CHAPTER

1

The Big Picture: Bioengineering Signals and Systems

1.1 WHY BIOMEDICAL ENGINEERS NEED TO ANALYZE SIGNALS AND SYSTEMS

With only slight exaggeration, just about everything we encounter can be called a system. Part of this is due to the broad definition of a system: a collection of processes or components that interact with a common purpose. In this book, we are interested in signals as well as systems, so for our purposes a system is defined as a collection of processes or components that operate on, or originate, one or more signals. The human body offers many examples of well-defined systems devoted to a common purpose. The cardiovascular system delivers oxygen-carrying blood to the peripheral tissues. The pulmonary system is devoted to exchanging gases (primarily O_2 and CO_2) between the blood and air. The mission of the renal system is to regulate water and ion balance and adjust the concentration of ions and molecules. Mass communication is the objective of the endocrine system with the distribution of signaling molecules through the blood stream, whereas the nervous system performs tightly controlled communication using neurons and axons to process and transmit information coded as electrical impulses.

Many medical specialties, particularly areas of internal medicine, emphasize a specific physiological system. Cardiologists specialize in the cardiovascular system, neurologists in the nervous system, ophthalmologists in the visual system, nephrologists in the kidneys, pulmonologists in the respiratory system, gastroenterologists in the digestive system, and endocrinologists in the endocrine system. Most other medical specialties are based on common tools or approaches such as surgery, radiology, and anesthesiology, whereas one specialty, pediatrics, is based on the type of patient.

Given this systems-based approach to physiology and medicine, it is not surprising that early bioengineers applied their engineering tools to some of these systems. Early applications of systems analysis included the quantification and explanation of the oscillatory respiratory patterns known as Cheyne–Stokes respiration, understanding the neural control factors underlying the speed and accuracy of the eye movements, and explaining the mechanism of pupil oscillations called hippus. The nervous system, with its apparent similarity to

Circuits, Signals, and Systems for Bioengineers
http://dx.doi.org/10.1016/B978-0-12-809395-5.00001-1

Copyright © 2018 Elsevier Inc. All rights reserved.

early computers, was another favorite target of bioengineers, as was the cardiovascular system with its obvious links to hydraulics and fluid dynamics. Some of these early efforts are described in the sections on system and analog models. As bioengineering has expanded into areas of molecular biology, systems on the cellular, and even subcellular, levels have utilized the tool of system analysis.

Irrespective of the type of biological system, it needs to interact with other systems and we bioengineers need a way of interacting with these systems. Signals, or more specifically "biosignals," carry information and mediate this intrasystem communication. By definition, signals carry information that we can use; if a "signal" contains no useful information, it is "noise."[1] (Quotes are used to highlight terms that are particularly important and should be an integral part of a bioengineer's vocabulary.) Physicians and biomedical researchers (who can be viewed as large, complex systems) use the information contained in biosignals to determine or diagnose the state of a physiological system. Such biosignals arise as changes in various biological or physiological variables. Common signals measured in diagnostic medicine include: electrical activity of the heart, muscles, and brain; blood pressure; heart rate; blood gas concentrations and concentrations of other blood components; and sounds generated by the heart and its valves.

Often, it is desirable to send signals into a biological system for purposes of experimentation or therapy. We often use the term "stimulus" for signals directed at a specific physiological system; if an output signal is evoked by these inputs, we term it a "response." In this scenario, the biological system acts like an input–output system, a classic model used in systems analysis and illustrated in Figure 1.1.

Examples of well-defined input–output physiological systems include the knee-jerk reflex, where the input is a mechanical force and the output is mechanical motion, and the pupillary light reflex, where the input is light and the output is a mechanical change in the iris muscles. Drug treatments can be included in this input–output description, where the input is the molecular configuration of the drug and the output is the therapeutic benefit (if any). Such system-analytic representations are further explored in the sections on systems and analog modeling.

Systems that produce an output without the need for an input stimulus, like the electrical activity of the heart, are considered biosignal "sources." (Although the electrical activity of the heart can be moderated by stimuli, such as exercise, the basic signal does not require a specific stimulus.) Input-only systems, like write-only memory, are not very useful since the purpose of an input signal is usually to produce some sort of response. An exception

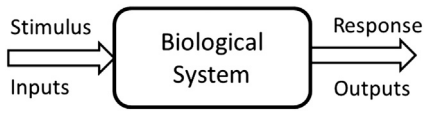


FIGURE 1.1 A classic systems view of a physiological system that receives an external stimulus or input signal that evokes a response or output.

[1]We sometimes use the term "random signal" which is a signal that does not carry information, so it really should be called "random noise." Put this down to sloppy terminology and just substitute "noise" for "signal" when you see it.

is the placebo, which is designed to produce no physiological response. (Nonetheless, it sometimes produces substantive results probably by interacting through complex, poorly understood neurological processes.)

Since all of our interactions with physiological systems are through biosignals, the characteristics of these signals are of great importance. Gleaning more information from these signals with signal processing tools is also very important in bioengineering. Indeed, much of modern medical technology is devoted to either extracting new physiological signals from the body or gaining more information from existing biosignals.

