CHAPTER 13

Communication systems simulation

13.1 Introduction

Most of the material in the preceding chapters has been concerned with the development of equations which can be used to predict the performance of digital communications systems. An obvious example is the error function formulae used to find the probability of bit error for ideal systems assuming zero ISI, matched filter receivers and additive, white, Gaussian noise. Such models are important in that they are simple, give instant results, and provide a series of 'reference points' in terms of the relative (or perhaps potential) performance of quite different types of system. In addition they allow engineers to develop a quantitative feel for how the performance of systems will vary as their parameters are changed. They also often provide theoretical limits on performance guarding the design engineer against pursuit of the unobtainable.

The principal limitation of such equations arises from the sometimes unrealistic assumptions on which they are based. Filters, for example, do not have rectangular amplitude responses, oscillators are subject to phase noise and frequency drift, carrier recovery circuits do not operate with zero phase error, sampling circuits are prone to timing jitter, etc.

Hardware prototyping during the design of systems avoids the limitation of idealised models in so far as real-world imperfections are present in the prototype. Designing, implementing and testing hardware, however, is expensive and time consuming, and is becoming more so as communications systems increase in sophistication and complexity. This is true to the extent that it would now usually be impossible to prototype all credible solutions to a given communications problem. Computer simulation of communications systems falls into the middle ground between idealised modelling using simple formulae and hardware prototyping. It occupies an intermediate location along all the following axes:

crude – accurate
simple – complex

cheap – expensive
quick – time consuming

In addition computer simulations are often able to account for system non-linearities which are notoriously difficult to model analytically.

Computer simulations of communications systems usually work as follows. The system is broken down into functional blocks each of which can be modelled mathematically by an equation, a rule, an input/output lookup table or in some other way. These subsystems are connected together such that the outputs of some blocks form the input of others and vice versa. An information source is then modelled as a random or pseudo random sequence of bits (see section 13.4). The signal is sampled and the samples fed into the input of the first subsystem. This subsystem then operates on these samples according to its system model and provides modified samples at its output. These samples then become the inputs for the next subsystem, and so on, typically until the samples represent the received, demodulated, information bit stream. Random samples representing noise and/or interference are usually added at various points in the system. Finally the received information bits are compared with the original information source bits to estimate the BER of the entire communications system. Intermediate results, such as the spectrum of the transmitted signal and the pdf of signal plus noise in the receiver, can also be obtained.

The functional definition of some subsystems, e.g. modulators, is easier in the time domain whilst the definition of others (e.g. filters) is easier in the frequency domain. Conversion between time and frequency domains is an operation which may be performed many times when simulations are run. Convolution, multiplication and discrete Fourier transforms are therefore important operations in communications simulation.

The accuracy of a well designed simulation, in the sense of how closely it matches a hardware prototype, depends essentially of the level of detail at which function blocks are defined. The more detailed the model the more accurate might be expected its results. The penalty paid, of course, is in the effort required to develop the model and the computer power needed to simulate the results in a reasonable time. Only an overview of the central issues involved in simulation is presented here. A detailed and comprehensive exposition of communication system simulation is given in [Jeruchim *et al.*]. In essence, however, the normal simulation process can be summarised as:

1. Derivation of adequate models for all input signals (including noise) and subsystems.
2. Conversion, where possible, of signals and subsystem models to their equivalent baseband form.
3. Sampling of all input signals at an adequately high rate.
4. Running of simulations, converting between time and frequency domains as necessary.
5. Use of Monte Carlo or quasi-analytic methods to estimate bit error rates.
6. Conversion of output signals back to passband form if necessary.
7. Display of intermediate signals, spectra, eye diagrams, etc. as required.

Much of the work in the preceding chapters (especially Chapters 2 to 4) has been directed at modelling the signals, noise and subsystems which commonly occur in digital communications systems. This chapter therefore concentrates on steps 2 to 7 above.

13.2 Equivalent complex baseband representations

Consider a microwave LOS communications system operating with a carrier frequency of 6 GHz and a bandwidth of 100 MHz. To simulate this system it might superficially appear that a sampling rate of $2 \times 6,050$ MHz would be necessary. This conclusion is, of course, incorrect as should be apparent from the bandpass sampling theorem discussed in Chapter 5. In fact, the most convenient way of representing this narrowband system for simulation purposes is to work with equivalent baseband quantities.

13.2.1 Equivalent baseband signals

A (real) passband signal can be expressed in polar (i.e. amplitude and phase) form by:

$$\begin{aligned} x(t) &= a(t) \cos\,[2\pi f_c t + \phi(t)] \\ &= \Re\left\{ a(t)\, e^{j2\pi f_c t}\, e^{j\phi(t)} \right\} \end{aligned} \tag{13.1(a)}$$

or, alternatively, in Cartesian (i.e. inphase and quadrature) form by:

$$x(t) = x_I(t) \cos 2\pi f_c t - x_Q(t) \sin 2\pi f_c t \tag{13.1(b)}$$

The corresponding complex baseband (or lowpass) signal is defined as:

$$x_{LP}(t) = a(t)\, e^{j\phi(t)} \tag{13.2(a)}$$

or:

$$x_{LP}(t) = x_I(t) + j\, x_Q(t) \tag{13.2(b)}$$

(Multiplying equation (13.2(b)) by $e^{j2\pi f_c t}$ and taking the real part demonstrates the correctness of the + sign here.) $x_{LP}(t)$ is sometimes called the *complex envelope* of $x(t)$. The baseband nature of $x_{LP}(t)$ is now obvious and the modest sampling rate required to satisfy Nyquist's theorem is correspondingly obvious. Notice that the transformation from the real passband signal of equation (13.1(a)) to the complex baseband signal of equation (13.2(a)) can be viewed as a two step process, i.e.:

1. $x(t)$ is made cisoidal (or *analytic*) by adding $j\, a(t) \sin[2\pi f_c t + \phi(t)]$.
2. The carrier is suppressed by dividing by $e^{j2\pi f_c t}$.

Since $a(t) \sin[2\pi f_c t + \phi(t)]$ is derived from $x(t)$ by shifting all positive frequency components by +90° and all negative frequency components −90°, step 1 corresponds to adding $j\hat{x}(t)$ to $x(t)$ where ˆ denotes the (time domain) Hilbert transform (see section 4.5). The relationship between $x(t)$ and $x_{LP}(t)$ can therefore be summarised as:

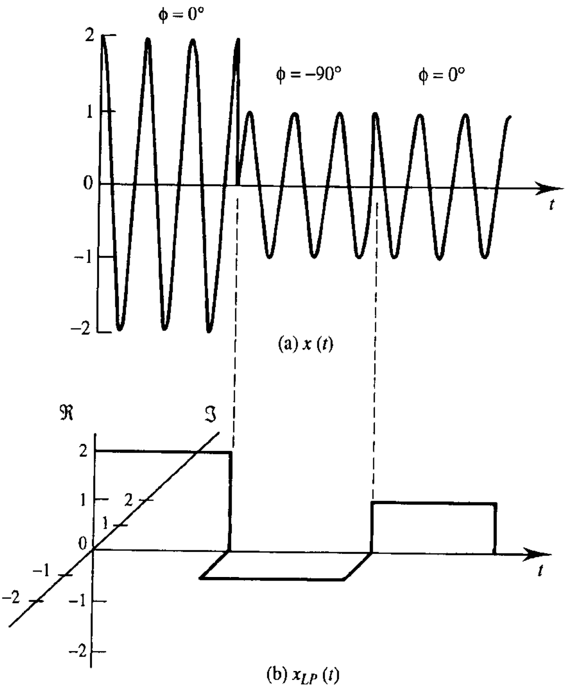


Figure 13.1 *Real passband signal $x(t)$ and its complex envelope $x_{LP}(t)$.*

$$x_{LP}(t) = [x(t) + j\,\hat{x}(t)]\,e^{-j2\pi f_c t} \tag{13.3}$$

$x(t)$ and $x_{LP}(t)$ are shown schematically, for an APK signal, in Figure 13.1.

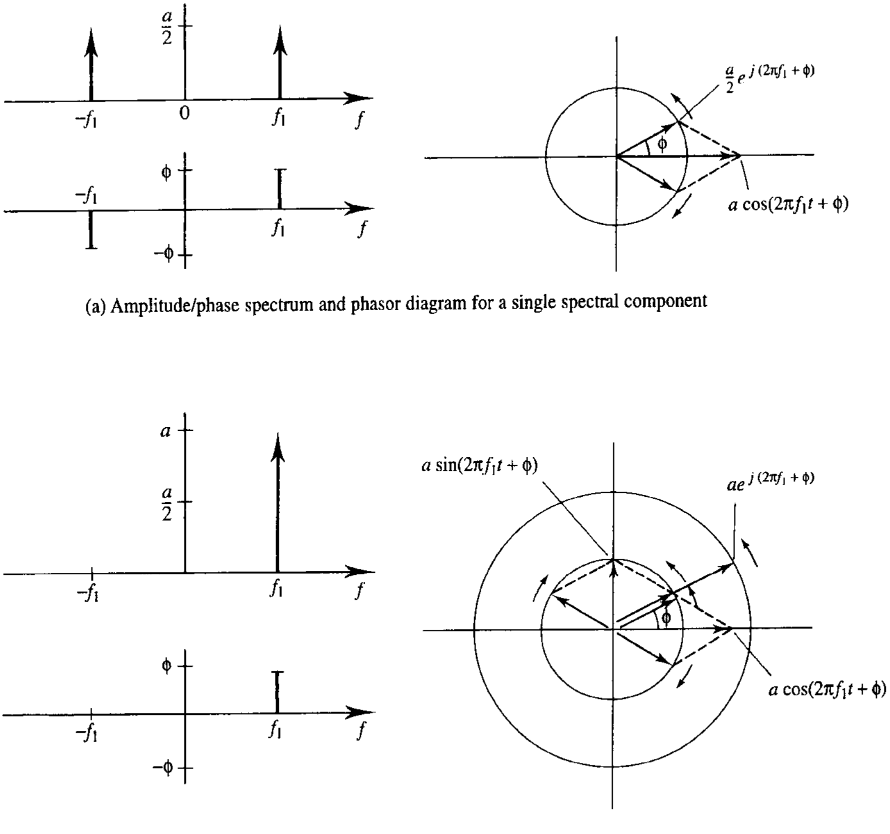
The spectrum, $X_{LP}(f)$, of $x_{LP}(t)$ can be found by applying the corresponding frequency domain steps to the spectrum, $X(f)$, of $x(t)$, i.e.:

1. $X(f)$ has its negative frequency components suppressed and its positive frequency components doubled. This is demonstrated using phasor diagrams, for a sinusoidal signal, in Figure 13.2.
2. The (doubled) positive frequency components are shifted to the left by f_c Hz.

Step 1 is more formally expressed as the addition to $X(f)$ of $j\hat{X}(f) = j[-j\text{sgn}(f)X(f)] = \text{sgn}(f)X(f)$. Step 2 follows from the Fourier transform frequency translation theorem. Figure 13.3 shows the relationship between $X(f)$ and $X_{LP}(f)$. Notice that the spectrum of the complex envelope does not have the Hermitian symmetry characteristic of real signals. Steps 1 and 2 together can be summarised by:

$$X_{LP}(f) = 2X(f + f_c)\,u(f + f_c) \tag{13.4}$$

The first factor on the RHS of equation (13.4) doubles the spectral components, the second moves the entire spectrum to the left by f_c Hz and the third factor suppresses all spectral components to the left of $-f_c$ Hz, Figure 13.4.



(a) Amplitude/phase spectrum and phasor diagram for a single spectral component

(b) Cancellation of negative frequency phasors, reinforcement of positive frequency phasors

Figure 13.2 *Phasor diagram demonstration of the equivalence between (a) the addition of an imaginary quadrature version of a signal and (b) the suppression of the negative frequency components plus a doubling of the positive frequency components.*

13.2.2 Equivalent baseband systems

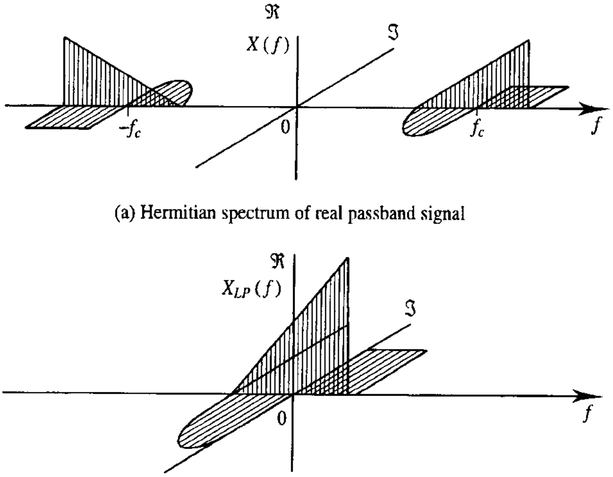
Equivalent baseband representations of passband systems can be found in the same way as for signals. A filter with a passband (Hermitian) frequency response $H(f)$ (Figure 13.5(a) to (c)), and (real) impulse response $h(t)$, has an equivalent baseband frequency response (Figure 13.5(d) to (f)):

$$H_{LP}(f) = H(f + f_c)\, u(f + f_c) \tag{13.5(a)}$$

and baseband impulse response:

$$h_{LP}(t) = \tfrac{1}{2}\,[h(t) + j\,\hat{h}(t)]\, e^{-j2\pi f_c t} \tag{13.5(b)}$$

Alternatively the complex baseband impulse response can be expressed in terms of its



(a) Hermitian spectrum of real passband signal

(b) Non-Hermitian spectrum of equivalent complex baseband signal

Figure 13.3 *Spectra of real passband signal and its complex envelope.*

inphase and quadrature components, i.e.:

$$h_{LP}(t) = \tfrac{1}{2}\,[h_I(t) + j\,h_Q(t)] \tag{13.6}$$

where the (real) passband response is:

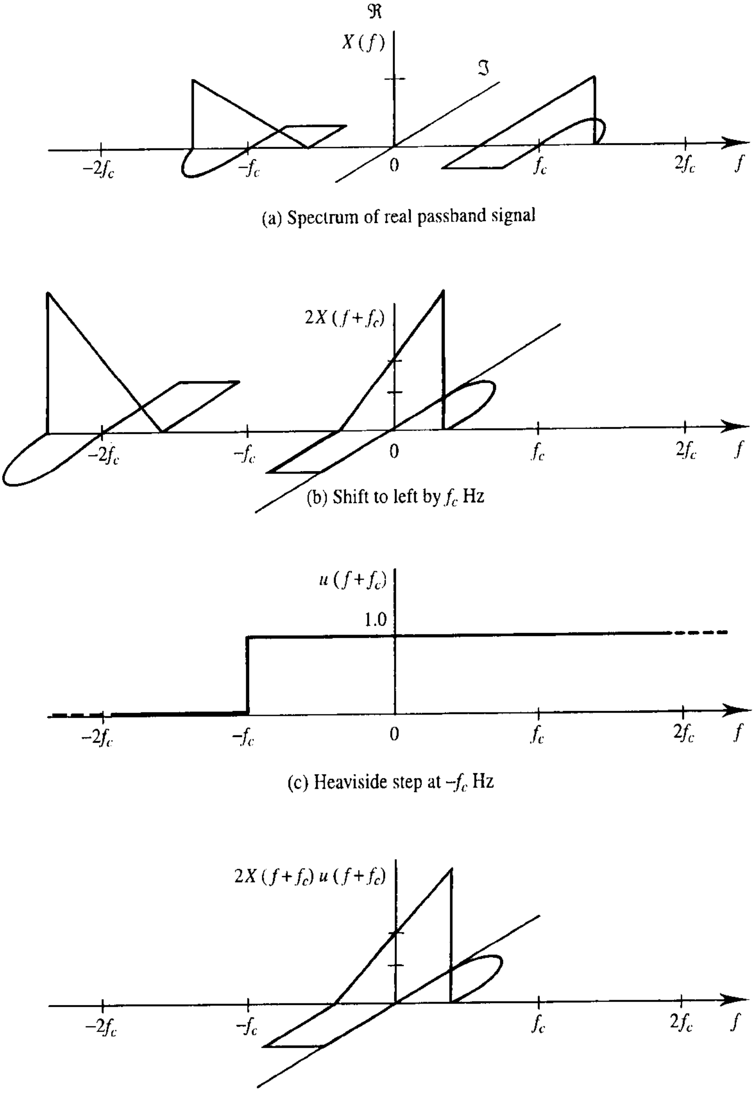
$$h(t) = h_I(t)\cos 2\pi f_c t - h_Q(t)\sin 2\pi f_c t \tag{13.7}$$

13.2.3 Equivalent baseband system output

The output of an equivalent lowpass linear system when excited by an equivalent lowpass signal is found, in the time domain, using convolution in the usual way (but taking care to convolve both real and imaginary components) and represents the equivalent lowpass output of the system, $y_{LP}(t)$, i.e.:

$$\begin{aligned} y_{LP}(t) &= h_{LP}(t) * x_{LP}(t) \\ &= \tfrac{1}{2}\,[h_I(t) + j\,h_Q(t)] * [x_I(t) + j\,x_Q(t)] \qquad (13.8) \end{aligned}$$

