Chapter 12

Design of a Digital Communication System

The power of digital signal processing can probably be best appreciated in the enormous progresses which have been made in the field of telecommunications. These progresses stem from three main properties of digital processing:

- The flexibility and power of discrete-time processing techniques, which allow for the low-cost deployment of sophisticated and, more importantly, *adaptive* equalization and filtering modules.
- The ease of integration between low-level digital processing and high-level information-theoretical techniques which counteract transmission errors.
- The *regenerability* of a digital signal: in the necessary amplification of analog signals after transmission, the noise floor is amplified as well, thereby limiting the processing gain. Digital signals, on the other hand, can be *exactly* regenerated under reasonable SNR conditions (Fig. 1.10).

The fruits of such powerful communication systems are readily enjoyable in everyday life and it suffices here to mention the fast ADSL connections which take the power of high data rates into the home. ADSL is actually a quantitative evolution of a humbler, yet extraordinarily useful device: the voiceband modem. Voiceband modems, transmitting data at a rate of up to 56 Kbit/sec over standard telephone lines, are arguably the crown achievement of discrete-time signal processing in the late 90's and are still the cornerstone of most wired telecommunication devices such as laptops and fax machines.

In this Chapter, we explore the design and implementation of a voiceband modem as a paradigmatic example of applied digital signal processing. In principle, the development of a fully-functional device would require the use of concepts which are beyond the scope of this book, such as adaptive signal processing and information theory. Yet we will see that, if we neglect some of the impairments that are introduced by real-world telephone lines, we are able to design a working system which will flawlessly modulates and demodulates a data sequence.

12.1 The Communication Channel

A telecommunication system works by exploiting the propagation of electromagnetic waves in a medium. In the case of radio transmission, the medium is the electromagnetic spectrum; in the case of land-line communications such as those in voiceband or ADSL modems, the medium is a copper wire. In all cases, the properties of the medium determine two fundamental constraints around which any communication system is designed:

- **Bandwith constraint:** data transmission systems work best in the frequency range over which the medium behaves linearly; over this *passband* we can rely on the fact that a signal will be received with only phase and amplitude distortions, and these are "good" types of distortion since they amount to a linear filter. Further limitations on the available bandwidth can be imposed by law or by technical requirements and the transmitter must limit its spectral occupancy to the prescribed frequency region.

- **Power constraint:** the power of a transmitted signal is inherently limited by various factors, including the range over which the medium and the transmission circuitry behaves linearly. In many other cases, such as in telephone or radio communications, the maximum power is strictly regulated by law. Also, power could be limited by the effort to maximize the operating time of battery-powered mobile devices. At the same time, all analog media are affected by noise, which can come in the form of interference from neighboring transmission bands (as in the case of radio channels) or of parasitic noise due to electrical interference (as in the case of AC hum over audio lines). The *noise floor* is the noise level which cannot be removed and must be reckoned with in the transmission scheme. Power constraints limit the achievable *signal to noise ratio* (SNR) with respect to the channel's noise floor; in turn, the SNR determines the reliability of the data transmission scheme.

These constraints define a communication channel and the goal, in the design of a communication system, is to maximize the amount of information which can be reliably transmitted across a given channel. In the design of a *digital* communication system, the additional goal is to operate entirely in the discrete-time domain up to the interface with the physical channel; this means that:

- at the transmitter, the signal is synthesized, shaped and modulated in the discrete-time domain and is converted to a continuous-time signal just prior to transmission;
- at the receiver, the incoming signal is sampled from the channel and demodulation, processing and decoding is performed in the digital domain.

12.1.1 The AM Radio Channel

A classic example of a regulated electromagnetic channel is commercial radio. Bandwidth constraints in the case of the electromagnetic spectrum are rigorously put in place because the spectrum is a scarce resource which needs to be shared amongst a multitude of users (commercial radio, amateur radio, cellular telephony, emergency services, military use, etc). Power constraints on radio emissions are imposed for human safety concerns. The AM band, for instance, extends from 530 kHz to 1700 kHz; each radio station is allotted an 8 kHz frequency slot in this range. Suppose that a speech signal $x(t)$, obtained with a microphone, is to be transmitted over a slot extending from $f_{\min} = 650$ kHz to $f_{\max} = 658$ kHz. Human speech can be modeled as a bandlimited signal with a frequency support of approximately 12 kHz; speech can, however, be filtered through a lowpass filter with cutoff frequency 4 kHz with little loss of intelligibility so that its bandwidth can be made to match the 8 kHz bandwidth of the AM channel. The filtered signal now has a spectrum extending from -4 kHz to 4 kHz; multiplication by a sinusoid at frequency $f_c = (f_{\max} + f_{\min})/2 = 654$ KHz shifts its support according to the continuous-time version of the *modulation theorem*: if $x(t) \overset{\text{FT}}{\longleftrightarrow} X(j\Omega)$ then:

$$x(t)\cos(\Omega_c t) \overset{\text{FT}}{\longleftrightarrow} \frac{1}{2}\left[X(j\Omega - j\Omega_c) + X(j\Omega + j\Omega_c)\right] \tag{12.1}$$

where $\Omega_c = 2\pi f_c$. This is, of course, a completely analog transmission system, which is schematically displayed in Figure 12.1.

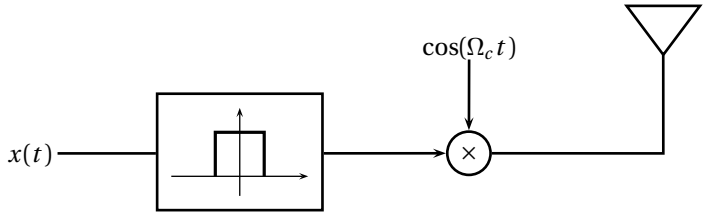

Figure 12.1 A simple AM radio transmitter.

12.1.2 The Telephone Channel

The telephone channel is basically a copper wire connecting two users. Because of the enormous number of telephone posts in the world, only a relatively small number of wires is used and the wires are *switched* between users when a call is made. The telephone network (also known as POTS, an acronym for "Plain Old Telephone System") is represented schematically in Figure 12.2. Each physical telephone is connected via a *twisted pair* (i.e. a pair of plain copper wires) to the nearest central office (CO); there are a lot of central offices in the network so that each telephone is usually no more than a few kilometers away. Central offices are connected to each other via the main lines in the network and the digits dialed by a caller are interpreted by the CO as connection instruction to the CO associated to the called number.

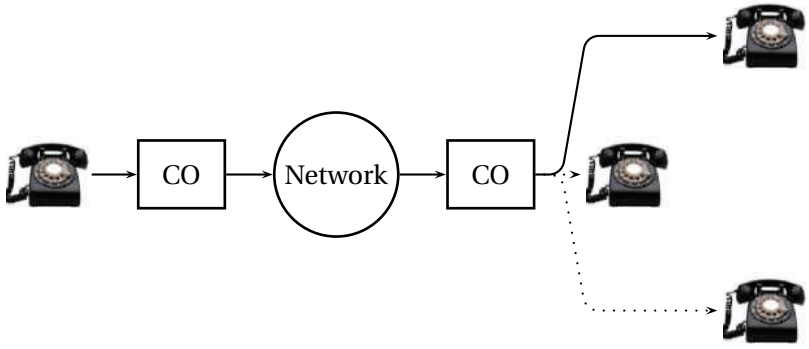

Figure 12.2 The Plain Old Telephone System (POTS).

To understand the limitations of the telephone channel we have to step back to the old analog times when COs were made of electromechanical switches and the voice signals traveling inside the network were boosted with simple operational amplifiers. The first link of the chain, the twisted pair to the central office, actually has a bandwidth of several MHz since it is just a copper wire (this is the main technical fact behind ADSL, by the

way). Telephone companies, however, used to introduce what are called *loading coils* in the line to compensate for the attenuation introduced by the capacitive effects of longer wires in the network. A side effect of these coils was to turn the first link into a lowpass filter with a cutoff frequency of approximately 4 kHz so that, in practice, the *official* passband of the telephone channel is limited between $f_{\min} = 300$ Hz and $f_{\max} = 3000$ Hz, for a total usable positive bandwidth $W = 2700$ Hz. While today most of the network is actually digital, the official bandwidth remains in the order of 8 KHz (i.e. a positive bandwidth of 4 KHz); this is so that many more conversations can be multiplexed over the same cable or satellite link. The standard sampling rate for a telephone channel is nowadays 8 KHz and the bandwidth limitations are imposed only by the antialiasing filters at the CO, for a maximum bandwidth in excess of $W = 3400$ Hz. The upper and lower ends of the band are not usable due to possible great attenuations which may take place in the transmission. In particular, telephone lines exhibit a sharp notch at $f = 0$ (also known as *DC level*) so that any transmission scheme *will have to use bandpass signals* exclusively.

The telephone channel is power limited as well, of course, since telephone companies are quite protective of their equipment. Generally, the limit on signaling over a line is 0.2 V rms; the interesting figure however is not the maximum signaling level but the overall signal-to-noise ratio of the line (i.e. the amount of unavoidable noise on the line *with respect to* the maximum signaling level). Nowadays, phone lines are extremely high-quality: a SNR of at least 28 dB can be assumed in all cases and one of 32-34 dB can be reasonably expected on a large percentage of individual connections.

12.2 Modem Design: The Transmitter

Data transmission over a physical medium is by definition analog; modern communication systems, however, place all of the processing in the digital domain so that the only interface with the medium is the final D/A converter at the end of the processing chain, following the signal processing paradigm of Section 9.7.

12.2.1 Digital Modulation and the Bandwidth Constraint

In order to develop a digital communication system over the telephone channel, we need to re-cast the problem in the discrete-time domain. To this end, it is helpful to consider a very abstract view of the data transmitter,

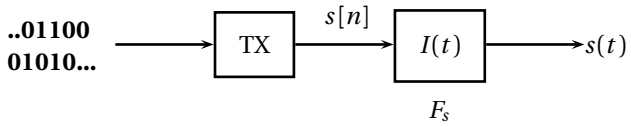

Figure 12.3 Abstract view of a digital transmitter.

as shown in Figure 12.3. Here, we neglect the details associated to the digital modulation process and concentrate on the digital-to-analog interface, represented in the picture by the interpolator $I(t)$; the input to the transmitter is some generic binary data, represented as a bit stream. The bandwidth constraints imposed by the channel can be represented graphically as in Figure 12.4. In order to produce a signal which "sits" in the prescribed frequency band, we need to use a D/A converter working at a frequency $F_s \geq 2f_{\max}$. Once the interpolation frequency is chosen (and we will see momentarily the criteria to do so), the requirements for the discrete-time signal $s[n]$ are set. The bandwidth requirements become simply

$$\omega_{\min,\max} = 2\pi \frac{f_{\min,\max}}{F_s}$$

and they can be represented as in Figure 12.5 (in the figure, for instance, we have chosen $F_s = 2.28\, f_{\max}$).

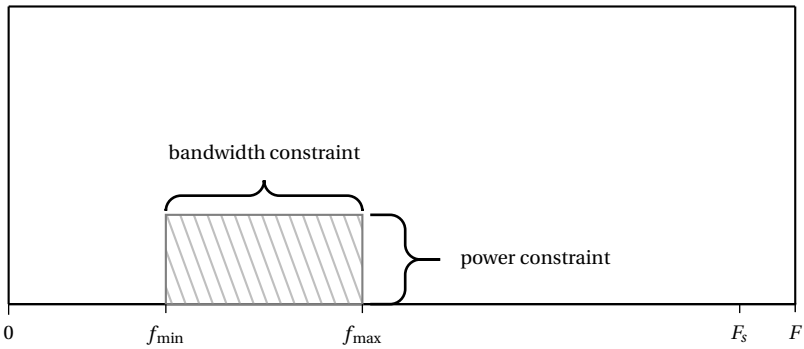

Figure 12.4 Analog specifications (positive frequencies) for the transmitter.