1.1.1 Goals of This Book

The tools of MATLAB make it possible for bioengineers to better analyze and understand signals and systems, the bread and butter of our professional lives. This book helps you to make the most of this powerful computer language in its applications to signal processing and systems analysis. The objective of this book is to give the reader a fundamental understanding of the field traditionally known as "linear systems analysis," but with concepts and applications in bioengineering. The basic ideas of linear systems analysis are well established and well understood. They can be divided into areas of signal analysis, system analysis, and the application of systems to signals known as signal processing.

The goal of signal analysis is to extract information from a signal by identifying features that are of particular interest. This includes basic and refined descriptions of a signal's time representation and a breakdown of its frequency content. This book covers the how and why of examining the correlations that occur between different portions of a signal. We find that correlation is a useful tool in signal analysis: we use it to determine similarities between two signals. When we use correlation to compare a signal and the (almost mystical) sinusoidal waveform, we discover that a bunch of sinusoids can give an alternative, yet complete representation of almost any signal. Moreover, applying a little algebra to this alternative "sinusoidal" representation can give us the "frequency spectrum" (or just "spectrum") of the signal.

In systems analysis, the basic goal is to be able to describe, quantitatively, the response of a system to a wide variety of stimuli. We will first learn to define system behavior analytically using differential equations. However, as is typical of engineers, we will find an easy way solve these equations using algebra. When digital systems are involved, they are described using difference equation rather than differential equations, but again we solve them algebraically. Surpassing these traditional analysis methods in ease and power is continuous system simulation implemented here with MATLAB's powerful system simulation tool, Simulink. This tool allows us to describe the behavior of very complex systems, even those that include nonlinear elements.

Mixing signal and system analysis, we use some special systems to alter signals. Applying these systems to a signal can produce an altered signal that is more useful to us. Often this increased utility is achieved by removing distracting components from the signal (i.e., noise). For example, we might remove frequencies in the signal that do not contribute to the signal's information content. Using a system to remove unwanted frequencies is known as "filtering" and the special system that does this is called, logically, a filter.

1.2 BIOSIGNAL: SIGNALS PRODUCED BY LIVING SYSTEMS

Biosignals are of utmost importance to us: much of our work, be it clinical or in research, involves the measurement, processing, analysis, and display of these signals. This section discusses some of the common physiological sources of biosignals and how these signals are measured.

1.2.1 Biological Signal Sources

Different physiological variations are supported by different energy forms. Table 1.1 summarizes the different energy forms that carry biological information, including the energy, the physiological variable that is changing, and the names of the biosignals associated with those variables.

Again, signals involve change. Within the body, communication between physiological systems uses signals encoded as changes in electrical or chemical energy. [2] Chemical energy encodes information by changing the concentration of the chemical within a "physiological compartment," for example, the concentration of a hormone in the blood. When speed is important, "bioelectric" signals are used. Since the body does not have many free electrons, it relies on ions, notably Na^+, K^+, and Cl^-, as the primary charge carriers.

TABLE 1.1 Energy Forms and Associated Information-Carrying Variables (i.e., Signals)

Energy	Variable Measured	Biosignal
Chemical	Chemical activity and/or concentration	Blood ions, O_2, CO_2, pH, hormonal concentrations, and other chemistry
Mechanical	Position	Muscle movement
	Force, torque, or pressure	Cardiovascular pressures, muscle contractility
		Valve and other cardiac sounds
Electrical	Voltage (potential energy of charge carriers)	EEG, ECG, EMG, EOG, ERG, EGG, GSR
	Current (energy in charge carrier flow)	
Thermal[a]	Temperature	Body temperature, thermography

ECG, electrocardiogram; *EEG*, electroencephalogram; *EGG*, electrogastrogram; *EMG*, electromyogram; *EOG*, electrooculogram; *ERG*, electroretinogram; *GSR*, galvanic skin response.
[a]*Thermal energy is the average kinetic energy of the molecules involved so it is actually mechanical energy.*

[2]Outside the body, information is commonly transmitted and processed as variations in electrical energy, although mechanical energy was used in the 18th and 19th centuries to send messages. The semaphore telegraph used the position of one or more large arms placed on a tower or high point to encode letters of the alphabet. These arm positions could be observed at some distance (on a clear day), and relayed onward if necessary. Information processing can also be accomplished mechanically, as in the early numerical processors constructed by Babbage in the early and mid-19th century. In the mid to late 20th century, mechanically based digital components were tried using variations in fluid flow.

1.2.2 Biotransducers and Common Physiological Measurements

Outside the body, electrically based signals are so useful that signals carried by other energy forms are usually converted to electrical energy when significant transmission or processing tasks are required. So even if we identify a physiological signal carrying information that we want, we usually need to convert that signal to one carried by electrical energy: an important, and often the first step in gathering information for clinical or research use. The energy conversion task is done by a device generally termed a "transducer," or "biotransducer."