Notice that when compared with the definition of equivalent baseband signals (equations (13.4), (13.3) and (13.2(b))) the equivalent baseband system definitions (equations (13.5(a)), (13.5(b)) and (13.6)) are smaller by a factor of ½. This is to avoid, for example, the baseband equivalent frequency response of a lossless passband filter having a voltage gain of 2.0 in its passband. (Many authors make no such distinction between the definitions of baseband equivalent signals and systems in which case the factor of ½ usually appears in the *definition* of equivalent baseband convolution.) Recognising that the equivalent baseband system output signal, $y_{LP}(t)$, can be expressed as inphase and



(a) Spectrum of real passband signal

(b) Shift to left by f_c Hz

(c) Heaviside step at $-f_c$ Hz

(d) Spectrum of complex equivalent baseband signal

Figure 13.4 *Relationship between passband and equivalent low-pass spectra.*

quadrature components of the passband output signal, $y(t)$, i.e.:

$$y_{LP}(t) = y_I(t) + j\, y_Q(t) \tag{13.9}$$

where:

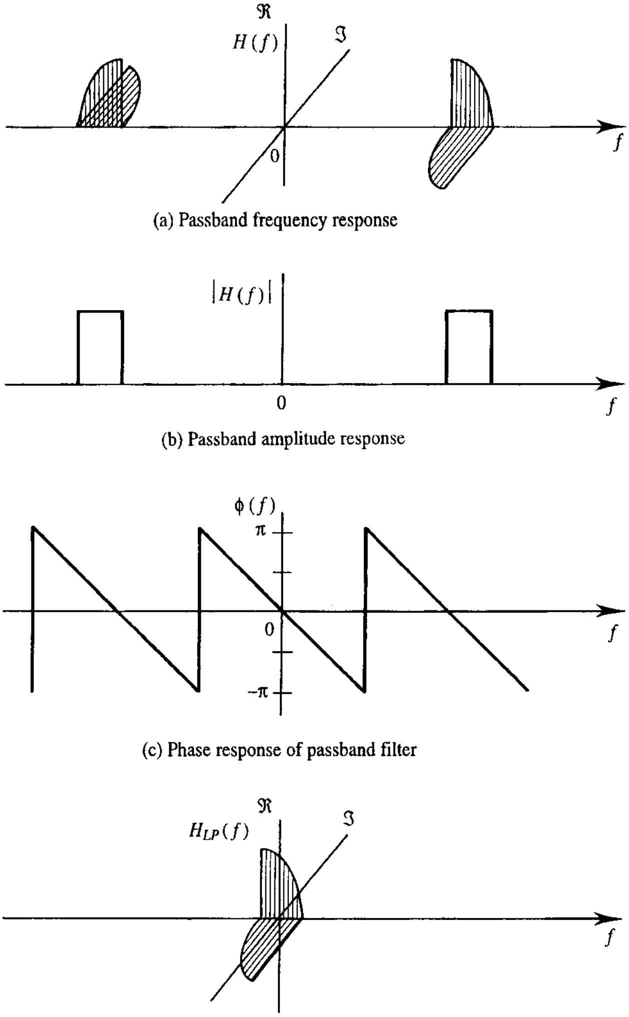
$$y(t) = y_I(t) \cos 2\pi f_c t - y_Q(t) \sin 2\pi f_c t \tag{13.10}$$

and equating real and imaginary parts in equations (13.8) and (13.9) gives:

$$y_I(t) = \tfrac{1}{2}\,[h_I(t) * x_I(t) - h_Q(t) * x_Q(t)] \qquad (13.11(a))$$

$$y_Q(t) = \tfrac{1}{2}\,[h_Q(t) * x_I(t) + h_I(t) * x_Q(t)] \qquad (13.11(b))$$

These operations are illustrated schematically in Figure 13.6. Equations (13.10) and (13.11) thus give the passband output of a system directly in terms of the inphase and quadrature baseband components of its passband input and impulse response. $y(t)$ can



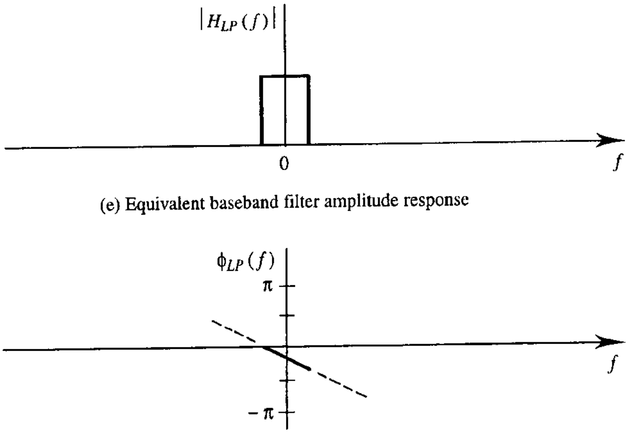
(a) Passband frequency response

(b) Passband amplitude response

(c) Phase response of passband filter

(d) Equivalent baseband filter frequency response

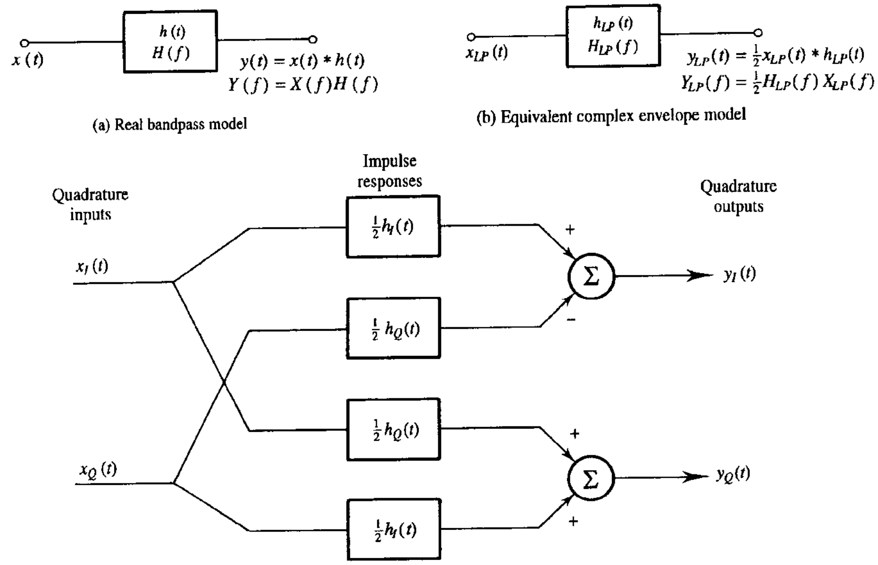
Figure 13.5 *Passband and equivalent lowpass frequency, amplitude and phase responses.*



(e) Equivalent baseband filter amplitude response

(f) Equivalent baseband filter phase response

Figure 13.5-ctd. *Equivalent lowpass amplitude and phase responses.*



(a) Real bandpass model

(b) Equivalent complex envelope model

(c) Inphase and quadrature baseband component model

Figure 13.6 *Convolution of inputs and impulse responses with equivalent baseband operations.*

also be found from $y_{LP}(t)$ by reversing the steps used in going from equations (13.1(a)) to (13.2(a)), i.e.:

$$y(t) = \Re\left\{ y_{LP}(t)\, e^{j2\pi f_c t} \right\}$$

$$= ½\left[y_{LP}(t)\, e^{j2\pi f_c t} + y_{LP}^*(t)\, e^{-j2\pi f_c t}\right] \tag{13.12(a)}$$

Alternatively, if $Y_{LP}(f)$ has been found from:

$$Y_{LP}(f) = H_{LP}(f)\, X_{LP}(f)$$

then $Y(f)$ can be obtained using the equivalent frequency domain quantities and operations of equation (13.12(a)) (see Figure 13.7), i.e.:

$$Y(f) = \text{Hermitian}\{Y_{LP}\ (f - f_c)\} \tag{13.12(b)}$$

$$= \frac{Y_{LP}\ (f - f_c) + Y_{LP}^*\ (-f - f_c)}{2}$$

(That any function, in this case $Y_{LP}(f - f_c)$, can be split into Hermitian and anti-Hermitian parts is easily demonstrated as follows:

$$X(f) = \frac{X(f) + X^*(-f)}{2} + \frac{X(f) - X^*(-f)}{2} \tag{13.13}$$

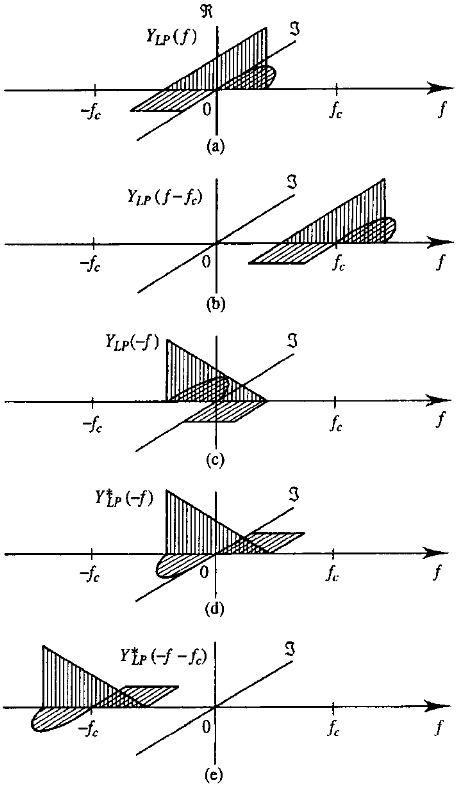


Figure 13.7 *Obtaining the spectrum of real passband signal, (b) plus (e), from the spectrum of an equivalent complex baseband signal, (a).*

The first term has even real part and odd imaginary part and is therefore Hermitian. The second term has odd real part and even imaginary part and is therefore anti-Hermitian.)

13.2.4 Equivalent baseband noise

Equivalent baseband noise, $n_{LP}(t)$, can be modelled in exactly the same way as an equivalent baseband signal, i.e.:

$$\begin{aligned} n_{LP}(t) &= r(t)\, e^{j\theta(t)} \\ &= n_I(t) + j\, n_Q(t) \end{aligned} \tag{13.14}$$

where the passband noise process is:

$$\begin{aligned} n(t) &= r(t)\, e^{j\theta(t)}\, e^{j2\pi f_c t} \\ &= n_I(t) \cos 2\pi f_c t - n_Q(t) \sin 2\pi f_c t \end{aligned} \tag{13.15}$$

Figure 13.8 illustrates the relationship between the passband and equivalent baseband processes in time and phase domains. For the special, but very important, case of a strict-sense, zero mean, Gaussian, narrowband noise process the properties of the baseband processes $n_I(t)$ and $n_Q(t)$ are summarised in Table 13.1.

Most of these properties are intuitively reasonable and are, therefore, not proved here. (Proofs can be found in [Taub and Schilling].) One anti-intuitive aspect of the equal variance property, however, is that each of the quadrature baseband processes alone contains the same power (i.e. has the same variance) as the passband noise process. (To the authors, at least, intuition would suggest that each of the baseband process would contain half the power of the passband process.) This problem is easily resolved, however, by considering the power represented by equation (13.15), i.e.:

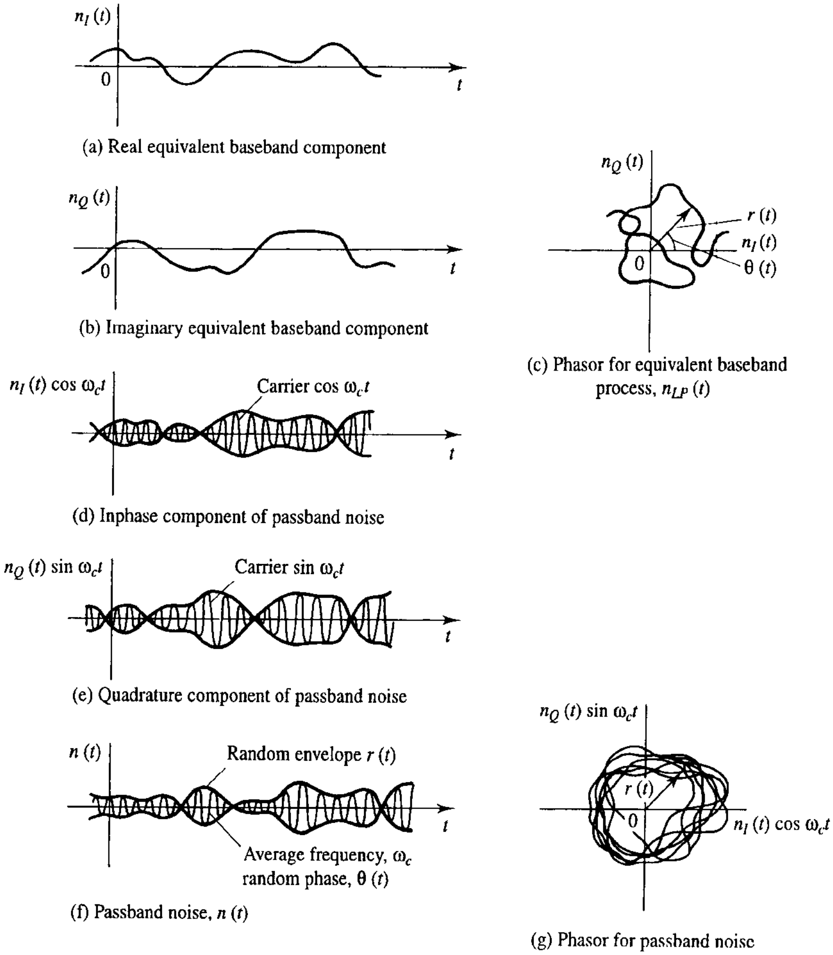
$$\begin{aligned} \langle n^2(t) \rangle &= \langle [n_I(t) \cos 2\pi f_c t - n_Q(t) \sin 2\pi f_c t]^2 \rangle \\ &= ½\langle n_I^2(t) \rangle + ½\langle n_Q^2(t) \rangle \end{aligned} \tag{13.16(a)}$$

And since $\langle n_I^2(t) \rangle = \langle n_Q^2(t) \rangle$ (acceptable on intuitive grounds) then:

$$\langle n^2(t) \rangle = \langle n_I^2(t) \rangle = \langle n_Q^2(t) \rangle \tag{13.16(b)}$$

Table 13.1 *Properties of equivalent baseband Gaussian noise quadrature processes.*

Property	*Definition*
Zero mean	$\langle n_I(t) \rangle = \langle n_Q(t) \rangle = 0$
Equal variance	$\langle n_I^2(t) \rangle = \langle n_Q^2(t) \rangle = \langle n^2(t) \rangle = \sigma^2$
Zero correlation	$\langle n_I(t)\, n_Q(t) \rangle = 0$
Gaussian quad. components	$p(n_I) = p(n_Q) = [1/(\sqrt{2\pi}\,\sigma)]\, e^{-\frac{n_x}{2\sigma^2}}$
Rayleigh amplitude	$p(r) = (r/\sigma^2)\, e^{-r^2/2\sigma^2}, \quad r \geq 0$
Uniform phase	$p(\theta) = 1/(2\pi), \quad \lvert\theta\rvert < \pi$



(a) Real equivalent baseband component

(b) Imaginary equivalent baseband component

(c) Phasor for equivalent baseband process, $n_{LP}(t)$

(d) Inphase component of passband noise

(e) Quadrature component of passband noise

(f) Passband noise, $n(t)$

(g) Phasor for passband noise process, $n(t)$

Figure 13.8 *Schematic illustration of passband and equivalent baseband noise processes with corresponding phasor trajectories.*

Figure 13.9 shows, in a systems context, how the passband process $n(t)$ could be generated from the baseband processes $n_I(t)$ and $n_Q(t)$. Notice that the power in each quadrature leg is halved after multiplication with the carrier.

Since noise processes do not have a well defined voltage spectrum (preventing equation (13.4) from being used to find an equivalent baseband spectrum) the (power) spectral (density) description of $n_I(t)$ and $n_Q(t)$ is found by translating the positive frequency components of $G_n(f)$ down by f_c Hz, translating the negative frequencies up by f_c Hz, and adding, i.e.:

$$G_{n_I}(f) = G_{n_Q}(f)$$

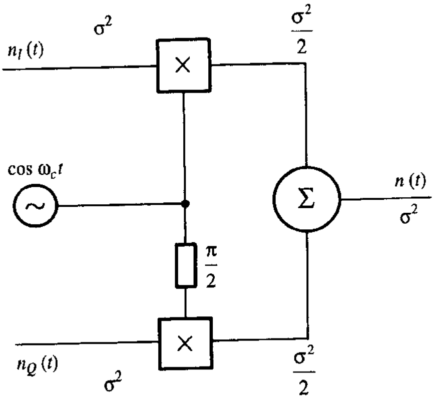


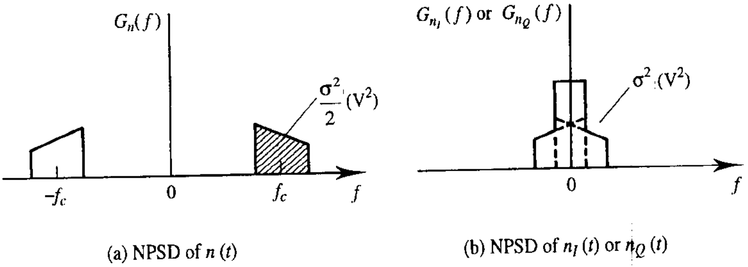
Figure 13.9 *Relationship between baseband quadrature noise components and the passband Gaussian noise process.*

$$= G_n(f + f_c)\, u(f + f_c) + G_n(f - f_c)\, u(-f + f_c) \qquad (13.16(c))$$

The relationship between the PSD of $n(t)$ and that of $n_I(t)$ and $n_Q(t)$ is illustrated in Figure 13.10.