We can now try to understand how to build a suitable $s[n]$ by looking more in detail into the input side of the transmitter, as shown in Figure 12.6. The input bitstream is first processed by a *scrambler*, whose purpose is to randomize the data; clearly, it is a pseudo-randomization since this operation needs to be undone algorithmically at the receiver. Please note how the implementation of the transmitter in the digital domain allows for a seamless integration between the transmission scheme and more abstract data

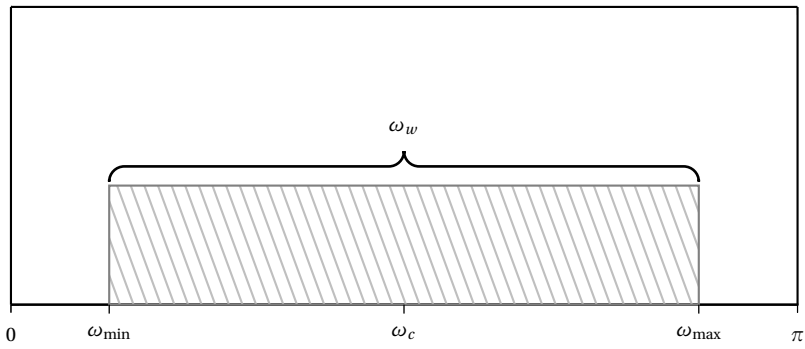

Figure 12.5 Discrete-time specifications (positive frequencies) for $F_s = 2.28\, f_{\max}$.

manipulation algorithms such as randomizers. The randomized bitstream could already be transmitted at this point; in this case, we would be implementing a binary modulation scheme in which the signal $s[n]$ varies between the two levels associated to a zero and a one, much in the fashion of telegraphic communications of yore. Digital communication devices, however, allow for a much more efficient utilization of the available bandwidth via the implementation of *multilevel* signaling. With this strategy, the bitstream is segmented in consecutive groups of M bits and these bits select one of 2^M possible signaling values; the set of all possible signaling values is called the *alphabet* of the transmission scheme and the algorithm which associates a group of M bits to an alphabet symbol is called the *mapper*. We will discuss practical alphabets momentarily; however, it is important to remark that the series of symbols *can be complex* so that all the signals in the processing chain up to the final D/A converter are complex signals.

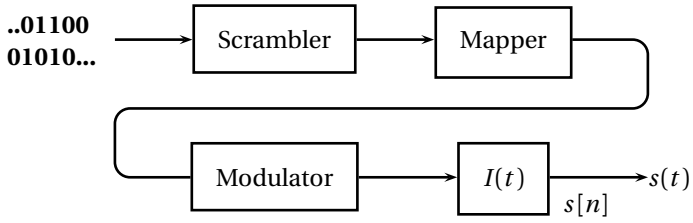

Figure 12.6 Data stream processing detail.

Spectral Properties of the Symbol Sequence. The mapper produces a sequence of symbols $a[n]$ which is the actual discrete-time signal which we need to transmit. In order to appreciate the spectral properties of this se-

quence consider that, if the initial binary bitstream is a maximum-information sequence (i.e. if the distribution of zeros and ones looks random and "fifty-fifty"), and with the scrambler appropriately randomizing the input bitstream, the sequence of symbols $a[n]$ can be modeled as a stochastic i.i.d. process distributed over the alphabet. Under these circumstances, the power spectral density of the random signal $a[n]$ is simply

$$P_A(e^{j\omega}) = \sigma_A^2$$

where σ_A depends on the design of the alphabet and on its distribution.

Choice of Interpolation Rate. We are now ready to determine a suitable rate F_s for the final interpolator. The signal $a[n]$ is a baseband, fullband signal in the sense that it is centered around zero and its power spectral density is nonzero over the entire $[-\pi, \pi]$ interval. If interpolated at F_s, such a signal gives rise to an analog signal with nonzero spectral power over the entire $[-F_s/2, F_s/2]$ interval (and, in particular, nonzero power at DC level). In order to fulfill the channel's constraints, we need to produce a signal with a bandwidth of $\omega_w = \omega_{\max} - \omega_{\min}$ centered around $\omega_c = \pm(\omega_{\max} + \omega_{\min})/2$. The "trick" is to *upsample* (and interpolate) the sequence $a[n]$, in order to narrow its spectral support.[1] Assuming ideal discrete-time interpolators, an upsampling factor of 2, for instance, produces a half-band signal; an upsampling factor of 3 produces a signal with a support spanning one third of the total band, and so on. In the general case, we need to choose an upsampling factor K so that:

$$\frac{2\pi}{K} \leq \omega_w$$

Maximum efficiency occurs when the available bandwidth is entirely occupied by the signal, i.e. when $K = 2\pi/\omega_w$. In terms of the analog bandwidth requirements, this translates to

$$K = \frac{F_s}{f_w} \tag{12.2}$$

where $f_w = f_{\max} - f_{\min}$ is the effective positive bandwidth of the transmitted signal; since K must be an integer, the previous condition implies that we must choose an interpolation frequency *which is a multiple of the positive*

[1] A rigorous mathematical analysis of multirate processing of stochastic signals turns out to be rather delicate and beyond the scope of this book; the same holds for the effects of modulation, which will appear later on. Whenever in doubt, we may simply visualize the involved signals as a deterministic realization whose spectral shape mimics the power spectral density of their generating stochastic process.

passband width f_w. The two criteria which must be fulfilled for optimal signaling are therefore:

$$\begin{cases} F_s \geq 2f_{\max} \\ F_s = Kf_w \qquad K \in \mathbb{N} \end{cases} \tag{12.3}$$

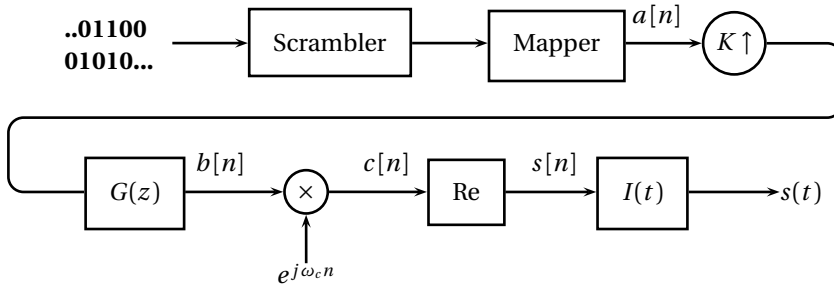

Figure 12.7 Complete digital transmitter.

The Baseband Signal. The upsampling by K operation, used to narrow the spectral occupancy of the symbol sequence to the prescribed bandwidth, must be followed by a lowpass filter, to remove the multiple copies of the upsampled spectrum; this is achieved by a lowpass filter which, in digital communication parlance, is known as the *shaper* since it determines the time domain shape of the transmitted symbols. We know from Section 11.2.1 that, ideally, we should use a sinc filter to perfectly remove all repeated copies. Since this is clearly not possible, let us now examine the properties that a practical discrete-time interpolator should possess in the context of data communications. The baseband signal $b[n]$ can be expressed as

$$b[n] = \sum_m a_{KU}[m]\, g[n-m]$$

where $a_{KU}[n]$ is the upsampled symbol sequence and $g[n]$ is the lowpass filter's impulse response. Since $a_{KU}[n] = 0$ for n not a multiple of K, we can state that:

$$b[n] = \sum_i a[i]\, g[n - iK] \tag{12.4}$$

It is reasonable to impose that, at multiples of K, the upsampled sequence $b[n]$ takes on the exact symbol value, i.e. $b[mK] = a[m]$; this translates to the following requirement for the lowpass filter:

$$g[mK] = \begin{cases} 1 & m = 0 \\ 0 & m \neq 0 \end{cases} \tag{12.5}$$

This is nothing but the classical interpolation property which we saw in Section 9.4.1. For realizable filters, this condition implies that the minimum frequency support of $G(e^{j\omega})$ cannot be smaller than $[-\pi/K, \pi/K]$.[2] In other words, there will always be a (controllable) amount of *frequency leakage* outside of a prescribed band with respect to an ideal filter.

To exactly fullfill (12.5), we need to use an FIR lowpass filter; FIR approximations to a sinc filter are, however, very poor, since the impulse response of the sinc decays very slowly. A much friendlier lowpass characteristic which possesses the interpolation property and allows for a precise quantification of frequency leakage, is the *raised cosine*. A raised cosine with nominal bandwidth ω_w (and therefore with nominal cutoff $\omega_b = \omega_w/2$) is defined over the positive frequency axis as

$$G(e^{j\omega}) = \begin{cases} 1 & \text{if } 0 < \omega < (1-\beta)\omega_b \\ 0 & \text{if } (1+\beta)\omega_b < \omega < \pi \\ \dfrac{1}{2} + \dfrac{1}{2}\cos\left(\pi \dfrac{\omega - (1-\beta)\omega_b}{2\beta\omega_b}\right) & \\ & \text{if } (1-\beta)\omega_b < \omega < (1+\beta)\omega_b \end{cases} \tag{12.6}$$

and is symmetric around the origin. The parameter β, with $0 < \beta < 1$, exactly defines the amount of frequency leakage as a percentage of the passband. The closer β is to one, the sharper the magnitude response; a set of frequency responses for $\omega_b = \pi/2$ and various values of β are shown in Figure 12.8. The raised cosine is still an ideal filter but it can be shown that its

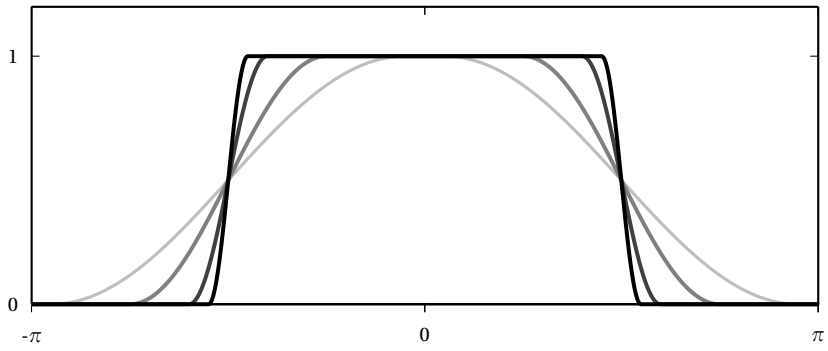

Figure 12.8 Frequency responses of a half-band raised-cosine filter for increasing values of β: from black to light gray, $\beta = 0.1$, $\beta = 0.2$, $\beta = 0.4$, $\beta = 0.9$.

[2] A simple proof of this fact can be outlined using multirate signal processing. Assume the spectrum $G(e^{j\omega})$ is nonzero only over $[-\omega_b, \omega_b]$, for $\omega_b < \pi/K$; $g[n]$ can therefore be subsampled by at least a factor of K without aliasing, and the support of the resulting spectrum is going to be $[-K\omega_b, K\omega_b]$, with $K\omega_b < \pi$. However, $g[Kn] = \delta[n]$, whose spectral support is $[-\pi, \pi]$.

impulse response decays as $1/n^3$ and, therefore, good FIR approximations can be obtained with a reasonable amount of taps using a specialized version of Parks-McClellan algorithm. The number of taps needed to achieve a good frequency response obviously increases as β approaches one; in most practical applications, however, it rarely exceeds 50.

The Bandpass Signal. The filtered signal $b[n] = g[n] * a_{KU}[n]$ is now a baseband signal with total bandwidth ω_w. In order to shift the signal into the allotted frequency band, we need to modulate[3] it with a sinusoidal *carrier* to obtain a complex bandpass signal:

$$c[n] = b[n]\, e^{j\omega_c n}$$

where the modulation frequency is the center-band frequency:

$$\omega_c = \frac{\omega_{\min} + \omega_{\max}}{2}$$

Note that the spectral support of the modulated signal is just the *positive* interval $[\omega_{\min}, \omega_{\max}]$; a complex signal with such a one-sided spectral occupancy is called an *analytic signal*. The signal which is fed to the D/A converter is simply the real part of the complex bandpass signal:

$$s[n] = \mathrm{Re}\{c[n]\} \tag{12.7}$$

If the baseband signal $b[n]$ is real, then (12.7) is equivalent to a standard cosine modulation as in (12.1); in the case of a complex $b[n]$ (as in our case), the bandpass signal is the combination of a cosine *and* a sine modulation, which we will examine in more detail later. The spectral characteristics of the signals involved in the creation of $s[n]$ are shown in Figure 12.9.

