CHAPTER 1

SIGNALS

A *signal* is a physical quantity, or quality, which conveys information. For example, the voice of my friend is a signal which causes me to perform certain actions or react in a particular way. My friend's voice is called an *excitation*, and my action or reaction is called a *response*. The evaluation (conversion) from excitation to response is called *signal processing*.

Signals have many different origins. They may be measurements of physical phenomenon such as propagation of seismic waves, traffic noise levels, voltage variations in an electric circuit, and pressure variations in a hydraulic control system. They may be produced by a digital computer, as in text-to-speech synthesis, electronic music, computer graphics, and synthetic images.

A typical reason for signal processing is to eliminate or reduce an undesirable signal. The undesirable signal may be electrical noise in a radio signal or the power supply sinusoidal signal at a frequency of 60 Hz (or 50 Hz) in measurements of traffic noise levels. The traffic noise levels would be the desired signal for a noise detector, but the undesirable signal in a mobile speech communications system.

From a signal processing perspective, a signal must be capable of being observed. For the description and processing of signals, it is important to have a variety of representations of signals. Regardless of the origin of a signal (e.g., mechanical, electrical, acoustic, and biological), we generally convert the original signal into a form that is suitable for further processing and observation.

One fundamental representation of a signal is as a function of at least one independent variable. The variation of the signal value as a function of the independent variable is called a *waveform*. The independent variable often represents *time*. Other possibilities include spatial distance or an index of a data vector. For example, recorded data on a magnetic tape or compact disc (CD) may be a signal that is a function of time. Or, instead, it may be a function of distance (from the beginning of the tape as shown on the numeric display of the tape player) or of the CD track number and the position in the track of the digitized signal on the CD.

In this book, we define a *signal* as a function of one independent variable that contains information about the behavior or nature of a phenomenon. We assume that the independent variable is time even in cases where the independent variable is a physical quantity other than time. The rest of this chapter discusses mathematical representations of signals and introduces the two computer environments, MATLAB [1] and *Mathematica* [2], which we will use to analyze and process signals.

1.1 SIGNAL CLASSIFICATION

1.1.1 Continuous or Analog Signals

We define the *continuous signal*, $x(t)$, as a signal that exists at every instant of time, t. In the jargon of the trade, a continuous signal is often referred to as *continuous time* or *analog*. The independent variable t is a *continuous* variable. The function $x(t)$ can assume any value over a *continuous* range of numbers. For example, the functions

$$x(t) = e^{-t}$$

$$x(t) = \sin(100\pi t)$$

$$x(t) = 3 + 10t + 2|\sin(100\pi t)|$$

are continuous signals.

Signals can be represented by graphs as shown in Fig. 1.1.

Note that x is the symbol representing the signal, t is the symbol representing the independent variable, and $x(t)$ means the *value* of the signal x at time t as shown in Fig. 1.1a. Figure 1.1a plots a continuous sinusoid, $x(t) = \sin(2\pi t)$, which is a periodic signal. Figure 1.1b plots a continuous signal $x(t)$ which is zero for $t < 0$ and $t > 4$. Figure 1.1c plots a continuous signal which is periodic, as is the sinusoidal signal in Fig. 1.1a, but not sinusoidal. For the signals plotted in Figs. 1.1a–c, we can find the values for $t < 0$ as well as for $t > 3$. Figure 1.1d shows a signal that is not periodic. From the graph, we can estimate its values on the interval $-1 \leq t \leq 3$. We cannot calculate its values outside this interval without previous knowledge of the mathematical formula describing the signal.

In recent years, tremendous advances have occurred in programmable, configurable, and dedicated digital hardware as well as in digital technology. The most suitable signal to be processed by a digital computer or digital hardware is a *digital signal*, that is, a discrete-time signal whose values are represented by digits (e.g., by binary or decimal numbers). For many applications a digital signal is more convenient than an analog signal, especially when signals are transmitted and stored. For these reasons it is of interest to transmit, store, and analyze digital signals in such a way that we can restore the initial analog signal without loss or with minimum acceptable loss of information.

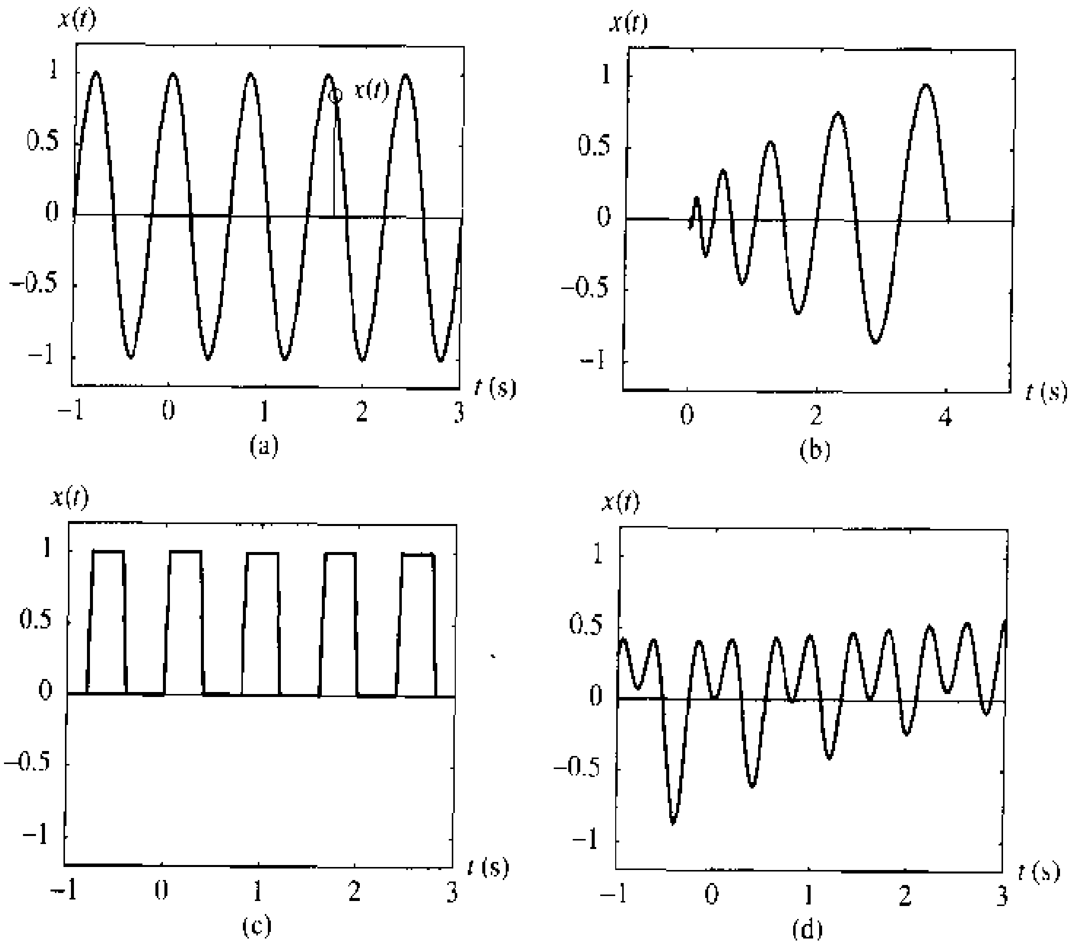

Figure 1.1 Continuous (analog) signals.

1.1.2 Discrete-Time Signals

A signal defined only for discrete values of time is called a *discrete-time signal* or simply a *discrete* signal. It may have been obtained by taking samples of an analog signal at discrete instants of time. In this book, we will only concern ourselves with uniform sampling by sampling every T units of time:

$$x(kT) = x(t)$$

$$t = 0, \pm T, \pm 2T, \pm 3T, \ldots$$

The discrete signal obtained from the continuous signal given in Fig. 1.1 is shown in Fig. 1.2.

The process of converting analog signals to digital signals takes a finite amount of time Δt at each sample, where $\Delta t \leq T$. Sampling produces a train of pulses of duration Δt, as shown in Fig. 1.3. Within the Δt time interval, the signal generally varies. If this interval is very short, as in Fig. 1.2, then the pulses are assumed to be rectangular and the pulse amplitude is uniquely defined by the instantaneous signal value. On the other hand, if Δt is long, the amplitudes of pulses are not uniquely defined and a measure for them must be specified. A typical measure is the signal value (a) at the beginning of, (b) in the middle of, (c) at the end of, or (d) the average signal value over the interval Δt.

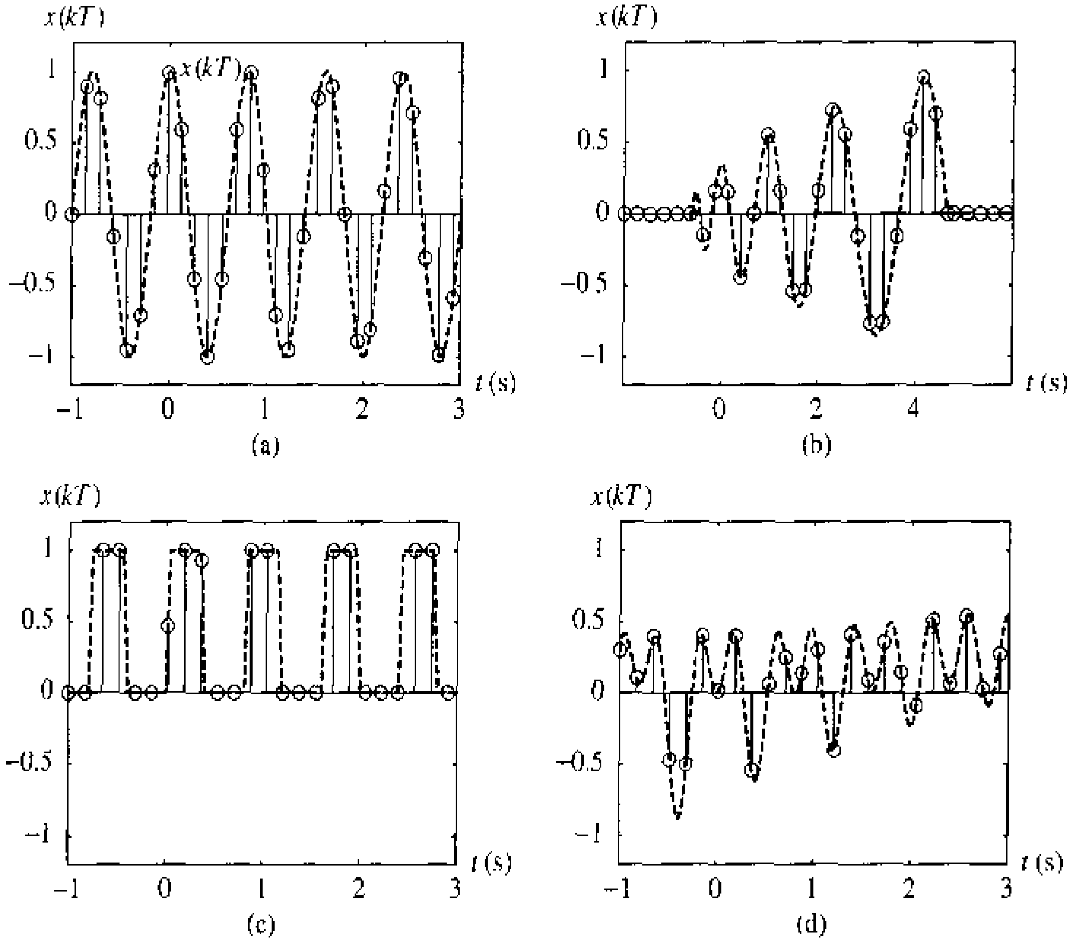

Figure 1.2 Discrete-time (discrete) signals, which are sampled but not quantized.

If the sampling process interval Δt is equal to the interval between successive samples T, and if we take the value of the signal at the beginning of Δt, the discrete-time signal looks like that in Fig. 1.4.

In this book, we will be using the MATLAB and *Mathematica* computer software environments for representing, analyzing, and manipulating signals. We used the MATLAB `plot` command to create the waveform plots in Fig. 1.1. The next three figures, Figs. 1.2, 1.3, and 1.4, were generated by the MATLAB `stem`, `bar` and `stairs` commands, respectively. We will discuss in more detail the use of MATLAB in Sections 1.5 and 1.6 and *Mathematica* in Sections 1.7 and 1.8.

All three discrete-time signals convey the same information: the values of the continuous signal $x(t)$ at regular intervals. The parameter T is called the *time step* or *sample interval*. A related quantity is the *sampling rate* or *sampling frequency* representing the number of samples per unit time

$$f_0 = \frac{1}{T}$$

When the units of T is seconds (s), the units of f_0 is Hertz (Hz).