13.3 Sampling and quantisation

Sampling and quantisation, as they affect communications systems generally, have been discussed in Chapter 5. Here these topics are re-examined in the particular context of simulation.



(a) NPSD of $n(t)$

(b) NPSD of $n_I(t)$ or $n_Q(t)$

Figure 13.10 *Relationship between PSD of each equivalent baseband quadrature component of $n(t)$ and PSD of $n(t)$.*

13.3.1 Sampling equivalent baseband signals

Nyquist's sampling theorem, if correctly interpreted, can be applied to any process, including an equivalent, complex, baseband process. In this case the baseband signal has a conventionally defined bandwidth which is only half the bandwidth, B, of the real passband signal, Figure 13.11. Thus a straightforward application of Nyquist's theorem gives a minimum sampling rate:

$$f_s \geq 2\,\frac{B}{2} = B \text{ Hz} \tag{13.17}$$

Superficially equation (13.17) looks wrong in that it suggests all the information present in a passband signal with bandwidth B Hz is preserved in only half the number of samples expected. This paradox is resolved by remembering that for a complex baseband signal there will be two real sample values for each sampling instant (i.e. an inphase, or real sample, and a quadrature, or imaginary, sample). The total number of real numbers characterising a given passband signal is therefore the same, whether or not an equivalent baseband representation is used.

Sampling at a rate of f_s Hz defines a simulation bandwidth of $f_s/2$ Hz in the sense that any spectral components which lie within this band will be properly simulated whilst spectral components outside this band will be aliased. The selection of f_s is therefore a compromise between the requirement to keep f_s low enough so that simulation can be carried out in a reasonable time with modest computer resources, and the requirement to keep f_s high enough for aliasing errors to be acceptably low. The aliasing errors are quantified in section 5.3.4 by a signal to distortion ratio (SDR) defined as the ratio of unaliased signal power to aliased signal power, Figure 13.12. SDR is clearly a function of the number of samples per symbol (i.e. f_s/R_s). Table 13.2 shows several corresponding pairs of SDR and f_s/R_s for the (worst) case of an unfiltered (i.e. rectangular pulse) symbol stream.

f_s/R_s is typically selected such that SDR is 10 dB greater than the best SNR to be simulated. For signals with significant pulse shaping the SDR for a given value of f_s/R_s is higher than that shown in Table 13.2. Eight samples per symbol, therefore, may often represent sufficiently rapid sampling for the simulation of realistic systems.

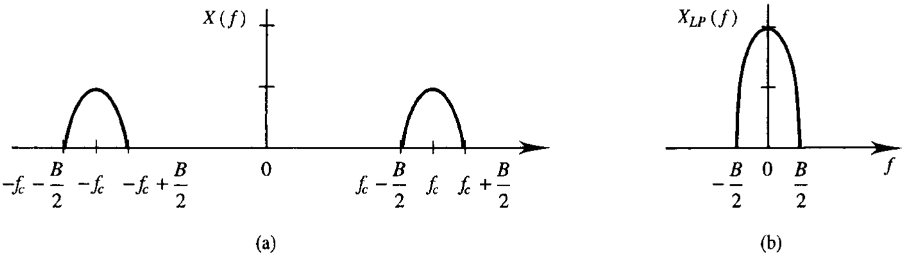
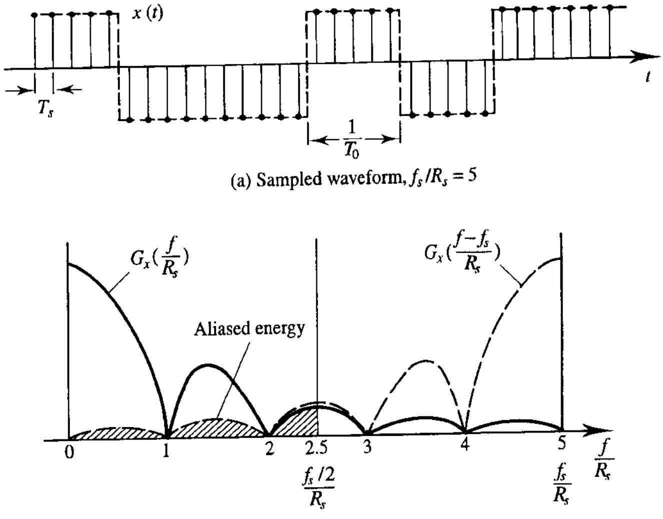


Figure 13.11 *(a) Passband and (b) equivalent baseband frequency spectra.*



(a) Sampled waveform, $f_s/R_s = 5$

(b) Aliasing effect

Figure 13.12 *Sampling of random binary waveform (source: Jeruchim et al., 1984, reproduced with permission of the IEEE).*

Table 13.2 *Worst case SDR for various numbers of samples per symbol. (After Jeruchim et.al.)*

f_s/R_s (samples/symbol)	*SDR* (dB)
4	15.8
8	18.7
10	19.8
16	21.9
20	22.9

13.3.2 Quantisation

Simulation quantisation errors can be equated with the limited precision with which a computer can represent numbers. The rounding errors due to this limited precision effectively add noise to the waveform being simulated. In digital communications the system's ADC itself introduces quantisation error (sections 5.5 and 5.6) quantified by a signal to quantisation noise ratio ($\text{SN}_\text{q}\text{R}$) and given approximately (for linearly quantised voice signals) by $6(n-1)$ dB where n is the number of bits (typically 8) representing each level, equation (5.23). The simulation induced $\text{SN}_\text{q}\text{R}$ will depend in a similar way on the binary word size which the computer uses to represent numbers. This word size would normally be at least 16 bits, typically 32-bits. Whilst this suggests that simulation induced quantisation error will be negligible with respect to system induced quantisation error it may sometimes be the case that simulation quantisation errors accumulate in calculations. For long simulations (perhaps millions of symbols) accumulated errors may become significant. Using double precision arithmetic, where possible, will obviously help in this respect.

13.4 Modelling of signals, noise and systems

One of the strengths of simulation is that it can often be interfaced to real signals and real systems hardware. A real voice signal, for example, might be rapidly sampled and those samples used as the information source in a simulation. Such a simulation might also include measured frequency responses of actual filters and measured input/output characteristics of non-linear amplifiers which are to be incorporated in the final system hardware. (In this respect the distinction between hardware prototyping and software simulation can become blurred.) Nevertheless, it is still a common requirement to model signals, noise and systems using simple (and rather idealised) assumptions. A variety of such models are discussed below.

13.4.1 Random numbers

Statistically independent, random, numbers with a uniform pdf are easily generated by computers. Many algorithms have been proposed, but one [Park and Miller] which appears to have gained wide acceptance for use on machines using 32-bit integer arithmetic is:

$$x(k) = 7^5\, x(k-1)\, (2^{31}-1) \tag{13.18}$$

$x(k)$ represents a sequence of integer numbers drawn from a uniform pdf with minimum and maximum values of 1 and $2^{31}-1$ respectively. To generate a sequence of numbers, $x_u(k)$, with uniform pdf between 0 and 1, equation (13.18) is simply divided by $2^{31}-1$. ($x_u(k)$ never takes on a value of exactly zero since the algorithm would then produce zero values indefinitely.) The initial value of the sequence $x(0) \neq 0$ is called the generator *seed* and is chosen by the user. Strictly speaking only the choice of seed is random since thereafter the sequence is deterministic, repeating periodically. The sequences arising from well designed algorithms such as equation (13.18), however, have many properties in common with truly random sequences and, from an engineering point of view, need not usually be distinguished from them. They are therefore referred to as pseudo-random (or pseudo-noise) sequences. A sequence $y(k)$ with a pdf $p_Y(y)$ can be derived from the sequence $x_u(k)$ using the target cumulative distribution of Y, Figure 13.13. The CD of Y (i.e. $P(Y \leq y)$) is first found (by integrating $p_Y(y)$ if necessary). The values of $x_u(k)$ are then mapped to $y(k)$ according to this curve as shown in Figure 13.13. For simple pdfs this transformation can sometimes be accomplished analytically resulting in a simple formula relating $y(k)$ and $x_u(k)$. Otherwise $P(Y \leq y)$ can be defined by tabulated values and the individual numbers transformed by interpolation.

The method described above is general and could be used to generate numbers with a Gaussian pdf. It is often easier, for this special case, however, to take advantage of the central limit theorem and add several independent, uniformly distributed, random

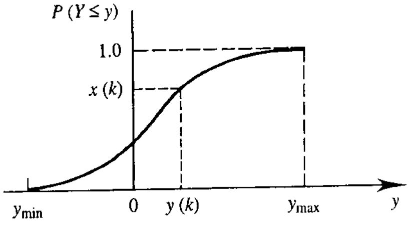


Figure 13.13 *Cumulative distribution used to transform a uniformly distributed random variable to a random variable Y with pdf $p_Y(y)$.*

sequences, i.e.:

$$y(k) = \sum_{i=1}^{N} x_{u,i}(k) - \frac{N}{2} \tag{13.19}$$

Using equation (3.16) we see that $\bar{X}_u$ is 0.5, and subtracting $N/2$ in equation (13.19) ensures that $\bar{Y} = 0$. Furthermore the variance of X_u is 1/12. Choosing $N = 12$ therefore ensures that $\sigma_y^2 = 1.0$ without the need for any additional scaling. (Higher values of N would, of course, improve the accuracy of the resulting Gaussian pdf.)

Correlated random numbers are easily generated from statistically independent random numbers by forming linear combinations of sequence pairs. For example, a random sequence, $y(k)$, with autocorrelation properties, $R_y(\kappa)$, (as defined in section 3.3.3) specified by:

$$R_y(0) = \sigma_Y^2 \tag{13.20(a)}$$

$$R_y(\pm 1) = \alpha\, \sigma_Y^2, \qquad \alpha < 1 \tag{13.20(b)}$$

$$R_y(\pm 2) = \beta\, \sigma_Y^2, \qquad \beta < 1 \tag{13.20(c)}$$

$$R_y(\pm \kappa) = 0, \qquad \kappa > 2 \tag{13.20(d)}$$

$$\overline{y(k)} = 0 \tag{13.20(e)}$$

can be formed from a zero mean, unit variance, statistically independent sequence, $s(k)$, using the linear transform:

$$y(k) = w_1 s(k) + w_2 s(k-1) + w_3 s(k-2) \tag{13.21}$$

Substituting equation (13.21) into equation (13.20(a)) gives:

$$\begin{aligned} \sigma_Y^2 &= \overline{y^2(k)} - [\overline{y(k)}]^2 \\ &= \overline{[w_1 s(k) + w_2 s(k-1) + w_3 s(k-2)]^2} - 0 \end{aligned} \tag{13.22(a)}$$

Since $s(k)$ is a sequence of uncorrelated numbers only terms, in the expansion of equation (13.22(a)), having factors with equal arguments of s give non-zero results. Therefore:

$$\sigma_Y^2 = w_1^2\,\overline{s^2(k)} + w_2^2\overline{s^2(k-1)} + w_3^2\overline{s^2(k-2)} \tag{13.22(b)}$$

And since the variance of $s(k)$ is 1.0 then:

$$\sigma_Y^2 = w_1^2 + w_2^2 + w_3^2 \tag{13.22(c)}$$

Similarly:

$$\begin{aligned}\alpha\,\sigma_Y^2 &= \overline{[w_1 s(k) + w_2 s(k-1) + w_3 s(k-2)]\,[w_1 s(k-1) + w_2 s(k-2) + w_3 s(k-3)]}\\ &= w_2 w_1 \overline{s^2(k-1)} + w_3 w_2 \overline{s^2(k-2)} \end{aligned}\tag{13.23(a)}$$

all other terms being zero. Thus:

$$\alpha\,\sigma_Y^2 = w_2 w_1 + w_3 w_2 \tag{13.23(b)}$$

and:

$$\begin{aligned}\beta\,\sigma_Y^2 &= \overline{[w_1 s(k) + w_2 s(k-1) + w_3 s(k-2)]\,[w_1 s(k-2) + w_2 s(k-3) + w_3 s(k-4)]}\\ &= w_3 w_1 \overline{s^2(k-2)} \end{aligned}\tag{13.24(a)}$$

i.e.:

$$\beta\,\sigma_Y^2 = w_3 w_1 \tag{13.24(b)}$$

Equations (13.22) to (13.24) generalise for a series with non-zero correlation over an N term window to:

$$R_y(\kappa) = \sum_{i=1}^{N-\kappa} w_i w_{\kappa+i} \tag{13.25}$$

Equation (13.20(e)) is automatically satisfied since $\overline{s(k)} = 0$. Equation (13.25) provides N independent equations which are solved simultaneously to give the appropriate weighing factors, $w_1, w_2, \cdots, w_N$.

The general problem of simultaneously obtaining a specified pdf and specified PSD is a difficult one. This is because the linear system represented by equation (13.21) generally changes the pdf in an unpredictable way. The exception to this, of course, is for random sequences with Gaussian pdf which can be filtered to realise a specified PSD without altering its Gaussian characteristic (see section 4.6.3).

13.4.2 Random digital symbol streams

Random digital symbol streams can be easily generated from a set of random numbers as follows. Consider a sequence of independent random numbers, $y(k)$, with the pdf $p_Y(y)$ shown in Figure 13.14. If three symbols represented by three voltage levels $v = 0, 1, 2$ V are required with probabilities of 0.5, 0.25, 0.25 respectively then $p_Y(y)$ is divided into three areas corresponding to those probabilities. This defines thresholds y_1 and y_2 (Figure 13.14). Each random variable sample can then be mapped to a random symbol using the rule:

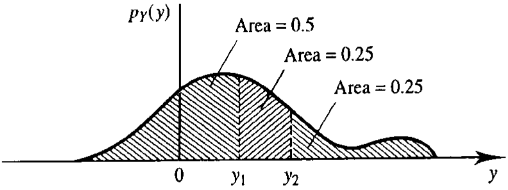


Figure 13.14 *pdf for a random variable used to generate a random digital symbol stream.*

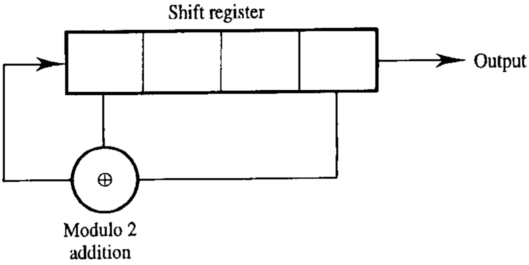


Figure 13.15 *Use of a shift register to generate pseudo-random bit sequences.*

$$v(k) = \begin{cases} 0, & y(k) \le y_1 \\ 1, & y_1 < y(k) \le y_2 \\ 2, & y(k) > y_2 \end{cases} \tag{13.26}$$

As an alternative to using a random number algorithm (such as equation (13.18)) followed by equation (13.26), pseudo-random digital symbol streams can be generated using shift registers with appropriate feedback connections. An example generator for a binary sequence is shown in Figure 13.15. Such generators are simple and easily implemented in hardware as well as software. (This makes them useful as signal sources for field measurements of BER when no secure, i.e. errorless, reference channel is available.) The properties of the pseudo-random bit sequences (PRBSs) generated in this way can be summarised as follows:

1. The sequence is periodic.
2. In each period the number of binary ones is one more than the number of binary zeros.
3. Among runs of consecutive ones and zeros one half of the runs of each kind are of length one, one quarter are of length two, one eighth are of length three, etc. as long as these fractions give meaningful numbers.
4. If a PRBS is compared, term by term, with any cyclical shift of itself the number of agreements differs from the number of disagreements by one.

The autocorrelation function of a K-bit, periodic, PRBS, $z(k)$, is defined by:

$$R_z(\tau) = \frac{1}{K}\sum_{k=1}^{K} z(k)z(k-\tau) \tag{13.27}$$

and is shown in Figure 13.16(b) for a polar sequence with amplitude ±1, Figure 13.16(a). Its voltage and power spectra are shown in Figure 13.16(c) and (d).

