(j)\|_1} \tag{4.33}$$

and

$$\boldsymbol{\beta}(j) = \frac{\boldsymbol{\Gamma}(j+1)\boldsymbol{\beta}(j+1)}{\|\boldsymbol{\Gamma}(j+1)\boldsymbol{\beta}(j+1)\|_1} \tag{4.34}$$

Together, these two equations define the algorithm for MAP decoding. In particular, the steps are as follows:

1. $\boldsymbol{\alpha}(0)$ and $\boldsymbol{\beta}(T)$ are initialized according to the trellis structure. For a trellis beginning and ending in the all-zero state, we have $\boldsymbol{\alpha}_0(0) = 1$ and $\boldsymbol{\alpha}_s(0) = 0$ for all $s \neq 0$, and similarly for $\boldsymbol{\beta}(T)$.
2. When $r(j)$ is received, the decoder computes $\boldsymbol{\Gamma}(j)$ and then $\boldsymbol{\alpha}(j)$, using Eq. (4.33). The computed values of $\boldsymbol{\alpha}(j)$ are stored for all j and s. Note that $\mathbf{r}(j)$ is a vector of length L, defined by
$$\mathbf{r}(j) = [r_{j(1)}, r_{j(2)}, \cdots r_{j(L)}]^T$$
where L is itself defined by the number of bits produced per transition by the encoder.
3. After the complete sequence $\mathbf{r}_{[1,T]}$ has been received, the decoder recursively computes $\boldsymbol{\beta}(j)$, using Eq. (4.34). Then, when the $\boldsymbol{\beta}(j)$ have been computed, they are multiplied by the appropriate $\boldsymbol{\alpha}(j)$ to obtain $\boldsymbol{\lambda}(j)$, using Eq. (4.30).

The MAP algorithm, also referred to as the *BCJR algorithm* in recognition of its originators,[9] involves a double recursion—one in the forward direction and the other in the reverse. Consequently, it has at least twice the complexity of the Viterbi algorithm in its most general form. However, it produces soft outputs, a feature that is the key to its usefulness in the turbo decoding algorithm.

4.13 COMPARISON OF CHANNEL-CODING STRATEGIES FOR WIRELESS COMMUNICATIONS

Channel-coding strategies, exemplified by convolutional codes and turbo codes, have established themselves as an essential design tool for reliable communications, each in its own way. Convolutional codes have a long history, dating back to the classic paper by Elias in the 1950s (see Note 3). In contrast, turbo codes are of recent origin, having been discovered in the early 1990s as a result of pragmatic experimentation by Berrou and Glavieux (see Note 8).

With classical convolutional codes covered in Section 4.9 and turbo codes in Section 4.12, it is apropos here that we discuss the relative merits of these two coding strategies for the correction of transmission errors. However, before proceeding with this discussion, we note that classical block codes such as Hamming codes, Bose–Chandhuri–Hocquenghem (BCH) codes, and Reed–Solomon codes, powerful as they are, have not been emphasized in the presentation largely because of their limited application to modern wireless channels. There are two reasons for not covering these classical block codes in this chapter:

- Block codes can be designed to handle error bursts that may occur on a wireless channel, but their ability to handle large numbers of distributed errors is rather limited.

- Block codes are typically algebraic in their formulation, using hard decisions that tend to destroy information; consequently, they do not obtain the performance advantage that is available through the use of soft decisions.

Nevertheless, there are some wireless channels to which block codes are applied. In high-SNR channels, for example, where the errors are few, block codes are often used to correct errors that do occur. In other wireless communication systems, block codes may be used as part of a concatenated coding scheme. In a two-level form of concatenation used in such systems, the block code is used as the outer code, and the inner code is typically a convolutional code. With such a configuration, the (inner) convolutional decoder in the receiver corrects the majority of the errors produced by the wireless channel. Then the (outer) block decoder corrects the few errors that remain at the output of the convolutional decoder.

To proceed with the task at hand, in what follows we present a comparative evaluation of classical convolutional codes and turbo codes in terms of encoding and decoding considerations, as well as other matters that pertain to signal transmission over wireless channels.