Baud Rate vs Bit Rate. The *baud rate* of a communication system is the number of *symbols* which can be transmitted in one second. Considering that the interpolator works at F_s samples per second and that, because of upsampling, there are exactly K samples per symbol in the signal $s[n]$, the baud rate of the system is

$$B = \frac{F_s}{K} = f_w \tag{12.8}$$

where we have assumed that the shaper $G(z)$ is an ideal lowpass. As a general rule, *the baud rate is always smaller or equal to the positive passband of the channel*. Moreover, if we follow the normal processing order, we can

[3] See footnote (1).

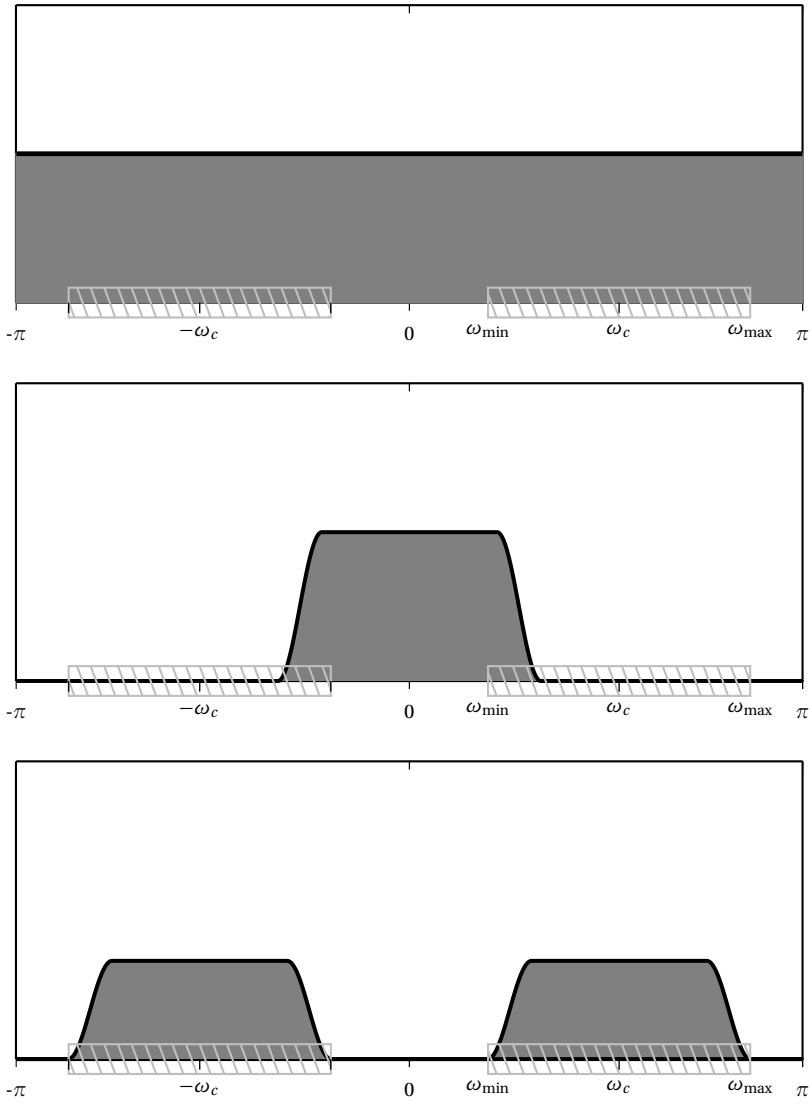

Figure 12.9 Construction of the modulated signal: PSD of the symbol sequence $a[n]$ (top panel); PSD of the upsampled and shaped signal $b[n]$ (middle panel); PSD of the real modulated signal $s[n]$ (bottom panel). The channel's bandwidth requirements are indicated by the dashed areas.

equivalently say that a symbol sequence generated at B symbols per second gives rise to a modulated signal whose positive passband is *no smaller than* B *Hz*. The effective bandwidth f_w depends on the modulation scheme and, especially, on the frequency leakage introduced by the shaper.

The total bit rate of a transmission system, on the other hand, is at most the baud rate times the log in base 2 of the number of symbols in the alphabet; for a mapper which operates on M bits per symbol, the overall bitrate is

$$R = MB \tag{12.9}$$

A Design Example. As a practical example, consider the case of a telephone line for which $f_{\min} = 450$ Hz and $f_{\max} = 3150$ Hz (we will consider the power constraints later). The baud rate can be at most 2700 symbols per second, since $f_w = f_{\max} - f_{\min} = 2700$ Hz. We choose a factor $\beta = 0.125$ for the raised cosine shaper and, to compensate for the bandwidth expansion, we deliberately reduce the actual baud rate to $B = 2700/(1+\beta) = 2400$ symbols per second, which leaves the effective positive bandwidth equal to f_w. The criteria which the interpolation frequency must fulfill are therefore the following:

$$\begin{cases} F_s \geq 2 f_{\max} = 6300 \\ F_s = KB = 2400\,K \qquad K \in \mathbb{N} \end{cases}$$

The first solution is for $K = 3$ and therefore $F_s = 7200$. With this interpolation frequency, the effective bandwidth of the discrete-time signal is $\omega_w = 2\pi(2700/7200) = 0.75\,\pi$ and the carrier frequency for the bandpass signal is $\omega_c = 2\pi(450+3150)/(2F_s) = \pi/2$. In order to determine the maximum attainable bitrate of this system, we need to address the second major constraint which affects the design of the transmitter, i.e. the power constraint.

12.2.2 Signaling Alphabets and the Power Constraint

The purpose of the mapper is to associate to each group of M input bits a value α from a given *alphabet* $\mathscr{A}$. We assume that the mapper includes a multiplicative factor G_0 which can be used to set the final gain of the generated signal, so that we don't need to concern ourselves with the absolute values of the symbols in the alphabet; the symbol sequence is therefore:

$$a[n] = G_0 \alpha[n], \qquad \alpha[n] \in \mathscr{A}$$

and, in general, the values α are set at integer coordinates out of convenience.

Transmitted Power. Under the above assumption of an i.i.d. uniformly distributed binary input sequence, each group of M bits is equally probable; since we consider only *memoryless* mappers, i.e. mappers in which no de-

pendency between symbols is introduced, the mapper acts as the source of a random process $a[n]$ which is also i.i.d. The power of the output sequence can be expressed as

$$\begin{aligned}\sigma_a^2 &= \mathrm{E}\left|a[n]\right|^2 \\ &= G_0^2 \sum_{\alpha \in \mathscr{A}} \left|\alpha\right|^2 p_a(\alpha) \qquad (12.10)\\ &= G_0^2 \sigma_\alpha^2 \qquad (12.11)\end{aligned}$$

where $p_a(\alpha)$ is the probability assigned by the mapper to symbol $\alpha \in \mathscr{A}$; the distribution over the alphabet $\mathscr{A}$ is one of the design parameters of the mapper, and is not necessarily uniform. The variance σ_α^2 is the intrinsic power of the alphabet and it depends on the alphabet size (it increases exponentially with M), on the alphabet structure, and on the probability distribution of the symbols in the alphabet. Note that, in order to avoid wasting transmission energy, communication systems are designed so that the sequence generated by the mapper is *balanced*, i.e. its DC value is zero:

$$\mathrm{E}[\alpha[n]] = \sum_{\alpha \in \mathscr{A}} \alpha p_a(\alpha) = 0$$

Using (8.25), the power of the transmitted signal, after upsampling and modulation, is

$$\sigma_s^2 = \frac{1}{\pi} \int_{\omega_{\min}}^{\omega_{\max}} \frac{1}{2} \left|G(e^{j\omega})\right|^2 G_0^2 \sigma_\alpha^2 \qquad (12.12)$$

The shaper is designed so that its overall energy over the passband is $G^2 = 2\pi$ and we can express this as follows:

$$\sigma_s^2 = G_0^2 \sigma_\alpha^2 \qquad (12.13)$$

In order to respect the power constraint, we have to choose a value for G_0 and design an alphabet $\mathscr{A}$ so that:

$$\sigma_s^2 \leq P_{\max} \qquad (12.14)$$

where $P_{\max}$ is the maximum transmission power allowed on the channel. The goal of a data transmission system is to maximize the *reliable* throughput but, unfortunately, in this respect the parameters σ_α^2 and G_0 act upon conflicting priorities. If we use (12.9) and boost the transmitter's bitrate by increasing M, then σ_α^2 grows and we must necessarily reduce the gain G_0 to fulfill the power constraint; but, in so doing, we impair the reliability of the transmission. To understand why that is, we must leap ahead and consider both a practical alphabet and the mechanics of symbol decoding at the transmitter.

QAM. The simplest mapping strategies are one-to-one correspondences between binary values and signal values: note that in these cases the symbol sequence is uniformly distributed with $p_a(\alpha) = 2^{-M}$ for all $\alpha \in \mathscr{A}$. For example, we can assign to each group of M bits $(b_0, \ldots, b_{M-1})$ the signed binary number $b_0 b_1 b_2 \cdots b_{M-1}$ which is a value between -2^{M-1} and 2^{M-1} (b_0 is the sign bit). This signaling scheme is called *pulse amplitude modulation* (PAM) since the amplitude of each transmitted symbol is directly determined by the binary input value. The PAM alphabet is clearly balanced and the inherent power of the mapper's output is readily computed as[4]

$$\sigma_\alpha^2 = \sum_{\alpha=1}^{2^{M-1}} 2^{-M} \alpha^2 = \frac{2^M(2^M+3)+2}{24}$$

Now, a pulse-amplitude modulated signal prior to modulation is a baseband signal with positive bandwidth of, say, ω_0 (see Figure 12.9, middle panel); therefore, the *total* spectral support of the baseband PAM signal is $2\omega_0$. After modulation, the total spectral support of the signal actually doubles (Fig. 12.9, bottom panel); there is, therefore, some sort of redundancy in the modulated signal which causes an underutilization of the available bandwidth. The original spectral efficiency can be regained with a signaling scheme called *quadrature amplitude modulation* (QAM); in QAM the symbols in the alphabet are complex quantities, so that *two* real values are transmitted simultaneously at each symbol interval. Consider a complex symbol sequence

$$a[n] = G_0\big(\alpha_I[n] + j\,\alpha_Q[n]\big) = a_I[n] + j\,a_Q[n]$$

Since the shaper is a real-valued filter, we have that:

$$b[n] = \big(a_{I,KU} * g[n]\big) + j\big(a_{Q,KU} * g[n]\big) = b_I[n] + j\,b_Q[n]$$

so that, finally, (12.7) becomes:

$$\begin{aligned} s[n] &= \mathrm{Re}\big\{b[n]\, e^{j\omega_c n}\big\} \\ &= b_I[n]\cos(\omega_c n) - b_Q[n]\sin(\omega_c n) \end{aligned}$$

In other words, a QAM signal is simply the linear combination of two pulse-amplitude modulated signals: a cosine carrier modulated by the real part of the symbol sequence and a sine carrier modulated by the imaginary part of the symbol sequence. The sine and cosine carriers are *orthogonal* signals, so that $b_I[n]$ and $b_Q[n]$ can be exactly separated at the receiver via a subspace projection operation, as we will see in detail later. The subscripts I

[4] A useful formula, here and in the following, is $\sum_{n=1}^{N} n^2 = N(N+1)(2N+1)/6$.

and Q derive from the historical names for the cosine carrier (the *in-phase* carrier) and the sine carrier which is the *quadrature* (i.e. the orthogonal carrier). Using complex symbols for the description of the internal signals in the transmitter is an abstraction which simplifies the overall notation and highlights the usefulness of complex discrete-time signal models.