The general algorithm implemented by an n-element shift register with modulo 2 feedback, Figure 13.17, can be expressed mathematically as:

$$v(k) = w_{n-1}v(k-1) \oplus w_{n-2}v(k-2) \oplus \cdots \oplus \; w_0 v(k-n) \tag{13.28}$$

where $\oplus$ denotes modulo 2 addition. w_i denotes the feedback weighting (1 or 0) associated with the register's $(i+1)$th element and corresponds in hardware to the

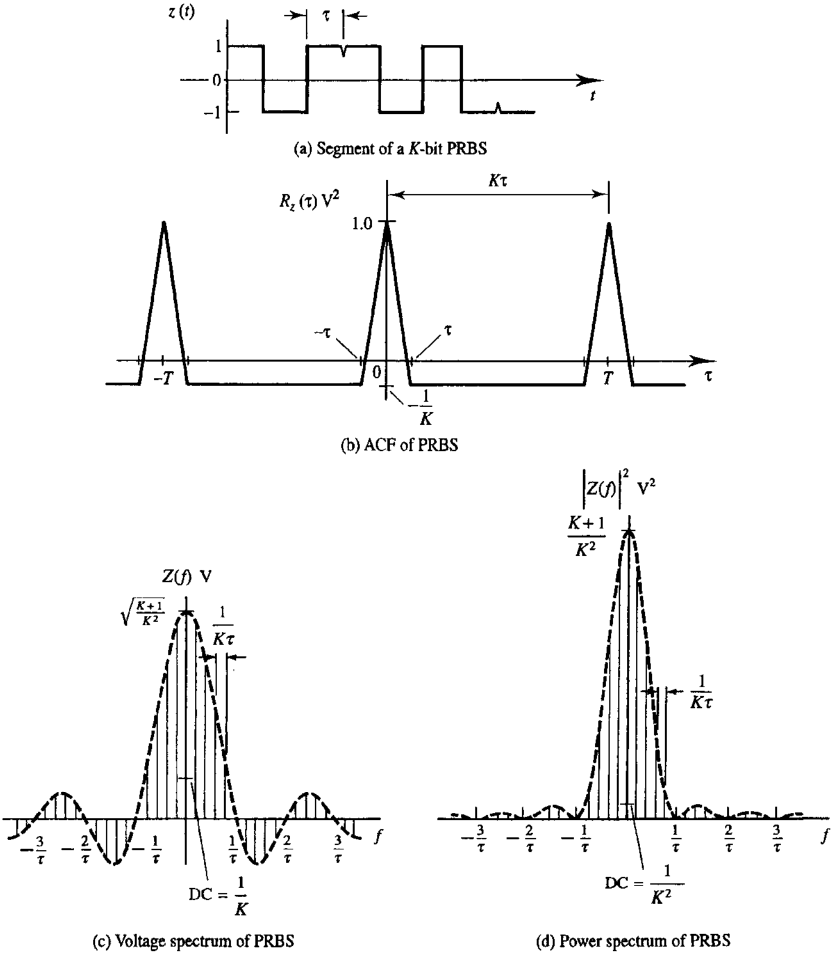


Figure 13.16 *Temporal and spectral properties of a rectangular pulsed polar, NRZ, PRBS.*

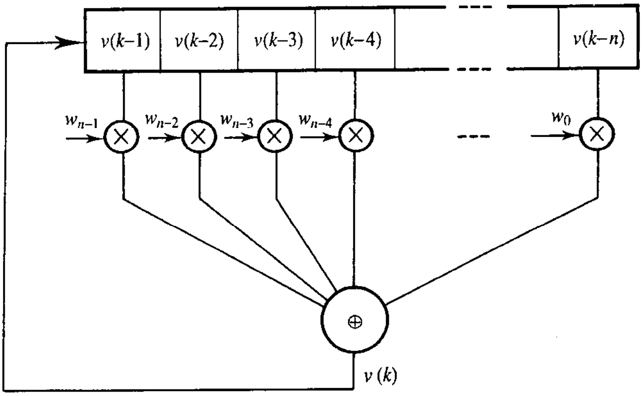


Figure 13.17 *General implementation of PRBS generator using an n-element shift register.*

presence or absence of a connection.

Table 13.3 *PRBS sequence feedback connections.*

Number of shift register elements, n	*Non-zero feedback taps in addition to* w_0	*Sequence length* $K = 2^n - 1$
4	w_3	15
5	w_3	31
6	w_5	63
7	w_6	127
8	w_6, w_5, w_4	255
9	w_5	511
10	w_7	1023
12	w_{11}, w_8, w_6	4095
14	w_{13}, w_8, w_4	16383
16	w_{15}, w_{13}, w_4	65535

The all-zeros state is prohibited in the PRBS, as generated above, since this would result in an endless stream of zeros thereafter. A maximal length PRBS is one in which all possible n-bit patterns (except the all-zeros pattern) occur once and once only in each period. The length of such a sequence is therefore $K = 2^n - 1$ bits. Not all arrangements of feedback connection give maximal length sequences. To establish whether a given set of connections will yield a maximal length sequence the polynomial:

$$f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_{n-1} x^{n-1} + x^n \qquad (13.29)$$

is formed and then checked to see if it is *irreducible*. (Note that $w_0 = 1$, otherwise the final element of the register is redundant.) An irreducible polynomial of degree n is one which cannot be factored as a product of polynomials with degree lower than n. A test for irreducibility is that such a polynomial will not divide exactly (i.e. leaving zero remainder) into $x^P + 1$ for all $P < 2^n + 1$.

Table 13.3 [adapted from Jeruchim *et al.*] gives a selection of shift register feedback connections which yield maximal length PRBSs.

13.4.3 Noise and interference

Noise is modelled, essentially, in the same way as the signals described in section 13.4.2. The following interesting point arises, however, in the modelling of white Gaussian noise. Because a simulation deals only with noise samples then, providing these noise samples are uncorrelated (and being Gaussian, statistically independent) no distinction can be made between any underlying (continuous) noise processes which satisfy:

$$R_n(kT_s) = 0 \tag{13.30}$$

where $T_s = 1/f_s$ is the simulation sampling period. The simulation is identical, then, whether the underlying noise process is strictly white with an impulsive autocorrelation function, or band limited to $f_s/2$ Hz with the sinc shaped autocorrelation function shown in Figure 13.18. Provided $f_s/2$ (sometimes called the simulation bandwidth) is large with respect to the bandwidth of the system being simulated then the results of the simulation will be unaffected by this ambiguity. White Gaussian noise is therefore, effectively, simulated by generating independent random samples from a Gaussian pdf with variance (i.e. normalised power), σ_n^2, given by:

$$\sigma_n^2 = \frac{N_0 f_s}{2} = \frac{N_0}{2\,T_s} \quad (\mathrm{V}^2) \tag{13.31}$$

where N_0 ($\mathrm{V}^2\mathrm{Hz}^{-1}$) is the required *one-sided* NPSD.

Impulsive noise is characterised by a transient waveform which may occur with random amplitude at random times, or with fixed amplitude periodically, or with some combination of the two, Figure 13.19. The noise may be generated at baseband or passband. In radio systems, however, the noise will be filtered by the receiver's front end

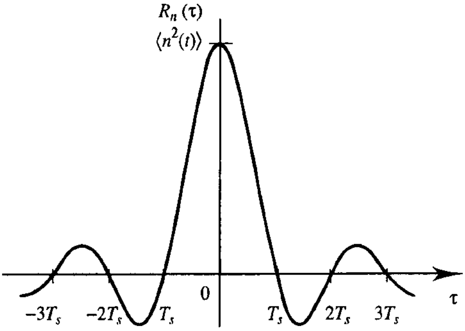


Figure 13.18 *ACF of bandlimited white noise with bandwidth $B = f_s/2$ Hz.*

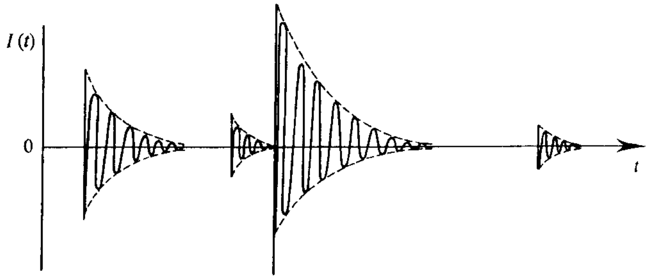


Figure 13.19 *Impulsive noise with random interpulse spacing and random pulse amplitude.*

and IF strip after which even baseband generated impulses will have bandpass characteristics, often resembling the receiver's impulse response. (Impulsive noise does not imply a sequence of impulses in the Dirac delta sense but usually does imply transient pulses with an effective duration which is short compared to the average interpulse spacing.) A useful way of modelling impulsive noise is to separate the statistical aspects from the deterministic pulse shape. This can be done by using a (complex) random number generator to model the amplitude and phase of the pulses, a Poisson counting process to model the arrival times, T_i, of the pulses and a deterministic function, $I(t)$, to model the pulse shape (for example $e^{-at} \cos \omega_c t \; u(t)$). Such a model would be specified at passband by:

$$I(t) = \sum_{i=1}^{N} A_i \, e^{-\frac{t-T_i}{\tau}} \cos(\omega_c t + \theta_i) \, u(t - T_i) \tag{13.32}$$

The equivalent baseband representation would be:

$$I_{LP}(t) = \sum_{i=1}^{N} A_i \, e^{j\theta_i} \, e^{-\frac{t-T_i}{\tau}} \, u(t - T_i) \tag{13.33}$$

Rewriting equation (13.33) as a convolution and expanding the exponential, i.e.:

$$I_{LP}(t) = e^{-\frac{t}{\tau}} u(t) * \sum_{i=1}^{N} A_i \, (\cos\theta_i + j \sin\theta_i) \, \delta(t - T_i) \tag{13.34}$$

emphasises the separation of pulse shape from pulse statistics. θ_i would normally be assumed to have a uniform pdf and a typical pdf for A_i might be log-normal. If the impulse noise is a Poisson process then the pdf of inter-arrival time between pulses is:

$$p_{\Delta T_i}(\Delta T_i) = \lambda \, e^{-\lambda \, \Delta T_i} \tag{13.35}$$

where $\Delta T_i = T_i - T_{i-1}$ and λ is the average pulse arrival rate (see Chapter 17).

Interference usually implies either an unwanted periodic waveform or an unwanted (information bearing) signal. In the former case pulse trains and sinusoids, for example, are easily generated in both passband and equivalent baseband form. In the latter case the interfering signal(s) can be generated in the same manner as wanted signals.

13.4.4 Time invariant linear systems

Linear subsystems, such as pulse shaping filters in a transmitter and matched filters in a receiver, are usually specified by their frequency response (both amplitude and phase). There is a choice to be made in terms of the appropriate level of idealisation when specifying such subsystems. An IF filter in a receiver may, for example, be represented by a rectangular frequency response at one extreme or a set of tabulated amplitude and phase values, obtained, across the frequency band, from measurements on a hardware prototype, at the other extreme. (In this case, unless the frequency domain points have the same frequency resolution as the simulation Fourier transform algorithm, interpolation and/or resampling with the correct resolution will be required.) In between these extreme cases analytical or tabulated models of classical filter responses (e.g. Butterworth, Chebyshev, Bessel, elliptic) may be used. Digital filter structures, typically implemented using tapped delay lines [Mulgrew and Grant], are especially easy to simulate, at least in principle. The effect of a frequency response on an input signal can be found by convolving the impulse response of the filter (which will be complex if equivalent baseband representations are being used) with its input. (The impulse response is obtained from the frequency response by applying an inverse FFT, see section 13.5.) Alternatively, block filtering can be applied in which the input time series is divided into many equal length segments, Fourier transformed using an FFT algorithm, multiplied by the (discrete) frequency response and then inverse transformed back to a time series. The implementation of block filtering, including important aspects such as appropriate zero padding, is described in [Strum and Kirk].

13.4.5 Non-linear and time varying systems

Amplitude compression, as used in companding (section 5.7.3), is a good example of baseband, memoryless, non-linear, signal processing. This type of non-linearity can be modelled either analytically (using, for example, equation (5.31)) or as a set of tabulated points relating instantaneous values of input and output. (Interpolation may be necessary in the latter case.)

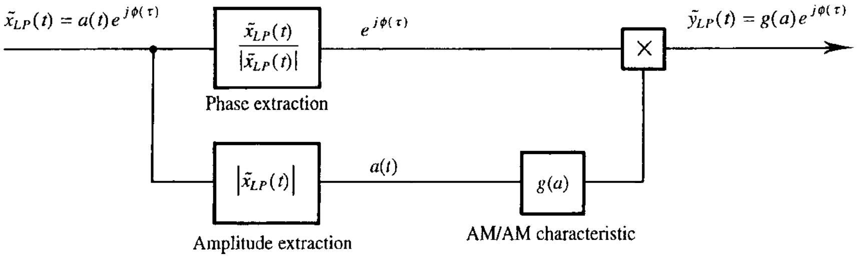

$\tilde{x}_{LP}(t) = a(t)e^{j\phi(\tau)}$
$\frac{\tilde{x}_{LP}(t)}{|\tilde{x}_{LP}(t)|}$
Phase extraction
$e^{j\phi(\tau)}$
×
$\tilde{y}_{LP}(t) = g(a)e^{j\phi(\tau)}$
$|\tilde{x}_{LP}(t)|$
$a(t)$
$g(a)$
Amplitude extraction
AM/AM characteristic

Figure 13.20 *Equivalent baseband model for a non-linear, memoryless, bandpass, system.*

For memoryless, non-linear, systems operating on narrow-band signals the useful equivalent baseband model shown in Figure 13.20 [developed from Tranter and Kosbar] can be used. This model works because, for a sinusoidal input, memoryless non-linearities produce outputs only at the frequency of the input and its harmonics. The harmonics can be filtered out at the non-linearity's output. The overall effect is of a 'linear' system with amplitude dependent gain (AM/AM conversion).

If a non-linearity has memory of intermediate length (i.e. memory which is significant with respect to, and may be many times, the carrier period but which is nevertheless short compared to changes in the carrier's complex envelope) then this too can be simulated as an equivalent baseband system, Figure 13.21. The amplitude of the signal (being related to the envelope) is changed in an essentially memoryless way. The phase of the signal, however, may now be affected by the non-linearity in a way which depends on the signal amplitude (AM/PM conversion). The power amplifier in satellite transponders can often be modelled in this way.

There are other types of non-linear bandpass system which fall into neither of the above special categories. In general it is not possible to obtain equivalent baseband models for these processes. Simulation of these non-linearities must normally, therefore, be executed at passband.

Adaptive equalisers (usually implemented as tapped delay lines with variable tap weights) and adaptive delta modulation (section 5.8.5) are examples of time varying, linear, subsystems. The simulation aspects of such subsystems are essentially straightforward, the design of algorithms to produce the required adaptive behaviour being the principal challenge. It is worth observing, however, that an equaliser operating on the demodulated I and Q components of a quadrature modulated signal generally comprises four separate tapped delay lines. Two lines operate in I and Q channels, individually controlling I and Q channel ISI, whilst two operate across I and Q channels controlling crosstalk, as in Figure 13.6(c). These four, real, tapped lines can be replaced by two lines, in which the weighting factors applied to each tap are complex, and which operate

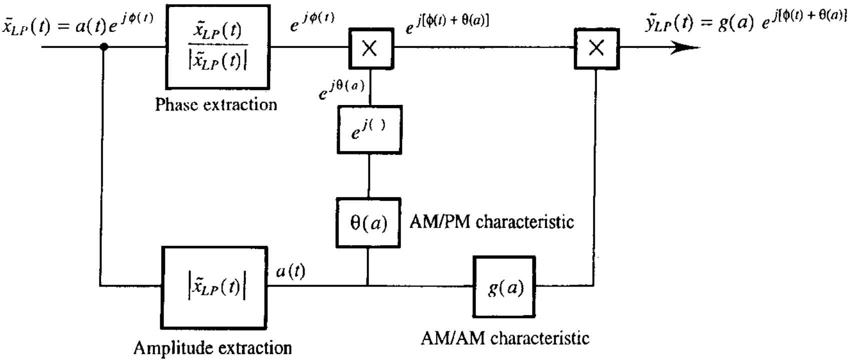

$\tilde{x}_{LP}(t) = a(t)e^{j\phi(t)}$
$\frac{\tilde{x}_{LP}(t)}{|\tilde{x}_{LP}(t)|}$
Phase extraction
$e^{j\phi(t)}$
×
$e^{j[\phi(t)+\theta(a)]}$
×
$\tilde{y}_{LP}(t) = g(a)\, e^{j[\phi(t)+\theta(a)]}$
$e^{j\theta(a)}$
$e^{j(\;)}$
$\theta(a)$
AM/PM characteristic
$|\tilde{x}_{LP}(t)|$
$a(t)$
$g(a)$
Amplitude extraction
AM/AM characteristic

Figure 13.21 *Equivalent baseband model for a non-linear, bandpass system, with memory of intermediate length.*

on the complex envelope of the I and Q channel signals.

13.5 Transformation between time and frequency domains

Both time domain and spectral quantities are of interest in the design and evaluation of communication systems. This alone would be sufficient reason to require transformation between domains when simulating systems. In fact transformation between domains is also desirable since some simulation operations can be implemented more efficiently in the frequency domain than in the time domain.

Computer simulations work with discrete samples of time waveforms (i.e. time series) and discrete samples of frequency spectra. The discrete Fourier transform (DFT) and its relative, the discrete Fourier series (DFS), are the sampled signal equivalent of the Fourier transform and Fourier series described in Chapter 2. From a mathematical point of view the DFT, DFS and their inverses can simply be defined as a set of consistent formulas without any reference to their continuous function counterparts. Almost always, however, in communications engineering the discrete time series and frequency spectra on which these transforms operate represent sampled values of underlying continuous functions. Their intimate and precise connection with continuous Fourier operations is therefore emphasised here.