We generally start observing a signal from an arbitrary point of time t_0. We will set $t_0 = 0$ and designate t to represent the time that has elapsed since t_0. We will uniformly sample continuous-time signals at $t = kT$, where $k = 0, \pm 1, \pm 2, \ldots$. We

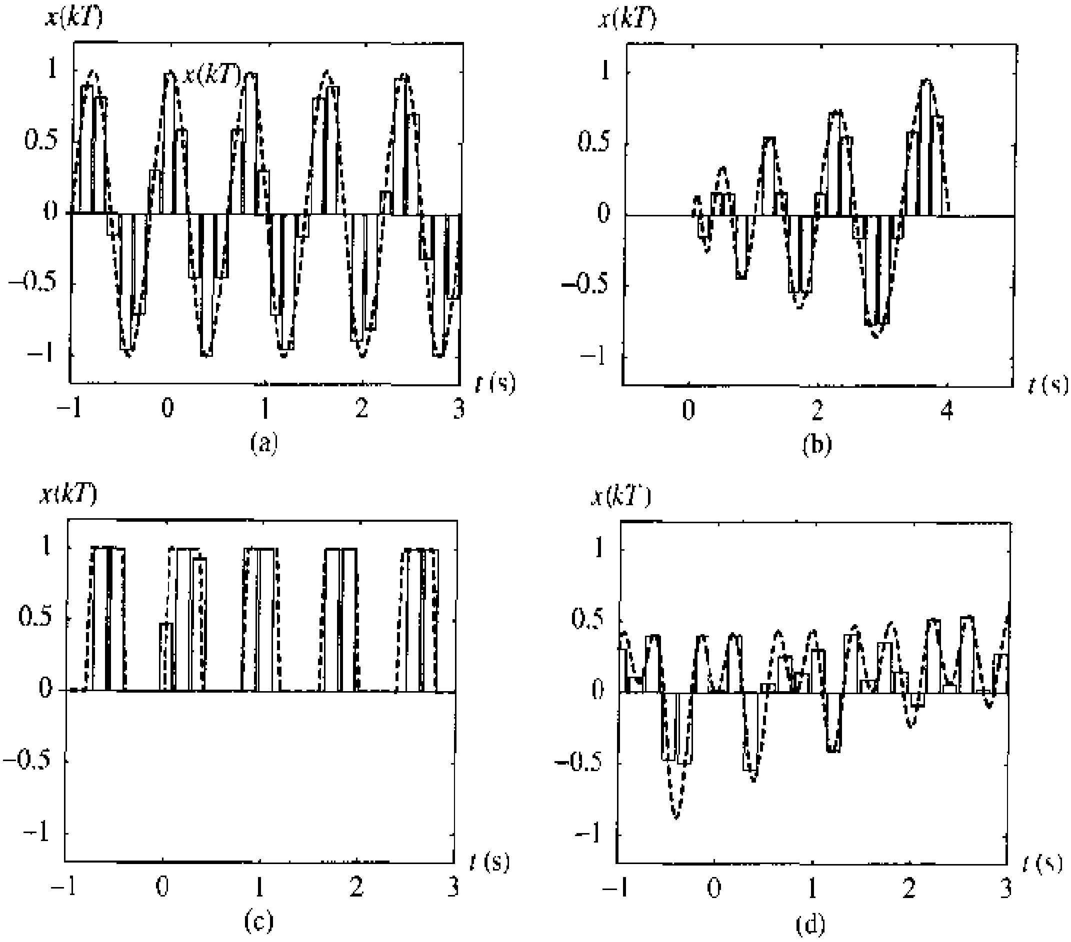

Figure 1.3 Discrete-time (discrete) signals, which are sampled but not quantized.

designate the corresponding sequence of samples $x(kT)$ by the set notation $\{x_{(k)}\}$. In order to simplify notation, we drop the curly braces and use $x(k)$ or $x_{(k)}$ or x_k rather than $x(kT)$ or $\{x_{(k)}\}$.

1.1.3 Digital Signals

The purpose of sampling a continuous signal is to transmit, store, or process a limited number of samples. It is also of interest to represent the values of samples by a limited number of digits. By using fewer digits we attain faster transmission and smaller storage requirements for the information. Thus, we utilize the quantized samples, $\hat{x}_k$, rather than the true samples of infinite accuracy, x_k,

$$\hat{x}_k = \text{quantize}(x_k) = Q[x_k]$$

When we quantize samples, we introduce an error in the representation. We define this quantization error as the difference between the quantized signal and the original signal, $e_k = \hat{x}_k - x_k$. For example, we quantize the discrete-time signals in Fig. 1.2 so that the possible values of $\hat{x}_k$ are from the set of five values $\{-1, -0.5, 0, 0.5, 1\}$:

$$\hat{x}_k \in \{-1, -0.5, 0, 0.5, 1\}$$

Figure 1.5 plots the quantized versions of these signals.

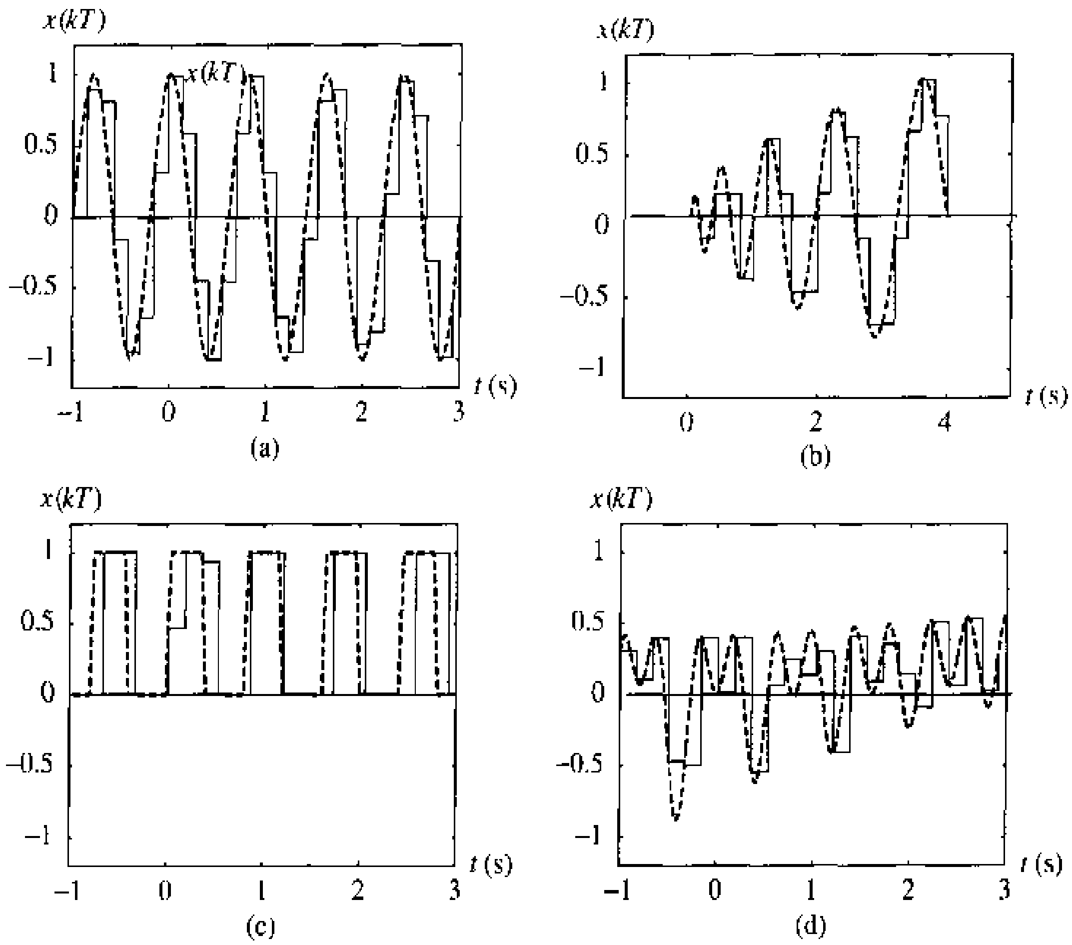

Figure 1.4 Discrete-time (discrete) signals, which are sampled but not quantized.

Each sample of $\hat{x}_k$ can be represented by its sign $\pm$ and the three magnitude values $\{0, 0.5, 1\}$ according to the following quantization strategy:

$$\begin{cases} \hat{x}_k = -1.0 & \text{for} \quad x_k < -0.6 \\ \hat{x}_k = -0.5 & \text{for} \quad -0.6 \le x_k < -0.1 \\ \hat{x}_k = 0.0 & \text{for} \quad -0.1 \le x_k \le 0.1 \\ \hat{x}_k = 0.5 & \text{for} \quad 0.1 < x_k \le 0.6 \\ \hat{x}_k = 1.0 & \text{for} \quad 0.6 < x_k \end{cases}$$

In binary notation, we can use three binary digits for representation: one for sign and two for representing the values $\{0, 0.5, 1\}$. This is known as sign-magnitude representation. Another three binary digit representation would be $110_2 = -1$, $111_2 = -0.5$, $000_2 = 0.0$, $001_2 = 0.5$, and $010_2 = 1.0$. If we extended this latter representation to include $101_2 = -1.5$, $100_2 = -2$, and $011_2 = 1.5$, then we would have a two's complement representation. The two's complement representation is the format most commonly used by programmable processors.

Although the resolution of the quantizer affects the accuracy of the data, as seen in Fig. 1.5, the error can be reduced by increasing the number of digits (possible values

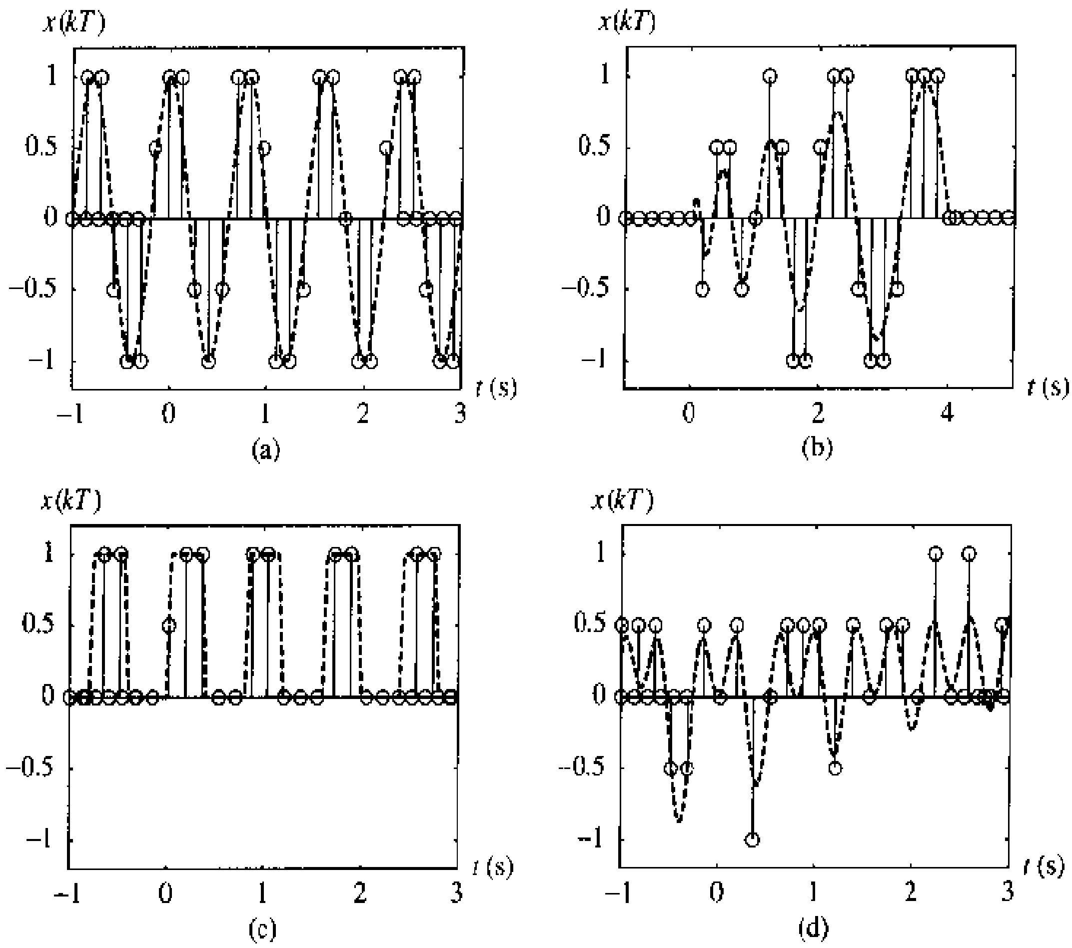

Figure 1.5 Digital signals, which are sampled and quantized.

for $\hat{x}_k$). The choice of the digits for representing the quantized signal is very important, and the quantization should be done properly: In transmitting, storing, and processing we prefer less digits, but with too small a number of digits we can lose information from the signal. The two opposing requirements must be satisfied: (1) Minimize number of digits to facilitate the signal transmission or storing, and (2) maximize number of digits to keep the quantization error as low as necessary in order to preserve the information contained in the signal.

The *quantization* is a many-to-one mapping. All the values that fall within a continuous band are mapped to the edge value of the band, as shown in Fig. 1.6. Figure 1.6 illustrates the quantization of a signal into 80 discrete levels in four different ways using the MATLAB commands `ceil`, `floor`, `round`, and `fix`. The maximum quantization error is lower than the amplitude of the undesirable signal, and the number of quantization levels is not very large. For digital signal processing, sampled values are typically mapped into an m-bit binary number. A sample of the signal shown on Fig. 1.6 can be represented by $m = 7$ bits or by 2^7 discrete values ($2^7 \geq 80$).

A *sample set* or *data sequence* is a collection of numbers which represents the quantized samples. In signal processing, we generally process finite sequences or finite portions of infinite sequences at any given time. The samples may be plotted versus the sample number k or time $t = kT$.