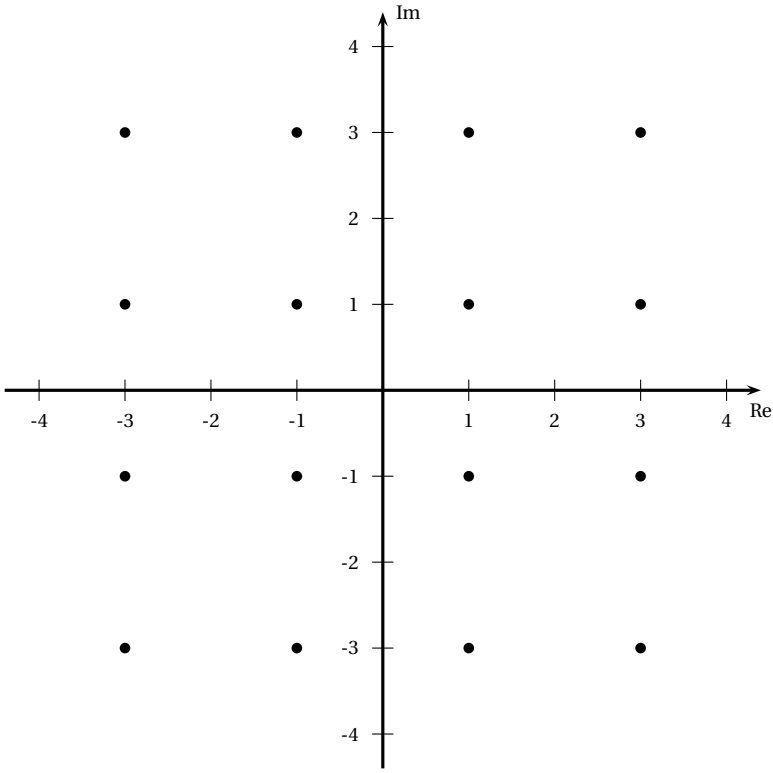

Figure 12.10 16-point QAM constellations ($M = 4$).

Constellations. The 2^M symbols in the alphabet can be represented as points in the complex plane and the geometrical arrangement of all such points is called the signaling *constellation*. The simplest constellations are upright square lattices with points on the odd integer coordinates; for M even, the 2^M constellation points α_{hk} form a square shape with $2^{M/2}$ points per side:

$$\alpha_{hk} = (2h-1) + j(2k-1), \qquad -2^{M/2-1} < h,k \leq 2^{M/2-1}$$

Such square constellations are called *regular* and a detailed example is shown in Figure 12.10 for $M = 4$; other examples for $M = 2, 6, 8$ are shown in Figure 12.11. The nominal power associated to a regular, uniformly distributed constellation on the square lattice can be computed as the second moment of the points; exploiting the fourfold symmetry, we have

$$\begin{aligned}\sigma_\alpha^2 &= 4 \sum_{h=1}^{2^{M/2-1}} \sum_{k=1}^{2^{M/2-1}} 2^{-M} \left[(2h-1)^2 + (2k-1)^2 \right] \\ &= \frac{2}{3}(2^M - 1) \end{aligned} \tag{12.15}$$

Square-lattice constellations exist also for alphabet sizes which are not perfect squares and examples are shown in Figure 12.12 for $M = 3$ (8-point constellation) and $M = 5$ (32-point). Alternatively, constellations can be defined on other types of lattices, either irregular or regular; Figure 12.13 shows an alternative example of an 8-point constellation defined on an irregular grid and a 19-point constellation defined over a regular hexagonal lattice. We will see later how to exploit the constellation's geometry to increase performance.

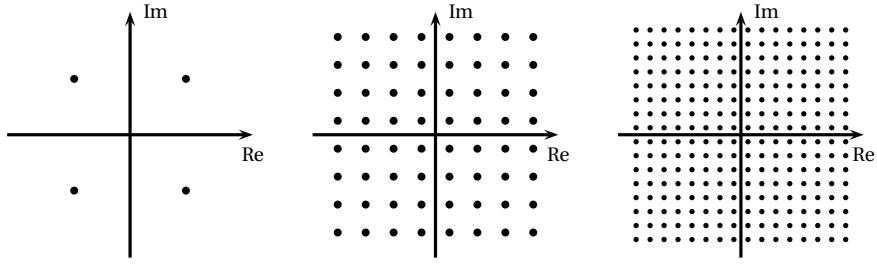

Figure 12.11 4-, 64- and 256-point QAM constellations (M bits/symbol for $M = 2$, $M = 6$, $M = 8$) respectively.

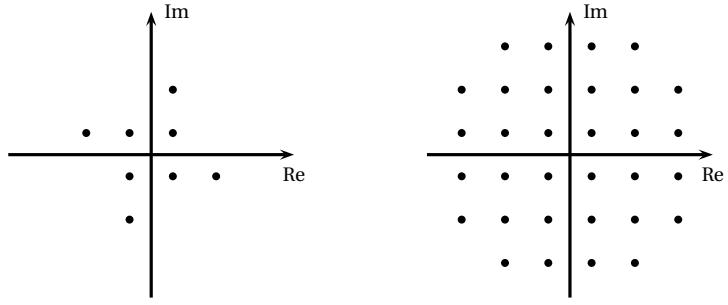

Figure 12.12 8- and 32-point square-lattice constellations.

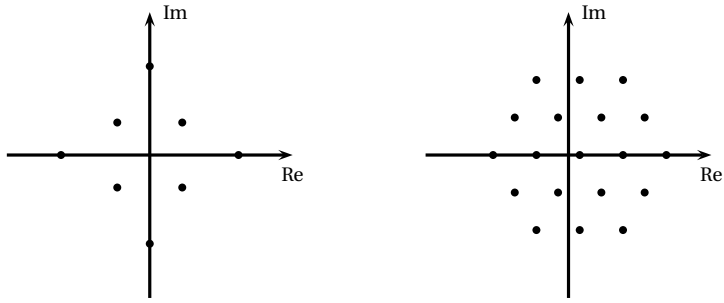

Figure 12.13 More exotic constellations: irregular low-power 8-point constellation (left panel) in which the outer point are at a distance of $1+\sqrt{3}$ from the origin ; regular 19-point hexagonal-lattice constellation (right panel).

Transmission Reliability. Let us assume that the receiver has eliminated all the "fixable" distortions introduced by the channel so that an "almost exact" copy of the symbol sequence is available for decoding; call this sequence $\hat{a}[n]$. What no receiver can do, however, is eliminate all the additive noise introduced by the channel so that:

$$\hat{a}[n] = a[n] + \eta[n] \tag{12.16}$$

where $\eta[n]$ is a *complex* white Gaussian noise term. It will be clear later why the internal mechanics of the receiver make it easier to consider a complex representation for the noise; again, such complex representation is a convenient abstraction which greatly simplifies the mathematical analysis of the decoding process. The real-valued zero-mean Gaussian noise introduced by the channel, whose variance is σ_0^2, is transformed by the receiver into complex Gaussian noise whose real and imaginary parts are independent zero-mean Gaussian variables with variance $\sigma_0^2/2$. Each complex noise sample $\eta[n]$ is distributed according to

$$f_\eta(z) = \frac{1}{\pi\sigma_0^2}\, e^{-\frac{|z|^2}{\sigma_0^2}} \tag{12.17}$$

The magnitude of the noise samples introduces a shift in the complex plane for the demodulated symbols $\hat{a}[n]$ with respect to the originally transmitted symbols; if this displacement is too big, a decoding error takes place. In order to quantify the effects of the noise we have to look more in detail at the way the transmitted sequence is retrieved at the receiver. A bound on the probability of error can be obtained analytically if we consider a simple QAM decoding technique called *hard slicing*. In hard slicing, a value $\hat{a}[n]$ is associated to the most probable symbol $\alpha \in \mathscr{A}$ by choosing the alphabet symbol at the minimum Euclidean distance (taking the gain G_0 into account):

$$\mathscr{D}\{\hat{a}[n]\} = \arg\min_{\alpha\in\mathscr{A}} \left\{ \left| \hat{a}[n] - G_0\alpha \right|^2 \right\}$$

The hard slicer partitions the complex plane into decision regions centered on alphabet symbols; all the received values which fall into the decision region centered on α are mapped back onto α. Decision regions for a 16-point constellation, together with examples of correct and incorrect hard slicing are represented in Figure 12.14: when the error sample $\eta[n]$ moves the received symbol outside of the right decision region, we have a decoding error. For square-lattice constellations, this happens when either the real or the imaginary part of the noise sample is larger than the minimum distance between a symbol and the closest decision region boundary. Said distance is $d_{\min} = G_0$, as can be easily seen from Figure 12.10, and therefore the probability of error at the receiver is

$$\begin{aligned} p_e &= 1 - \mathrm{P}\Big[\Big(\mathrm{Re}\{\eta[n]\} < G_0\Big) \wedge \Big(\mathrm{Im}\{\eta[n]\} < G_0\Big)\Big] \\ &= 1 - \int_D f_\eta(z)\,dz \end{aligned}$$

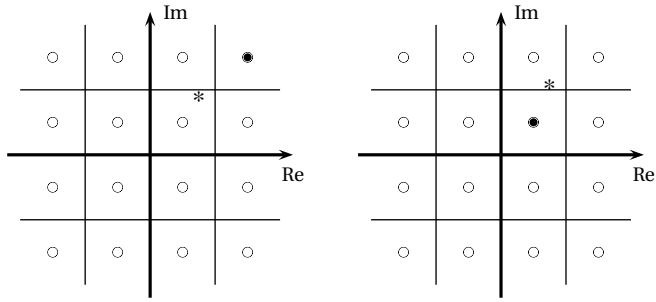

Figure 12.14 Decoding of noisy symbols: transmitted symbol is black dot, received value is the star. Correct decoding (left) and decoding error (right).

where $f_\eta(x)$ is the pdf of the additive complex noise and D is a square on the complex plane centered at the origin and $2d_{\min}$ wide. We can obtain a closed-form expression for the probability of error if we approximate the decision region D by the inscribed circle of radius $d_{\min}$ (Fig. 12.15), so:

$$\begin{aligned} p_e &= 1 - \int_{|z|<G_0} f_\eta(z)\,dz \\ &= 1 - \int_0^{2\pi} d\theta \int_0^{G_0} \frac{\rho}{\pi\sigma_0^2}\, e^{-\frac{\rho^2}{\sigma_0^2}} \\ &= e^{-\frac{G_0^2}{\sigma_0^2}} \end{aligned} \tag{12.18}$$

where we have used (12.17) and the change of variable $z = \rho\, e^{j\theta}$. The probability of error decreases exponentially with the gain and, therefore, with the power of the transmitter.

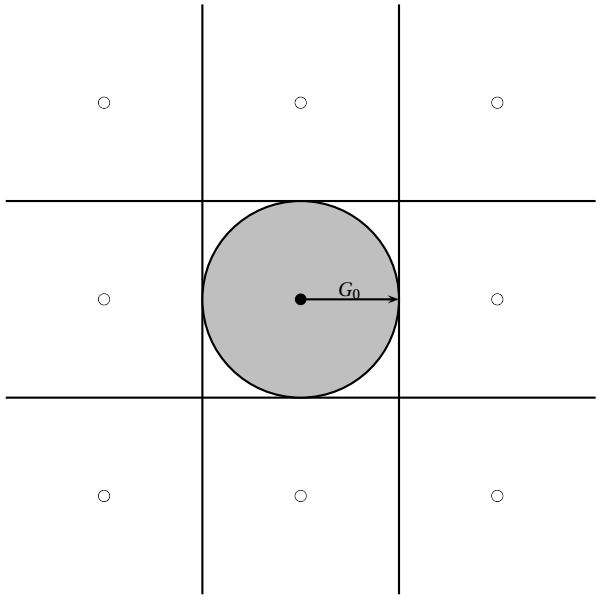

Figure 12.15 Decision region and its circular approximation.