13.5.1 Discrete Fourier transform

Consider the N-sample time domain signal (or time series), $v_s(t)$, shown in Figure 13.22(a). The sample values can be represented by a series of weighted impulses, Figure 13.22(b), and expressed mathematically by:

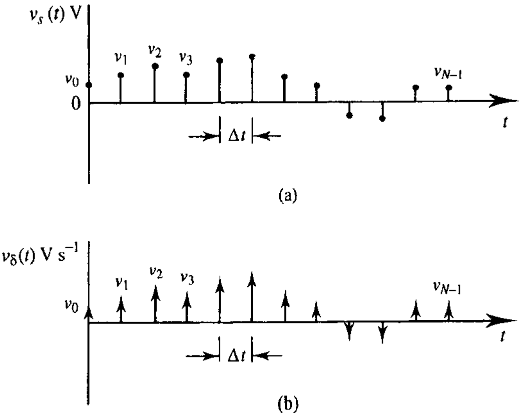


Figure 13.22 *(a) Sampled continuous signal or time series, and (b) its representation as a sum of impulses or delta functions.*

$$v_\delta(t) = v_0\,\delta(t) + v_1\,\delta(t-\Delta t) + v_2\,\delta(t-2\Delta t) + \cdots \quad \text{(V/s)} \tag{13.36}$$

Notice that $v_\delta(t)$ has units of V/s. This is easily demonstrated by using the sampling property of impulse functions under integration to recover the original time series which has units of V, i.e.:

$$\begin{aligned} v_s(t) &= \left\{ \int_{-\infty}^{\infty} v_0\delta(t)\,dt,\ \int_{-\infty}^{\infty} v_1\delta(t-\Delta t)\,dt,\ \int_{-\infty}^{\infty} v_2\delta(t-2\Delta t)\,dt, \cdots \right\} \\ &= \left\{ v_0\int_{-\infty}^{\infty} \delta(t)\,dt,\ v_1\int_{-\infty}^{\infty} \delta(t-\Delta t)\,dt,\ v_2\int_{-\infty}^{\infty} \delta(t-2\Delta t)\,dt, \cdots \right\} \\ &= \{v_0, v_1, v_2, \cdots, v_{N-1}\} \quad \text{(V)} \end{aligned} \tag{13.37}$$

(Since the sample values, v_i, have units of V this means that the impulses, $\delta(t-\Delta t)$, have units of s^{-1}, i.e. their area, or strength, is dimensionless.) The voltage spectrum of $v_\delta(t)$ is given by:

$$\begin{aligned} V_\delta(f) &= \text{FT}\,\{v_\delta(t)\} \\ &= v_0 \int_{-\infty}^{\infty} \delta(t)e^{-j2\pi ft}\,dt \;+\; v_1 \int_{-\infty}^{\infty} \delta(t-\Delta t)e^{-j2\pi ft}\,dt + \cdots \quad \text{(V)} \end{aligned} \tag{13.38}$$

(Notice that because equation (13.38) is the transform of a sequence of weighted impulses with units of V/s the voltage spectrum has units of V only, not V/Hz as usual.) Using the sampling property of $\delta(t)$ under integration equation (13.38) becomes:

$$V_\delta(f) = v_0e^{-j0} + v_1e^{-j2\pi f\Delta t} + v_2e^{-j2\pi f2\Delta t} + \cdots \quad \text{(V)} \tag{13.39}$$

$V_\delta(f)$ in equation (13.39) is a *continuous* function, i.e. it is defined for all values of f. Using the summation notation equation (13.39) can be written more succinctly as:

$$V_\delta(f) = \sum_{\tau=0}^{N-1} v_\tau\ e^{-j2\pi f(\tau\Delta t)} \quad \text{(V)} \tag{13.40}$$

The values of the spectrum, $V_\delta(f)$, at the discrete frequencies f_0, f_1, f_2, etc. are given by:

$$V_\delta(f_\nu) = \sum_{\tau=0}^{N-1} v_\tau\ e^{-j2\pi f_\nu(\tau\Delta t)} \quad \text{(V)} \tag{13.41}$$

and if the frequencies of interest are equally spaced by Δf Hz then:

$$V_\delta(f_\nu) = \sum_{\tau=0}^{N-1} v_\tau\ e^{-j2\pi(\nu\Delta f)(\tau\Delta t)} \quad \text{(V)} \tag{13.42}$$

where $\nu = 0, 1, 2, 3, \ldots$ etc. Since the time series contains N samples it represents a signal with duration, T, given by:

$$T = N\Delta t \quad \text{(s)} \tag{13.43}$$

Nyquist's sampling theorem (Chapter 5) asserts that the lowest observable frequency (excluding DC) in the time series is:

$$f_1 = \frac{1}{N\Delta t} = \Delta f \quad \text{(Hz)} \tag{13.44}$$

Thus:

$$\Delta f \; \Delta t = \frac{1}{N} \tag{13.45}$$

and equation (13.42) can be written as:

$$V_\delta(f_\nu) = \sum_{\tau=0}^{N-1} v_\tau \, e^{-j2\pi \frac{\nu}{N} \tau} \quad \text{(V)} \tag{13.46}$$

Equation (13.46) is the definition, adopted here, for the *forward* DFT. Comparing this with the conventional (i.e. continuous) FT:

$$V(f) = \int_{-\infty}^{\infty} v(t)\, e^{-j2\pi ft}\, dt \quad \text{(V/Hz)} \tag{13.47}$$

the difference is seen, essentially, to be the absence of a factor corresponding to dt. $V_\delta(f_\nu)$ is therefore related to $V(f)$ by:

$$V(f)\,|_{f=f_\nu} \approx V_\delta(f_\nu)\,\Delta t \quad \text{(V/Hz)} \tag{13.48}$$

The reason why an approximation sign is used in equation (13.48) will become clear later. A note of caution is appropriate at this point. Equation (13.46) as a definition for the DFT is not universal. Sometimes a factor of $1/N$ and sometimes a factor of $1/\sqrt{N}$ is included in the formula. If the absolute magnitude of a voltage spectrum is important it is essential, therefore, to know the definition being used. Furthermore proprietary DFT software may not include an implementation of equation (13.48). Care is therefore needed in correctly interpreting the results given by DFT software.

13.5.2 Discrete Fourier series

If $v(t)$ is periodic then its FT should represent a *discrete* voltage spectrum (in contrast to discrete values taken from a continuous spectrum). Comparing the DFT (equation (13.46) with the formula for a set of Fourier series coefficients (Chapter 2):

$$\tilde{C}_\nu = \frac{1}{T}\int_0^T v(t)\, e^{-j2\pi ft}\, dt \quad \text{(V)} \tag{13.49}$$

and remembering that the length of the time series is given by $T = N\Delta t$ then, to make $V_\delta(f_\nu)$ reflect $\tilde{C}_\nu$ properly, an extra factor $1/N$ is required, i.e.:

$$\tilde{C}_\nu = \frac{1}{N\Delta t} V_\delta(f_\nu)\,\Delta t = \frac{1}{N} V_\delta(f_\nu) \quad \text{(V)} \tag{13.50}$$

The (forward) *discrete Fourier series* (DFS) is therefore defined by:

$$\tilde{C}(f_\nu) = \frac{1}{N}\sum_{\tau=0}^{N-1} v_\tau \, e^{-j2\pi \frac{\nu}{N}\tau} \quad \text{(V)} \tag{13.51}$$

The need for the factor $1/N$ in equation (13.51) is seen most easily for the DC ($\nu = 0$) value which is simply the time series average.

13.5.3 DFS spectrum and rearrangement of spectral lines

Providing all the samples in the time series, v_τ, are real then the spectrum defined by equation (13.51) has the following properties:

1. Spectral lines occurring at $\nu = 0$ and $\nu = N/2$ are real. All others are potentially complex.
2. $\tilde{C}(f_{N-\nu}) = \tilde{C}^*(f_\nu)$, i.e. the DFS amplitude spectrum is even and the DFS phase spectrum is odd.

Figure 13.23 illustrates these properties for a 16-sample time series. The harmonic number, ν, along the x-axis of Figure 13.23 is converted to conventional frequency f (in Hz) using:

$$f_\nu = \nu f_1 = \frac{\nu}{N\Delta t} \quad (Hz) \tag{13.52}$$

This leads, superficially, to a paradox in that f_{N-1} appears to correspond to a frequency of (almost) $1/\Delta t$ Hz yet Nyquist's sampling theorem asserts that frequencies no higher than $1/(2\Delta t)$ Hz can be observed. The paradox is resolved by recognising that no additional information is contained in the harmonics $N/2 < \nu < N$ since these are conjugates of the harmonics $0 < \nu < N/2$. A satisfying interpretation of the 'redundant' harmonics ($\nu > N/2$) is as the negative frequency components of a double sided spectrum. The conventional frequency spectrum is therefore constructed from the DFS (or DFT) by shifting all the lines from the top half of the DFS spectrum down in frequency by $N\Delta f$ Hz as shown in Figure 13.24. (Half the component at $\nu = N/2$ can also be shifted by

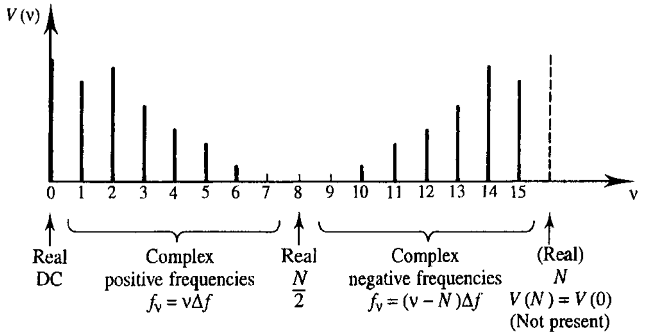


Figure 13.23 *Schematic illustration of discrete Fourier series for a 16-sample time series.*

$-N\Delta f$ Hz to retain overall symmetry, if desired.) The highest observable frequency is then given by:

$$f_{N/2} = \frac{1}{2\,\Delta t} \quad \text{(Hz)} \tag{13.53}$$

as expected.

13.5.4 Conservation of information

In the time domain an N-sample real time series is represented by N real numbers. In the frequency domain the same signal is represented by $N-2$ complex numbers and 2 real numbers (i.e. $2(N-2)+2$ real numbers). Half of the complex numbers, however, are complex conjugates of the other half. Thus in the frequency domain, as in the time domain, the signal is represented by N *independent* real numbers.

13.5.5 Phasor interpretation of DFS

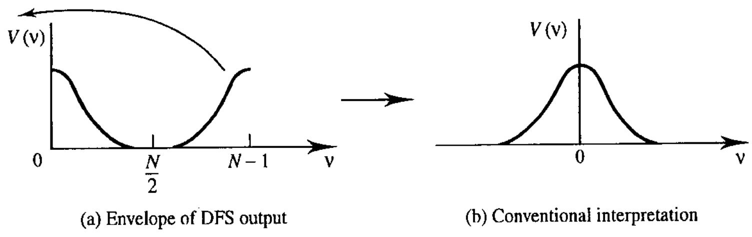
Consider equation (13.51). Each (complex) spectral line, $\tilde{C}(f_3)$ for example, is a sum of N phasors (one arising from each time sample), all with identical frequency, f_3 in this case, but each with a different phase, $\theta_\tau = -2\pi(3/N)\tau$. Figure 13.25(a) and (b) illustrate the phasor diagrams for $\tilde{C}(f_1)$ and $\tilde{C}(f_2)$ respectively corresponding to an 8-sample time series. In Figure 13.25(a) (where $\nu = 1$) the phasors advance by $2\pi(1/8)$ rad = 45° each time τ is incremented. In Figure 13.25(b) (where $\nu = 2$) the phasors advance by $2\pi(2/8)$ rad = 90°. (There is an additional π rad phase change when the corresponding time series sample is negative.)

13.5.6 Inverse DFS and DFT

Consider the forward DFS:

$$\tilde{C}(f_\nu) = \frac{1}{N}\sum_{\tau=0}^{N-1} v_\tau\, e^{-j2\pi\frac{\nu}{N}\tau} \quad \text{(V)} \tag{13.54}$$

After rearranging, this represents N cisoids (or phasors) each rotating with a different



(a) Envelope of DFS output

(b) Conventional interpretation

Figure 13.24 *Interpretation of DFS (or DFT) components $\nu > N/2$ as negative frequencies.*

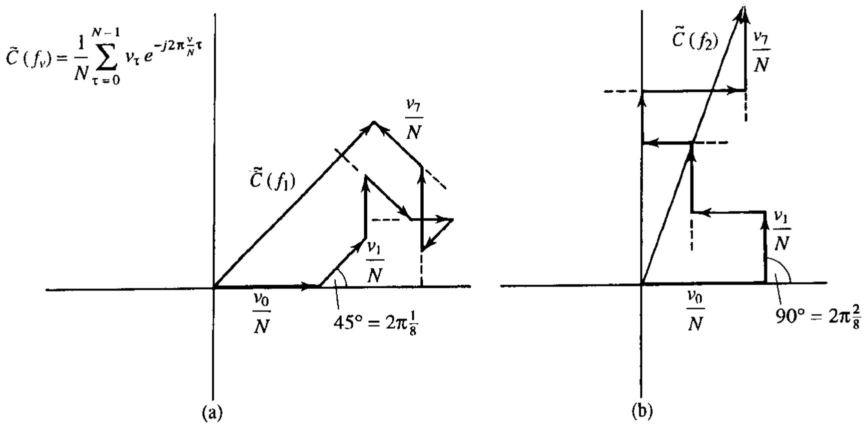


Figure 13.25 *Phasor interpretation for DFS showing composition of: (a) fundamental ($\nu = 1$) and (b) second harmonic ($\nu = 2$) components of an 8-sample time series.*

frequency, i.e.:

$$\tfrac{1}{2}\,\tilde{C}(f_{-N/2})\,, \cdots, \tilde{C}(f_{-2}), \tilde{C}(f_{-1}), \tilde{C}(f_0), \tilde{C}(f_1), \tilde{C}(f_2), \cdots, \tfrac{1}{2}\,\tilde{C}(f_{N/2})$$

Each time sample, v_τ, is the sum of these cisoids evaluated at time $t = \tau\Delta t$. Figure 13.26, for example, shows the phasor diagram for $N = 8$ at the instant $t = 3\Delta t$ (i.e. $\tau = 3$). The resultant on this diagram corresponds to the third sample of the time series. (The diagram is referred to as a phasor diagram, here, even though different phasor pairs are rotating at different frequencies.) By inspection of Figure 13.26 the *inverse* DFS can be seen to be:

$$v_\tau = \sum_{\nu=0}^{N-1} \tilde{C}(f_\nu)\, e^{j\frac{2\pi}{N}\nu\tau} \quad \text{(V)} \tag{13.55}$$

Similarly the inverse DFT is:

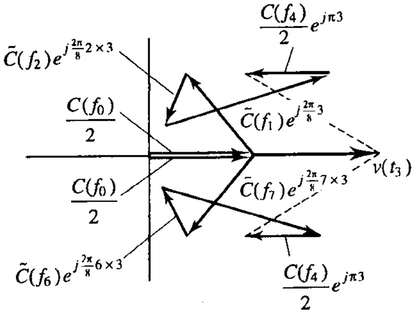

$\tilde{C}(f_2)e^{j\frac{2\pi}{8}2\times 3}$
$\frac{C(f_0)}{2}$
$\frac{C(f_0)}{2}$
$\tilde{C}(f_6)e^{j\frac{2\pi}{8}6\times 3}$
$\frac{C(f_4)}{2}e^{j\pi 3}$
$\tilde{C}(f_1)e^{j\frac{2\pi}{8}3}$
$v(t_3)$
$\tilde{C}(f_7)e^{j\frac{2\pi}{8}7\times 3}$
$\frac{C(f_4)}{2}e^{j\pi 3}$

Figure 13.26 *'Phasor' diagram at the instant $t = 3\Delta t$ (i.e. $\tau = 3$) for a real, 8-sample, time series.*

$$v_\tau = \frac{1}{N} \sum_{\nu=0}^{N-1} V_\delta(f_\nu)\, e^{j\frac{2\pi}{N}\nu\tau} \quad \text{(V)} \tag{13.56}$$

(Notice that the factors of $1/N$ are arranged in the DFS, DFT and their inverses, such that applying a given discrete transform, followed immediately by its inverse, results in no change.)

13.5.7 DFT accuracy

Having calculated a spectrum using a DFT, and rearranged the spectral lines as described in section 13.5.3, the question might be asked – how well do the resulting values represent the continuous Fourier transform of the continuous time function underlying the time series? This question is now addressed.

Sampling and truncation errors

There are two problems which have the potential to degrade accuracy. These are:

1. sampling;
2. truncation.

Aliasing, due to sampling, is avoided providing sampling is at the Nyquist rate or higher, i.e.:

$$\frac{1}{\Delta t} \geq 2 f_H \quad \text{(Hz)} \tag{13.57}$$

The truncation problem arises because it is only possible to work with time series of finite length. A series with N samples corresponds to an infinite series multiplied by a rectangular window, $\Pi(t/T)$, of width $T = N\Delta t$ s, Figure 13.27. This is equivalent to convolving the FT of $v_\delta(t)$ with the function $T\text{sinc}(Tf)$, Figure 13.28. The convolution smears, or smooths, $V(f)$ on a frequency scale of approximately $2/T$ Hz (i.e. the width of the smoothing function's main lobe). It might be argued that a time series for which the underlying function is strictly time limited to $N\Delta t$ s does not suffer from truncation error. In this case, however, the function cannot be bandlimited and the Nyquist sampling rate would be ∞ Hz making it impossible to avoid aliasing errors. Conversely if the underlying function is strictly bandlimited (allowing the possibility of zero aliasing error)

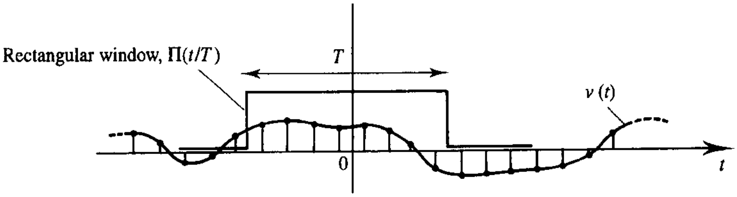


Figure 13.27 *A finite (8-sample) time series interpreted as the product of an infinite series and a rectangular window.*

then it cannot be time limited, making truncation errors unavoidable. (It is, of course, possible to reduce both types of error by sampling at a faster rate and increasing the length of the time series. This means increasing the overall number of samples, however, requiring greater computer resources.)