A transducer is a device that converts energy from one form to another. By this definition, a light bulb or a motor is a transducer. In signal processing applications, the purpose of energy conversion is to transfer information, not to transform energy as with a light bulb or a motor. In physiological measurement systems, all transducers are so-called input transducers: they convert nonelectrical energy into an electronic signal. An exception to this is the electrode, a transducer that converts electrical energy from ionic to electronic form. The output of a biotransducer is a voltage (or current) whose amplitude is proportional to the measured energy. Figure 1.2 shows a transducer that converts acoustic sounds from the heart to electric signals. This "cardiac microphone" uses piezoelectric elements to convert the mechanical sound energy to electrical energy.

The energy that is converted by the input transducer may be generated by the physiological process itself as with heart sounds, or it may be energy that is indirectly related to the physiological process, although sometimes the energy being produced is only indirectly related to the system of interest, Figure 1.3A. For example, cardiac internal pressures are usually measured using a pressure transducer placed on the tip of a catheter, which is introduced into the appropriate chamber of the heart. The measurement of electrical activity in the heart, muscles, or brain provides other examples of direct measurement of physiological energy. For

FIGURE 1.2 A cardiac microphone that converts the mechanical sound energy produced by the beating heart into an electrical signal is shown. The device uses a piezoelectric disk that produces a voltage when it is deformed by movement of the patient's chest. The white foam pad is specially designed to improve the coupling of energy between the chest and the piezoelectric disk.

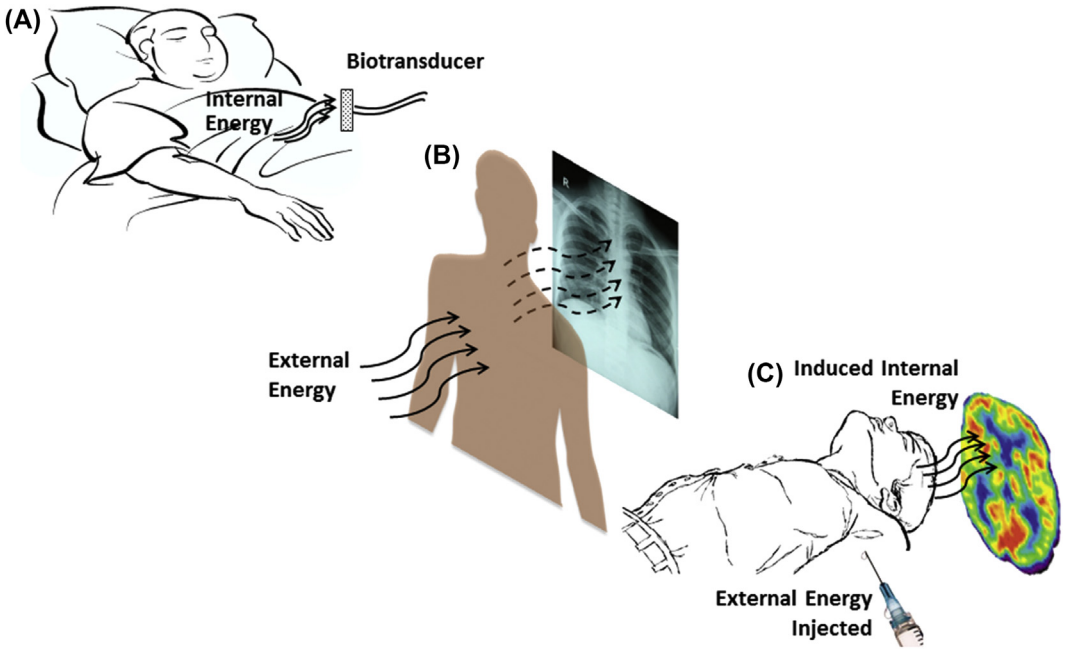


FIGURE 1.3 Three strategies for obtaining measurements from the body (or other physiological systems). (A) The body generates the energy, such as in heart sounds or electrically excitable tissue. (B) External energy is passed through the body, which is differentially blocked. For example, x-rays are differentially absorbed by body tissue and bone. In another example, external pressure applied to the arm induces sound generation used in the standard measurement of blood pressure. (C) The body is converted to a signal source by an external substance or energy source. For example, in positron emission tomography (PET), a radioisotope is introduced into the body. In magnetic resonance imaging, electromagnetic radiation and a strong magnetic field convert tissue protons into miniature radio transmitters.

these measurements, the bioelectric energy is already electrical and only needs to be converted from ionic to electronic current using an "electrode."[3] An early device for monitoring the electrocardiography (ECG) is shown in Figure 1.4. The interface between the body and the electrical monitoring equipment, the electrode, is buckets filled with saline ("E" in Figure 1.4).

In another approach, energy is injected from an external source and a difference in tissue absorption or reflection is measured by external transducers, Figure 1.3B. Examples include x-rays, computer-aided tomography, and ultrasound scans.