The concept of "reliability" is quantified by the probability of error that we are willing to tolerate; note that this probability can never be zero, but it can be made arbitrarily low – values on the order of $p_e = 10^{-6}$ are usually taken as a reference. Assume that the transmitter transmits at the maximum permissible power so that the SNR on the channel is maximized. Under these conditions it is

$$\text{SNR} = \frac{\sigma_s^2}{\sigma_0^2} = G_0^2 \frac{\sigma_\alpha^2}{\sigma_0^2}$$

and from (12.18) we have

$$\text{SNR} = -\ln(p_e)\sigma_\alpha^2 \tag{12.19}$$

For a regular square-lattice constellation we can use (12.15) to determine the maximum number of bits per symbol which can be transmitted at the given reliability figure:

$$M = \log_2\left(1 - \frac{3}{2}\frac{\text{SNR}}{\ln(p_e)}\right) \tag{12.20}$$

and this is how the power constraint ultimately affects the maximum achievable bitrate. Note that the above derivation has been carried out with very specific hypotheses on both the signaling alphabet and on the decoding algorithm (the hard slicing); the upper bound on the achievable rate on the channel is actually a classic result of information theory and is known under the name of Shannon's capacity formula. Shannon's formula reads

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

where C is the absolute maximum capacity in bits per second, B is the available bandwidth in Hertz and S/N is the signal to noise ratio.

Design Example Revisited. Let us resume the example on page 339 by assuming that the power constraint on the telephone line limits the maximum achievable SNR to 22 dB. If the acceptable bit error probability is $p_e = 10^{-6}$, Equation (12.20) gives us a maximum integer value of $M = 4$ bits per symbol. We can therefore use a regular 16-point square constellation; recall we had designed a system with a baud rate of 2400 symbols per second and therefore the final reliable bitrate is $R = 9600$ bits per second. This is actually one of the operating modes of the V.32 ITU-T modem standard.[5]

12.3 Modem Design: The Receiver

The analog signal $s(t)$ created at the transmitter is sent over the telephone channel and arrives at the receiver as a distorted and noise-corrupted signal $\hat{s}(t)$. Again, since we are designing a purely digital communication system, the receiver's input interface is an A/D converter which, for simplicity, we assume, is operating at the same frequency F_s as the transmitter's D/A converter. The receiver tries to undo the impairments introduced by the channel and to demodulate the received signal; its output is a binary sequence which, in the absence of decoding errors, is identical to the sequence injected into the transmitter; an abstract view of the receiver is shown in Figure 12.16.

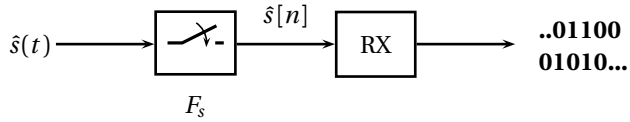

Figure 12.16 Abstract view of a digital receiver.

[5] ITU-T is the Standardization Bureau of the International Telecommunication Union.

12.3.1 Hilbert Demodulation

Let us assume for the time being that transmitter and receiver are connected back-to-back so that we can neglect the effects of the channel; in this case $\hat{s}(t) = s(t)$ and, after the A/D module, $\hat{s}[n] = s[n]$. Demodulation of the incoming signal to a binary data stream is achieved according to the block diagram in Figure 12.17 where all the steps in the modulation process are undone, one by one.

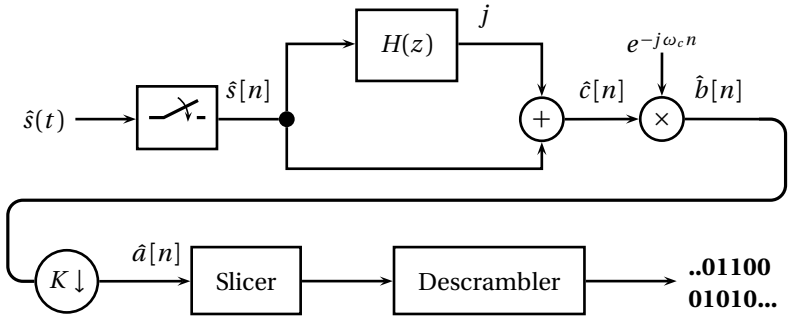

Figure 12.17 Complete digital receiver.

The first operation is retrieving the complex bandpass signal $\hat{c}[n]$ from the real signal $\hat{s}[n]$. An efficient way to perform this operation is by exploiting the fact that the original $c[n]$ is an analytic signal and, therefore, its imaginary part is completely determined by its real part. To see this, consider a complex analytic signal $x[n]$, i.e. a complex sequence for which $X(e^{j\omega}) = 0$ over the $[-\pi, 0]$ interval (with the usual 2π-periodicity, obviously). We can split $x[n]$ into real and imaginary parts:

$$x[n] = x_r[n] + j x_i[n]$$

so that we can write:

$$x_r[n] = \frac{x[n] + x^*[n]}{2}$$
$$x_i[n] = \frac{x[n] - x^*[n]}{2} j$$

In the frequency domain, these relations translate to (see (4.46)):

$$X_r(e^{j\omega}) = \frac{\left[X(e^{j\omega}) + X^*(e^{-j\omega})\right]}{2} \tag{12.21}$$
$$X_i(e^{j\omega}) = \frac{\left[X(e^{j\omega}) - X^*(e^{-j\omega})\right]}{2} j \tag{12.22}$$

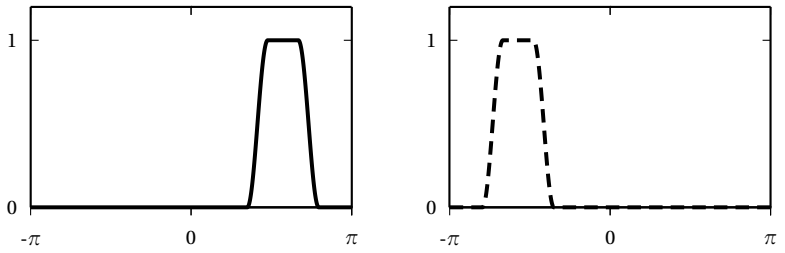

Figure 12.18 Magnitude spectrum of an analytic signal $x[n]$. $\left|X^*(e^{-j\omega})\right|$ (left) and $\left|X^*(e^{-j\omega})\right|$ (right).

Since $x[n]$ is analytic, by definition $X(e^{j\omega})=0$ for $-\pi\le\omega<0$, $X^*(e^{-j\omega})=0$ for $0<\omega\le\pi$ and $X(e^{j\omega})$ does not overlap with $X^*(e^{-j\omega})$ (Fig. 12.18). We can therefore use (12.21) to write:

$$X(e^{j\omega})=\begin{cases}2X_r(e^{j\omega}) & \text{for } 0\le\omega\le\pi\\ 0 & \text{for } -\pi<\omega<0\end{cases} \tag{12.23}$$

Now, $x_r[n]$ is a real sequence and therefore its Fourier transform is conjugate-symmetric, i.e. $X_r(e^{j\omega})=X_r^*(e^{-j\omega})$; as a consequence

$$X^*(e^{-j\omega})=\begin{cases}0 & \text{for } 0\le\omega\le\pi\\ 2X_r(e^{j\omega}) & \text{for } -\pi<\omega<0\end{cases} \tag{12.24}$$

By using (12.23) and (12.24) in (12.22) we finally obtain:

$$X_i(e^{j\omega})=\begin{cases}-jX_r(e^{j\omega}) & \text{for } 0\le\omega\le\pi\\ +jX_r(e^{j\omega}) & \text{for } -\pi<\omega<0\end{cases} \tag{12.25}$$

which is the product of $X_r(e^{j\omega})$ with the frequency response of a Hilbert filter (Sect. 5.6). In the time domain this means that the imaginary part of an analytic signal can be retrieved from the real part only via the convolution:

$$x_i[n]=h[n]*x_r[n]$$

At the demodulator, $\hat{s}[n]=s[n]$ is nothing but the real part of $c[n]$ and therefore the analytic bandpass signal is simply

$$\hat{c}[n]=\hat{s}[n]+j\left(h[n]*\hat{s}[n]\right)$$

In practice, the Hilbert filter is approximated with a causal, $2L+1$-tap type III FIR, so that the structure used in demodulation is that of Figure 12.19.

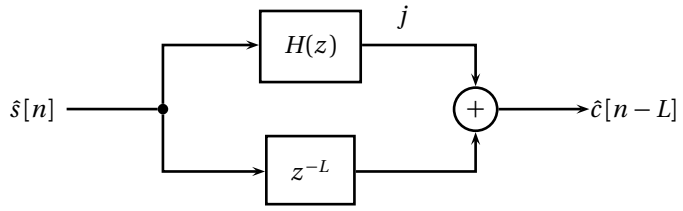

Figure 12.19 Retrieving the complex baseband signal with an FIR Hilbert filter approximation.

The delay in the bottom branch compensates for the delay introduced by the causal filter and puts the real and derived imaginary part back in sync to obtain:

$$\hat{c}[n] = \hat{s}[n-L] + j\big(h[n] * \hat{s}[n]\big)$$

Once the analytic bandpass signal is reconstructed, it can be brought back to baseband via a complex demodulation with a carrier with frequency $-\omega_c$:

$$\hat{b}[n] = \hat{c}[n]\, e^{-j\omega_c n}$$

Because of the interpolation property of the pulse shaper, the sequence of complex symbols can be retrieved by a simple downsampling-by-K operation:

$$\hat{a}[n] = \hat{b}[nK]$$

Finally, the slicer (which we saw in Section 12.2.2) associates a group of M bits to each received symbol and the descrambler reconstructs the original binary stream.

12.3.2 The Effects of the Channel

If we now abandon the convenient back-to-back scenario, we have to deal with the impairments introduced by the channel and by the signal processing hardware. The telephone channels affects the received signal in three fundamental ways:

- it adds noise to the signal so that, even in the best case, the signal-to-noise ratio of the received signal cannot exceed a maximum limit;
- it distorts the signal, acting as a linear filter;
- it delays the signal, according to the propagation time from transmitter to receiver.

Distortion and delay are obviously both linear transformations and, as such, their description could be lumped together; still, the techniques which deal with distortion and delay are different, so that the two are customarily kept separate. Furthermore, the physical implementation of the devices introduces an unavoidable lack of absolute synchronization between transmitter and receiver, since each of them runs on an independent internal clock. Adaptive synchronization becomes a necessity in all real-world devices, and will be described in the next Section.

Noise. The effects of noise have already been described in Section 12.2.2 and can be summed up visually by the plots in Figure 12.20 in each of which successive values of $\hat{a}[n]$ are superimposed on the same axes. The analog noise is transformed into discrete-time noise by the sampler and, as such, it leaks through the demodulation chain to the reconstructed symbols sequence $\hat{a}[n]$; as the noise level increases (or, equivalently, as the SNR decreases) the shape of the received constellation progressively loses its tightness around the nominal alphabet values. As symbols begin to cross the boundaries of the decision regions, more and more decoding errors, take place.

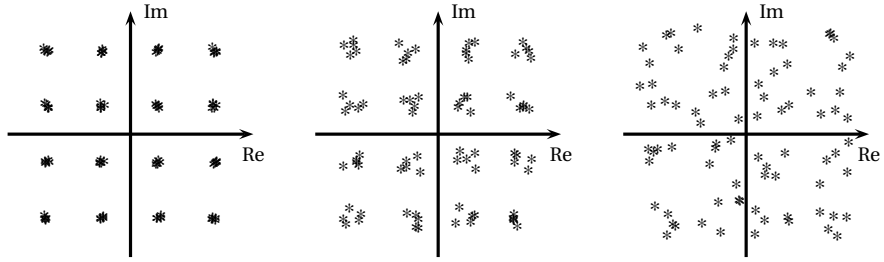

Figure 12.20 Noisy constellation for decreasing SNR.