4.13.1 Encoding

A convolutional encoder can assume one of two forms:

- *Nonrecursive nonsystematic*, in which form the convolutional encoder distinguishes itself by, first, the use of feedforward paths only and, second, the information bits losing their distinct identity as a result of the convolution process.
- *Recursive systematic*, in which form, first, the convolutional encoder uses feedforward as well as feedback paths and, second, the k-tuple of information bits appears as a subset of the n-tuple of output bits.

On the one hand, for historical reasons, nonrecursive nonsystematic schemes have been advocated for classical convolutional encoders. On the other hand, turbo codes use recursive systematic convolutional (RSC) encoders, as discussed in Section 4.12. Comparing turbo codes with convolutional codes, we find some strengths, weaknesses, and similarities between the two codes:

- By virtue of the fact that the turbo code uses a pseudorandom interleaver to separate its own two convolutional encoders, the turbo code is closer to the random code originally advocated by Shannon in his formulation of communication theory than it is to the classical convolutional code. However, unlike Shannon's random codes, turbo codes are decodable, hence their practical importance.
- Experimentation shows that turbo codes work better than classical convolutional codes when the code rates are high or the signal-to-noise ratios are low.
- Both types of codes require the use of flush bits to return them to the state 0 at the end of the incoming information bits. With the parallel encoding structure of turbo codes, it is not straightforward to flush the second encoder, so flushing is often not done.
- Unlike convolutional codes, turbo codes have an error floor. That is, the BER drops very quickly at the beginning, but eventually levels off and decreases at a

much slower rate. The leveling-off point is often in the BER range of 10^{-5} to 10^{-8}, depending on factors such as how flushing is done, how long the block size is, and which turbo interleaver is being used. Achieving very low error rates (less than 10^{-10}) is difficult with a turbo code; but then, this kind of error performance is rarely required for wireless communications, except perhaps in computer communications, which demand a high degree of reliability over a wireless channel.

- Both types of codes perform better with soft decisions. Turbo codes rely on soft inputs to work. Convolutional codes can work with either soft-decision or hard-decision inputs, but the penalty for using the latter in an AWGN channel is approximately 2 dBs.

4.13.2 Decoding

The traditional approach to decoding a convolutional code is to use the Viterbi algorithm, which operates as a maximum-likelihood state estimator. The complexity of the Viterbi algorithm is directly proportional to the number of states, which in turn depends exponentially on v, denoting the number of memory units in the shift register of the code. Since the minimum free distance (i.e., the error-correction capability) of the convolutional code increases with v, there is a trade-off between performance and decoder complexity. Current wireless systems often use $v = 6$ or 8 (constraint length $K = 7$ or 9) convolutional codes, corresponding to 64-state and 256-state decoders, respectively.

By contrast, turbo decoding relies on the exchange of extrinsic information between two soft-input, soft-output (SISO) decoding stages on an iterative basis. The decoder may use a maximum a posterior probability (MAP) algorithm or its approximation. The advantage of turbo codes is that they can achieve large coding gains with very simple component codes typically, 8-state or 16-state codes. The MAP decoder complexity is proportional to the number of states and is approximately twice the complexity of the Viterbi algorithm for the same number of states; the multiplying factor of two is due to the fact that the MAP algorithm, or its approximation, operates in the forward as well as backward direction. The overall complexity of the turbo decoder is equal to the computational complexity of one complete decoding iteration, multiplied by the number of iterations. In practice, between four and eight iterations usually are employed, although optimum performance often would require twice that number of iterations or more.