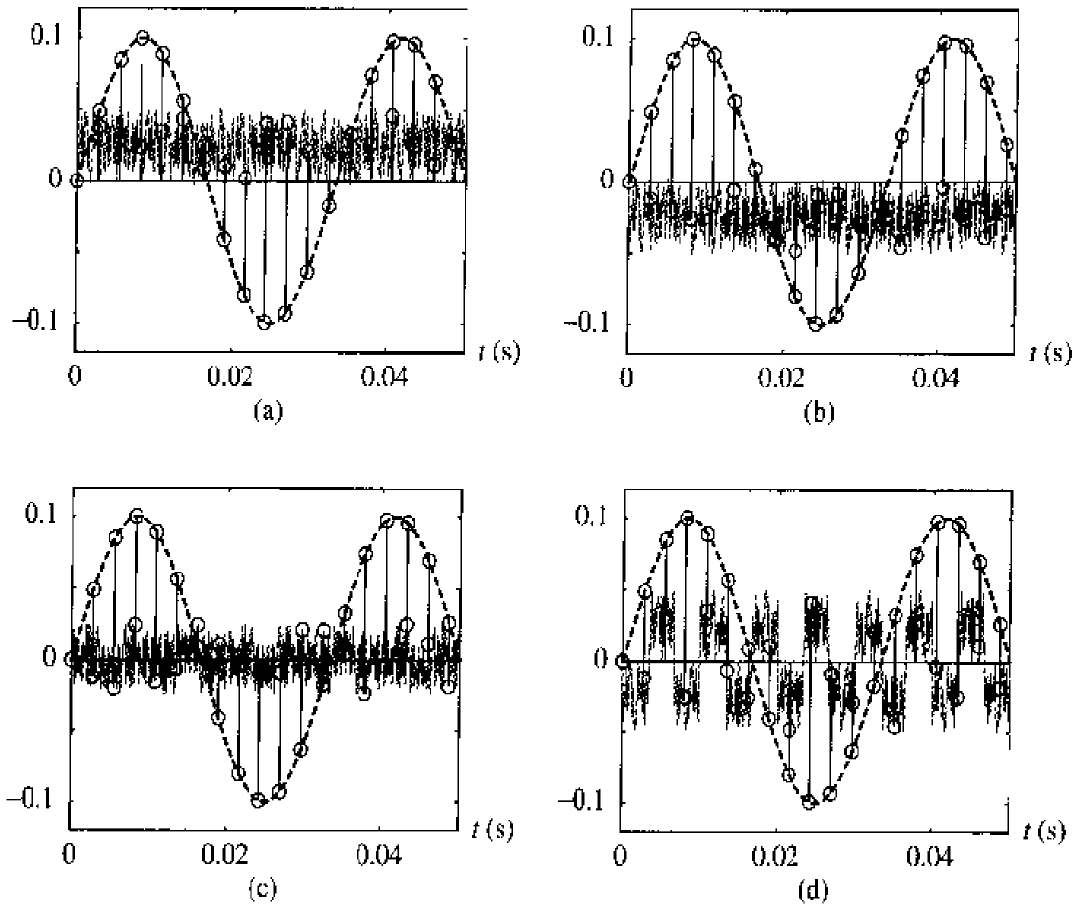

Figure 1.6 Undesirable signal $0.1\sin(60\pi t)$ and quantization error: (a) $\hat{x}(t) \geq x(t)$, (b) $\hat{x}(t) \leq x(t)$, (c) $|\hat{x}(t)| \leq |x(t)|$, (d) $|\hat{x}(t)| \geq |x(t)|$.

1.1.4 Deterministic Signals

A signal that can be described by an explicit mathematical form is *deterministic*. A deterministic signal can be *periodic* or *aperiodic*. A periodic signal consists of a basic "shape" of finite duration, T_{basic}, that is replicated infinitely as shown in Figs. 1.1a and 1.1c. Figure 1.1c plots a signal that is periodic and can be represented as an infinite sum of a constant and sinusoidal signals,

$$x(t) = X_0 + \sum_{i=1}^{+\infty} X_i \sin(i\, 2\pi f_{basic} t + \phi_i) \qquad \text{where} \qquad f_{basic} = \frac{1}{T_{basic}}$$

The sinusoidal terms $X_i \sin(i\, 2\pi f_{basic} t + \phi_i)$ are referred to as *harmonics*. An *aperiodic signal* does not have a repetitive form, such as a single rectangular pulse or the signals plotted in Figs. 1.1b and 1.1d.

1.1.5 Random (Nondeterministic) Signals

A signal that cannot be described in an explicit mathematical form is called *random*, also known as *nondeterministic* or *stochastic*. For example, the difference between a sinusoidal signal $x(t)$ and the quantized signal $\hat{x}(t)$ is a random signal as shown in Fig. 1.6. This difference we call the *quantization noise* or simply *noise*. Although we cannot explicitly describe a random signal, some of its properties can be useful in signal processing. For example, we can conclude that the *mean* (average) values of the signals in Figs. 1.6c and 1.6d are zero:

$$\mu_x = \text{mean}\, x(t) = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} x(t)\, dt$$

On the other hand, the mean value is positive for Fig. 1.6a and negative for Fig. 1.6b. The *variance*

$$\sigma_x = \text{variance}\, x(t) = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} (x(t) - \mu_x)^2\, dt$$

of the signal from Fig. 1.6c is lower than the variance of the signal from Fig. 1.6d. The mean and variance are important properties for random signals in the design of signal processing systems.

1.2 THE SAMPLING THEOREM

Signals occurring in nature are generally continuous. If we choose to use a digital system such as a computer to process signals, then we will need to convert the continuous signals into digital signals before processing and, possibly, convert the resulting digital signals to continuous signals. Our digital system should eliminate undesirable and interfering signals.

Prior to processing digital signals, we have to make a choice of the sampling period T and the number of quantization levels. The wrong choice of the sampling period can produce serious errors and loss of information. A key theorem, referred to as the *Sampling Theorem*, gives the guidelines to select the required T.

In order to give a reader an intuitive feel for the importance of the sampling theorem we will consider a single sinusoidal signal. The sampling theorem requires that a continuous sinusoidal signal of frequency f_a, $x_a(t) = \sin(2\pi f_a t + \phi_a)$, be sampled at a rate f_0 greater than twice f_a:

$$f_0 = \frac{1}{T} \qquad \text{such that} \qquad f_0 > 2 f_a$$

The samples of the sinusoidal signal $x_a(t)$ are

$$x_{a,k} = \sin((2\pi f_a)kT + \phi_a)$$

or

$$x_{a,k} = \sin\left(2\pi \frac{f_a}{f_0} k + \phi_a\right)$$

where ϕ_a designates the initial phase for $t = 0$, so $x_a(0) = \sin(\phi_a)$.

For the case $f_0 = 2f_a$, we have

$$x_k = \sin(\pi k + \phi_a)$$

resulting in the data sequence

$$x_k = (-1)^k \sin\phi_a, \qquad k = 0, 1, 2, \ldots$$

as shown in Fig. 1.7. The sample values x_k depend on ϕ_a and take the maximum value $(x_k)_{\max} = 1$ for $\phi_a = \frac{\pi}{2}$. For $\phi_a = 0$, as shown in Fig. 1.7a, we might conclude that there is no signal at all. For this reason, the condition for properly sampling an analog sinusoidal signal is $f_0 > 2f_a$.

In the next example we will consider two signals: (1) a signal $x_a(t)$ with $f_a > \frac{1}{2}f_0$ and (2) a signal $x_b(t) = \sin(2\pi f_b t + \phi_b)$ with $f_b < \frac{1}{2}f_0$. We assume that $f_a = \frac{1}{2}f_0 + \Delta f$ and $f_b = \frac{1}{2}f_0 - \Delta f$, where $\Delta f > 0$. The corresponding sample sequences are

$$x_{a,k} = \sin\left(2\pi\left(\frac{f_0}{2} + \Delta f\right)kT + \phi_a\right)$$

$$x_{b,k} = \sin\left(2\pi\left(\frac{f_0}{2} - \Delta f\right)kT + \phi_b\right)$$

which are shown in Fig. 1.8a, for $\Delta f = 0.1 f_0$, and Fig. 1.8b, for $\Delta f = 0.25 f_0$.

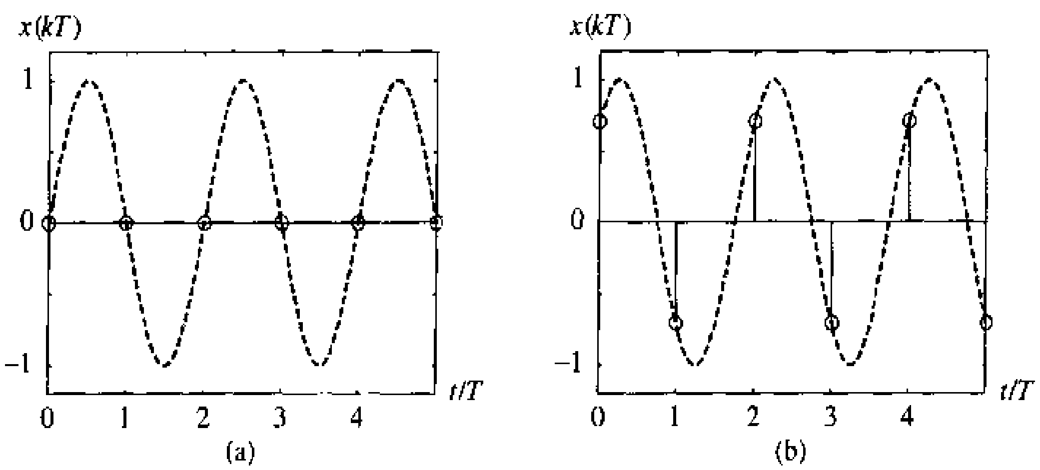

Figure 1.7 Sinusoidal signal samples for $f_a = \frac{1}{2}f_0$, and (a) $\phi_a = 0$, (b) $\phi_b = \pi/4$.

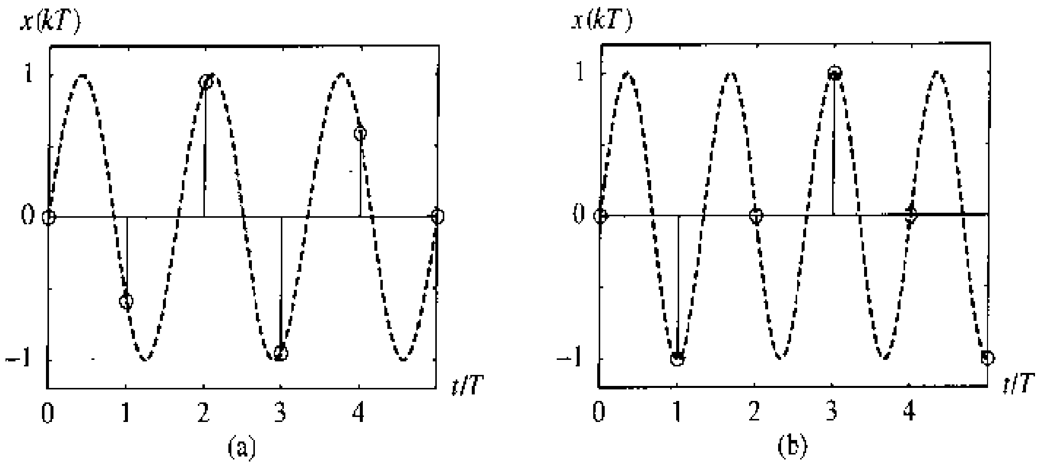

Figure 1.8 Sinusoidal signal samples for (a) $x(t) = \sin(2\pi(1/2 \pm 0.1) f_0 t)$, (b) $x(t) = \sin(2\pi(1/2 \pm 1/4) f_0 t)$.

From the samples shown in Fig. 1.8a, we cannot reconstruct the original continuous-time signal. An infinite number of continuous-time signals exist that yield the same sequence. For example, we can reconstruct the following three continuous-time signals:

$$y_a(t) = \sin\left(2\pi\left(\frac{f_0}{2} + \Delta f\right)t + \phi_a\right)$$

$$y_b(t) = \sin\left(2\pi\left(\frac{f_0}{2} - \Delta f\right)t + \phi_b\right)$$

$$y_c(t) = \frac{1}{2}\sin\left(2\pi\left(\frac{f_0}{2} + \Delta f\right)t + \phi_a\right) + \frac{1}{2}\sin\left(2\pi\left(\frac{f_0}{2} - \Delta f\right)t + \phi_b\right)$$

We use the symbol y for the reconstructed signal and use x for the sampled signal.

From Fig. 1.8b, we find that the first two sample values are $x_0 = 0$ and $x_1 = -1$. The values of the reconstructed signals are $y_a(0) = \sin(\phi_a) = x_0 = 0$, yielding $\phi_a = 0$. Next, $y_a(T) = \sin(2\pi(\Delta f + f_0/2)T) = x_1 = -1$, and we obtain $2\pi\Delta f T + \pi = 3\pi/2 \Rightarrow \Delta f = f_0/4$. Finally, $y_a(t) = \sin(1.5\pi f_0 t)$. In the same way, we find $\phi_b = 0$ and $y_b(t) = \sin(0.5\pi f_0 t)$:

$$y_a(t) = \sin(1.5\pi f_0 t)$$

$$y_b(t) = \sin(0.5\pi f_0 t)$$

$$y_c(t) = \frac{1}{2}y_a(t) + \frac{1}{2}y_b(t)$$

Since a sinusoidal signal can be represented by its amplitude, frequency, and phase, the knowledge of three adjacent samples gives us an opportunity to recover the signal from a set of three equations. In our case, we already know the amplitude to be one, so we only need two adjacent samples to recover the frequency and phase values.

In the first case, when the sampling rate (frequency) is exactly twice the sinusoidal frequency, the continuous signal cannot be reconstructed. In the second case, when the sampling rate is less than twice the sinusoidal frequency, the continuous signal cannot be uniquely reconstructed because there is an infinite number of sinusoidal signals fitting the sample sequence. The only possibility to reconstruct the original signal is to have *a priori* information about the range of expected signal frequencies. In this example, we could uniquely reconstruct the original signal if we would know that the signal frequency ranges from $f_{\min} = \frac{1}{2} f_0$, up to $f_{\max} = f_0$. Then, we would be able to identify $y_a(t)$ as the original signal.

In Fig. 1.9, the sinusoidal signal is sampled with a sampling rate satisfying the sampling theorem. The straight-line interpolation has been used to connect adjacent samples. If the signal frequency, f_a, gets closer to $\frac{1}{2} f_0$ we cannot visually identify the sine function from the simple linear interpolated diagram. Nevertheless, the continuous signal can be reconstructed exactly.

The basic results for defining relationships between continuous-time and discrete-time signals were presented by H. Nyquist, J. M. Whittaker, D. Gabor, and C. E. Shannon. At this point, we will express these results in a heuristic and intuitive way. The rigorous formulation, called the *Sampling Theorem*, will be given in Chapter 3 in Section 3.3.