Equalization. We saw previously that the passband of a communication channel is determined by the frequency region over which the channel introduces only linear types of distortion. The channel can therefore be modeled as a continuous-time linear filter $D_c(j\Omega)$ whose frequency response is unknown (and potentially time-varying). The received signal (neglecting noise) is therefore $\hat{S}(j\Omega) = D_c(j\Omega)S(j\Omega)$ and, after the sampler, we have

$$\hat{S}(e^{j\omega}) = D(e^{j\omega})S(e^{j\omega})$$

where $D(e^{j\omega})$ represents the combined effect of the original channel and of the anti-aliasing filter at the A/D converter. To counteract the channel distortion, the receiver includes an *adaptive equalizer* $E(z)$ right after the A/D

converter; this is an FIR filter which is modified on the fly so that $E(z) \approx 1/D(z)$. While adaptive filter theory is beyond the scope of this book, the intuition behind adaptive equalization is shown in Figure 12.21. In fact, the demodulator contains an exact copy of the modulator as well; if we assume that the symbols produced by the slicer are error-free, a perfect copy of the transmitted signal $s[n]$ can be generated locally at the receiver. The difference between the equalized signal and the reconstructed original signal is used to adapt the taps of the equalizer so that:

$$d[n] = \hat{s}_e[n] - s[n] \longrightarrow 0$$

Clearly, in the absence of a good initial estimate for $D(e^{j\omega})$, the sliced values $\hat{a}[n]$ are nothing like the original sequence; this is obviated by having the transmitter send a pre-established *training sequence* which is known in advance at the receiver. The training sequence, together with other synchronization signals, is sent each time a connection is established between transmitter and receiver and is part of the modem's *handshaking protocol*. By using a training sequence, $E(z)$ can quickly converge to an approximation of $1/D(z)$ which is good enough for the receiver to start decoding symbols correctly and use them in driving further adaptation.

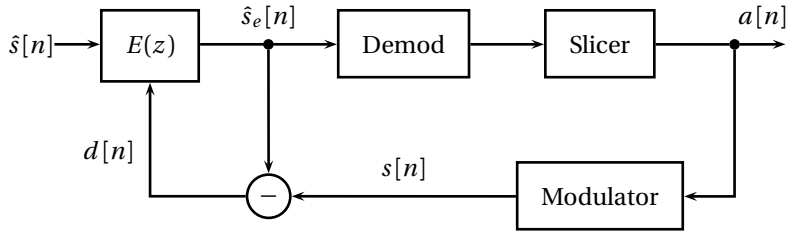

Figure 12.21 Adaptive equalization: based on the estimated symbols the receiver can synthesize the perfect desired equalizer output and use the difference to drive the adaptation.

Delay. The continuous-time signal arriving at the receiver can be modeled as

$$\hat{s}(t) = (s * v)(t - t_d) + \eta(t) \tag{12.26}$$

where $v(t)$ is the continuous-time impulse response of the channel, $\eta(t)$ is the continuous-time noise process and t_d is the *propagation delay*, i.e. the time it takes for the signal to travel from transmitter to receiver. After the sampler, the discrete-time signal to be demodulated is $\hat{s}[n] = \hat{s}(nT_s)$; if we neglect the noise and distortion, we can write

$$\hat{s}[n] = s(nT_s - t_d) = s\big((n - n_d)T_s - \tau T_s\big) \tag{12.27}$$

where we have split the delay as $t_d = (n_d + \tau)T_s$ with $n_d \in \mathbb{N}$ and $|\tau| \leq 1/2$. The term n_d is called the *bulk delay* and it can be estimated easily in a full-duplex system by the following handshaking procedure:

1. System A sends an impulse to system B at time $n = 0$; the impulse appears on the channel after a known processing delay t_{p1} seconds; let the (unknown) channel propagation delay be t_d seconds.

2. System B receives the impulse and sends an impulse back to A; the processing time t_{p2} (decoding of the impulse and generation of response) is known by design.

3. The response impulse is received by system A after t_d seconds (propagation delay is symmetric) and detected after a processing delay of t_{p3} seconds.

In the end, the total round-trip delay measured by system A is

$$t = 2t_d + t_{p1} + t_{p2} + t_{p3} = 2t_d + t_p$$

since t_p is known exactly in terms of the number of samples, t_d can be estimated *to within a sample*. The bulk delay is easily dealt with at the receiver, since it translated to a simple z^{-n_d} component in the channel's response. The fractional delay, on the other hand, is a more delicate entity which we will need to tackle with specialized machinery.

12.4 Adaptive Synchronization

In order for the receiver to properly decode the data, the discrete-time signals inside the receiver must be synchronous with the discrete-time signals generated by the transmitter. In the back-to-back operation, we could neglect synchronization problems since we assumed $\hat{s}[n] = s[n]$. In reality, we will need to compensate for the propagation delay and for possible clock differences between the D/A at the transmitter and the A/D at the receiver, both in terms of time offsets and in terms of frequency offsets.

12.4.1 Carrier Recovery

Carrier recovery is the modem functionality by which any phase offset between carriers is estimated and compensated for. Phase offsets between the transmitter's and receiver's carriers are due to the propagation delay and to the general lack of a reference clock between the two devices. Assume that

the oscillator in the receiver has a phase offset of θ with respect to the transmitter; when we retrieve the baseband signal $\hat{b}[n]$ from $\hat{c}[n]$ we have

$$\hat{b}[n] = \hat{c}[n]\, e^{-j(\omega_c n - \theta)} = c[n]\, e^{-j(\omega_c n - \theta)} = b[n]\, e^{j\theta}$$

where we have neglected both distortion and noise and assumed $\hat{c}[n] = c[n]$. Such a phase offset translates to a rotation of the constellation points in the complex plane since, after downsampling, we have $\hat{a}[n] = a[n]\, e^{j\theta}$. Visually, the received constellation looks like in Figure 12.22, where $\theta = \pi/20 = 9°$. If we look at the decision regions plotted in Figure 12.22, it is clear that in the rotated constellation some points are shifted closer to the decision boundaries; for these, a smaller amount of noise is sufficient to cause slicing errors. An even worse situation happens when the receiver's carrier frequency is slightly different than the transmitter's carrier frequency; in this case the phase offset changes over time and the points in the constellation start to rotate with an angular speed equal to the difference between frequencies. In both cases, data transmission becomes highly unreliable: carrier recovery is then a fundamental part of modem design.

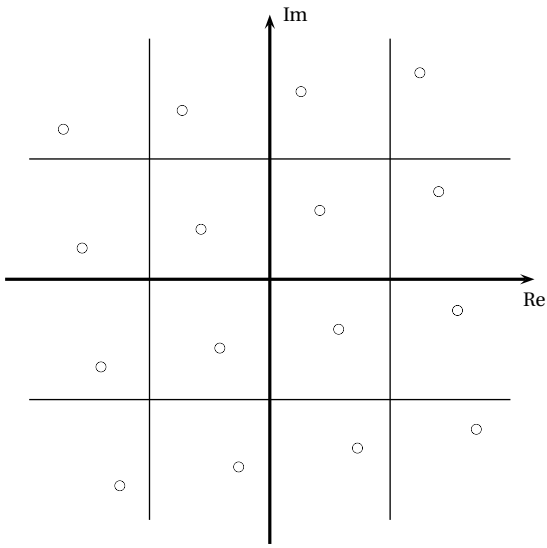

Figure 12.22 Rational effect of a phase offset on the received symbols.

The most common technique for QAM carrier recovery over well-behaved channels is a *decision directed loop*; just as in the case of the adaptive equalizer, this works when the overall SNR is sufficiently high and the distortion is mild so that the slicer's output is an almost error-free sequence of symbols. Consider a system with a phase offset of θ; in Figure 12.23 the

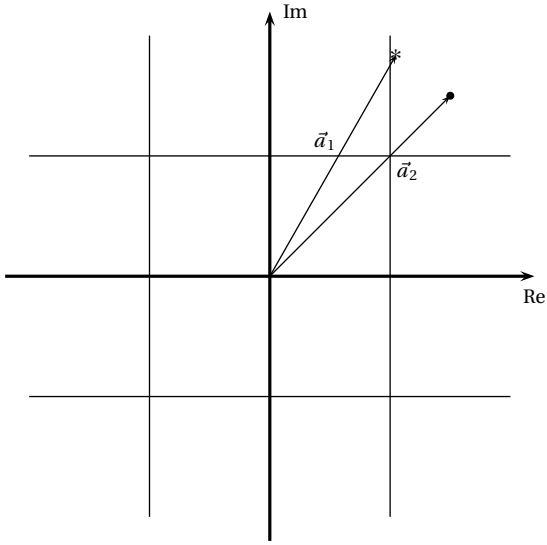

Figure 12.23 Estimation of the phase offset.

rotated symbol $\hat{\alpha}$ (indicated by a star) is sufficiently close to the transmitted value α (indicated by a dot) to be decoded correctly. In the z plane, consider the two vectors $\vec{a}_1$ and $\vec{a}_2$, from the origin to $\hat{\alpha}$ and α respectively; the magnitude of their vector product can be expressed as

$$\left|\vec{a}_1 \times \vec{a}_2\right| = \mathrm{Re}\{\hat{\alpha}\}\,\mathrm{Im}\{\alpha\} - \mathrm{Im}\{\hat{\alpha}\}\,\mathrm{Re}\{\alpha\} \tag{12.28}$$

Moreover, the angle between the vectors is θ and it can be computed as

$$\left|\vec{a}_1 \times \vec{a}_2\right| = \left|\vec{a}_1\right|\left|\vec{a}_2\right|\sin(\theta) \tag{12.29}$$

We can therefore obtain an estimate for the phase offset:

$$\sin(\theta) = \frac{\mathrm{Re}\{\hat{\alpha}\}\,\mathrm{Im}\{\alpha\} - \mathrm{Im}\{\hat{\alpha}\}\,\mathrm{Re}\{\alpha\}}{|\vec{a}_1||\vec{a}_2|} \tag{12.30}$$

For small angles, we can invoke the approximation $\sin(\theta) \approx \theta$ and obtain a quick estimate of the phase offset. In digital systems, oscillators are realized using the algorithm we saw in Section 2.1.3; it is easy to modify such a routine to include a time-varying corrective term derived from the estimate of θ above so that the resulting phase offset is close to zero. This works also in the case of a slight frequency offset, with θ converging in this case to a nonzero constant. The carrier recovery block diagram is shown in Figure 12.24.

This decision-directed feedback method is almost always able to "lock" the constellation in place; due to the fourfold symmetry of regular square

constellations, however, there is no guarantee that the final orientation of the locked pattern be the same as the original. This difficulty is overcome by a mapping technique called *differential encoding*; in differential encoding the first two bits of each symbol actually encode the *quadrant offset* of the symbol with respect to the previous one, while the remaining bits indicate the actual point within the quadrant. In so doing, the encoded symbol sequence becomes independent of the constellation's absolute orientation.

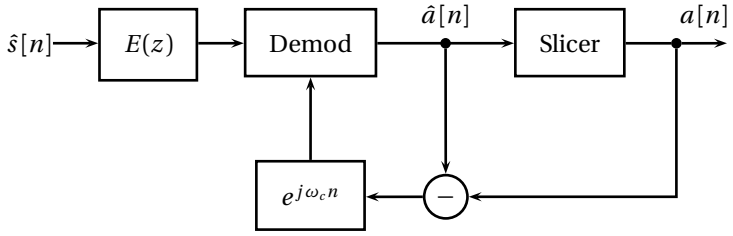

Figure 12.24 Carrier recovery by decision-directed loop.