With the decoding process of a turbo code being iterative in nature, it would be highly desirable to have at our disposal a tool for describing the convergence analysis of the code and its design. The *extrinsic information transfer (EXIT) chart* provides such a tool in graphical form. The development of the EXIT chart assumes that the size of the turbo interleaver is infinitely large. In particular, the chart describes the exchange of extrinsic information between the first stage and second stage in the turbo decoder, and vice versa. For each constituent decoding stage, the EXIT chart is defined as *the function that maps the prior information to the extrinsic information applied to the decoder in question*, with the information capacity of its communication channel treated as a parameter.[10]

4.13.3 AWGN Channel

For an additive white Gaussian noise (AWGN) channel, which, for example, is closely approximated by a line-of-sight satellite communication link, turbo codes outperform convolutional codes by a significant margin in their ability to achieve near-optimum performance. Optimality is defined in terms of theoretical limits imposed by Shannon's information capacity theorem. The theoretical limit for a block of information bits that is infinitely long is measured in terms of E_b/N_0 required for a prescribed code rate and minimum bit error rate (BER). When information blocks of a finite size are considered, the performance is measured in terms of the increase in E_b/N_0 (measured in dBs with respect to the theoretical limit) required for prescribed values of the code rate, block size, and frame error rate (FER). The FER is the number of frames that have at least one bit error, divided by the total number of frames used in the calculation.

For example, for a code rate of 2/3 and minimum bit error rate of 10^{-4}, the theoretical limit (assuming an information block of infinite size) is about –0.8 dB. For the same code rate, but a block size of 500 information bits, using a turbo code, we need to increase E_b/N_0 by about 1.3 dB (i.e., a net value for E_b/N_0, equal to $1.3 - 0.8 = 0.5$ dB) in order to maintain an FER of 10^{-4}. This increase in E_b/N_0 is indeed modest in comparison with that for convolutional codes.

4.13.4 Fading Wireless Channels

Wireless channels are prone to error bursts, while both convolutional and turbo decoders work best with independent error events. Hence, both types of codes require the use of interleavers in the transmitter and corresponding deinterleavers in the receiver, as illustrated in Fig. 4.1, to combat error bursts that occur on wireless channels. In the absence of problems such as loss of synchronization, nonstationary characteristics of the channel, or other disruptions in the communications link, larger interleavers imply better performance for both decoding techniques.

The performance curves of turbo codes, which plot BER against E_b/N_0 in dBs, have a brick-wall shape. The corresponding performance curves of convolutional codes exhibit a slow roll-off characteristic, which matches the behavior of fading wireless channels reasonably well; whether this feature provides an advantage or not depends on the ratio of the data rate and block length to the fading rate. Moreover, for short block lengths, which are most robust for communication over fading wireless channels, the improvement offered by turbo codes over convolutional codes is usually small.

Both types of codes are present in third-generation (3G) wireless standards.

4.13.5 Latency

The issue of latency refers to the delay incurred by a channel decoder in processing the received signal in order to recover the original sequence of information bits. Latency is particularly serious in the transmission of voice signals over a wireless channel, as close-to-real-time communication is a necessary requirement, but perhaps less so in the transmission of data.

Latency is proportional to the interleaver size, since the decoder usually must receive the complete interleaved block of bits before it can start decoding. For turbo codes that include two interleavers, namely, a turbo interleaver and a channel interleaver, the delay is proportional to the larger of the two. Latency may also be related to decoder speed, but for both types of codes, practical decoders ordinarily operate at the channel rate or faster, and this is usually not a distinguishing feature.

Smaller block sizes mean smaller latency. Convolutional codes usually achieve their maximum gain at a smaller block size. Turbo codes, in contrast, require large block sizes to achieve their maximum gain, which is usually greater than that attained by convolutional codes.

4.13.6 Joint Equalization and Decoding[11]

An elegant feature of the turbo coding principle is that it provides an iterative basis for the joint implementation of channel equalization and channel decoding in the receiver in a way that makes a significant difference to the overall receiver performance. This issue is discussed in a theme example in Section 4.18.

1. Turbo codes typically have a very small free distance, but, upon decoding, they have far fewer error events at this free distance than convolutional codes have. It is this latter characteristic that gives rise to the brick-wall nature of turbo code performance. However, asymptotically, the performance of turbo codes is still characterized by their minimum free distance. Since this distance is often small, it causes the BER performance of the turbo decoder to level off at high SNR and then appear as an error floor compared with the initial brick-wall performance.
2. The convolutional encoders used in the turbo encoder are recursive, which are therefore non-return-to-zero encoders in that they return to their initial state only with probability 2^{-v}, where v is the number of memory units in the shift register of the code. Furthermore, the turbo code may be endowed with a cyclic trellis, which means that the temporal representation of the states assumed by the encoder for time steps $j = 1$ to $j = k$, where k, the number of information bits, repeats itself in a cyclic manner. Consequently, the turbo code becomes a block code, and, most important, the non-return-to-zero sequence produced by the encoder affects all of the redundant bits introduced by the encoder in such a way that the probability of the turbo decoder failing to recover the original non-return-to-zero sequence is negligibly small.