Frequency sampling, smoothing, leakage and windowing

Discreteness and periodicity are the corresponding properties of a function expressed in frequency and time domains. This is summarised in Table 13.4.

Table 13.4 *Relationship between periodic and discrete signals.*

Time domain	*Frequency domain*
Periodic	Discrete
Discrete	Periodic

Since the DFT gives discrete samples of $V(f)$ (at least approximately) then the function of which it is the precise FT (or FS) is periodic. Furthermore, since the time series, $v_\delta(t)$, is discrete it should have a periodic spectrum. This implies that a replicated version of a DFT, such as that shown in Figure 13.29(b), is the exact FT (or FS) of a function such as that shown in Figure 13.29(a).

Figure 13.30 illustrates how the implicit periodicity of both the underlying time series and the exact spectrum of this time series impacts on truncation errors. It can be seen that in addition to smoothing of the baseband spectrum, energy leaks into the DFT from the higher frequency spectral replicas via the window spectrum with which the exact spectrum is convolved. This type of error is called leakage and can be reduced by decreasing the sidelobes of the window function's spectrum. Shaping of the time domain window function to realise low sidelobes in its spectrum tends, however, to increase the width of the spectral main lobe. There is, consequently, a trade-off to be made between leakage and smoothing errors. The optimum shape for a time domain window depends on the particular application, the data being transformed and engineering judgement. The commonly encountered window functions are discussed in [Brigham].

The effect of crude windowing, with a rectangular shape for example, does not, *necessarily*, lead to poor accuracy in the calculated spectrum of a time series. If the

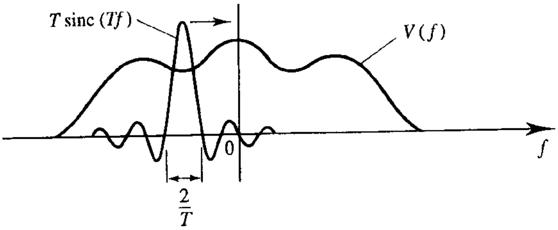


Figure 13.28 *Convolution in frequency domain of FT {v(t)} with FT {Π(t/T)}, corresponding to rectangular windowing in Figure 13.27.*

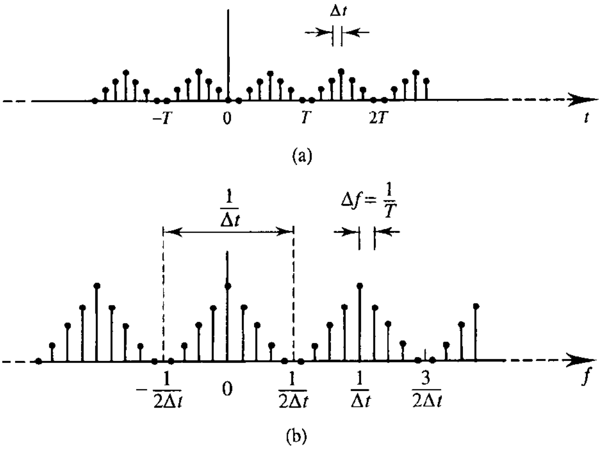


Figure 13.29 *Sampled and replicated versions of a signal in (a) time and (b) frequency domain forming a precise FT or FS pair.*

spectrum of the unwindowed time series is slowly changing on the frequency scale of the oscillations of the window's sidelobes then cancellation tends to occur between the leakage from adjacent sidelobes, Figure 13.31. A corollary of this is a frequency domain manifestation of Gibb's phenomenon, Figure 13.32. If the time series spectrum changes rapidly on the frequency scale of window spectrum oscillations, then as the window spectrum slides across the time series spectrum the convolution process results in window spectrum oscillations being reproduced in the spectrum of the windowed time series. This problem is at its worst in the region of time series spectral discontinuities where it can lead to significant errors. The same effect occurs in the region of any impulses present in the time series spectrum. In particular this means that the DC level (i.e. average value) of a time series should be removed before a DFT is applied. If this is overlooked it is possible that the window spectrum reproduced by convolution with the 0 Hz impulse may obscure, and be mistaken for, the spectrum of the time series data, Figure 13.33. (If impulses in the spectrum of the time series are important they should be identified, and removed, before application of a DFT, and subsequently reinserted into the calculated spectrum.)

Trailing zeros

The frequency spacing of spectral values obtained from a DFT is given by:

$$\Delta f = f_1 = \frac{1}{T} \text{ (Hz)} \tag{13.58}$$

where $T = N\Delta t$ is the (possibly windowed) length of the time series. Δf is sometimes called the resolution of the DFT. (This does not imply that all spectral features on a scale of Δf are necessarily resolved since resolution in this sense may be limited by

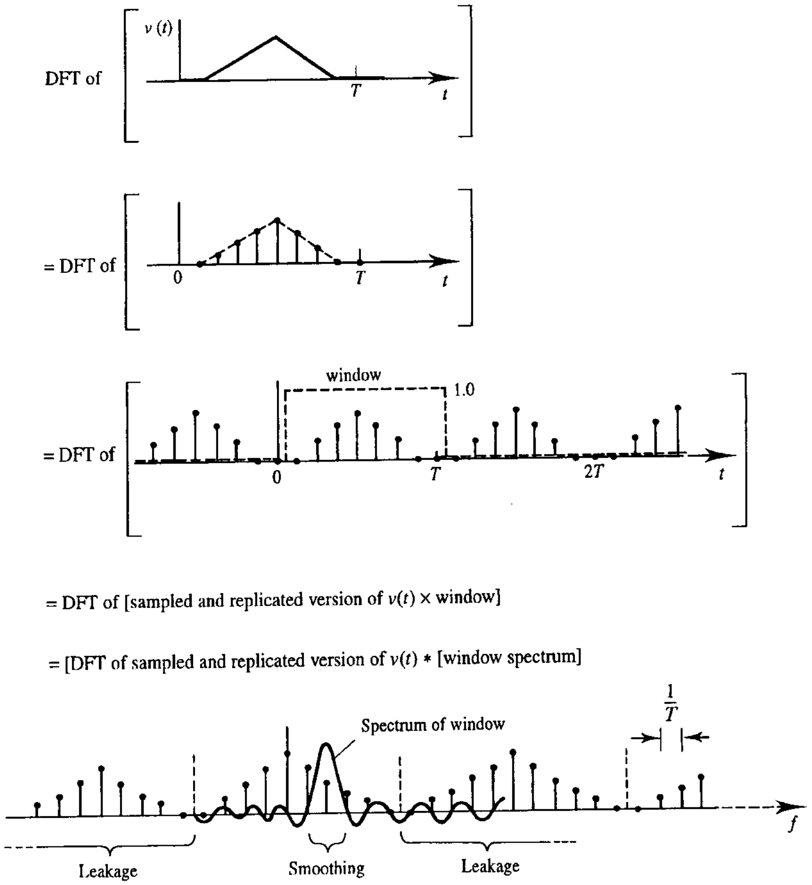

DFT of
v (t)
T
t
= DFT of
0
T
t
= DFT of
window
1.0
0
T
2T
t
= DFT of [sampled and replicated version of v(t) × window]
= [DFT of sampled and replicated version of v(t) * [window spectrum]
Spectrum of window
1/T
f
Leakage
Smoothing
Leakage

Figure 13.30 *Origin of smoothing and leakage errors in a DFT.*

windowing effects.) Δf can only be made smaller by increasing the length of the time series data record, i.e. increasing T (or N). The alternative is to artificially extend the data record with additional trailing zeros. This is illustrated in Figure 13.34 and is called zero padding. Notice that the highest observable frequency, $f_{N/2}$, determined by the sampling period, Δt, is unaffected by zero padding but, since more samples are included in the DFT operation, there are more samples in the output display, giving finer sampling in the frequency domain. Since the genuine data record has not been extended, however, the underlying resolution is unaltered, the zero padding samples having simply allowed the DFT output to be more finely interpolated [Mulgrew and Grant].

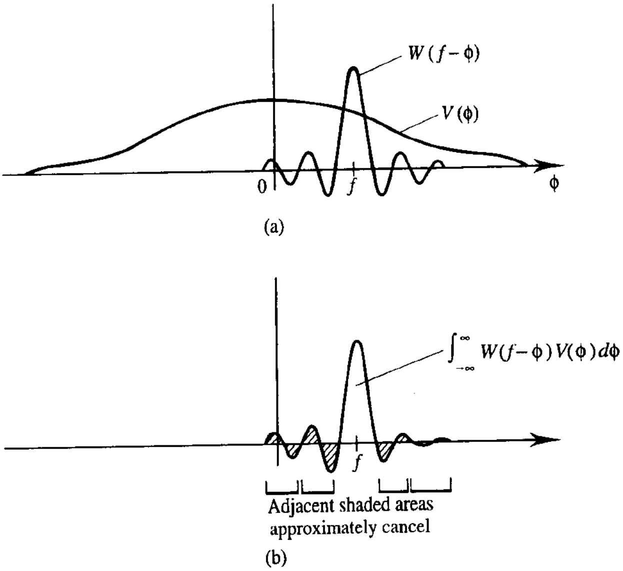


Figure 13.31 *Approximate cancellation of leakage errors for a windowed signal with slowly changing spectrum: (a) convolution of underlying spectrum with oscillating window spectrum; (b) result of convolution at frequency shift $\phi = f$.*

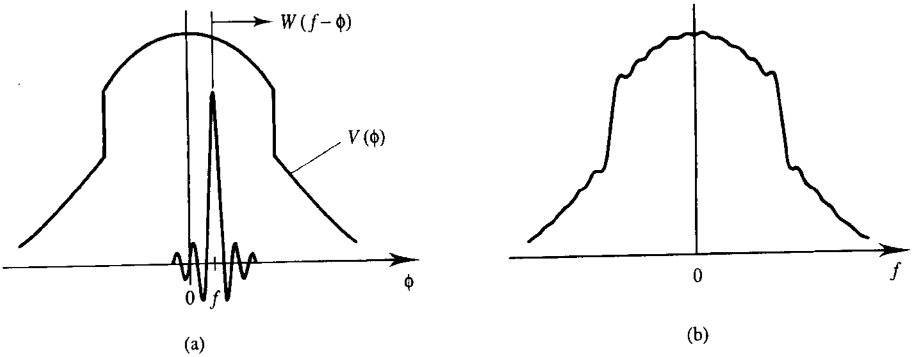


Figure 13.32 *Frequency manifestation of Gibb's phenomenon: (a) convolution of a spectrum containing rapidly changing region with oscillating spectrum of a windowing function; (b) resulting oscillations in spectrum of windowed function.*

Random data and spectral estimates

Consider the stationary random process, $v_s(t)$, illustrated in Figure 13.35(a). If several segments of this random time series are windowed and transformed using a DFT then the result is a number of voltage spectra, each with essentially meaningless phase

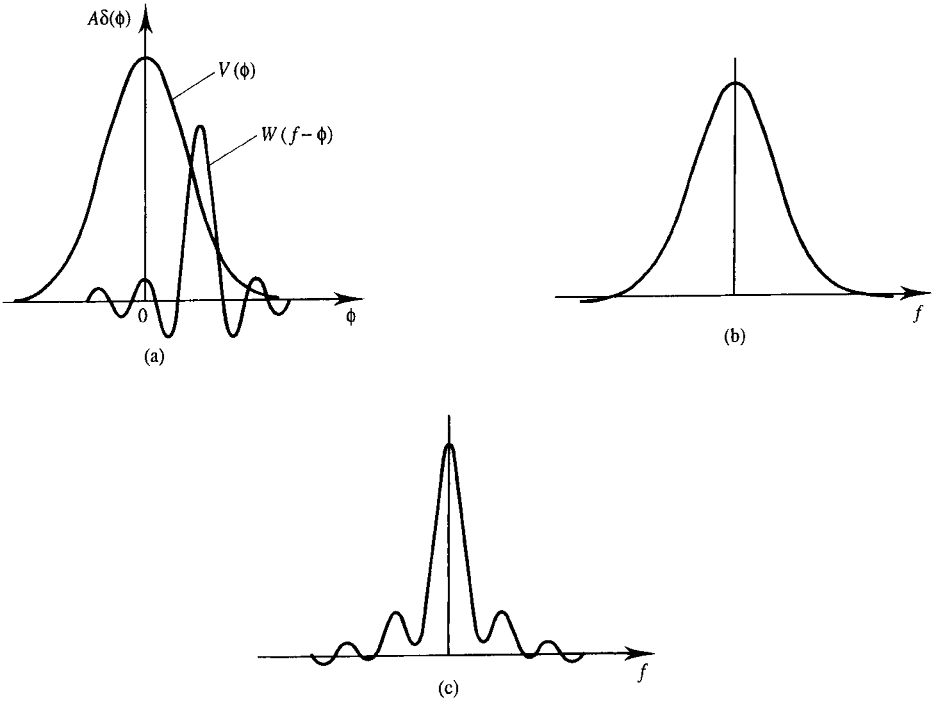


Figure 13.33 *(a) Convolution of a signal spectrum containing an impulse at 0 Hz with a sinc function reflecting rectangular windowing of a signal with a large DC value, (b) result of convolution excluding DC impulse, (c) convolution including DC impulse.*

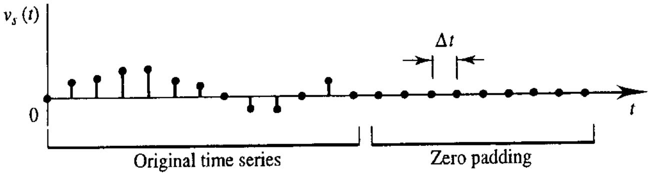


Figure 13.34 *Illustration of zero padding a time series with trailing zeros.*

information. If power spectra, $G_i(f_v)$, are obtained using:

$$G_i(f_v) = \frac{1}{T} \mid \text{DFT}\,[v_\tau\ w_i(\tau)]\Delta t \mid^2 \quad (\text{V}^2\text{Hz}^{-1}) \tag{13.59}$$

where $w_i(\tau)$ is the window applied to the ith segment of the data and $G_i(f_v)$ is the power spectrum of the ith segment, then the size of a spectral line at a given frequency will fluctuate randomly from spectrum to spectrum, Figure 13.35(b). Perhaps surprisingly, the random error in the calculated samples of the power spectral density does not decrease if

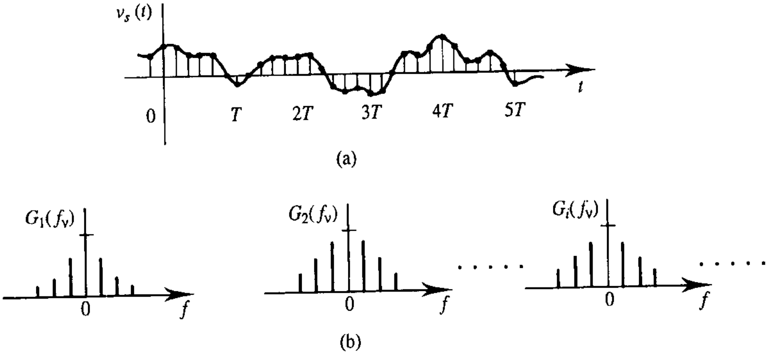


Figure 13.35 *(a) Stationary random process divided into T-second segments, (b) power spectra calculated from each time series segment.*

the length of time series segments is increased. This is because the frequency 'resolution' improves proportionately. An example makes this clear. If the length of a time series segment is doubled the spacing of its spectral lines is decreased by a factor of 2. Twice the number of spectral lines are generated by the DFT and thus the *information per spectral line remains constant.* The random errors in such a spectrum can be decreased by:

1. Averaging the corresponding values (lines) over the spectra found from independent time segments, i.e. using:

$$G(f_v) = \frac{1}{M} \sum_{i=1}^{M} G_i(f_v) \quad (\mathrm{V^2Hz^{-1}}) \tag{13.60}$$

2. Averaging adjacent values in a single spectra.

An alternative approach is to use the Wiener-Kintchine theorem and transform the autocorrelation function (ACF) of the time series, i.e.:

$$G(f_v) = \mathrm{DFT}\,\{\mathrm{ACF}\,[v_\tau\ w(\tau)]\}\,\Delta t \quad (\mathrm{V^2Hz^{-1}}) \tag{13.61}$$

where the units of the discrete ACF are V^2. The number of spectral lines generated by equation (13.61) now depends on the maximum temporal displacement (or lag) used in the ACF rather than the length of the time series used.