In the third approach, external energy is introduced in the physiological system, which then becomes an energy source that can be monitored externally, Figure 1.3C. Such energy can be in the form of short-lived radioisotopes introduced into the body that then accumulate

[3]Measurements of bioelectric energy are usually named using the rubric ExG, where the "x" represents the physiological process that produces the electrical energy: ECG, electrocardiogram; EEG, electroencephalogram; EMG, electromyogram; EOG, electrooculogram; ERG, electroretinogram; and EGG, electrogastrogram. An exception to this terminology is the galvanic skin response, GSR, the electrical activity generated by the skin.

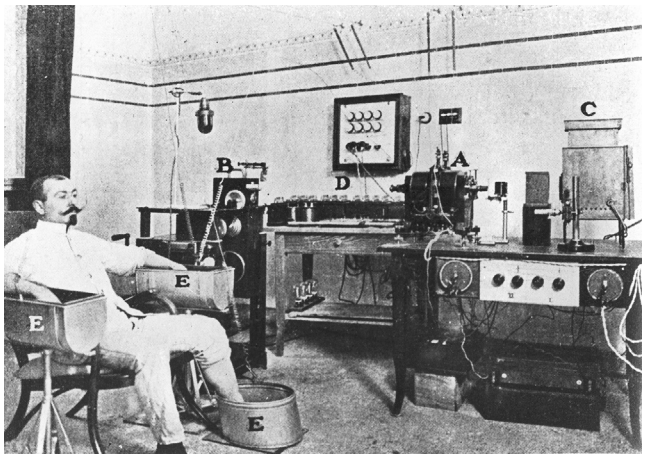


FIGURE 1.4 An early electrocardiography machine.

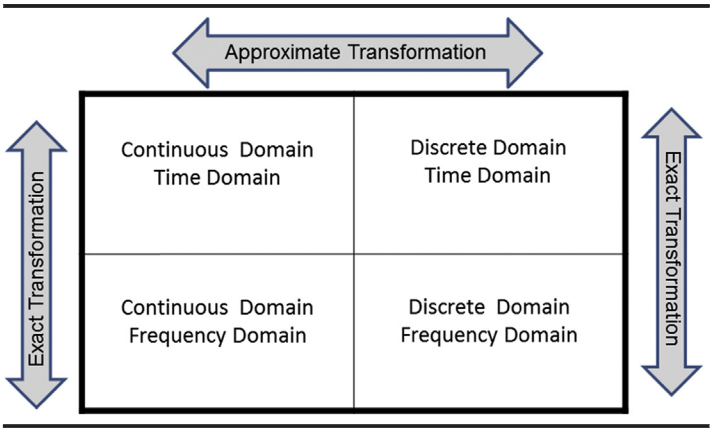
in a target system to be detected by radiation detectors (PET). In the best known example of this approach, body protons are induced to become radio transmitters by placing them in strong magnetic fields and exciting them with external radio frequency (rf) radiation, an approach known as magnetic resonance imaging (MRI).

The biotransducer is often the most critical element in the system, since it defines the accuracy or resolution of the measurement and acts as an interface between the life process and the rest of the system. The transducer establishes the risk, or "invasiveness," of the overall system. For example, an imaging system based on differential absorption of x-rays, such as a computed tomography (CT) scanner, is considered more invasive than an imaging system based on ultrasonic reflection, since CT uses ionizing radiation that may have an associated risk. (The actual risk of ionizing radiation is still an open question, and imaging systems based on x-ray absorption are considered minimally invasive.) Both ultrasound and x-ray imaging would be considered less invasive than, for example, monitoring internal cardiac pressures through cardiac catheterization in which a small catheter is threaded into the heart chamber. Indeed, fame and fortune await the bioengineer who solves any of the many outstanding problems in biomedical measurement; for example, the noninvasive measurement of internal cardiac pressures or intracranial pressure.

1.2.3 Signal Representation: Continuous (Analog) Signals Versus Discrete (Digital) Signals

Signals fall into two broad categories: one-dimensional signals, where the dimension is usually time, and two-dimensional signals, where the dimensions are usually spatial. Here we are primarily concerned with one-dimensional signals, but many of the operations we apply to one-dimensional signals are easily extended to two-dimensional signals. Both types of signals can be further classified based on the way they are represented. These signal

TABLE 1.2 Signal Representations



representations are unique: a signal is represented in either the continuous or discrete domain, and in either the time or frequency domain. This results in the possible combinations shown in Table 1.2. Methods for transferring between time and frequency domains are described for continuous signals in Chapter 3 and for discrete signals in Chapter 4. These transformations are exact: there is no loss of signal information through the transfer. Transformation between continuous and discrete signals is applied only in the time domain and is described later. This transformation is only approximate and, depending on the way it is done, there could be a loss of signal information.

Signals also have some important properties that we need to understand before we analyze them. Important signal properties include deterministic (or chaotic) versus stochastic; linear versus nonlinear; and stationary or time-invariant versus nonstationary. These properties are not necessarily unique: a signal can be basically linear but still have some nonlinearity properties and the same could be said of the other properties. Usually, signal properties are due to the system that created the signal, so we discuss these properties in Section 1.4 along with system properties.