12.4.2 Timing Recovery

Timing recovery is the ensemble of strategies which are put in place to recover the synchronism between transmitter and receiver at the level of discrete-time samples. This synchronism, which was one of the assumptions of back-to-back operation, is lost in real-world situations because of propagation delays and because of slight hardware differences between devices. The D/A and A/D, being physically separate, run on independent clocks which may exhibit small frequency differences and a slow drift. The purpose of timing recovery is to offset such hardware discrepancies in the discrete-time domain.

A Digital PLL. Traditionally, a Phase-Locked-Loop (PLL) is an analog circuit which, using a negative feedback loop, manages to keep an internal oscillator "locked in phase" with an external oscillatory input. Since the internal oscillator's parameters can be easily retrieved, PLLs are used to accurately measure the frequency and the phase of an external signal with respect to an internal reference.

In timing recovery, we use a PLL-like structure as in Figure 12.25 to compensate for sampling offsets. To see how this PLL works, assume that the discrete-time samples $\hat{s}[n]$ are obtained by the A/D converter as

$$\hat{s}[n] = \hat{s}(t_n) \tag{12.31}$$

where the sequence of sampling instants t_n is generated as

$$t_{n+1} = t_n + T[n] \tag{12.32}$$

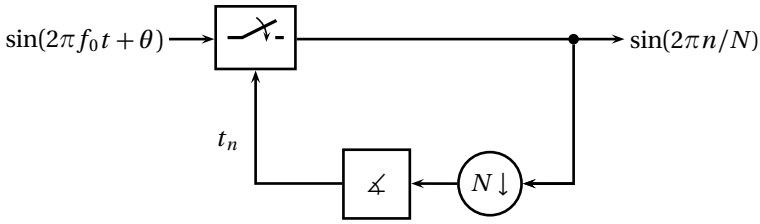

Figure 12.25 A digital PLL with a sinusoidal input.

Normally, the sampling period is a constant and $T[n] = T_s = 1/F_s$ but here we will assume that we have a special A/D converter for which the sampling period can be dynamically changed at each sampling cycle. Assume the input to the sampler is a zero-phase sinusoid of *known* frequency $f_0 = F_s/N$ for $N \in \mathbb{N}$ and $N \geq 2$:

$$x(t) = \sin(2\pi f_0 t)$$

If the sampling period is constant and equal to T_s and if the A/D is synchronous to the sinusoid, the sampled signal are simply:

$$x[n] = \sin\left(\frac{2\pi}{N} n\right)$$

We can test such synchronicity by downsampling $x[n]$ by N and we should have $x_{ND}[n] = 0$ for all n; this situation is shown at the top of Figure 12.26 and we can say that the A/D is *locked* to the reference signal $x(t)$.

If the local clock has a time lag τ with respect to the reference time of the incoming sinusoid (or, alternatively, if the incoming sinusoid is delayed by τ), then the discrete-time, downsampled signal is the constant:

$$x_{ND}[n] = \sin(2\pi f_0 \tau) \tag{12.33}$$

Note, the A/D is still locked to the reference signal $x(t)$, but it exhibits a phase offset, as shown in Figure 12.26, middle panel. If this offset is sufficiently small then the small angle approximation for the sine holds and $x_{ND}[n]$ provides a direct estimate of the corrective factor which needs to be injected into the A/D block. If the offset is estimated at time n_0, it will suffice to set

$$T[n] = \begin{cases} T_s - \tau & \text{for } n = n_0 \\ T_s & \text{for } n > n_0 \end{cases} \tag{12.34}$$

for the A/D to be locked to the input sinusoid.

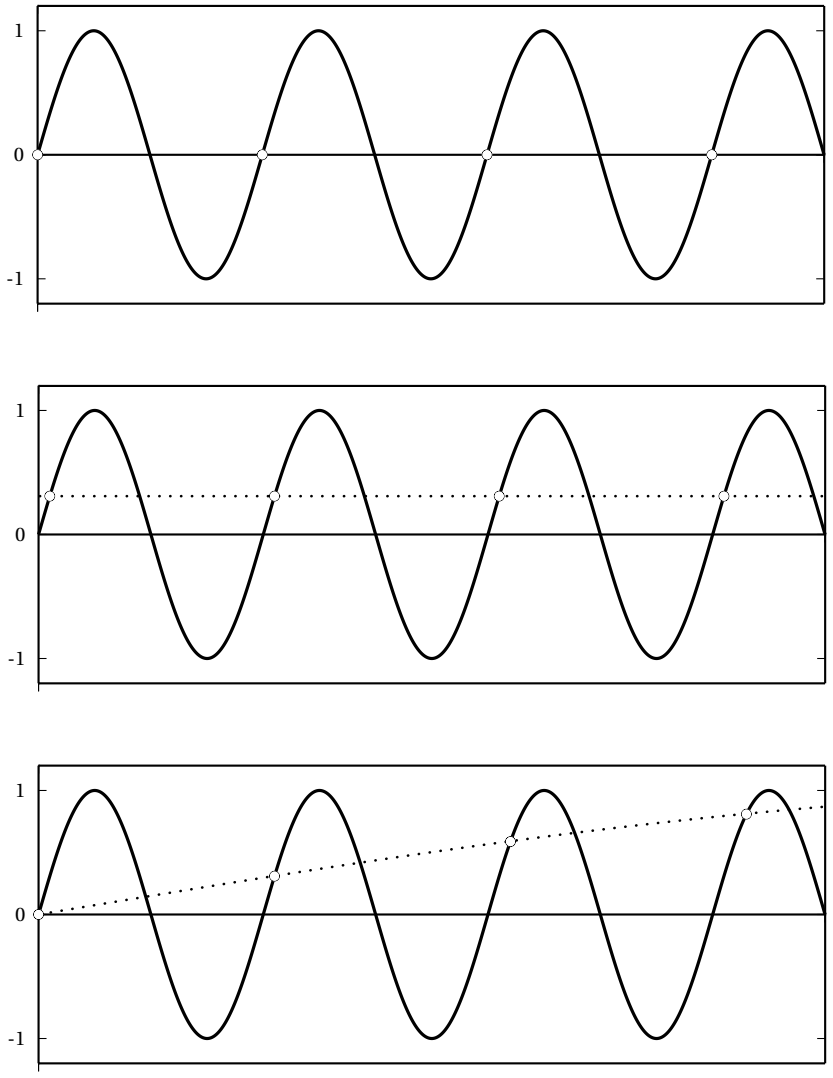

Figure 12.26 Timing recovery from a continuous-time sinusoid, with reference samples drawn as white circles: perfect locking (top); phase offset (middle) and frequency drift (bottom). All plots are in the time reference of the input sinusoid.

Suppose now that the the A/D converter runs slightly slower than its nominal speed or, in other words, that the effective sampling frequency is $F_s' = \beta F_s$, with $\beta < 1$. As a consequence the sampling period is $T_s' = T_s/\beta > T_s$ and the discrete-time, downsampled signal becomes

$$x_{ND}[n] = \sin\big((2\pi\beta)n\big) \tag{12.35}$$

i.e. it is a sinusoid of frequency $2\pi\beta$; this situation is shown in the bottom panel of Figure 12.26. We can use the downsampled signal to estimate β and we can re-establish a locked PLL by setting

$$T[n] = \frac{T_s}{\beta} \tag{12.36}$$

The same strategy can be employed if the A/D runs faster than normal, in which case the only difference is that $\beta > 1$.

A Variable Fractional Delay. In practice, A/D converters with "tunable" sampling instants are rare and expensive because of their design complexity; furthermore, a data path *from* the discrete-time estimators *to* the analog sampler would violate the digital processing paradigm in which all of the receiver works in discrete time and the one-way interface from the analog world is the A/D converter. In other words: the structure in Figure 12.25 is not a truly digital PLL loop; to implement a completely digital PLL structure, the adjustment of the sampling instants must be performed in discrete time via the use of a programmable fractional delay.

Let us start with the case of a simple time-lag compensation for a continuous-time signal $x(t)$. Of the total delay t_d, we assume that the bulk delay has been correctly estimated so that the only necessary compensation is that of a fractional delay τ, with $|\tau| \leq 1/2$. From the available sampled signal $x[n] = x(nT_s)$ we want to obtain the signal

$$x_\tau[n] = x(nT_s + \tau T_s) \tag{12.37}$$

using discrete-time processing only. Since we will be operating in discrete time, we can assume $T_s = 1$ with no loss of generality and so we can write simply:

$$x_\tau[n] = x(n+\tau)$$

We know from Section 9.7.2 that the "ideal" way to obtain $x_\tau[n]$ from $x[n]$ is to use a fractional delay filter:

$$x_\tau[n] = d_\tau[n] * x[n]$$

where $D_\tau(e^{j\omega}) = e^{j\omega\tau}$. We have seen that the problem with this approach is that $D_\tau(e^{j\omega})$ is an ideal filter, and that its impulse response is a sinc, whose slow decay leads to very poor FIR approximations. An alternative approach relies on the *local interpolation* techniques we saw in Section 9.4.2. Suppose $2N+1$ samples of $x[n]$ are available around the index $n = n_0$; we could easily build a local continuous-time interpolation around n_0 as

$$\hat{x}(n_0; t) = \sum_{k=-N}^{N} x[n_0 - k]\, L_k^{(N)}(t) \tag{12.38}$$

where $L_k^{(N)}(t)$ is the k-th Lagrange polynomial of order $2N$ defined in (9.14). The approximation

$$\hat{x}(n_0;t) \approx x(n_0+t)$$

is good, at least, over a unit-size interval centered around n_0, i.e. for $|t| \le 1/2$ and therefore we can obtain the fractionally delayed signal as

$$x_\tau[n_0] = \hat{x}_\tau(n_0;\tau) \tag{12.39}$$

as shown in Figure 12.27 for $N=1$ (i.e. for a three-point local interpolation).

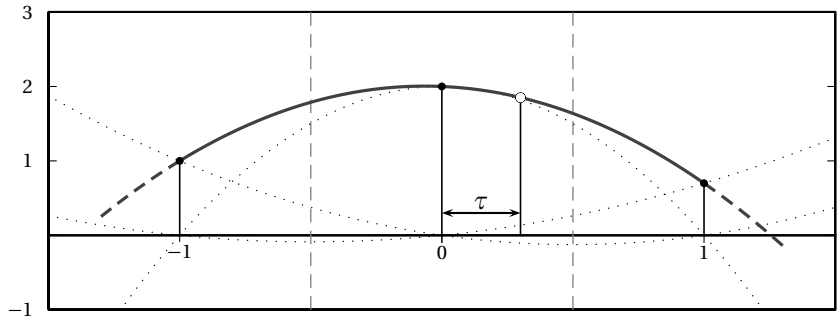

Figure 12.27 Local interpolation around $n_0 = 0$ and $T_s = 1$ for time lag compensation. The Lagrange polynomial components are plotted as dotted lines. The dashed lines delimit the good approximation interval. The white dot is the fractionally delayed sample for $n = n_0$.

Equation (12.39) can be rewritten in general as

$$x_\tau[n] = \sum_{k=-N}^{N} x[n-k]\, L_k^{(N)}(\tau) = \hat{d}_\tau[n] * x[n] \tag{12.40}$$

which is the convolution of the input signal with a $(2N+1)$-tap FIR whose coefficients are the values of the $2N+1$ Lagrange polynomials of order $2N$ *computed in* $t=\tau$. For instance, for the above three-point interpolator, we have

$$\begin{aligned}
\hat{d}_\tau[-1] &= \tau\frac{\tau-1}{2} \\
\hat{d}_\tau[0] &= -(\tau+1)(\tau-1) \\
\hat{d}_\tau[1] &= \tau\frac{\tau+1}{2}
\end{aligned}$$

The resulting FIR interpolators are expressed in noncausal form purely out of convenience; in practical implementations an additional delay would make the whole processing chain causal.