First, we assume that a continuous signal consists of one or more sinusoidal signals and that the highest frequency of these sinusoidal signals is f_{max}. The continuous signal can be uniquely represented by its equally spaced samples if the sampling frequency f_0 is at least twice the highest frequency f_{max}. The original continuous signal can be reconstructed from the sample sequence by passing the sequence through a system having the property to reject sinusoidal signals of frequencies higher than f_{max}.

The minimal sampling frequency is $f_0 = 2f_{max}$, and one-half of the sampling frequency is called the *Nyquist frequency*, also called the *folding frequency*:

$$\text{Nyquist frequency} = \frac{f_0}{2} = \frac{1}{2T}$$

A sinusoidal signal with the frequency, $f_a < f_0$, which is above the Nyquist frequency, $\frac{1}{2} f_0$, is aliased by sampling into a discrete sinusoidal signal below the Nyquist frequency.

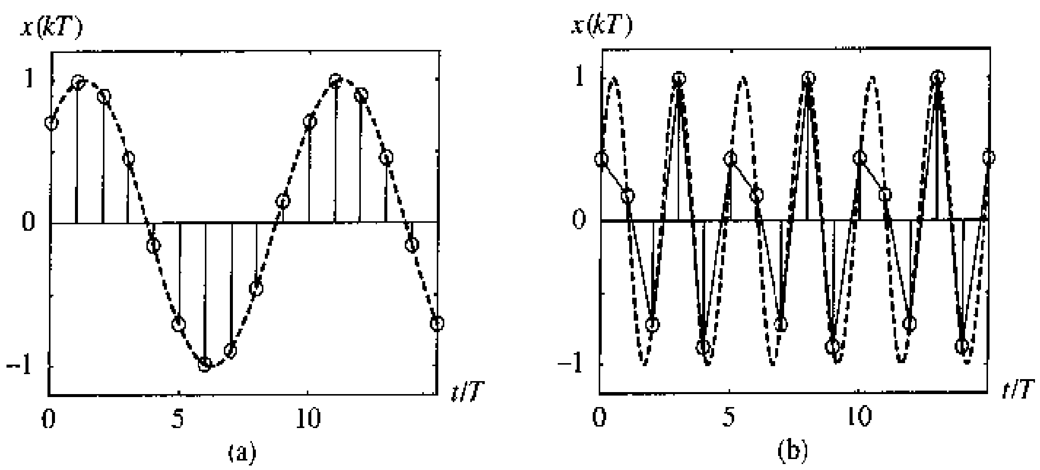

Figure 1.9 Frequency of signal is less than a half of sampling frequency.

The aliased frequency appears as if two signals existed, one at f_a and the other at $f_0 - f_a$, hence its name. The aliasing effect is eliminated by a system called a lowpass continuous *anti-aliasing* filter, which is often used before of discretizing the continuous signal. In practical applications, the actual sampling frequency is often selected to be four times the maximum expected signal frequency f_a.

1.3 BASIC CONTINUOUS-TIME SIGNALS—FUNCTIONS

In practical applications the excitation or response of a system can be represented by a combination of basic signals. In this section, all signals are assumed to be functions that depend on time t over the range $-\infty < t < +\infty$.

1.3.1 Sinusoidal Signals

A *sinusoidal* signal is defined by the sinusoidal function

$$x_s(t) = X_s \sin(2\pi f_s t + \phi_s)$$

Alternatively

$$x_s(t) = X_s \sin\left(\frac{2\pi}{T_s} t + \phi_s\right)$$

where T_s is a period of sinusoidal signal

$$T_s = \frac{1}{f_s}$$

The unit of frequency for f_s is Hertz (Hz) or cycles per second when t represents time in seconds. The period $T_s = 1/f_s$ is generally given in seconds (s). The phase shift ϕ_s is in radians (rad). The *radian* frequency $\omega_s = 2\pi f_s$ is in rad/s. The derivative or integral of a sinusoidal function is, again, a sinusoidal function.

1.3.2 Real-Valued Exponential Signals

A *real-valued exponential* signal is defined by exponential function

$$x_e(t) = X_e e^{bt}$$

where e is the Naperian constant ($e \approx 2.718$), and X_e and b are real constants. Three different signals can be derived from the exponential function:

- an increasing function of time for $b > 0$,
- a constant $x_e(t) = X_e$ for $b = 0$, and
- a decreasing function of time for $b < 0$.

In practical applications, a signal obtained as a combination of the above can be used; for example,

$$x_b(t) = X_e\left(1 - e^{bt}\right), \qquad b < 0$$

1.3.3 Unit Step Signal

A *unit step* signal, $u(t)$, is a signal that turns on from 0 to a constant (unit) value at $t = 0$:

$$u(t) = \begin{cases} 1, & t > 0 \\ 0, & t \leq 0 \end{cases}$$

The unit step signal is a continuous function with a single point of discontinuity at $t = 0$ where it jumps in amplitude from 0 to 1.

The unit step signal can be scaled in time and amplitude:

$$x_u(t) = X_u u(t - t_1) = \begin{cases} X_u, & t > t_1 \\ 0, & t \leq t_1 \end{cases}$$

Step signals are used to select portions of other signals. For example, $x(t)(u(t) - u(t - t_2))$ is a portion of $x(t)$ between $t = 0$ and t_2.

1.3.4 Pulse Signals

A *pulse* signal can be obtained from unit step signals as

$$x_p(t) = X_p(u(t - t_1) - u(t - t_2))$$

For $t_2 > t_1$ the pulse signal is constant over the range $t_1 < t \leq t_2$, $x_p(t) = X_p$, and zero otherwise. A unit pulse of width t_w can be represented by $u(t) - u(t - t_w)$.

We frequently use rectangular pulses whose area under the pulse equals 1. For this reason, we define the *pulse function* $p_\epsilon(t)$ by

$$p_\epsilon(t) = \begin{cases} 0, & t \leq 0 \\ \frac{1}{\epsilon}, & 0 < t \leq \epsilon \\ 0, & \epsilon < t \end{cases}$$

In other words, $p_\epsilon(t)$ is a pulse of height $1/\epsilon$, of width ϵ, and starting at $t = 0$. Whatever the value of the positive parameter ϵ, the area under $p_\epsilon(t)$ is 1. Note that

$$p_\epsilon(t) = \frac{u(t) - u(t - \epsilon)}{\epsilon}$$

Pulse signals are used to select portions of other signals. For example, $x(t)x_p(t)$ is a portion of $x(t)$ between $t = t_1$ and t_2, assuming $X_p = 1$.

1.3.5 Unit Ramp Signal

A *ramp* signal, $x_r(t)$, is defined as a continuous function which rises in amplitude linearly at a rate given by the constant X_r

$$x_r(t) = X_r\, t\, u(t)$$

A ramp signal shifted by t_1 is defined as

$$x_r(t - t_1) = X_r(t - t_1)u(t - t_1)$$

A *unit ramp* signal is zero for $t \leq 0$ and has a unity rate of increase in amplitude, $X_r = 1$:

$$r(t) = \begin{cases} t, & t > 0 \\ 0, & t \leq 0 \end{cases}$$

The unit ramp can be expressed in terms of the unit step function

$$r(t) = t\,u(t)$$

Also, the unit ramp signal results from the unit step signal by integration

$$r(t) = \int_{-\infty}^{t} u(\tau)\,\mathrm{d}\tau$$

As a consequence, the unit step signal is the derivative of the unit ramp signal

$$u(t) = \begin{cases} \dfrac{\mathrm{d}r(t)}{\mathrm{d}t}, & t \neq 0 \\ 0, & t = 0 \end{cases}$$

Notice that the derivative $\mathrm{d}r/\mathrm{d}t$ is not defined for $t = 0$ because $u(t)$ is discontinuous at $t = 0$.

A unit step signal with a linear rise from 0 to 1 can be obtained as a combination of two unit ramp signals:

$$x_{ur}(t) = \frac{r(t) - r(t - t_2)}{t_2}$$

The rise time is t_2.

1.3.6 Unit Impulse Signal

The *unit impulse* signal, $\delta(t)$, also called the *Dirac delta*, is not a function in a strict mathematical sense. We proceed intuitively [3] because a detailed exposition would require the theory of distributions. A more detailed discussion of the Dirac delta can be found in references 4–6. Our following definition of the unit impulse function, however, is satisfactory for the purposes of this book.

For our purposes, we state that

$$\delta(t) = \begin{cases} 0, & t \neq 0 \\ \text{undefined}, & t = 0 \end{cases}$$

and that the area under the unit impulse is one:

$$\int_{-\infty}^{\infty} \delta(t)\,\mathrm{d}t = 1$$

Hence, one can consider the area at the origin to be one. Intuitively, we may think of the impulse function $\delta(t)$ as the limit, as $\epsilon \to 0$, of the pulse function $p_\epsilon(t)$. From the definition of $\delta(t)$ and $u(t)$, we can formally obtain

$$u(t) = \int_{-\infty}^{t} \delta(\tau)\,\mathrm{d}\tau$$

and

$$\frac{du(t)}{dt} = \delta(t)$$

Another frequently used property of the unit impulse is the *sifting property*. Letting $x(t)$ be a continuous signal, we obtain

$$\int_{-\infty}^{\infty} x(t)\,\delta(t - t_0)\,dt = x(t_0)$$

This property is important when used in integral transforms such as the Laplace and Fourier transform, which we will cover in a subsequent chapter.

1.3.7 Causal Signals

Formally, a signal is said to be *causal* if it is zero for $t < 0$. Causal signals are readily created by multiplying any continuous signal by the unit step $u(t)$. In practical applications we often consider signals of finite duration. The instant when the signal begins is called the starting time. We usually take the starting time to be $t = 0$. In most cases, we analyze the signal from beginning at the starting time, and we consider causal signals only.

1.3.8 Combining Signals

By multiplication, addition, and subtraction of basic signals, we can generate combined signals. We have already used the multiplication by unit step. Some interesting signals that can appear in signal processing are:

- *Exponentially modulated sinusoidal* signals

$$x(t) = X_m e^{bt} \sin(2\pi f_a t + \phi_a)$$

- *Sinusoidally modulated sinusoidal* signals

$$x(t) = X_m \sin(2\pi f_b t + \phi_b) \sin(2\pi f_a t + \phi_a)$$

  which can be expressed as the sum of two sinusoidal signals

$$x(t) = \frac{1}{2} X_m(-\cos(2\pi(f_a + f_b)t + \phi_a + \phi_b) + \cos(2\pi(f_a - f_b)t + \phi_a - \phi_b))$$

- A sum of m sinusoidal signals

$$x(t) = \sum_{i=1}^{m} X_i \sin(2\pi f_i t + \phi_i)$$

  For example, if we choose $X_i = \frac{4}{\pi}\frac{1}{2i-1}$, $f_i = (2i - 1) f_a$, $\phi_i = 0$, we obtain an approximation of a pulse train shown in Fig. 1.10, [5]. Similarly, for $X_i =$

$\frac{2}{\pi}\frac{(-1)^{i+1}}{i}$, $f_i = if_a$, $\phi_i = 0$, we obtain an approximation of a sawtooth periodic signal shown in Fig. 1.11.

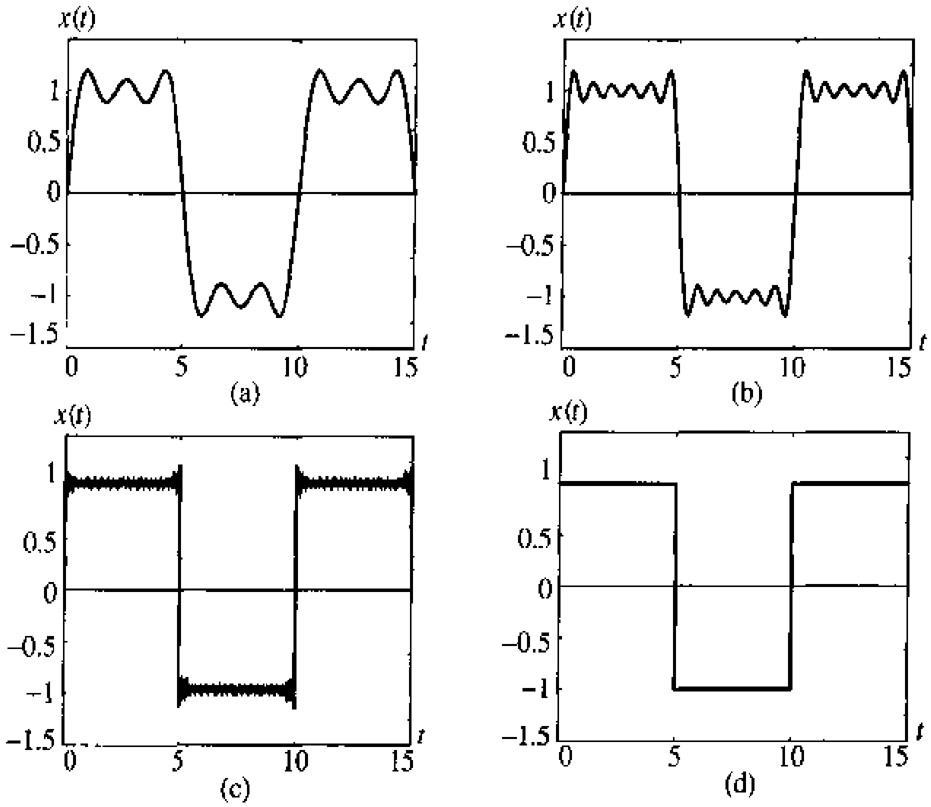

Figure 1.10 The summation of m sinusoidal signals

$x(t) = \sum_{i=1}^{m} \frac{4}{\pi}\frac{1}{2i-1}\sin(2\pi(2i-1)f_a t)$, for: (a) $m = 3$, (b) $m = 6$, (c) $m = 500$, (d) $m \to \infty$.