As the name implies, continuous signals vary in time and amplitude in a continuous manner, whereas discrete signals have discrete levels of amplitude and exist at discrete moments in time. Continuous signals are often called "analog signals" and are commonly found in the real world. Discrete signals, often called "digital signals," are found inside computers or in discrete logic circuits.[4] Analog or digital signals can be defined by their mathematical domain. In the continuous domain, information is represented by a signal's amplitude (i.e., the value), which can change continuously over time.[5] In the digital domain, a signal is

[4]The central nervous system uses both. Action potentials transmitted through the axons are discrete, whereas the nerve body maintains a continuously variable, analog voltage.

[5]Analog signals may also be encoded as a continuous change in frequency, so-called frequency modulation (FM), or change in phase, termed phase modulation, but such encoding schemes are not common in nature and are not found in biosignals.

represented by numbers and different encoding schemes can be used. Signals are usually represented as a series of discrete numbers sampled at discrete points in time, so they are doubly discrete: in time and in value.

1.2.3.1 *Analog Signals*

An analog signal can be mathematically represented by the equation:

$$x(t) = f(t) \tag{1.1}$$

where $f(t)$ is some function of time and can be quite complex. For an electronic signal, $x(t)$ is the value of the voltage (occasionally current) at a given time. Some signals are so complicated that it is impossible to find a mathematical expression for $f(t)$ and the signal must be presented graphically. Again we emphasize that all signals are by nature "time-varying," since a time-invariant value contains no information: it is just some meaningless number.[6]

Often, an analog signal encodes the information as a linear change in signal amplitude. For example, a temperature transducer might encode room temperature into voltage as shown in Table 1.3 below:

This encoding could be defined by the equation:

$$\text{temperature} = 2 \times \text{voltage amplitude} - 10 \tag{1.2}$$

Analog signals and linear encoding are common in consumer electronics, such as within hi-fi amplifiers, although many applications that traditionally used analog encoding, such as television and sound and video recording, now primarily use digital signals. An interesting exception is the resurgence of music recorded as an analog signal on vinyl (i.e., records), including analog playback, as it is thought by some to have better sound characteristics. In addition, analog encoding remains important to the biomedical engineer because many physiological systems use analog encoding, and most biotransducers generate

TABLE 1.3 Relationship Between Temperature and Voltage of a Hypothetical Temperature Transducer

Temperature (°C)	Voltage (volts)
−10	0.0
0.0	5.0
+10	10.0
+20	15.0

[6]Modern information theory makes explicit the difference between information and meaning. The latter depends on the receiver, that is, the device or person for which the information is intended. Many students have attended lectures with a considerable amount of information that, for them, had little meaning. This text strives valiantly for both information and meaning.

analog signals. In living systems, not all analog signals are linearly encoded, although we often make that assumption as an approximation to allow us to use advanced signal analysis methods. In this book, the assumption is that all analog signals are linearly encoded unless otherwise stated.

1.2.3.2 Digital Signals

Signals may originate in the analog domain, but they usually need to be represented, processed, and stored in a digital computer. To transform a signal from the analog to digital domain requires that the continuous analog signal be converted to a series of numbers through a process known as "sampling." Since digital numbers can only represent discrete or specific amplitudes, the analog signal must also be sliced up in amplitude, a process called "quantization." Hence digitizing or sampling an analog signal requires slicing the signal two ways: in time and in amplitude. The details of the sampling process and its limitations are discussed in the next two sections, but essentially the continuous analog signal, $x(t)$, is sliced up into a sequence of discrete numbers, usually at equal time intervals, T_s. Such time intervals are called the "sampling interval" as shown in Figure 1.5.

1.2.3.2.1 TIME SAMPLING

A time-sampled, quantized signal, also referred to as a "digitized signal" or simply "digital signal," can be easily stored in a digital computer. A digital signal, $x[k]$, is just a series of discrete numbers: $x[k] = x_1, x_2, x_3, \ldots x_N$. These sequential numbers approximate, after

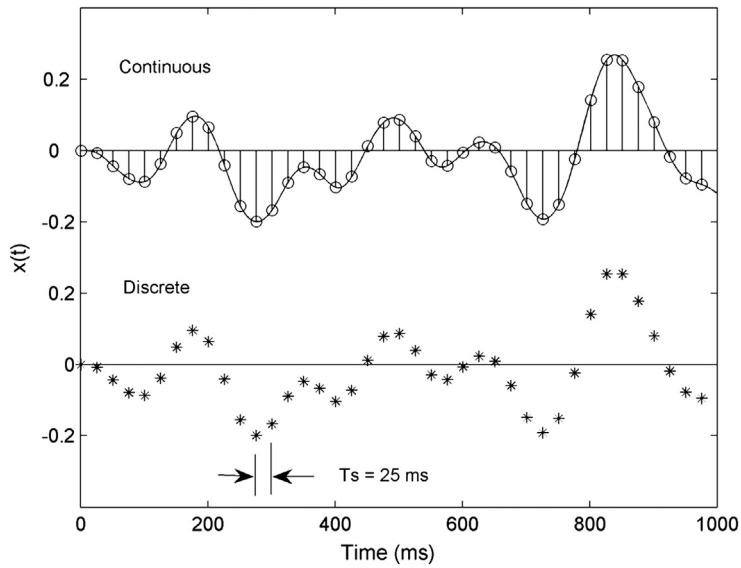


FIGURE 1.5 Digitizing a continuous signal, upper curve, requires slicing the signal in time and amplitude. The result is a series of discrete numbers (* points) that approximate the original signal, but can be stored in a computer. Time slicing is known as sampling and amplitude slicing (or rounding) as quantization. Here the interval between time slices, the sample interval, is 25 ms.