4.14 RF MODULATION REVISITED

With source (speech) encoding and channel encoding in place, the next operation that is performed in the transmitter is modulation to prepare the signal for transmission over the wireless channel. Here, we have a wide range of strategies to consider, as discussed in Chapter 3. However, with *spectral efficiency* of narrowband wireless communications being an issue of primary concern, it is necessary that adjacent-channel interference (i.e., the spillage of the spectrum of a modulated signal into adjacent channels) be minimized.

FIGURE 4.19 Block diagram of Gaussian minimum-shift keying (GMSK) signal generator.

Partial-response modulation provides one approach to accomplishing this practical requirement. The basic idea of partial-response modulation is to ensure that the *phase response of the modulated signal is spread over several symbol periods.* Stated another way, the phase response in any signaling interval is "partial"—hence the term "partial-response modulation." Note, however, that the use of partial-response modulation has the effect of a deliberate introduction of *controlled* intersymbol interference (ISI), which typically requires the use of an equalizer at the receiver for its removal. *Gaussian minimum-shift keying* (GMSK) is an example of partial-response modulation. (See Section 3.7 for its detailed description.) A block diagram of the GMSK signal generator is shown in Fig. 4.19.

4.15 BASEBAND PROCESSING FOR CHANNEL ESTIMATION AND EQUALIZATION[12]

The generated RF modulated signal $s(t)$ is transmitted over the narrowband wireless channel whose midband is centered on the carrier frequency f_c. Thereupon, the transmitted RF signal is convolved with the impulse response of the channel, $h(t)$. (The use of convolution to describe signal transmission over the channel is justified, since the wireless channel is essentially linear, as discussed in Chapter 2.) The signal resulting from this convolution operation plus additive white Gaussian noise (AWGN) $w(t)$ at the channel output constitutes the received RF signal

$$x(t) = s(t) \otimes h(t) + w(t) \tag{4.35}$$

where, again, the symbol $\otimes$ denotes convolution. The signal $x(t)$ is ready for processing by the receiver.

With the ever-increasing availability of digital signal-processing devices powered by continuing improvements in computer hardware (in the form of silicon chips) and software, the trend nowadays is to convert the received RF signal $x(t)$ into *baseband* form. Simply put, not only is the use of digital signal processing cost effective, but it also provides flexibility unmatched by analog devices. The RF-to-baseband conversion is accomplished with the *quadrature demodulator*, depicted in Fig. 4.20, in accordance with the discussion of the complex representation of band-pass signals and systems presented in Section 3.4. The demodulator yields a *complex baseband signal* $\tilde{x}(t)$ which is equivalent to the RF modulated signal $x(t)$ in that there is no loss of information. The baseband signal $\tilde{x}(t)$ is complex in that it consists of two orthogonal components, as shown by the formula

$$\tilde{x}(t) = x_I(t) + x_Q(t) \tag{4.36}$$

where $x_I(t)$ is the *in-phase component* and $x_Q(t)$ is the *quadrature component*. Note that both $x_I(t)$ and $x_Q(t)$ are low-pass real signals whose common bandwidth is one-half that of the RF-modulated signal $x(t)$.

Recall from the discussion presented in Section 3.4 that RF-to-baseband conversion has an effect on the narrowband wireless channel centered on the carrier frequency f_c, which is similar to the channel output $x(t)$. Specifically, the real impulse response of the channel, $h(t)$, is transformed into the complex equivalent baseband form

$$\tilde{h}(t) = h_I(t) + jh_Q(t) \tag{4.37}$$

where $h_I(t)$ is the in-phase component and $h_Q(t)$ is the quadrature component. In a manner similar to the baseband representation of the modulated signal $x(t)$, both $h_I(t)$ and $h_Q(t)$ are real low-pass impulse responses occupying the same frequency band as the complex baseband signal $\tilde{x}(t)$.