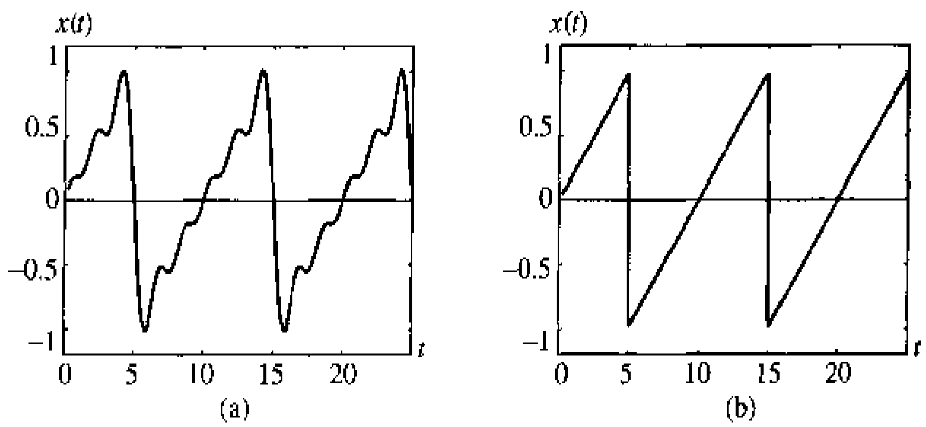

Figure 1.11 The summation of m sinusoidal signals

$x(t) = \sum_{i=1}^{m} \frac{2}{\pi}\frac{(-1)^{i+1}}{i}\sin(2\pi i f_a t)$, for (a) $m = 3$, and (b) $m \to \infty$.

1.4 BASIC DISCRETE-TIME SIGNALS—SEQUENCES

A *sequence*, also known as a *data sequence* or *sample set*, is a collection of ordered samples x_k. For the general case, the sample set is defined over the entire range from $k = -\infty$ to $k = +\infty$. In practical applications, we generally process finite sequences defined over the interval $0 \leq k \leq N-1$, where N denotes the total number of data samples in a sequence. The finite sequence is often obtained by extracting a finite portion of an existing, possibly infinite sequence. The existing sequence is often a sampled version of a continuous signal. For the purposes of analysis, we represent sequences as a combination of basic sequences which are better understood and easier to manipulate.

1.4.1 Sinusoidal Sequences

A sinusoidal sequence may be described as

$$x_{s,k} = X_s \sin\left(2\pi \frac{1}{N_s} k + \phi_s\right)$$

where X_s is the amplitude of the sinusoidal sequence (a positive real number), N_s is the period, and ϕ_s is the phase. When the sinusoidal sequence is obtained by sampling a continuous signal $x_s(t) = X_s \sin(2\pi f_s t + \phi_s)$, the resulting sample set is

$$x_{s,k} = X_s \sin\left(2\pi \frac{f_s}{f_0} k + \phi_s\right)$$

1.4.2 Real-Valued Exponential Sequences

A *real-valued exponential* sequence is defined by the exponential function

$$x_{e,k} = X_e e^{bTk}$$

where e is the exponential constant (base of natural logarithms) with an approximate numerical value of 2.71828, X_e and b are real constants, and T is the sampling period. By rewriting the above expression,

$$x_{e,k} = X_e a^k$$

where $a = e^{bT}$.

Three different sequences can be derived from the real exponential sequence:

- an increasing exponential sequence, for $a > 1$,
- a constant $x_{e,k} = X_e$, for $a = 1$,
- a decreasing exponential sequence, for $0 < a < 1$.

We combine the above sequences to form a sequence like $x_k = X_e(1 - a^k)$, $a < 1$.

1.4.3 Unit Step Sequence

A *unit step* sequence is defined as

$$u_k = \begin{cases} 1, & k \geq 0 \\ 0, & k < 0 \end{cases}$$

We can shift and scale the unit step sequence and obtain

$$X_u u_{k-k_0} = \begin{cases} X_u, & k \geq k_0 \\ 0, & k < k_0 \end{cases}$$

1.4.4 Unit Ramp Sequence

A *ramp* sequence is defined as

$$x_{r,k} = X_r \, k \, u_k$$

and the shifted ramp sequence is defined as

$$x_{r,(k-m)} = X_r \, (k - m) \, u_{k-m}$$

A *unit ramp* sequence is zero for $k < 0$:

$$r_k = \begin{cases} k, & k \geq 0 \\ 0, & k < 0 \end{cases}$$

The unit ramp sequence can be expressed in terms of the unit step sequence

$$r_k = k \, u_k$$

Also, the unit ramp sequence results from the unit step sequence by summation:

$$r_k = \sum_{m=-\infty}^{k} u_m$$

As a consequence, the unit step sequence is the difference of two adjacent samples of unit ramp sequences:

$$u_k = r_{k+1} - r_k$$

1.4.5 Unit Impulse Sequence

A *unit impulse* sequence is defined as

$$\delta_k = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

The unit impulse sequence can be obtained as the difference of the two adjacent samples of unit step sequences:

$$\delta_k = u_k - u_{k-1}$$

Also, the unit step sequence results from the unit impulse sequence by summation:

$$u_k = \sum_{m=-\infty}^{k} \delta_m$$

The unit impulse sequence is an important signal for analysis of discrete-time systems. The impulse acts to sample a single value, x_m, of a sequence, x_k, which it multiplies:

$$x_k \delta_{k-m} = x_m \delta_{k-m} = x_m$$

for $-\infty < k < +\infty$. An arbitrary sequence x_k can be expressed in terms of the unit impulse sequence as

$$x_k = \sum_{m=-\infty}^{+\infty} x_m \delta_{k-m}$$

1.4.6 Causal Sequences

A sequence x_k that is nonzero only over a finite interval $k_{min} < k < k_{max}$ is called a *finite-length* sequence. A sequence containing nonzero samples for $k \geq 0$ is said to be *causal*. An *anti-causal* sequence has nonzero samples only for $k < 0$. An example of a noncausal sequence follows:

$$x_k = u_{-k-1} = \begin{cases} 1, & k < 0 \\ 0, & k \geq 0 \end{cases}$$

1.5 CONTINUOUS-TIME SIGNALS IN MATLAB

The original design of MATLAB was to provide an interactive, command-line interface to a variety of numerical analysis libraries. Arithmetic is performed using a double-precision floating-point format. MATLAB has evolved over the years from a matrix laboratory into a general-purpose programming language, an algorithm development environment through its many toolboxes, a visual programming environment through Simulink, and a graphical user interface programming environment. Throughout the book, we will draw heavily on the MATLAB Signal Processing Toolbox and the Optimization Toolbox.

For our purposes, we will focus on the matrix laboratory aspects of MATLAB. As a matrix laboratory, MATLAB primarily works with numbers, which are usually represented as sequences and stored as arrays. Therefore, we cannot use continuous signals or operate on them in an analytic manner. We can describe a continuous-time signal with sequences of numbers; that is, we sample the signal and assume that the time step between two adjacent samples is appropriate. It is expected that the signal digitalization introduces no loss of information conveyed by the signal.

Consider the signal $x(t) = X^a e^{-at}\sin(2\pi ft + \phi)$. It can be coded in MATLAB like this: `x(t) = X^a * exp(-a*t) .*sin(2*pi*f*t+phi)`. For convenience, we often use `pi` to denote π, `phi` for ϕ, `w` for $\omega = 2\pi f$, `exp(a*t)` to denote e^{at}, and `*` to denote multiplication.

A MATLAB script for plotting this signal follows:

```
X = 1;
a = 0.1;
f = 0.53;
phi = 0;
t = 0.:0.05:15;
x = X^a * exp(-a*t) .* sin(2*pi*f*t+phi);
plot(t,x)
ylabel('x(t)')
xlabel('t (s)')
grid
```

In this script, there are two types of multiplication: `*` and `.*`. When applied to two vectors of the same length, the `*` operator will give the dot product between the two vectors, whereas the `.*` operator will produce a new vector whose ith element is the product of the ith elements of the two vectors. Hence, the `.*` operator is an example of an elementwise operator. In the script, the independent variable `t` is an array, and so the expressions `exp(-a*t)` and `sin(2*pi*f*t+phi)` also create arrays because `a` and `2*pi*f` are scalars. We use the string `t (s)` as the label on the horizontal axis of the plot to denote that time is in seconds—note the space between `'t'` and `'(s)'`. No space appears when we designate a signal x as a function of time, `x(t)`. The plot of $x(t)$ obtained by executing the script is shown in Fig. 1.12.

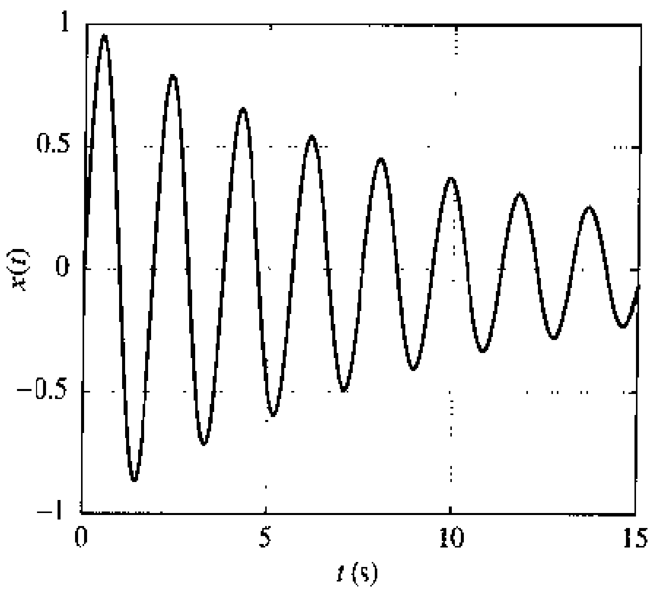

Figure 1.12 $x(t) = (1^{0.1}e^{-0.1t}\sin(2\pi 0.53t))$.

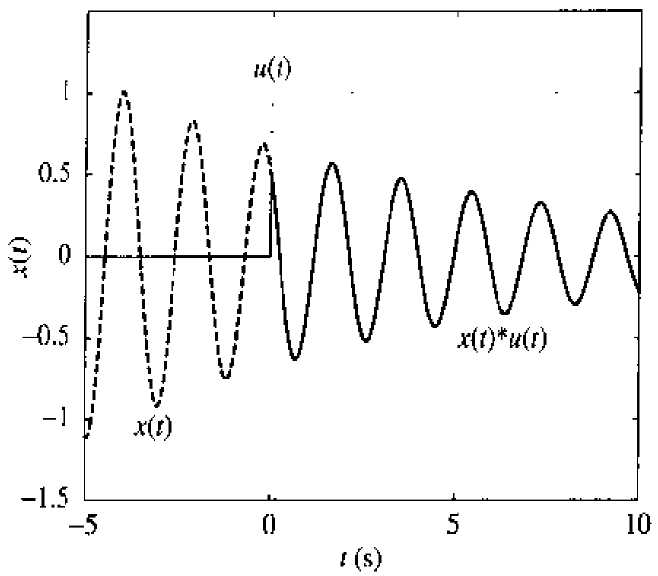

Figure 1.13 $u(t), x(t)=(0.02^{0.1}e^{-0.1t}\sin(2\pi 0.53t+3\pi/4))$ and $x(t)u(t)=(0.02^{0.1}e^{-0.1t}\sin(2\pi 0.53t+3\pi/4))u(t)$.

In the next example, we generate a causal signal from $x(t)$. First, we calculate the unit step signal $u(t)$ by executing the MATLAB command `u=(t>0)`. For a given array `t`, this command produces the unit step sequence `u` of the same length as `t`. The logical expression `t>0` maps the ith component of `t` into 1, when `t>0` evaluates true, or into 0 when `t>0` is false. The signal is multiplied by the unit step: $x(t)u(t)$. Again, we have to use `.*` rather than `*` because `x` and `u` are arrays.

The signals $x(t)$, $u(t)$, and $x(t)u(t)$ are plotted in Fig. 1.13, and the MATLAB script that is used to create the plot is as follows:

```
X = 0.02;
a = 0.1;
f = 0.53;
phi = 3*pi/4;
t = -5.:0.05:10;
x = X^a * exp(-a*t) .* sin(2*pi*f*t+phi);
xu = (X^a * exp(-a*t) .* sin(2*pi*f*t+phi)).*(t>0);
u = (t> 0);
plot(t,x,'--',t,u,':',t,xu)
ylabel('x(t)')
xlabel('t (s)')
text(0,1.2,'u(t)')
text(-4.,-1.1,'x(t)')
text(5,-.6,'x(t)*u(t)')
axis([t(1) t(length(t)) -1.5 1.5])
```

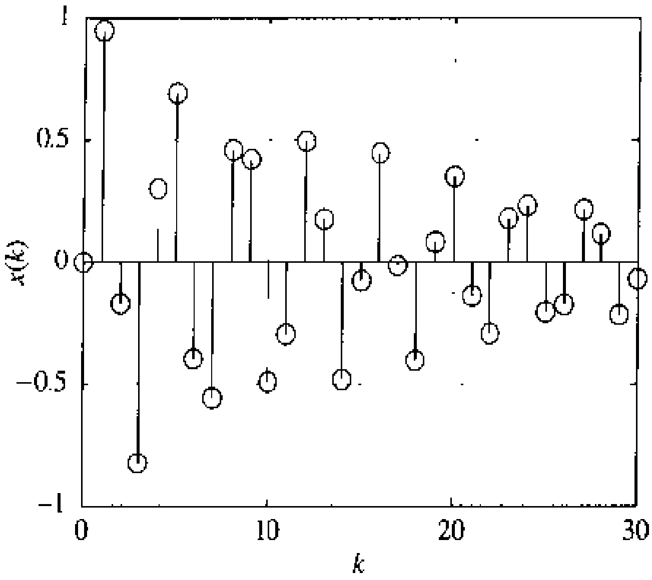

Figure 1.14 $x(k) = e^{-0.05\ k}\sin(\pi\ 0.53\ k)$.