rounding, the value of the analog signal at a discrete point in time determined by the sample interval, T_s, or by the sampling frequency, f_s:

$$t = nT_s = n/f_s \tag{1.3}$$

where n is the position of the number in the sequence, t is the signal time represented by the number, and f_s is the inverse of the sample interval:

$$f_s = 1/T_s \tag{1.4}$$

Usually this series of numbers would be stored in sequential memory locations with $x[1]$ followed by $x[2]$, then $x[3]$, etc. As is common, we use brackets to identify a discrete or digital variable, $x[k]$, whereas parentheses are used for continuous variables, $x(t)$. [7]

In some advanced signal processing techniques, it is useful to think of the sequence of n numbers representing a signal as a single vector pointing to the combination of n numbers in an n-dimensional space. Fortunately, this challenging concept (imagine a 256-dimensional space, or even a space with only 5 dimensions) is not used here except in passing, but it gives rise to use of the term "vector" when describing a series of numbers such as $x[k]$. This terminology is also used by MATLAB, so in this text the term "vector" translates to "sequence or array of numbers" often representing a time signal.

Time slicing samples the continuous waveform, $x(t)$, at discrete points in time, nT_s, where T_s is the sample interval and $n = 1,2,3,\ldots$. The consequences of time slicing depend on the signal characteristics and the sampling time, and are discussed in Chapter 4.

1.2.3.2.2 QUANTIZATION

Slicing the signal amplitude in discrete levels, quantization, is shown in Figure 1.6. The equivalent number can only approximate the level of the analog signal, and the degree of approximation depends on the range of numbers used and the amplitude of the analog signal. For example, if the signal is converted into an 8-bit binary number, the range of numbers is 2^8 or 256 discrete values. If the analog signal amplitude ranges between 0.0 and 5.0 V, then the quantization interval is 5/256 or 0.0195 V. If the analog signal is continuous in value, as is usually the case, it can only be approximated by a series of binary numbers representing the approximate analog signal level at discrete points in time as seen in Figure 1.6. The errors associated with quantization are described in Chapter 3.

[7]Note that the MATLAB programming language used throughout this text uses parentheses to index a variable (i.e., `x(1), x(2) x(3)`…) even though MATLAB variables are clearly digital. MATLAB reserves brackets for concatenation of numbers or variables. In addition, MATLAB uses the integer 1, not zero, to indicate the first number in a variable sequence. This can be a source of confusion when programming some equations that use zero as the index of the first number in a series. Hence x[0] in an equation translates to `x(1)` in a MATLAB program. (All MATLAB variables and statements used in the book are typeset in a `sans serif typeface` for improved clarity.)

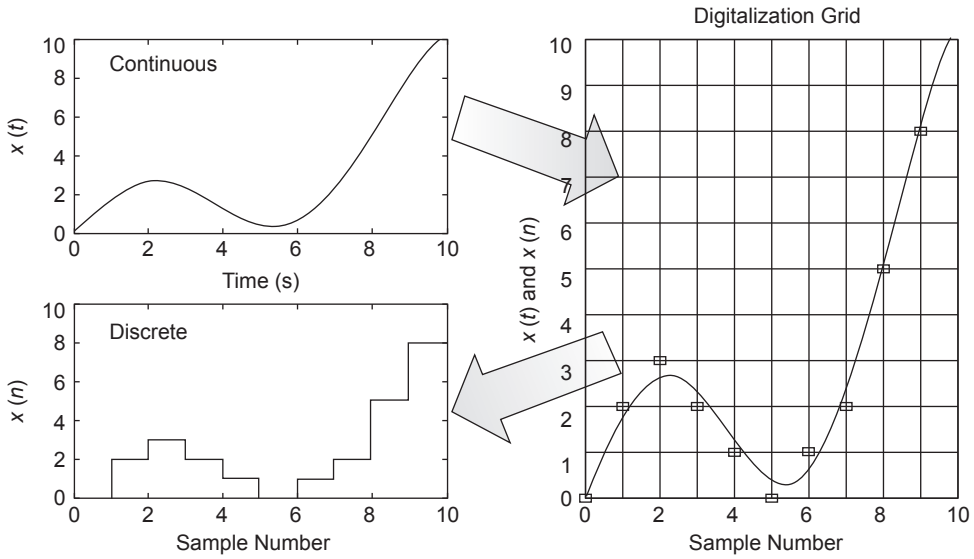


FIGURE 1.6 Digitizing a continuous signal, upper left, requires slicing the signal in time and amplitude, right. The result, lower left, is a series of numbers that approximate the original signal as a series of discrete levels at discrete time values. This digitizing operation is also known as analog-to-digital conversion.