The analog baseband quadrature signals $x_I(t)$ and $x_Q(t)$ are converted into digital form, whereafter the baseband signal processing is carried out in the receiver. (The analog-to-digital converter is not shown in Fig. 4.20, as it is considered part of the RF-to-baseband converter.)

With the digitized in-phase and quadrature components of the received signal at hand, the baseband signal processing can begin in the receiver. The objective of this processing is to compute a set of *transition metrics* that, in turn, are employed in a Viterbi equalizer designed to remove the effects of any controlled ISI (e.g., due to the partial-response modulation) and the channel-induced ISI, thereby facilitating data recovery. To that end, the baseband processor solves two intermediate problems, with the solution to the first leading to the solution of the second. The first problem is that of estimating the unknown impulse response of the channel. The second problem involves generating all the possible signals that would emanate from a channel having

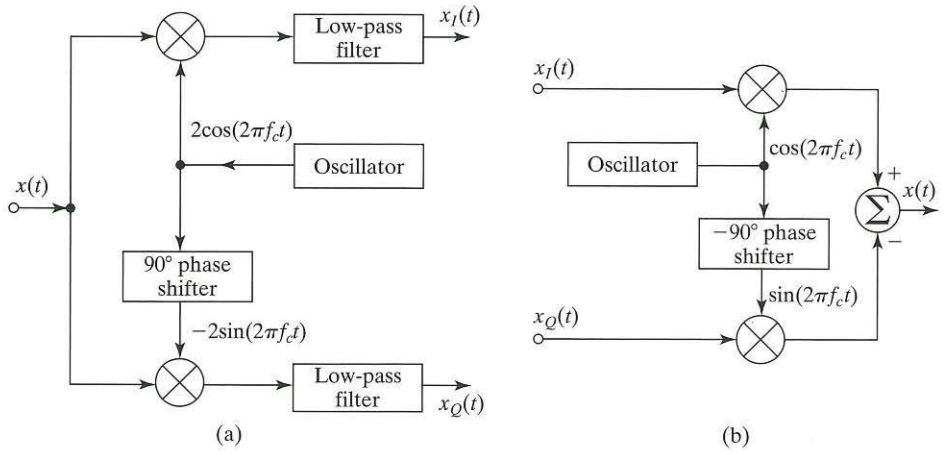

FIGURE 4.20 (a) Scheme for deriving the in-phase and quadrature components of RF-modulated signal $x(t)$. (b) Scheme for reconstructing the RF-modulated signal from its in-phase and quadrature components.

such an estimate for its impulse response. The related issues of channel estimation and estimated signal generation are addressed in what follows in that order.

4.15.1 Channel Estimation

To proceed with the estimation of the channel impulse response, the baseband signal $\tilde{x}(t)$ is first *demultiplexed* into two parts, as illustrated in Fig. 4.21. The part denoted by $\tilde{x}_{\text{channel}}(t)$ relates to the sounding signal employed for channel estimation. The other part, denoted by $\tilde{x}_{\text{data}}(t)$, contains information on the original source signal $m(t)$.

Before reaching the baseband processor, the channel-probing signal is introduced into the "packetizer" in the transmitter and is then modulated on the RF carrier, transmitted over the narrowband channel, and, finally, converted into complex baseband form; typically, the probing signal has an impulselike autocorrelation function, for reasons to be outlined in the discourse that follows. To simplify the discussion, we ignore the effect of channel noise. Under this assumption, the signal $\tilde{x}_{\text{channel}}(t)$ may be expressed as the convolution of two complex time functions:

$$\tilde{x}_{\text{channel}}(t) = \tilde{c}(t) \otimes \tilde{h}(t) \qquad w(t) = 0 \tag{4.38}$$