1.6 SEQUENCES IN MATLAB

A continuous signal in MATLAB is in fact represented by ordered pairs (*time*, *value*). A digital signal is a set of numbers arranged in an array. Instead of time, we speak about index of an array element. If an analytic expression for the element of an array is known, we can generate a sequence of numbers. For example, given an index, `k`, we compute the array element `x(k) = X^a * exp(-a*T*k) * sin(2*pi*T*f*k+phi)` of an exponentially modulated sinusoidal sequence. The MATLAB script that is used to generate and plot the sequence follows, and the corresponding sequence is shown in Fig. 1.14:

```
X = 1;
a = 0.1;
f = 0.53;
T = 0.5;
phi = 0;
k = 0:1:30;
x = X^a * exp(-a*T*k) .* sin(2*pi*f*T*k+phi);
stem(k,x)
ylabel('x(k)')
xlabel('k')
grid
```

In the next example, we generate a causal sequence from x_k. First, we calculate the unit step sequence u_k: `u = (k>=0)`. In fact, as in the case of continuous signals, we use a logical expression so that the value of the kth component of `u` is 1 when the logical expression `k>=0` is true, and 0 when the expression `k>=0` is false.

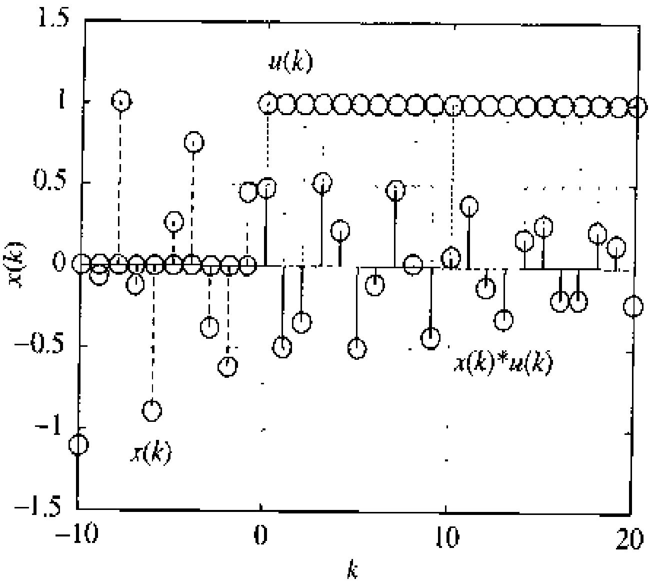

Figure 1.15 $u(k)$, $x(k) = (0.02^{0.1}e^{-0.05\ k}\sin(\pi\ 0.53\ k + 3\pi/4))$ and $x(k)\ u(k) = (0.02^{0.1}e^{\ 0.05\ k}\sin(\pi\ 0.53\ k + 3\pi/4))u(k)$.

The causal signal is obtained by multiplying x_k by u_k. Figure 1.15 illustrates the causal signal and its components. The MATLAB script is as follows:

```
X = 0.02;
a = 0.1;
f = 0.53;
T=.5;
phi = 3*pi/4;
k = -10:1:20;
x = X^a * exp(-a*T*k) .* sin(2*pi*f*T*k+phi);
ux = (X^a * exp(-a*T*k) .* sin(2*pi*f*T*k+phi)).*(k>=0);
u = (k>=0);
stem(k,x,'--')
hold on
stem(k,u,':')
stem(k,ux)
hold off
ylabel('x(k)')
xlabel('k')
text(0,1.3,'u(k)')
text(-8.,-1.12,'x(k)')
text(10,-.65,'x(k)*u(k)')
axis([k(1) k(length(k)) -1.5 1.5])
```

The primary difference between continuous and discrete-time signals in MATLAB is in the type of the corresponding independent variable. For continuous signals, it is a 'time' array, t, and can be any number; however, in the case of digital signals it is always an integer, k, and may take a positive, negative, or zero value.

1.7 CONTINUOUS-TIME SIGNALS IN MATHEMATICA

Mathematica is a system for doing mathematics by computer. Its forte is the algebra analysis and manipulation of functions and operators. One can work in an arbitrary number of dimensions and convert algebraic expressions to numbers with arbitrary precision. It has a graphical user interface which can load and store work sessions in an ASCII notebook format. When it first appeared, *Mathematica* offered a fundamentally different but complementary approach for analyzing signals and algorithms than MATLAB. *Mathematica* has a variety of *Application Packs* that extend the analysis to different fields. Throughout the book, we will draw heavily on the *Mathematica Signals and Systems Pack* [7].

Using *Mathematica*, the processing of continuous signals can be maintained in exact symbolic form. The transform of continuous signals involving integrals and derivatives are easier in symbolic form. Therefore, in symbolic processing, the signal is represented on a computer as a formula instead of a sequence of numbers. The aforementioned examples worked in MATLAB will be reconsidered here, but with the aid of *Mathematica* and the *Signals and Systems Pack* (SSP).

The *Mathematica* script that is used to create a signal is as follows:

```
X=1;
a=0.1;
f=0.53;
phi = 0;
x[t_] := X^a Exp[-a*t] Sin[2 Pi f t + phi];
Plot[ x[t], {t, 0, 14},
      AxesLabel -> {"t (s)", "x(t)"},
      PlotRange -> {-1, 1},
      GridLines -> Automatic ];
```

Two features of *Mathematica* may be readily apparent: Spaces between terms in an expression are interpreted as multiplication, and square brackets are used to delimit functions instead of parentheses. In the script, we define a function $x(t)$. In *Mathematica* notation, `x[t_]` defines a function `x` that takes any independent variable labeled `t`. For the exponential function, we can use either `Exp[-a*t]` or `E^(-a*T)`. We use `Sin[.]` to denote the sine function and use `Pi` for π. All of the built-in, protected *Mathematica* functions and constants begin with an uppercase letter.

We assign numerical values to X, a, `f`, and `phi` only for the purpose of obtaining a plot. In a symbolic algebra system such as *Mathematica*, we can work with symbols rather than with numbers. The actual plot command is `Plot[x[t],{t,0,14}]`, in which we define the independent variable to be t and its range from $t = 0$ to $t = 14$. The plot option `AxesLabel->{"t (s)","x(t)"}` adds labels to the horizontal and vertical axes, respectively. The grid lines are drawn by `GridLines->Automatic`, while the plot range is specified by `PlotRange->{-1,1}`. The resulting plot is in Fig. 1.16.

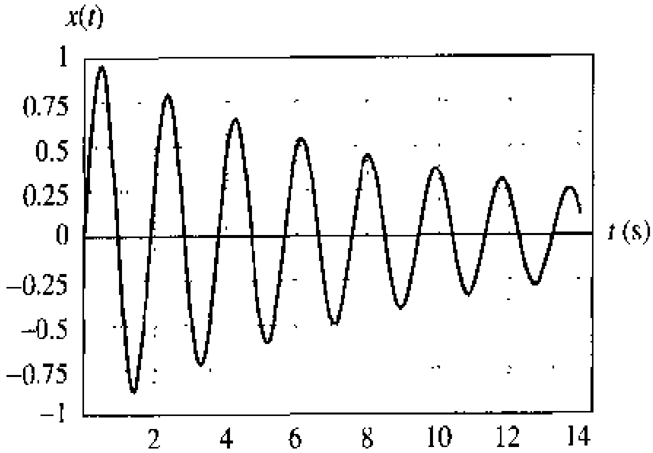

Figure 1.16 $x(t) = (1^{0.1} e^{(-0.1t)} \sin(2\pi 0.53t))$.

The second example of representing and plotting the three signals is expressed in *Mathematica* by

```
X=0.02;
a=0.1;
f=0.53;
phi = 3*Pi/4;
x[t_] :=  X^a Exp[-a*t] Sin[2*Pi*f*t+phi];
SignalPlot[{x[t]*UnitStep[t],x[t],UnitStep[t]},{t,-5,10},
      AxesLabel->{"t (s)","x(t)"},
      PlotRange->{-1.5,1.5},
      AxesOrigin->{-5,-1.5},
      PlotStyle->{GrayLevel[0],
                    Dashing[{.05}],Dashing[{.01}]}];
```

The signals are written in the *Mathematica* notation. For instance, `Sin[2 Pi f t]` designates $\sin(2\pi f t)$, and `Exp[-a*t]` represents e^{-at}. The unit step, $u(t)$, is represented by `UnitStep[t]`. Additional options define the position of the axis origin, `AxesOrigin->{-5,-1.5}`, and the type of lines, `PlotStyle->{GrayLevel[0]}, Dashing[{.05}], ...` (see Fig. 1.17). We use the *Mathematica* built-in function `Plot` in Fig. 1.16 and `SignalPlot` from the SSP in Fig. 1.17. SSP contains functions for frequently used signals. For example, `ContinuousPulse[width,t]` represents a rectangular pulse with the unit height, existing for `0<t<width`. Various SSP signals are shown in Figs. 1.18–1.26. A continuous pulse signal starting at time $t = 0.4$ s and with the width of 0.2 s is plotted in Fig. 1.18.

```
SignalPlot[ContinuousPulse[.2,t-0.4],{t,0,1},
      PlotStyle->{Thickness[.01]},
      AxesLabel->{"t (s)","ContinuousPulse[.2,t-0.4]"}];
```

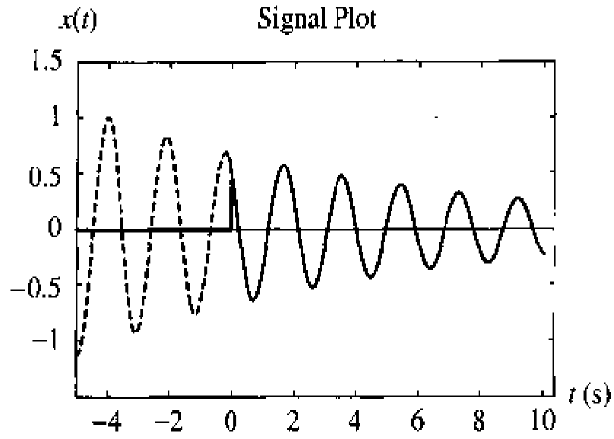

Figure 1.17 $u(t)$, $x(t) = (0.02^{0.1}e^{-0.1t}\sin(2\pi 0.53t + 3\pi/4))$ and $x(t)u(t) = (0.02^{0.1}e^{-0.1t}\sin(2\pi 0.53t + 3\pi/4))u(t)$.

The unit impulse signal can be plotted as *Dirac delta* (`DiracDelta[t-0.75]`), *continuous impulse* (`ContinuousImpulse[t-0.75]`), or as 0th-order *unit singularity function* (`UnitFunction[0][t-0.75]`, the order of the singularity is 0 and the time shift is 0.75). The height of the unit impulse represents the area under the delta function ($\int_{-\infty}^{+\infty}\delta(t)\,dt = 1$). Because $\delta(t)$ has zero width and infinite amplitude at $t = 0$ but unity total area, we draw arrows at the unit height. If we multiply the unit impulse signal by 2, then the area is also 2, $\int_{-\infty}^{+\infty}2\delta(t)\,dt = 2$. We draw $2\delta(t)$ taller impulse with arrow at the doubled height. We label the arrow with a number called the *weight* of the impulse. The weight does not indicate the height of the impulse, but it indicates the area:

$$\int_{-\infty}^{+\infty} weight\,\delta(t)\,dt = weight$$

```
SignalPlot[DiracDelta[t-0.75],{t,0,2},
    AxesLabel->{"t (s)","DiracDelta[t-0.75]"}];
```

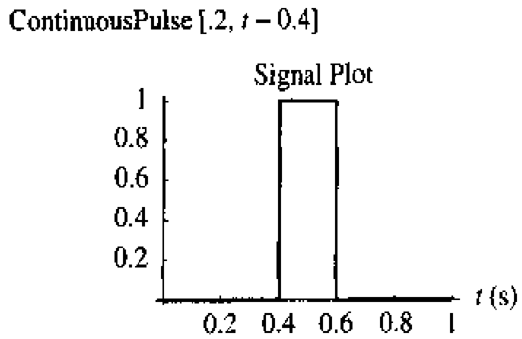

Figure 1.18 An example of a continuous unit pulse, $u(t-0.4) - u(t-0.6)$.

DiracDelta [t – 0.75]

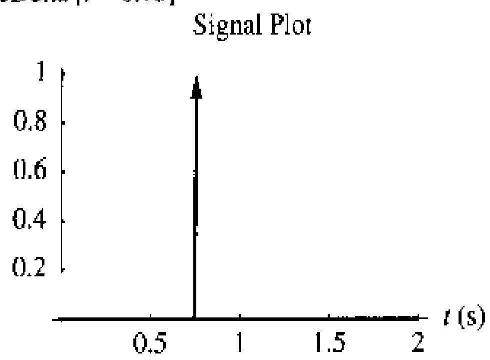

Figure 1.19 The Dirac delta signal, $\delta(t)$.

UnitFunction [0][t – 0.75]

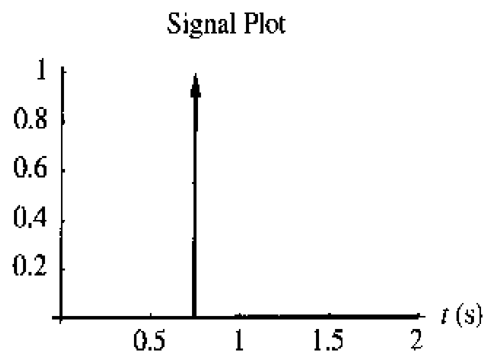

Figure 1.20 The Dirac delta signal, $\delta(t)$.