EXAMPLE 1.1

A 12-bit analog-to-digital converter (ADC) advertises an accuracy of ± the least significant bit (LSB). If the input range of the ADC is 0 to 10 V, what is the accuracy of the ADC in analog volts?

Solution: If the input range is 10 V, then the analog voltage represented by the LSB is:

$$V_{LSB} = \frac{V_{\max}}{2^{\text{Nu bits}} - 1} = \frac{10}{2^{12} - 1} = \frac{10}{4095} = .0024 \text{ V}$$

Hence the accuracy would be ±.0024 V.

It is relatively easy, and common, to convert between the analog and digital domains using electronic circuits specially designed for this purpose. Many medical devices acquire the physiological information as an analog signal, then convert it to digital format using an "analog-to-digital converter" ("ADC") for subsequent computer processing. For example, the electrical activity produced by the heart can be detected using properly placed electrodes, and the resulting signal, the electrocardiogram (ECG), is an analog encoded signal. This signal might undergo some "preprocessing" or "conditioning" using analog electronics, but would eventually be converted to a digital signal using an ADC for more complex, computer-based processing and storage. In fact, conversion to digital format is usually done even when the data are only stored for later use.

Transformation from the digital to the analog domain is possible using a "digital-to-analog" converter ("DAC"). Most PCs include both ADCs and DACs as part of a sound

card. This circuitry is specifically designed for the conversion of audio signals, but can be used for other analog signals. Data transformation cards and USB-driven devices designed as general-purpose ADCs and DACs are readily available and offer greater flexibility in sampling rates and conversion gains. These devices generally provide multichannel ADCs (usually 8−16 channels) and several DAC channels.

In this text, the basic concepts that involve signals are often introduced or discussed in terms of analog signals, but most of these concepts apply equally well to the digital domain, assuming that the digital representation of the original analog signal is accurate. The equivalent digital domain equation is presented alongside the analog equation to emphasize the equivalence. Many of the problems and examples use a computer, so they are necessarily implemented in the digital domain even if they are presented as analog domain problems.

Is a signal that has been transformed from the continuous to the discrete domain the same? Clearly not; just compare the two different signals in Figure 1.5. Yet in signal analysis we often operate on discrete signals converted from an analog signal with the expectation (or assumption) that the discrete version is essentially the same as the original continuous signal. If they are not the same, is there at least some meaningful relationship between the two? The definitive answer is, maybe. The conditions necessary for the existence of a meaningful relationship between a continuous signal and its discrete version are described in Chapter 4. For now we will assume that all computer-based signals used in examples and problems are accurate representations of their associated continuous signals. In Chapter 4 we look at the consequences of the analog-to-digital conversion process in more detail and establish rules for when a digitized signal can be taken as a truthful representation of the original analog signal.

1.2.3.3 Time and Frequency Domains

We are already familiar with time domain signals from our discussion of the transformation of time domain signals between analog and digital domains. Frequency domain signals can also exist, or at least be analyzed, in either domain. Signals are represented in the frequency domain as two functions of frequency: a magnitude and a phase.

Figure 1.7 shows a section of an electroencephalography (EEG) signal in the time domain (left) and frequency domain (right). To completely represent the signal in the frequency domain, two functions are required, a magnitude (upper right) and a phase (lower right). The time domain representation is not easy to interpret; it looks like a disorganized bunch of squiggles. You could say the same for the phase; it is usually difficult to interpret. Because it is needed to fully represent the time domain signal, it is determined, even though it is not informative. On the other hand, the magnitude does show some interesting features. Specifically, there are peaks around 1−3 Hz and 16−18 Hz, and a large peak around 7−8 Hz. These peaks in frequency are well known to neurologists: the 1−3 Hz peaks are called theta waves, the 16−18 Hz peaks are called beta waves, and the 7−8 Hz peaks are called alpha waves, which have been associated with states of meditation. The important point is that in this case, these features could not be easily detected in the time domain representation (although they are there), but are very easy to spot in the magnitude of the frequency spectrum.

Because the frequency domain signals are functions of frequency, there are called spectra, the magnitude spectrum and phase spectrum. Although the time and frequency domain representation hold the same information, they do not store it in the same way so that one or the other may be more diagnostically useful in a given situation. It is fairly easy to transform from one domain to the other using techniques presented in Chapters 3 and 4.