13.6 Discrete and cyclical convolution

For sampled data such as that used in computer simulations the (discrete) convolution between two time series $x_0, x_1, x_2, \cdots, x_{N-1} = \{x_\tau\}$ and $y_0, y_1, y_2, \cdots, y_{M-1} = \{y_\tau\}$ is defined by:

$$z_n = \sum_{\tau=0}^{n} x_\tau\, y_{n-\tau} \quad (\mathrm{V^2}) \tag{13.62}$$

where τ is the (integer) sample number of (both) time series (i.e. $x_\tau = x(\tau\Delta t)$), Δt is the sampling period and $n = 0, 1, 2, \cdots, N + M - 2$ is the shift in sample numbers between x_τ and the time reversed version of y_τ. (If the discrete convolution is being used to evaluate the convolution of a pair of underlying continuous signals, then equation (13.62) must be multiplied by the sampling period, Δt. The result will then have units of V^2s as expected.) The question may be asked, 'does the precise timing of the sampling instants affect the result of the numerical (or discrete) convolution defined in equation (13.62)?'. The slightly surprising answer is generally yes. Figure 13.36 shows a signal sampled at 10 Hz. Dots represent samples which start at the origin (defined by the start of the continuous signal) and crosses represent samples which start half a sample period after the origin. Since convolution involves integration it is usually the case that the set of sampling points which best represent the area of the underlying function will give the best result in the sense that application of equation (13.62) will give an answer closest to the analytical convolution of the underlying functions. This implies that the crosses in Figure 13.36 represent superior sampling instants which is also consistent with an intuitive feeling that a sample should be at the centre of the function segment which it represents.

The crosses in Figure 13.36 also have the advantage that none fall on a point of discontinuity. If sampling at such points cannot be avoided then an improvement in terms of area represented is obtained by assigning a value to that sample equal to the mean of the function value on either side of the discontinuity. In Figure 13.36, therefore, the sample (dot) for the point $t = 1.0$ would be better placed at 0.5 V than 0 V. (This is consistent with the fact that physical signals do not contain discontinuities and that bandlimited signals, corresponding to truncated Fourier series, converge at points of discontinuity, to the mean of the signal value on either side of that discontinuity, see section 2.2.3.)

In practice convolution is often implemented by taking the inverse DFT of the product of the DFTs of the individual time series, i.e.:

$$\{x_\tau\} * \{y_\tau\} \equiv \mathrm{DFT}^{-1}\left[\mathrm{DFT}\{x_\tau\}\,\mathrm{DFT}\{y_\tau\}\right] \quad (\mathrm{V}^2) \tag{13.63}$$

For equation (13.63) to yield sensible results the sampling period and length of both time series must be the same. (Zero padding can be used to equalise series lengths if necessary.) Writing out equation (13.63) explicitly (using equations (13.56) and (13.46)), and using primes and double primes to keep track of sample numbers in the different time

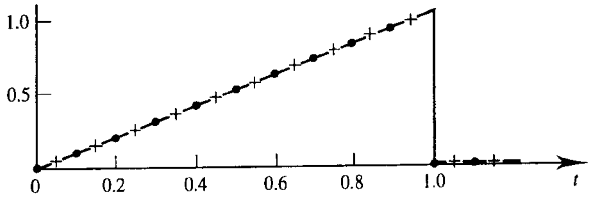


Figure 13.36 *Sampled signal showing alternative sampling instants.*

series:

$$\begin{aligned} z(\tau) &= \{x_{\tau'}\} * \{y_{\tau''}\} = \frac{1}{N}\sum_{\nu=0}^{N-1}\left[\sum_{\tau'=0}^{N-1} x_{\tau'}\, e^{-j2\pi\frac{\nu}{N}\tau'} \sum_{\tau''=0}^{N-1} y_{\tau''}\, e^{-j2\pi\frac{\nu}{N}\tau''}\right] e^{j\frac{2\pi}{N}\nu\tau} \\ &= \frac{1}{N}\sum_{\nu=0}^{N-1}\left[\sum_{\tau'=0}^{N-1}\sum_{\tau''=0}^{N-1} x_{\tau'}\, y_{\tau''}\, e^{-j2\pi\frac{\nu}{N}(\tau'+\tau'')}\right] e^{j\frac{2\pi}{N}\nu\tau} \\ &= \frac{1}{N}\sum_{\tau'=0}^{N-1}\sum_{\tau''=0}^{N-1} x_{\tau'}\, y_{\tau''}\left[\sum_{\nu=0}^{N-1} e^{j2\pi\frac{\nu}{N}(\tau-\tau'-\tau'')}\right] \end{aligned} \qquad (13.64)$$

The square bracket on the last line of equation (13.64) is zero unless:

$$\tau - \tau' - \tau'' = nN \quad (\text{for } \textit{any integer } n) \qquad (13.65)$$

in which case it is equal to N. Equation (13.64) can therefore be rewritten as:

$$\begin{aligned} z(\tau) &= \sum_{\tau'=0}^{N-1}\sum_{\tau''=0}^{N-1} x_{\tau'}\, y_{\tau''} \qquad (\tau'' = \tau - \tau' - nN \text{ only}) \\ &= \sum_{\tau'=0}^{N-1} x_{\tau'}\, y_{\tau-\tau'-nN} \end{aligned} \qquad (13.66)$$

The implication of equation (13.66) is that the τth sample in the convolution result is the same for any n, i.e.:

$$\begin{aligned} z(\tau) &= \ldots \\ &= \sum_{\tau'=0}^{N-1} x_{\tau'} y_{\tau-\tau'+N} \\ &= \sum_{\tau'=0}^{N-1} x_{\tau'} y_{\tau-\tau'} \\ &= \sum_{\tau'=0}^{N-1} x_{\tau'} y_{\tau-\tau'-N} \\ &= \ \ldots \end{aligned} \qquad (13.67)$$

This means that $y_{\tau-\tau'}$ (i.e. $y_{\tau''}$) is cyclical with period N as shown in Figure 13.37. Alternative interpretations are that as the time shift variable, τ, changes, elements of the time series, $y_{\tau''}$, lying outside the (possibly windowed) time series, $x_{\tau'}$, are recycled as shown in Figure 13.38, or that the time series are arranged in closed loops as shown in Figure 13.39. Figures 13.38 and 13.39 both define the cyclical convolution operation which is the equivalent time series operation to the multiplication of DFTs.

The cyclic (or periodic) convolution of two N-element time series is contained in a series which has, itself, N elements. (For normal, or aperiodic, convolution the series has

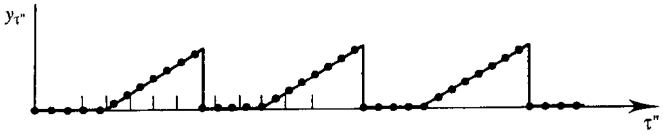


Figure 13.37 *Periodic interpretation of $y_{\tau''}$ resulting in cyclic convolution when using $x_\tau * y_\tau = \mathrm{DFT}^{-1}[\mathrm{DFT}(x_\tau)\mathrm{DFT}(y_\tau)]$ to implement discrete convolution.*

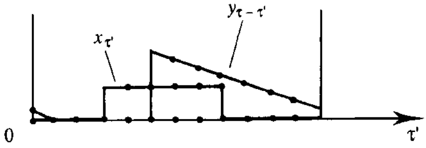


Figure 13.38 *Sample recycling interpretation of y_τ for cyclic convolution.*

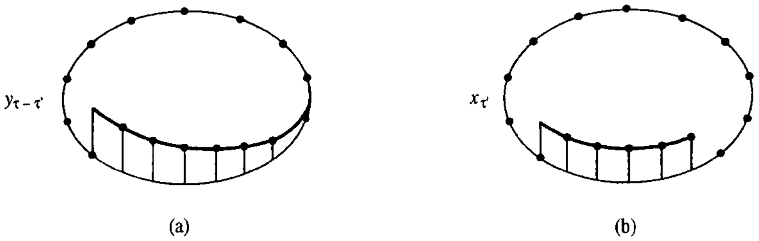


Figure 13.39 *Closed loop interpretation of x_τ and y_τ for cyclic convolution.*

$2N - 1$ elements.) A consequence is that if two functions are convolved using equation (13.63) (thus implying cyclical convolution) then enough zero padding must be used to ensure that one period of the result is visible, isolated by zeros on either side. If insufficient leading and/or trailing zeros are present in the original time series then the convolved sequences will have overlapping ends, Figure 13.40, making normal interpretation difficult. (In general, to avoid this, the number of leading plus trailing zeros required prior to cyclical convolution is equal to the number of elements in the functions to be convolved from the first non-zero element to the last non-zero element.)

13.7 Estimation of BER

For digital communications the quantity most frequently used as an objective measure of performance is symbol error rate, SER, or equivalently probability of symbol error, P_e. Sometimes more detailed information is desirable, for example it might be important to know whether errors occur independently of each other or whether they tend to occur in

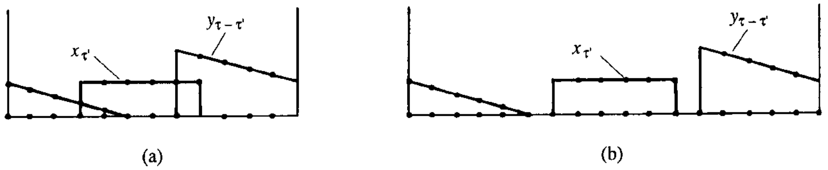


Figure 13.40 *Use of extra zeros to pad x_τ and y_τ in order to avoid overlapping of discrete convolution replicas in cyclical convolution result: (a) insufficient leading/trailing zeros leading to overlapping of replicas; (b) padded functions avoiding overlapping.*

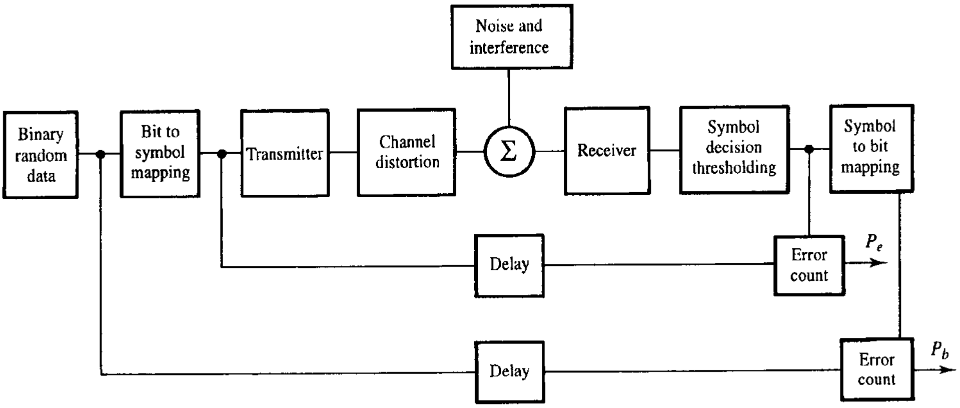


Figure 13.41 *Principles of Monte Carlo evaluation of SER and BER.*

bursts. Here we give a brief outline of two of the methods by which SER (and, if necessary, second order performance information) can be estimated using simulation. These are, the Monte Carlo method and the quasi-analytical method. To some extent these methods represent examples of SER estimation techniques located at opposite ends of a spectrum of techniques. Other methods exist which have potential advantages under particular circumstances. A detailed discussion of many of these methods is given in [Jeruchim *et al.*].

13.7.1 Monte Carlo simulation

This is the conceptually simplest and most general method of estimating SERs. The detected symbol sequence at the receiver is compared symbol by symbol with the (error free) transmitted sequence and the errors are counted, Figure 13.41. The estimated P_e is then given by:

$$P_e = \frac{\text{error count}}{\text{total symbol count}} \tag{13.68}$$

and the SER is given in equation (6.13) as:

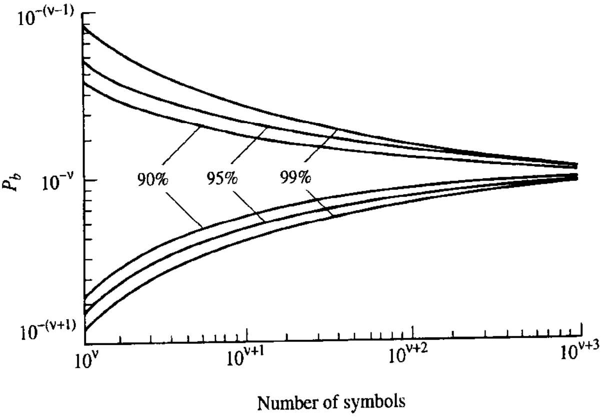


Figure 13.42 *Confidence bands on P_b when observed value is 10^{-v} for the Monte Carlo technique based on the normal approximation (source: Jeruchim et al., 1984, reproduced with permission of the IEEE).*

$$\text{SER} = P_e R_s \quad \text{(error/s)} \tag{13.69}$$

where R_s is the baud rate. Direct Monte Carlo estimation of BER is effected in a similar way, Figure 13.41. The penalty paid for the simplicity and generality of Monte Carlo simulation is that if SER is to be measured with both precision and confidence for low error rate systems, then large numbers of symbols must be simulated. This implies large computing power and/or long simulation times. If errors occur independently of each other, and a simulation is sufficiently long to count at least 10 errors, then the width of the P_e interval in which we can be 90%, 95% and 99% confident that the true P_e lies may be found from Figure 13.42. If errors are not independent (for example they may occur in bursts due to the presence of impulsive noise) then each error in a given burst clearly yields less information, on average, about the error statistics than in the independent error case. (At its most obvious each subsequent error in a burst is less surprising, and therefore less informative, than the first error.) It follows that a greater number of errors would need to be counted for a given P_e confidence interval in this case than in the independent error case.

13.7.2 Quasi-analytic simulation

Quasi-analytic (QA) simulation can dramatically reduce the required computer power and/or run time compared with Monte Carlo methods. This is because the QA method simulates only the effect of system induced distortion occurring in the signal rather than including the effects of additive noise. It does depend, however, on a knowledge of the total noise pdf at the decision circuit input. The essence of QA simulation is best illustrated by a binary signalling example. Figure 13.43(a) shows an (unfiltered) baseband binary signal, $v_s(t)$, representing the output of a binary information source.

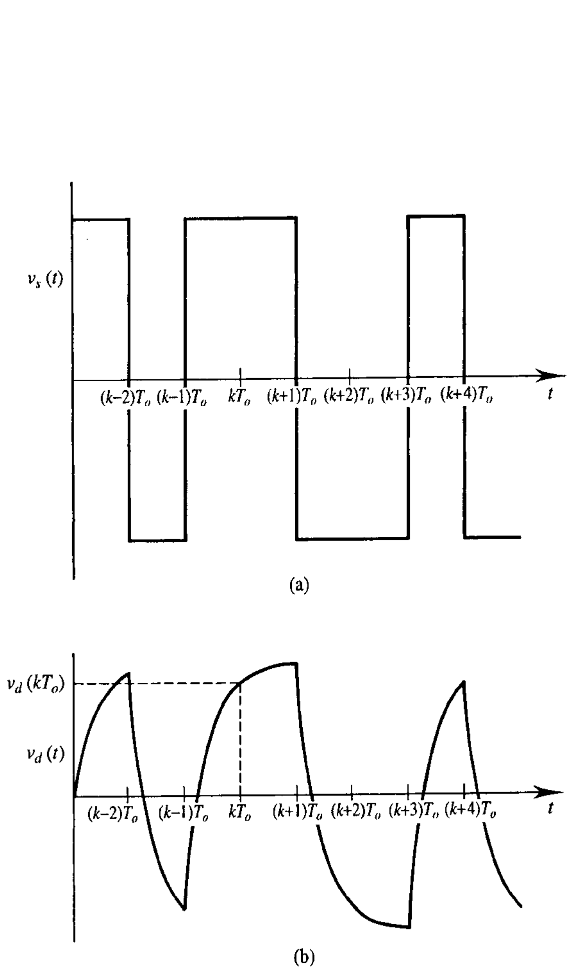


Figure 13.43 *Signals at: (a) transmitter binary source output; (b) receiver decision circuit input after transmission through a distorting but noiseless channel.*

Figure 13.43(b) shows the demodulated, but distorted, signal, $v_d(t)$, at the decision circuit input. If purely additive noise with pdf $p_n(v)$ is present at the decision circuit input the total pdf of signal plus noise at the kth sampling instant is given by:

$$p_{s+n}(v) = p_n\,[v - v_d(kT_0)] \tag{13.70}$$

where $v_d(kT_0)$ is the decision instant signal. $v_d(kT_0)$ will depend on the history of the bit sequence via the intersymbol interference due to the impulse response (i.e. distorting effect) of the entire system.

The probability, $P_e(k)$, that the noise (if present) would have produced an error at the kth bit sampling instant is:

$$P_e(k) = \int_{-\infty}^{0} p_n[v - v_d(kT_0)]\,dv = \int_{-\infty}^{-v_d(kT_0)} p_n(v)\,dv \tag{13.71(a)}$$

if $v_s(kT_0) > 0$ (i.e. v_s represents a transmitted digital 1), and is:

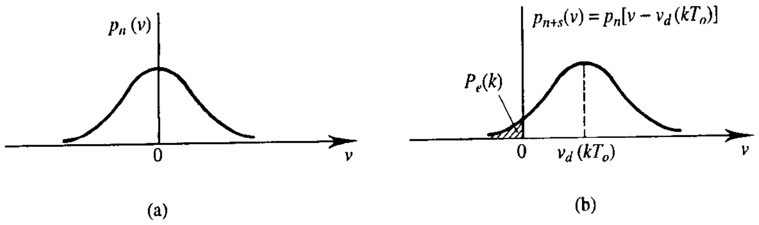


Figure 13.44 *Pdfs of: (a) noise only; (b) signal plus noise at kth sampling instant.*

$$P_e(k) = \int_0^{\infty} p_n[v - v_d(kT_0)]\, dv = \int_{-v_d(kT_0)}^{\infty} p_n(v)\, dv \qquad (13.71(b))$$

if $v_s(kT_0) < 0$ (i.e. v_s represents a transmitted digital 0). $p_n(v)$, $p_n[v - v_d(kT_0)]$, and $v_d(kT_0)$ are shown in Figure 13.44. If $p_n(v)$ is Gaussian then the evaluation of the integrals is particularly easy using error function look-up tables or series approximations.