ContinuousImpulse [t – 0.75]

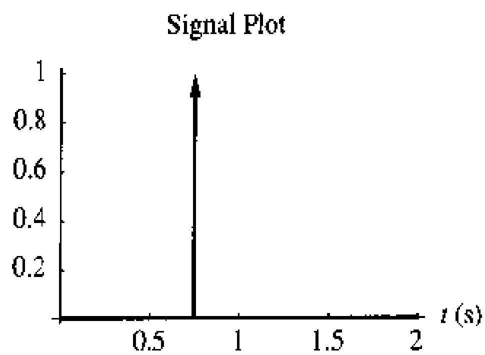

Figure 1.21 The Dirac delta signal, $\delta(t)$.

UnitFunction[−1][t − 3.5]

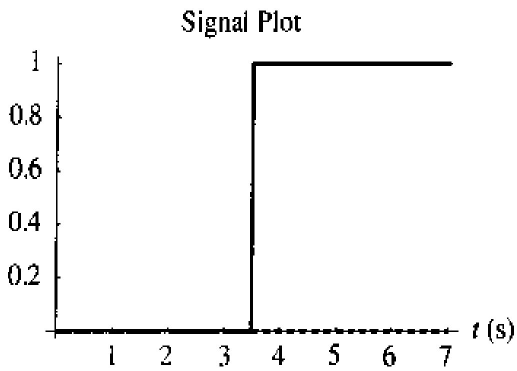

Figure 1.22 The continuous unit step signal, $u(t)$.

ContinuousStep[t − 3.5]

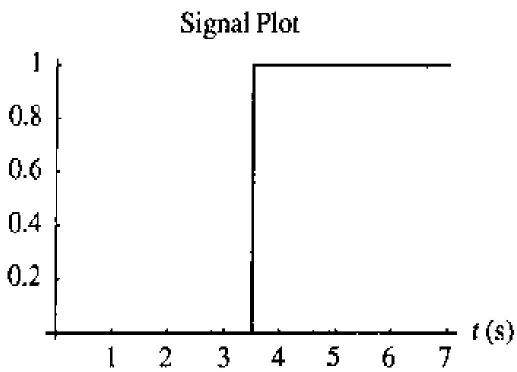

Figure 1.23 The continuous unit step signal, $u(t)$.

UnitFunction[−2][t]

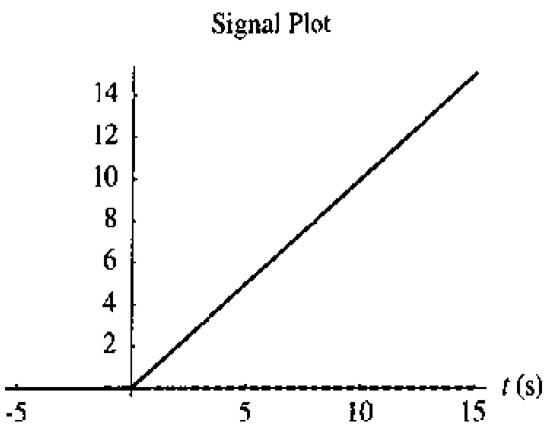

Figure 1.24 The continuous unit ramp signal, $r(t)$.

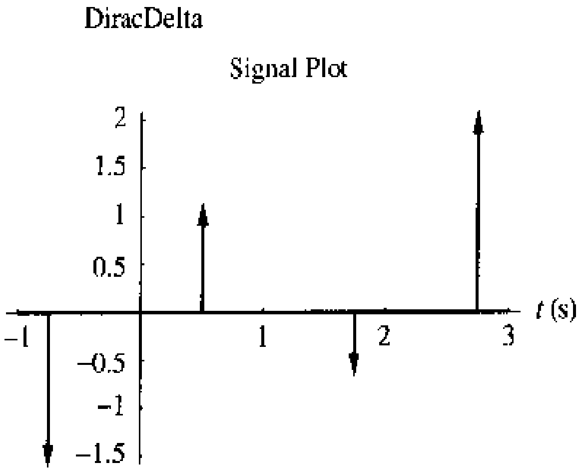

Figure 1.25 The Dirac delta signal, $\delta(t)$.

```
SignalPlot[UnitFunction[0][t-0.75],{t,0,2},PlotRange->All,
     AxesLabel->{"t (s)","UnitFunction[0][t-0.75]"}];

SignalPlot[ContinuousImpulse[t-0.75],{t,0,2},PlotRange->All,
     AxesLabel->{"t (s)","ContinuousImpulse[t-0.75]"}];
```

We draw the unit step function as `ContinuousStep[t]` or as the unit function with the order of the singularity −1, `UnitFunction[-1][t]`. When the order of the singularity of the unit function is −2, we draw the unit ramp signal.

```
SignalPlot[UnitFunction[-1][t-3.5],{t,0,7},
     PlotRange->All,PlotStyle->{Thickness[.01]},
     AxesLabel->{"t (s)","UnitFunction[-1][t-3.5]"}];

SignalPlot[ContinuousStep[t-3.5],{t,0,7},
     PlotRange->All,PlotStyle->{Thickness[.01]},
     AxesLabel->{"t (s)","ContinuousStep[t-3.5]"}];
```

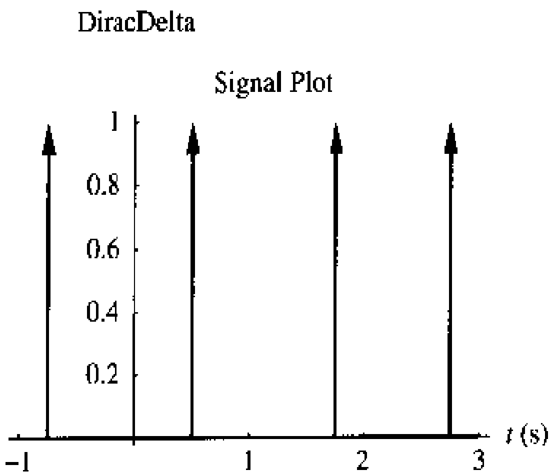

Figure 1.26 The Dirac delta signal, $\delta(t)$.

```
SignalPlot[UnitFunction[-2][t],{t,-5,15},
      PlotRange->All,PlotStyle->{Thickness[.01]},
      AxesLabel->{"t (s)","UnitFunction[-2][t]"}];
```

The weights of the impulse signal can be positive or negative. The negative impulse is with arrow toward the $-\infty$. If we represent by impulse height the value of the impulse, not the area, then the height is always zero because it makes no sense to draw taller and shorter arrows for impulse of infinite value. In SSP, we can draw impulse with always unit value by using `DiracDeltaScaling -> False`.

```
SignalPlot[{1.0 DiracDelta[t-0.5] -
            1.5 DiracDelta[t+0.75] -
            0.5 DiracDelta[t-1.75] +
            2.0 DiracDelta[t-2.75]},
           {t,-1,3},
      DiracDeltaScaling -> True,
      AxesLabel->{"t (s)","DiracDelta"}];

SignalPlot[{1.0 DiracDelta[t-0.5] -
            1.5 DiracDelta[t+0.75] -
            0.5 DiracDelta[t-1.75] +
            2.0 DiracDelta[t-2.75]},
           {t,-1,3},
      DiracDeltaScaling -> False,
      AxesLabel->{"t (s)","DiracDelta"}];
```

1.8 SEQUENCES IN MATHEMATICA

We can express a sequence in analytical form. For example, a discrete signal $x_k = e^{-0.05\,k}\sin(\pi\,0.53\,k)$ we can describe by the *Mathematica* script as follows:

```
X=1;
a=0.1;
f=0.53;
t0=0.5;
phi = 0;
x[k_] := X^a * E^(-a t0 k) Sin[2 Pi f t0 k + phi];
DiscreteSignalPlot[x[k],{k,0,30},
     AxesLabel->{"k","x(k)"},
     PlotRange->{-1,1},
     AxesOrigin->{0,-1},
GridLines->{{0,0},{0,0}}];
```

Instead of `SignalPlot` we use `DiscreteSignalPlot`. The same example was considered in the MATLAB script. The resulting plot is in Fig. 1.27.

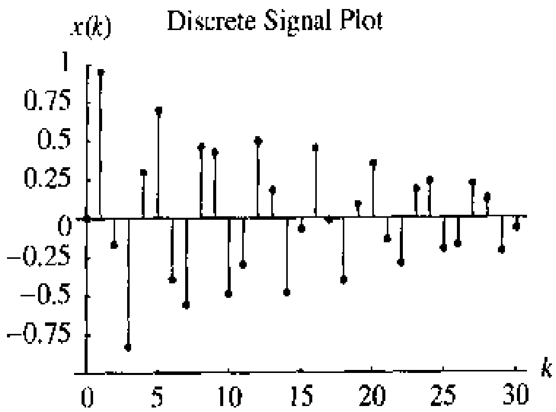

Figure 1.27 $x(k) = e^{-0.05\,k}\sin(\pi\ 0.53\ k)$.

The second example of a discrete-time signal is described by the *Mathematica* script:

```
X=0.02;
a=0.1;
t0=0.5;
f=0.53;
phi = 3*Pi/4;
x[k_] :=  X^a * E^(-a t0 k) Sin[2 Pi f t0 k + phi];
DiscreteSignalPlot[{x[k]*DiscreteStep[k],
                        x[k],DiscreteStep[k]},{k,-10,20},
      AxesLabel->{"k","x(k)"},
      PlotRange->{-1.5,1.5},
      AxesOrigin->{-10,-1.5},
      PlotStyle->{{Thickness[.01],GrayLevel[0]},
                  Dashing[{.025}],Dashing[{.01}]},
GridLines->{{-10,0},{0,0}}];
```

The resulting plot is in Fig. 1.28. A new command `DiscreteStep[k]` is used for representing discrete-time step u_k.

We use `DiscretePulse[length,k]` to draw the discrete unit pulse signal (Fig 1.29). The number of discrete unit impulses is equal to the `length` with the first impulse at k=0.

```
DiscreteSignalPlot[DiscretePulse[3,k-4],{k,0,10},
      PlotStyle->{Thickness[.01]},
      AxesLabel->{"k","DiscretePulse[3,k-4]"}];
```

A unit discrete impulse is invoked by `KroneckerDelta[k]` (Fig. 1.30). It has a unit value for k=0 and zero value everywhere else.

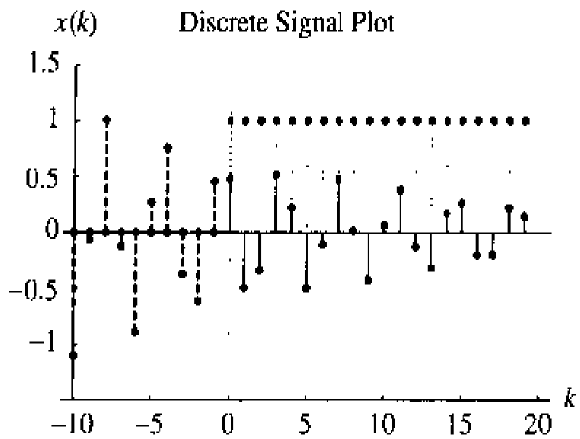

Figure 1.28 $u(k)$, $x(k) =$
$(0.02^{0.1} e^{-0.05\ k} \sin(\pi\ 0.53\ k + 3\pi/4))$ and
$x(k)\ u(k) =$
$(0.02^{0.1} e^{-0.05\ k} \sin(\pi\ 0.53\ k + 3\pi/4))u(k)$.

DiscretePulse [3, *k* – 4]

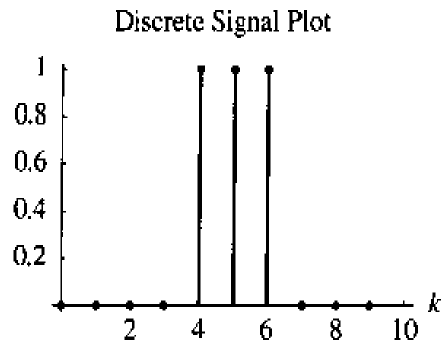

Figure 1.29 The discrete pulse signal.

KroneckerDelta [*k* – 7]

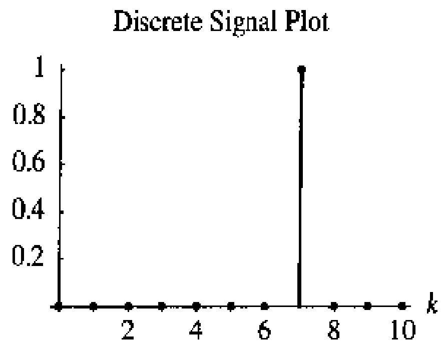

Figure 1.30 The unit impulse sequence, $\delta(k)$.

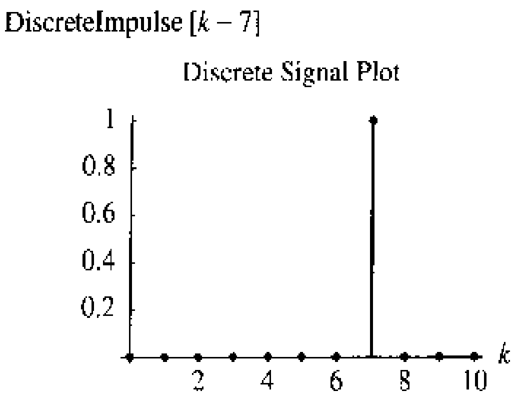

Figure 1.31 The unit impulse sequence, $\delta(k)$.

```
DiscreteSignalPlot[KroneckerDelta[k-7],{k,0,10},
    AxesLabel->{"k","KroneckerDelta[k-7]"}];
```

`KronekerDelta` is also called `DiscreteImpulse` as counterpart of `ContinuousImpulse` (Fig. 1.31).

```
DiscreteSignalPlot[DiscreteImpulse[k-7],{k,0,10},
    AxesLabel->{"k","DiscreteImpulse[k-7]"}];
```

A unit step sequence is invoked by `DiscreteStep[k]` (Fig. 1.32). It is zero for negative k and it has a unit value for k=0 and $k > 0$. We assume that the index of the first nonzero element in the step sequence is zero. This is opposite to standard *Mathematica* and MATLAB usage where the first index is 1.

```
DiscreteSignalPlot[DiscreteStep[k-3],{k,0,10},
    PlotRange->All,PlotStyle->{Thickness[.01]},
    AxesLabel->{"k","DiscreteStep[k-3]"}];
```

Sequences of data may be read from a file on disk or generated by a formula. We use the function `ToSampledData[function]` to create a special data structure to store the signal values (Fig. 1.33).