Here, $\tilde{c}(t)$ and $\tilde{h}(t)$ are, respectively, the baseband versions of the probing signal $c(t)$ and channel impulse response $h(t)$. The signal $\tilde{x}_{\text{channel}}(t)$ is applied to a *matched filter*, whose impulse response, $\tilde{q}(t)$, is equal to the complex conjugate of a time-reversed version of $\tilde{c}(t)$. Appropriately delayed to satisfy causality (see Appendix D), the impulse response is

$$\tilde{q}(t) = \tilde{c}^*(T_c - t) \tag{4.39}$$

where T_c is the duration of the sounding signal $c(t)$ and the asterisk denotes complex conjugation. The matched-filter output is therefore

$$\begin{aligned} \tilde{z}(t) &= \tilde{q}(t) \otimes \tilde{x}_{\text{channel}}(t) \\ &= \tilde{c}^*(T_c - t) \otimes \tilde{c}(t) \otimes \tilde{h}(t) \end{aligned} \tag{4.40}$$

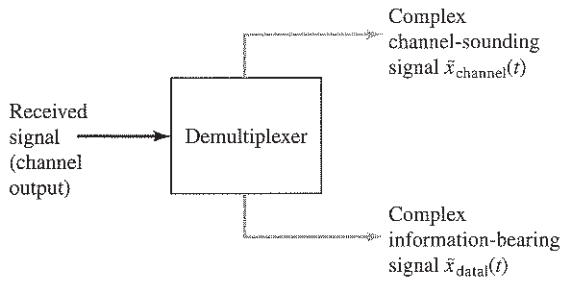

FIGURE 4.21 Block diagram of demultiplexer at the receiver input, following the quadrature demodulator.

The following is an important property of matched filters (see Appendix D):

> *The output of a matched filter, in response to an input signal to which the filter is matched, is equal to the autocorrelation function of the input signal.*

In the context of our present discussion, this statement means that

$$\tilde{c}^*(T_c - t) \otimes \tilde{c}(t) = r_{\tilde{c}}(T_c - t) \tag{4.41}$$

where $r_{\tilde{c}}(\tau)$ is the *autocorrelation function* of the complex envelope $\tilde{c}(t)$ of the probing signal for lag $\tau = T_c - t$. Hence, Eq. (4.40) reduces to

$$\tilde{z}(t) = r_{\tilde{c}}(T_c - t) \otimes \tilde{h}(t) \tag{4.42}$$

In general, the autocorrelation function of a complex signal has a complex value. However, by having the original sounding signal $c(t)$ exhibit *even symmetry* about its midpoint $t = T_c/2$, the autocorrelation function $r_{\tilde{c}}(\tau)$ assumes a *real* value, which simplifies the design of the baseband processor considerably. To emphasize this point, we set

$$r_{\tilde{c}}(\tau) = \rho(\tau) \qquad \text{for all } \tau \tag{4.43}$$

where $\rho(\tau)$ is a real autocorrelation function that is an even function of τ; that is,

$$\rho(-\tau) = \rho(\tau) \tag{4.44}$$

Accordingly, we may rewrite Eq. (4.42) in terms of this new function as

$$\tilde{z}(t) = \rho(t - T_c) \otimes \tilde{h}(t) \tag{4.45}$$

In words, the output $\tilde{z}(t)$ of the matched filter is equal to the real-valued, delayed autocorrelation function $\rho(t - T_c)$, convolved with the complex baseband impulse response $\tilde{h}(t)$ of the channel.

Suppose now that the autocorrelation function $\rho(\tau)$ not only is real, but also is in the form of a *delta function* $\delta(\tau)$. Then Eq. (4.45) simplifies further to the ideal result

$$\begin{aligned} \tilde{z}(t) &= \delta(t - T_c) \otimes \tilde{h}(t) \\ &= \tilde{h}(t - T_c) \end{aligned} \tag{4.46}$$

where we have used the fact that the convolution of a time function with a delta function leaves that time function unchanged, except for a possible time shift.

The result of Eq. (4.46) is idealized in that it is derived under two special conditions:

1. The channel noise $w(t)$ is zero.
2. The probing signal $c(t)$ is long enough for the autocorrelation function of its complex envelope $\tilde{c}(t)$ to approach a delta function.