The fact that the coefficients $\hat{d}_\tau[n]$ are expressed in closed form as a polynomial function of τ makes it possible to efficiently compensate for a time-varying delay by recomputing the FIR taps on the fly. This is actually the case when we need to compensate for a frequency drift between transmitter and receiver, i.e. we need to *resample* the input signal. Suppose that, by using the techniques in the previous Section, we have estimated that the actual sampling frequency is either higher or lower than the nominal sampling frequency by a factor β which is very close to 1. From the available samples $x[n] = x(nT_s)$ we want to obtain the signal

$$x_\beta[n] = x\left(\frac{nT_s}{\beta}\right)$$

using discrete-time processing only. With a simple algebraic manipulation we can write

$$x_\beta[n] = x\left(nT_s - n\frac{1-\beta}{\beta}T_s\right) = x(nT_s - n\tau T_s) \tag{12.41}$$

Here, we are in a situation similar to that of Equation (12.37) but in this case the delay term is linearly increasing with n. Again, we can assume $T_s = 1$ with no loss of generality and remark that, in general, β is very close to one so that it is

$$\tau = \frac{1-\beta}{\beta} \approx 0$$

Nonetheless, regardless of how small τ is, at one point the delay term $n\tau$ will fall outside of the good approximation interval provided by the local interpolation scheme. For this, a more elaborate strategy is put in place, which we can describe with the help of Figure 12.28 in which $\beta = 0.82$ and therefore $\tau \approx 0.22$:

1. We assume initial synchronism, so that $x_\beta[0] = x(0)$.

2. For $n = 1$ and $n = 2$, $0 < n\tau < 1/2$; therefore $x_\beta[1] = x_\tau[1]$ and $x_\beta[2] = x_{2\tau}[2]$ can be computed using (12.40).

3. For $n = 3$, $3\tau > 1/2$; therefore *we skip* $x[3]$ and calculate $x_\beta[3]$ from a local interpolation around $x[4]$: $x_\beta[3] = x'_\tau[4]$ with $\tau' = 1 - 3\tau$ since $|\tau'| < 1/2$.

4. For $n = 4$, again, the delay 4τ makes $x_\beta[4]$ closer to $x[5]$, with an offset of $\tau' = 1 - 4\tau$ so that $|\tau'| < 1/2$; therefore $x_\beta[4] = x'_\tau[5]$.

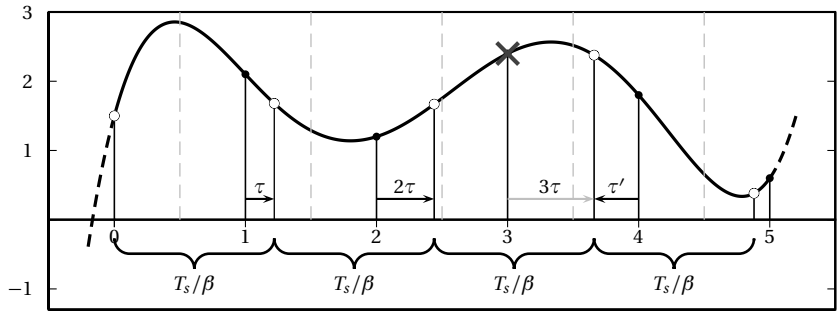

Figure 12.28 Sampling frequency reduction ($T_s = 1$, $\beta = 0.82$) in the discrete-time domain using a programmable fractional delay; white dots represent the resampled signal.

In general the resampled signal can be computed for all n using (12.40) as

$$x_\beta[n] = x_{\tau_n}[n+\gamma_n] \tag{12.42}$$

where

$$\tau_n = \text{frac}\left(n\tau + \frac{1}{2}\right) - \frac{1}{2} \tag{12.43}$$

$$\gamma_n = \left\lfloor n\tau + \frac{1}{2} \right\rfloor \tag{12.44}$$

It is evident that, τ_n is the quantity $n\tau$ "wrapped" over the $[-1/2, 1/2]$ interval[6] while γ_n is the number of samples skipped so far. Practical algorithms compute τ_n and $(n+\gamma_n)$ incrementally.

Figure 12.29 shows an example in which the sampling frequency is too slow and the discrete-time signal must be resampled at a higher rate. In the figure, $\beta = 1.28$ so that $\tau \approx -0.22$; the first resampling steps are:

1. We assume initial synchronism, so that $x_\beta[0] = x(0)$.

2. For $n = 1$ and $n = 2$, $-1/2 < n\tau$; therefore $x_\beta[1] = x_\tau[1]$ and $x_\beta[2] = x_{2\tau}[2]$ can be computed using (12.40).

3. For $n = 3$, $3\tau < -1/2$; therefore *we fall back on* $x[2]$ and calculate $x_\beta[3]$ from a local interpolation around $x[2]$ once again: $x_\beta[3] = x'_\tau[2]$ with $\tau' = 1 + 3\tau$ and $|\tau'| < 1/2$.

[6] The frac function extracts the fractionary part of a number and is defined as $\text{frac}(x) = x - \lfloor x \rfloor$.

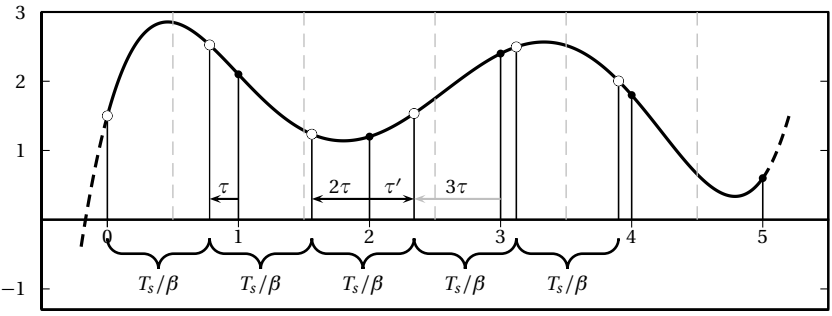

Figure 12.29 Sampling frequency increase ($T_s = 1$, $\beta = 1.28$) in the discrete-time domain using a programmable fractional delay.

4. For $n = 4$, the delay 4τ makes $x_\beta[4]$ closer to $x[3]$, with an offset of $\tau' = 1 + 4\tau$ so that $|\tau'| < 1/2$; therefore $x_\beta[4] = x'_\tau[5]$.

In general the resampled signal can be computed for all n using (12.40) as

$$x_\beta[n] = x_{\tau_n}[n - \gamma_n] \tag{12.45}$$

where τ_n and γ_n are as in (12.43) and (12.44).

Nonlinearity. The programmable delay is inserted in a PLL-like loop as in Figure 12.30 where $\mathscr{S}\{\cdot\}$ is a processing block which extracts a suitable sinusoidal component from the baseband signal.[7] Hypothetically, if the transmitter inserted an explicit sinusoidal component $p[n]$ in the baseband with a frequency equal to the baud rate and with zero phase offset with re-

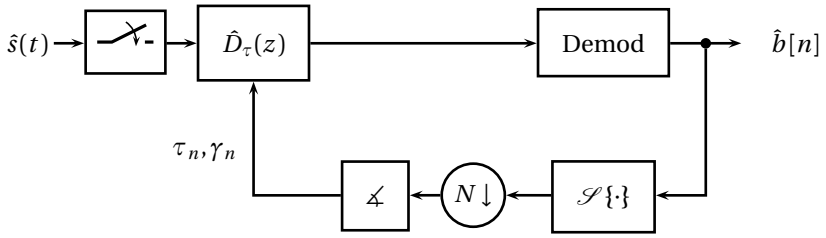

Figure 12.30 A *truly* digital PLL for timing recovery.

[7] Note that timing recovery is performed in the baseband signal since in baseband everything is slower and therefore easier to track; we also assume that equalization and carrier recovery proceed independently and converge before timing recovery is attempted.

spect to the symbol times, then this signal could be used for synchronism; indeed, from

$$p[n] = \sin\left(\frac{2\pi f_b}{F_s}\, n\right) = \sin\left(\frac{2\pi}{K}\, n\right)$$

we would have $p_{KD}[n] = 0$. If this component was present in the signal, then the block $\mathscr{S}\{\cdot\}$ would be a simple resonator $\mathscr{R}$ with peak frequencies at $\omega = \pm 2\pi/K$, as described in Section 7.3.1.

Now, consider more in detail the baseband signal $b[n]$ in (12.4); if we always transmitted the same symbol a, then $b[n] = a\sum_i g[n - iK]$ would be a periodic signal with period K and, therefore, it would contain a strong spectral line at $2\pi/K$ which we could use for synchronism. Unfortunately, since the symbol sequence $a[n]$ is a balanced stochastic sequence we have that:

$$\mathrm{E}\big[b[n]\big] = \mathrm{E}[a[n]] \sum_i g[n - iK] = 0 \tag{12.46}$$

and so, even on average, no periodic pattern emerges.[8] The way around this impasse is to use a fantastic "trick" which dates back to the old days of analog radio receivers, i.e. we process the signal through a *nonlinearity* which acts like a diode. We can use, for instance, the square magnitude operator; if we process $b[n]$ with this nonlinearity, it will be

$$\mathrm{E}\Big[\big|b[n]\big|^2\Big] = \sum_h \sum_i \mathrm{E}\big[a[h]a^*[i]\big]\, g[n - hK]\, g[n - iK] \tag{12.47}$$

Since we have assumed that $a[n]$ is an uncorrelated i.i.d. sequence,

$$\mathrm{E}\big[a[h]a^*[i]\big] = \sigma_a^2\, \delta[h - i]$$

and, therefore,

$$\mathrm{E}\big[\mathscr{S}\{b[n]\}\big] = \sigma_a^2 \sum_i \big(g[n - iK]\big)^2 \tag{12.48}$$

The last term in the above equation is periodic with period K and this means that, on average, the squared signal contains a periodic component at the frequency we need. By filtering the squared signal through the resonator above (i.e. by setting $\mathscr{S}\{x[n]\} = \mathscr{R}\{\big|x[n]\big|^2\}$), we obtain a sinusoidal component suitable for use by the PLL.

[8] Again, a rigorous treatment of the topic would require the introduction of cyclostationary analysis; here we simply point to the intuition and refer to the bibliography for a more thorough derivation.

Further Reading

Of course there are a good number of books on communications, which cover the material necessary for analyzing and designing a communication system like the modem studied in this Chapter. A classic book providing both insight and tools is J. M. Wozencraff and I. M. Jacobs's *Principles of Communication Engineering* (Waveland Press, 1990); despite its age, it is still relevant. More recent books include *Digital Communications* (McGraw Hill, 2000) by J. G. Proakis; *Digital Communications: Fundamentals and Applications* (Prentice Hall, 2001) by B. Sklar, and *Digital Communication* (Kluwer, 2004) by E. A. Lee and D. G. Messerschmitt.

Exercises

Exercise 12.1: Raised cosine. Why is the raised cosine an ideal filter? What type of linear phase FIR would you use for its approximation?

Exercise 12.2: Digital resampling. Use the programmable digital delay of Section 12.4.2 to design an *exact* sampling rate converter from CD to DVD audio (Sect. 11.3). How many different filters $h_\tau[n]$ are needed in total? Does this number depend on the length of the local interpolator?

Exercise 12.3: A quick design. Assume the specifications for a given telephone line are $f_{\min} = 300$ Hz, $f_{\max} = 3600$ Hz, and a SNR of at least 28 dB. Design a set of operational parameters for a modem transmitting on this line (baud rate, carrier frequency, constallation size). How many bits per second can you transmit?

Exercise 12.4: The shape of a constellation. One of the reasons for designing non-regular constellations, or constellation on lattices, different than the upright square grid, is that the energy of the transmitted signal is directly proportional to the parameter σ_α^2 as in (12.10). By arranging the same number of alphabet symbols in a different manner, we can sometimes reduce σ_α^2 and therefore use a larger amplification gain while keeping the total output power constant, which in turn lowers the probability of error. Consider the two 8-point constellations in the Figure 12.12 and Figure 12.13 and compute their intrinsic power σ_α^2 for uniform symbol distributions. What do you notice?