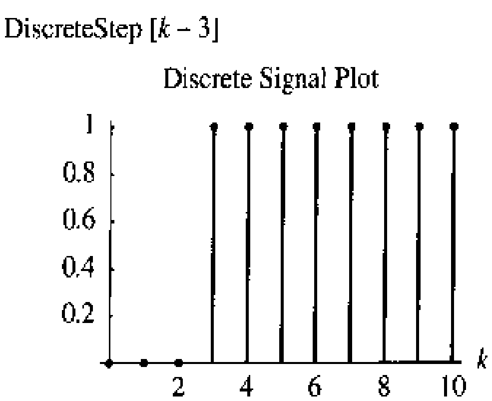

Figure 1.32 The discrete unit step sequence, $u(k)$.

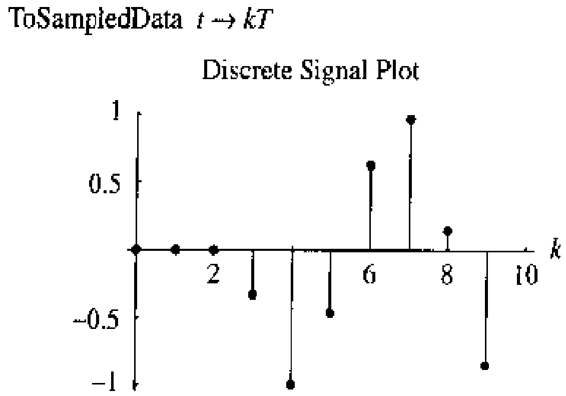

Figure 1.33 $\sin(2 \times 0.58k)u(2k-6)$.

```
DiscreteSignalPlot[
   N[ToSampledData[Sin[0.58 k 2]*ContinuousStep[k 2 - 6],
     {k,0,10}]], {k,0,10},
     PlotRange->All,AxesLabel->{"k","ToSampledData  t -> k T"}];
```

Continuous-time and discrete-time signals represented by formulas are easily manipulated as expressions. This enables us to simplify or rearrange expressions until they take a desired form even in the case when the signal duration is infinite.

PROBLEMS

1.1 Which ones of the following signals are continuous signals?

(a) $x(t) = e^{-t}$,

(b) $x(t) = \sin(2\pi t)$,

(c) $x(t) = \sin(4\pi t + \pi/2)$,

(d) $x(t) = \frac{1}{t}\sin(2\pi t)$,

(e) $x(t) = \left(\frac{(t+1)^2}{t} - \frac{(t-1)^2}{t}\right)\cos(2\pi t)$,

(f) $x(t) = \sin(4\pi t)\ |\cos(2\pi t)| + t - 1$,

(g) $x(t) = \mathrm{F}(t)$, and $\mathrm{F}(|t|) = |t|$, $\mathrm{F}(-|t|) = 0$,

(h) $x(t) = \mathrm{F}(\sin(2\pi t + \pi/4))$, and $\mathrm{F}(0) = 0$, $\mathrm{F}(|a|) = 1$, $\mathrm{F}(-|a|) = -1$.

1.2 Sketch the signals given in Problem 1.1 for values of t in the range $-2 < t < 2$.

1.3 Which ones of the signals given in Problem 1.1 are periodic signals?

1.4 Determine the period of the signal

$$x(t) = \left(\frac{(t+1)^2}{t} - \frac{(t-1)^2}{t}\right)\sin(2\pi t), \quad t \neq 0$$
$$x(0) = \lim_{t\to 0} x(t)$$

1.5 Sketch the following signals:

(a) $u(t-2)$,

(b) $-\frac{1}{2}u(t-3)$,

(c) $\frac{4}{5}u(t+4)$,

(d) $u(t-2)-u(t-6)$.

1.6 Sketch the following signals:

(a) $e^{-t}u(t-2)$,

(b) $2\left(1-e^{-t}\right)u(t-3)$,

(c) $e^{-t}\cos(2\pi t+\frac{\pi}{4})u(t)$,

(d) $e^{t}\cos(\pi t)\,u(t)$.

1.7 Which ones of the following signals are digital signals for $k \in \{-4,-3,-2,-1,0,1,2,3,4\}$?

(a) $x(k)=e^{-4k}$,

(b) $x(k)=\frac{k(k+1)(k+2)}{3}\sin(\pi k/2+\pi/4)$,

(c) $x(k)=\frac{1}{3}\sin(\pi k+\pi/2)$,

(d) $x(k)=\frac{1}{k}\sin(\pi k/2)$,

(e) $x(k)=\left(\frac{(k+4)^2}{k}-\frac{(k-4)^2}{k}\right)\sin(\pi k/2)$,

(f) $x(k)=\sin(\pi k)\ |\cos(\pi k/2)|+5k-1$,

(g) $x(k)=\sin(\pi k/2)\ |\cos(\pi k)|+4k-1$,

(h) $x(k)=\mathrm{F}(k)$, and $\mathrm{F}(|k|)=|k|$, $\mathrm{F}(-|k|)=0$,

(i) $x(k)=\mathrm{F}(\sin(\pi k/2))$, and $\mathrm{F}(0)=0$, $\mathrm{F}(|k|)=1$, $\mathrm{F}(-|k|)=-1$.

1.8 Sketch the following sequences:

(a) $u(k-2)$,

(b) $-\frac{1}{2}u(k-3)$,

(c) $\frac{4}{5}u(k+4)$,

(d) $u(k-2)-u(k-6)$.

1.9 Sketch the following sequences:

(a) $e^{-k}u(k-2)$,

(b) $2\left(1-e^{-k/5}\right)u(k-3)$,

(c) $e^{-k}\cos(\pi k+\frac{\pi}{4})u(k)$,

(d) $e^{k}\cos(\frac{\pi}{4}k)\,u(k)$.

1.10 Sketch the sawtooth signal $x(t) = \frac{2}{3}t\,(u(t) - u(t-3))$. Sketch the periodic sawtooth signal $y_1(t) = \sum_{k=-\infty}^{+\infty} x(t-3k)$. Determine the period of the signals $y_1(t) = \sum_{k=-\infty}^{+\infty} x(t-3k)$, $y_2(t) = \sum_{k=-\infty}^{+\infty} x(t+4k)$, and $y_3(t) = \sum_{k=-\infty}^{+\infty} x(t-5k)$.

1.11 Sketch the triangular pulse

$$x(t) = \begin{cases} 0, & t \le 0 \\ 4t, & 0 < t \le 1 \\ 5-t, & 1 < t \le 5 \\ 0, & 5 < t \end{cases}$$

Sketch the periodic triangular signal $y_1(t) = \sum_{k=-\infty}^{+\infty} x(t+5k)$. Determine the period of the signals $y_1(t) = \sum_{k=-\infty}^{+\infty} x(t-5k)$, $y_2(t) = \sum_{k=-\infty}^{+\infty} x(t+10k)$, and $y_3(t) = \sum_{k=-\infty}^{+\infty} x(t-15k)$. Use the unit step signal to describe analytically the signal $x(t)$.

1.12 Sketch the sequence $x(k) = u(k) - u(k-3)$. Sketch the sequences $y_1(k) = \sum_{m=-\infty}^{+\infty} x(k+3m)$, $y_2(k) = \sum_{m=-\infty}^{+\infty} x(k+4m)$, $y_3(k) = \sum_{m=-\infty}^{+\infty} x(k+5m)$. Determine the period of the sequences $y_1(k)$, $y_2(k)$, and $y_3(k)$.

1.13 Sketch the sequence

$$x(k) = \begin{cases} 0, & k < 0 \\ k, & 0 \le k < 2 \\ 2, & 2 \le k < 4 \\ 6-k, & 4 \le k < 6 \\ 0, & 6 \le k \end{cases}$$

Sketch the sequence $y_1(k) = \sum_{m=-\infty}^{+\infty} x(k-10m)$. Determine the period of the sequences $y_1(k) = \sum_{m=-\infty}^{+\infty} x(k-10m)$, $y_2(k) = \sum_{m=-\infty}^{+\infty} x(k-6m)$, and $y_3(k) = \sum_{m=-\infty}^{+\infty} x(k-4m)$. Use the unit step sequence to describe analytically the sequence $x(k)$.

1.14 Sketch the signal $x(t) = \sum_{k=1}^{3} \frac{4}{\pi}\frac{1}{2k-1}\sin(2\pi(2k-1)t + m\frac{\pi}{2})$. (a) $m = 0$; (b) $m = 1$; (c) $m = k$; (d) $m = k/5$.

MATLAB EXERCISES

1.1 Write a MATLAB script to plot the signals given in Problem 1.1 for values of t in the range $-2 < t < 2$.

1.2 Write a MATLAB script to plot the signals given in Problem 1.5 for values of t in the range $-5 < t < 15$.

1.3 Write a MATLAB script to plot the sequences given in Problem 1.7.

1.4 Write a MATLAB script to plot the sequences given in Problem 1.8 for values of k in the range $-10 \le k \le 20$.

1.5 Write a MATLAB script to plot the signals given in Problem 1.10 for values of t in the range $-20 < t < 20$.

1.6 Write a MATLAB script to plot the signals given in Problem 1.11 for values of t in the range $-20 \le t \le 40$

1.7 Write a MATLAB script to plot the sequences given in Problem 1.12 for values of k in the range $0 < k < 40$.

1.8 Write a MATLAB script to plot the sequences given in Problem 1.13 for values of k in the range $0 < k < 40$.

1.9 Define $x(t)$ as

$$x(t) = \sin(2\pi f t + \phi)$$

Write a MATLAB script to generate and plot sinusoidal sequences.

(a) Assume that $f = 0.1$, $\phi = \pi/3$, $t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$.

(b) Assume that $f = \frac{1}{2}$, $\phi = \pi/2$, $t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$.

(c) Assume that $f = \frac{1}{2}$, $\phi = \pi$, $t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$.

(d) Assume that $f = \frac{1}{2}$, $\phi = \pi/4$, $t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$.

Use the MATLAB functions `stem`, `bar`, and `stairs`.

1.10 Define $x(t)$ as

$$x(t) = A^{-t}\sin(2\pi f t + \phi), \quad t > 0$$

Write a MATLAB script to generate the quantized sequence by using the functions `ceil`, `floor`, `round`, and `fix`. Plot the sequence using the function `stem`. Assume that $f = 0.1$, $\phi = \pi/3$, $A = 1.1$, $t = 0, 1, 2, 3, \ldots, 20$. Each sample can be represented by its sign and the magnitude values

$$\{-1.0, -0.9, \ldots, -0.1, 0, 0.1, \ldots, 0.9, 1.0\}.$$

1.11 Define $x(t)$ as

$$x(t) = A\sin(2\pi f t + \phi)$$

Write a MATLAB script that generates the sinusoidal sequence from $x(t)$. Plot the sequence using the built-in function `stem`. Assume that $x(0) = 0$, $x(1) = -1$, $x(2) = 0$, $x(3) = 1$.

MATHEMATICA EXERCISES

1.1 Write a *Mathematica* code to plot the signals given in Problem 1.1 for values of t in the range $-2 < t < 2$.

1.2 Write a *Mathematica* code to plot the signals given in Problem 1.5 for values of t in the range $-10 < t < 20$.

1.3 Write a *Mathematica* code to plot the sequences given in Problem 1.7.

1.4 Write a *Mathematica* code to plot the sequences given in Problem 1.8 for values of k in the range $-10 \leq k \leq 20$.

1.5 Write a *Mathematica* code to plot the signals given in Problem 1.10 for values of t in the range $-10 < t < 20$.

1.6 Write a *Mathematica* code to plot the sequences given in Problem 1.12 for values of k in the range $-10 < k < 20$.

1.7 Define $x(t)$ as

$$x(t) = \sin(2\pi f t + \phi)$$

Write a *Mathematica* code to generate sinusoidal sequences from $x(t)$ and plot them using the function `ListPlot`.

(a) Assume that $f = 0.1, \phi = \pi/3, t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$;

(b) Assume that $f = \frac{1}{2}, \phi = \pi/2, t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$;

(c) Assume that $f = \frac{1}{2}, \phi = \pi, t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$;

(d) Assume that $f = \frac{1}{2}, \phi = \pi/4, t = \{-5, -4, \ldots, 0, \ldots, 4, 5\}$.

1.8 Define $x(t)$ as

$$x(t) = A^{-t}\sin(2\pi f t + \phi), \quad t > 0$$

Write a *Mathematica* code to generate a quantized sequence from $x(t)$ for $f = 0.1$, $\phi = \pi/3$, $A = 1.1$, $t = 0, 1, 2, 3, \ldots, 20$. Use the functions `Round`, `Floor`, and `Ceiling`. Assume that each sample can be represented by its sign and the magnitude value from $\{-1.0, -0.9, \ldots, -0.1, 0, 0.1, \ldots, 0.9, 1.0\}$. Plot the sequence using the function `ListPlot`.

1.9 Define $x(t)$ as

$$x(t) = A\sin(2\pi f t + \phi)$$

Write a *Mathematica* code to generate a sinusoidal sequence from $x(t)$. Assume that $x(0) = 0, x(1) = -1, x(2) = 0, x(3) = 1$. Find A, ϕ, and f. Plot the sequence using the function `ListPlot`.