The overall probability of error is found by averaging equations (13.71) over many (N, say) bits, i.e.:

$$P_e = \frac{1}{N} \sum_{k=1}^{N} P_e(k) \qquad (13.72)$$

N must be sufficiently large to allow essentially all possible combinations of bits, in a time window determined by the memory of the system, to occur. This ensures that all possible distorted signal patterns will be accounted for in the averaging process.

EXAMPLE 13.1

As an example of the power and utility of proprietary simulation packages a simplified model of a satellite communications system is analysed here using Signal Processing WorkSystem (SPW)™ marketed by Comdisco Systems, Inc. SPW is immensely powerful as a simulation tool and only a

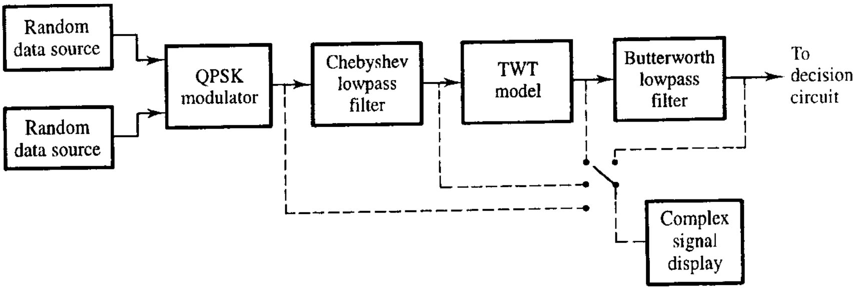


Figure 13.45 *Block diagram for example simulation.*

fraction of its facilities are illustrated in this example. The system is shown in Figure 13.45 which has been redrawn to simplify that produced using SPW's block diagram editor. It consists of a pair of (independent) random binary data sources which provide the input to a QPSK modulator, Figure 11.23. The symbol rate of each source is 1.0 baud.

The output of the modulator is an equivalent baseband signal and is therefore complex. The transmitted, uplink, signal is filtered by a 6-pole Chebyshev filter. The bandwidth of this (equivalent baseband) filter is 1.1 Hz. A satellite channel is modelled analytically using a non-linear input/output characteristic typical of a travelling wave tube (TWT) amplifier operating with 3.0 dB of input back-off which accounts for both AM/AM and AM/PM distortion (see section 14.3.3). A parameter of the TWT model is its average input power which is set to 2.0 W. The received, downlink, signal is filtered by a 6-pole Butterworth filter. Signal sink blocks are used to record the time series data generated as the simulation progresses. It is these signal records which are analysed to produce the required simulation results.

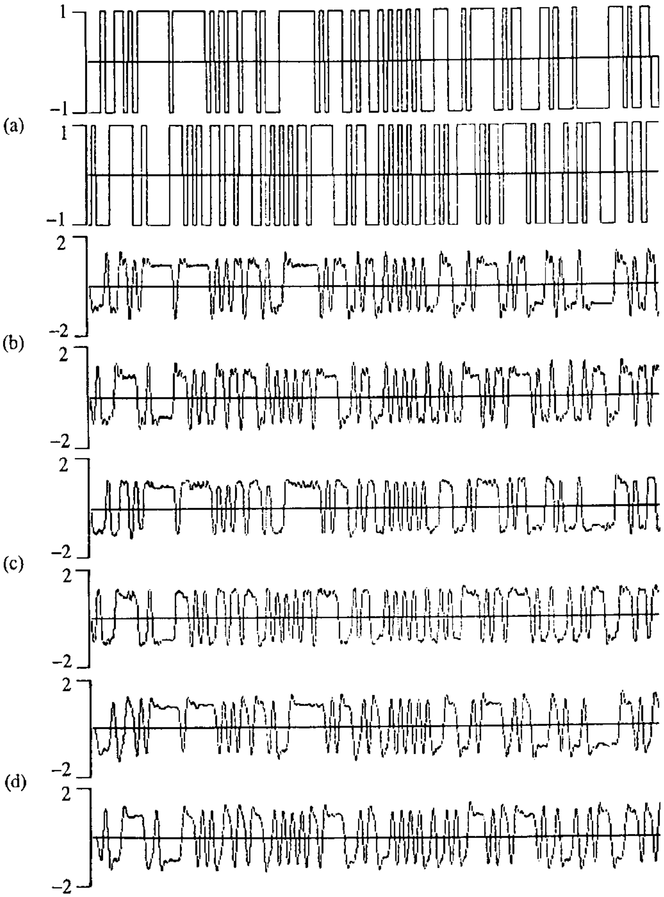


Figure 13.46 *In-phase (upper) and quadrature (lower) time series data at (a) QPSK modulator output and the outputs of: (b) Chebyshev filter; (c) TWT; (d) Butterworth filter.*

The simulation sampling rate used is 16 Hz which, for a baud rate of 1.0, corresponds to 16 sample/symbol. The number of samples simulated (called iterations in SPW) is 3,000. Unrealistic parameters in simulations (e.g. unit baud rate and filter bandwidths of the order of 1 Hz) are typical. It is, usually, necessary only for parameters to be correct relative to each other for the simulation to give useful results. (The interpretation of those results must, of course, be made in the proper context of the parameters used.) Figure 13.46(a), (b), (c) and (d) show the simulated (complex) time series data at the input to the QPSK modulator, and the outputs from the Chebyshev filter, the TWT and the Butterworth filter. Note that the 'random' component of the signals in Figure 13.46(b), (c) and (d) is not due to noise but is due entirely to the distortion introduced by the channel.

Figure 13.47(a) and (b) show the magnitude and phase spectra of the pre- and post-filtered transmitted uplink signals respectively, calculated using a 1024 (complex) point FFT. (The time series were windowed with a Hamming function prior to the FFT in this case.) Figure 13.47(c) shows the spectrum of the TWT output. It is interesting to note the regenerative effect that the TWT non-linearity has on the sidelobes of the QPSK signal, previously suppressed by the Chebyshev filter. (This effect was briefly discussed in section 11.4.4.) The frequency axes of the spectra are easily calibrated remembering that the highest observable frequency (shown as 0.5 in the SPW output) corresponds to half the sampling rate, i.e. 16/2 = 8 Hz, see section 13.5.3.

A QA simulation routine available in SPW has been run to produce the SER versus E_b/N_0 curve shown in Figure 13.48. QA simulation is applicable if the system is linear between the point at which noise is added and the point at which symbol decisions are made. In this example we assume that noise is added at the satellite (TWT) output. This corresponds to the, often realistic, situation in which the downlink dominates the CNR performance of a satellite communication system, see section 14.3.3.

The way in which SPW implements QA simulation is as follows:

1. The equivalent noise bandwidth, B_N, of the system segment between the noise injection point and the receiver decision circuit is estimated from a (separate) simulation of this segment's impulse response, $h(t)$, i.e.:

$$B_N = \int_0^{\infty} |h(t)|^2 \, dt \qquad (13.73)$$

2. The signal power at the input to the receiver, C (in this example the input to the Butterworth filter) is estimated from the time series contained in the appropriate signal sink file (i.e. iagsig/twtout.sig, see Figure 13.46(c)).
3. The required range and interval values of E_b/N_0 are specified and, for each value of E_b/N_0, the normalised (Gaussian) noise power, N (= σ^2), is found using:

$$\sigma^2 = N_0 B_N \quad (\mathrm{V}^2) \qquad (13.74(\mathrm{a}))$$

where:

$$N_0 = \frac{CT_b}{(E_b/N_0)} = \frac{C/(R_s H)}{(E_b/N_0)} \quad (\mathrm{V}^2/\mathrm{Hz}) \qquad (13.74(\mathrm{b}))$$

(Here H is the number of binary digits/symbol, i.e. it is the entropy assuming zero redundancy. Thus in this example $H = 2$.)

4. If *centre* point sampling is required then the distances, d_1 and d_2, from the decision thresholds of the (complex) sample at the *centre* of each received symbol is calculated, see Figure 13.49. (For MPSK and MQAM systems SPW assumes constellations which are regular, thus allowing

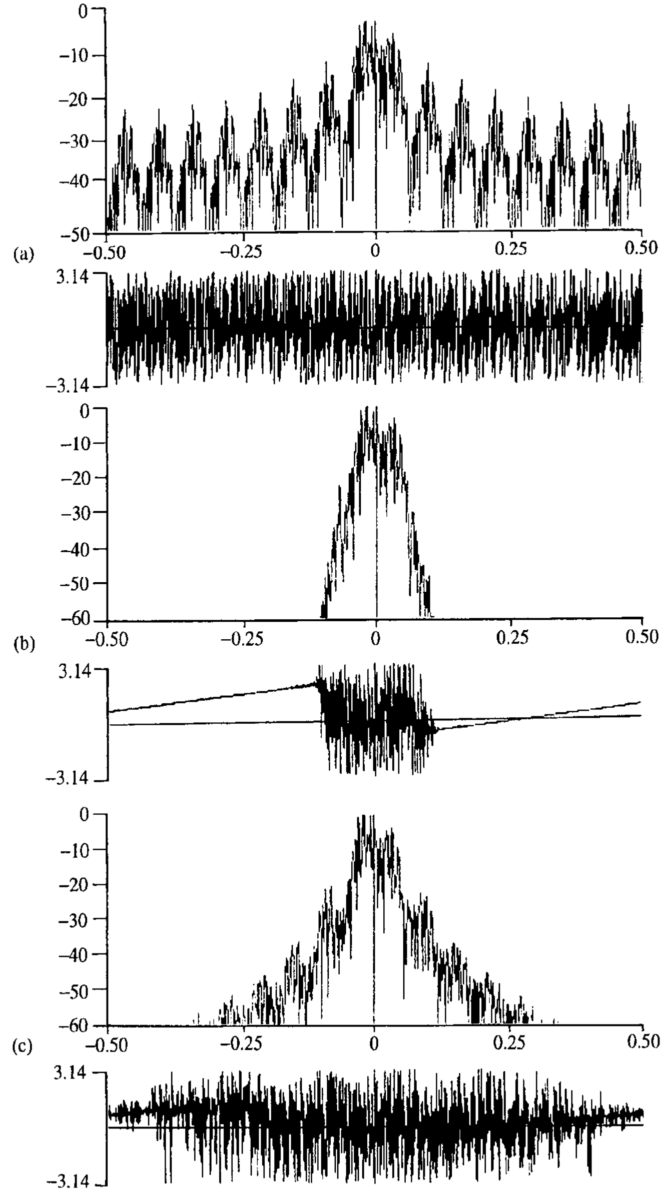


Figure 13.47 *Amplitude (dB) and phase (rad) spectra for signal at outputs of: (a) QPSK modulator; (b) Chebyshev filter; (c) TWT.*

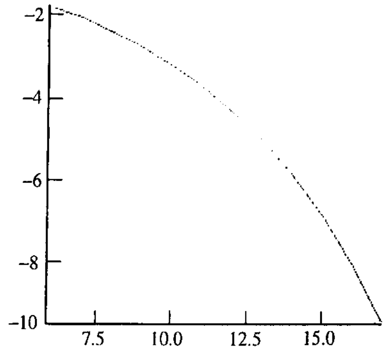


Figure 13.48 *Result of P_e versus E_b/N_0 (dB) for a quasi-analytic (QA) analysis of system shown in Figure 13.45. (P_e is plotted on a logarithmic scale so 10^{-2} is represented by −2.)*

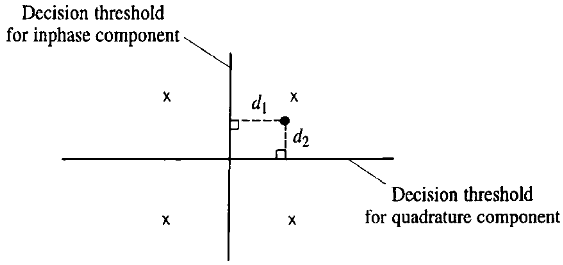


Figure 13.49 *Definition of QPSK received sample distances d_1 and d_2 from the decision thresholds (× represents transmitted constellation point and • a received sample).*

all constellation points to be folded into the first quadrant before this calculation is carried out.) Although centre point sampling is typical, SPW allows the user to specify which sample within the symbol is to be used as the decision sample.

5. The probability of error for the particular sampling point selected, in the particular symbol being considered, is then calculated using d_1, d_2, σ and the error function.
6. The probability of error is averaged over many (N) symbols. (N is typically a few hundred but should be large enough to allow all sequences of symbols, possible in a time window equal to the duration of the impulse response estimated in 1, to occur at least once.)

Many other analysis routines are provided in SPW, for example the scatter plot of Figure 13.50 which shows the scatter of simulation points, at the Butterworth filter output (decision circuit input), about the nominal points of the QPSK constellation diagram. Figure 13.51 shows the eye diagram (across two symbols) of the received signal's inphase components at the Butterworth filter output, on a much expanded timescale compared to Figure 13.46.

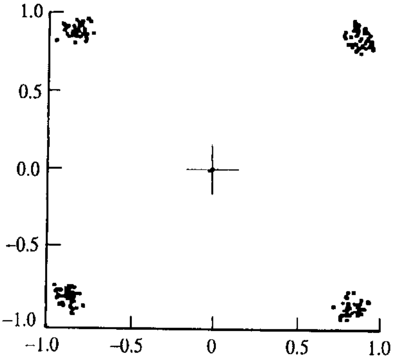


Figure 13.50 *Scatter of simulation QPSK constellation points at Butterworth filter output (decision circuit input) in Figure 13.45.*

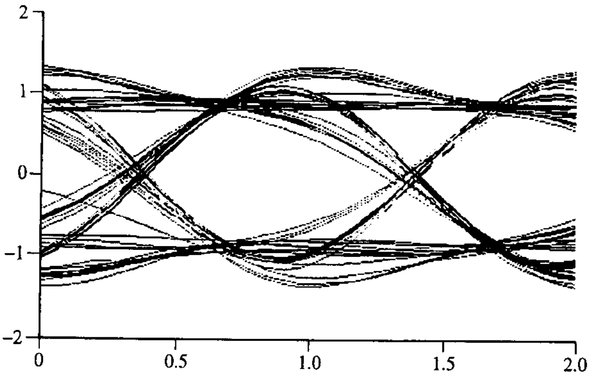


Figure 13.51 *Eye diagram for in-phase component (I) of signal at Butterworth filter output.*

13.8 Summary

Simulation is now a vital part of the design process for all but the simplest of communications systems. It enables performance to be assessed in the presence of noise, interference, and distortion, and allows alternative design approaches to be compared before an expensive construction phase is implemented. Since simulation is based on discrete samples of underlying continuous signals, it is usual for passband systems (with the exception of those which are non-linear and have long memory) to be simulated as equivalent (complex) baseband processes. The distortion introduced by the sampling and quantisation, required for simulation, needs to be carefully considered to ensure it does not significantly alter the simulation results. Noise modelling using random numbers or pseudo-random bit sequences must also be implemented carefully to ensure that simulated noise samples faithfully represent both the spectral and pdf properties of the actual noise present in the system.

Transformation between time and frequency domains during a simulation is frequently required. This is because some simulation processes are more easily (or more efficiently) implemented in one domain rather than the other, and also because some simulation results are more easily interpreted in one domain rather than the other. Digital signal processing algorithms (principally the FFT) are therefore used to translate between domains. An adequate understanding of effects such as smoothing, leakage, windowing and zero padding is required if these algorithms are to be used to best effect. It is also important that the output of DFT/DFS software is properly interpreted and processed if a numerically accurate representation of a power, or energy, spectral density is required.

Finally, in the assessment of most types of digital communications system the principal objective measure of performance is BER. For systems which are linear, between the point at which noise is introduced and the point at which symbol decisions are made, quasi-analytic (QA) simulation is extremely efficient in terms of the computer resources required. QA simulation does require the noise pdf at the decision circuit input to be known, however. The most general method of estimating BER uses Monte-Carlo simulation in which neither a linearity restriction (on the system through which the noise passes prior to detection), nor any apriori knowledge of the noise characteristics at the decision circuit input, is needed. Monte-Carlo simulation can become very expensive in terms of computer power and/or run time, however, if accurate estimates of small error probabilities are required.

Part Three

Applications

Part 3 shows how the principles described in Part Two are applied in a selection of fixed and mobile data applications for voice and video transmission.

The link budget analysis presented in Chapter 12 is extended in Chapter 14 to less idealised fixed service, terrestrial and satellite, microwave communication systems, and includes important propagation effects such as rain fading, multipath fading and signal scintillation. The special problems posed by the exceptionally long range of satellite systems are also discussed as are frequency allocations, multiplexing and multiple accessing schemes.

Mobile, cellular and paging applications are described in Chapter 15 including examples of current systems such as personal cordless, GSM 900, DCS 1800 and DECT. These systems all limit their transmissions to small geographical areas, or cells, permitting frequency reuse in close proximity, without incurring intolerable levels of interference. This maximises the number of active users per cell who can be accommodated per MHz of allocated bandwidth. This chapter also includes evolving standards for CDMA spread spectrum cellular radio, and future satellite-mobile systems.

Finally, Chapter 16 discusses the specific requirements of digitisation, transmission and storage for video applications. It includes examples of HDTV development as well as low bit rate video compression coders using transform and model based coding techniques, as employed in MPEG and other video coding standards.