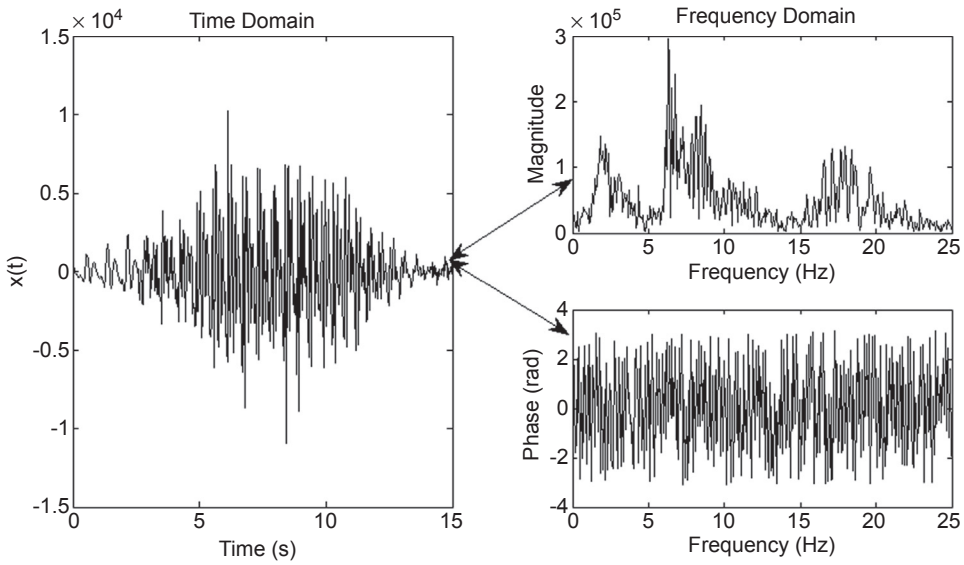


FIGURE 1.7 Time and (left) frequency domain representations of a 15-s segment of an EEG signal. The two representations are equivalent in terms of the information they contain, but they do present this information differently. Features seen in the frequency domain magnitude plot are not obvious in the time domain. These features, distinct peaks, have clinical relevance.

1.2.4 Two-Dimensional Signals ~ Images

The signals we have discussed thus far are one-dimensional functions of time that are represented in MATLAB as arrays or vectors. It is possible to have multiple signals that have some common origin and they should be treated together. A common biomedical example is the multilead EEG signal, where the electrical activity of the brain is recorded from 16, 32, or even 64 different locations on the head. Some analyses described in the next chapter treat these signals in combination, looking at features common to these signals as well as analyzing them individually. Multiple signals are often represented as a matrix where each row is a different signal.

Images in the digital domain, images that have been sampled, can be considered two-dimensional signals and can be represented as matrices. Each number in the matrix represents the intensity of a small region of space. These regions are usually square and are called pixels, so each number in the matrix represents a pixel. The distance between pixels is the spatial interval, the two-dimensional equivalent of the sample interval of a digitized signal.

MATLAB has a special toolbox for image processing that provides a number of useful routines for manipulating images, but image processing can be done using standard MATLAB routines. As an example, we generate a simple image: one consisting of vertical bars that vary sinusoidally in the horizontal direction. When considered and displayed as an image, a matrix consisting of identical rows of sinusoids looks like a sinusoidally varying pattern of vertical bars. Such an image is called a "sinusoidal grating" and is used in experiments on the human visual system.

EXAMPLE 1.2

Construct the image of sine wave bars having four cycles across the image. Make the image 100 by 100 pixels.

Solution: The strategy is straightforward; construct a single sine wave in a 100-point array, then make the vertical bars by copying that sine wave into sequential rows of a matrix. If you are not a MATLAB guru, constructing the sine wave is a nontrivial achievement, but very worthwhile as we will be doing similar operations throughout this book. If your MATLAB is a little rusty, jump ahead and check out Example 1.6, where the only goal it to construct and display a sine wave.

To generate the 100-point, four-cycle sine wave, we take advantage of MATLAB's ability to operate on a string of numbers (i.e., arrays) with a single statement. To construct a one-cycle sine wave, all we need is an array of numbers that go from 0 to 2π. If that array is labeled `n`, then the command, `x = sin(n);` would give us a one-cycle sine wave in array `x` where `x` is the same length as `n`.[8] Since we want a 4-cycle sine wave, array `n` should range between 0 and 8π. Since we want that 4-cycle sine wave to be 100 points long, we need `n` to be 100 points long. We can generate a 100-point array whose numbers range between 0 and 8π with the command: `n = (0:99)*8*pi/100;`. To be more general, we could define the array for a variable length, `N`.

```
N = 100;               % Number of points in the array, 100 points
n = (0:N-1)8*pi/N ;    % N-point array with numbers between 0 and 0.99
```

Then:

```
x = sin(n);     % Construct array with 4-cycle sine wave
```

More commonly, we would put the 8*pi in the sin argument because as we see later, that makes the frequency of the sine wave explicit. So the instructions to construct the sine wave become:

```
N = 100;              % Number of points in the array, 100 points
n = (0:N-1)/N ;       % N-point array with numbers between 100-point array
x = sin(8*pi*n);      % Construct array with 4-cycle sine wave
```

Duplicating that sine wave over 100 rows of a matrix is straightforward. If we call the matrix `I` (for image):

```
for k = 1:N
  I(k,:) = y;  % Duplicate 100 times
end
```

When displayed as an image, a matrix consisting of identical rows of sinusoids would look like a sinusoidally varying pattern of vertical bars. Such an image is called a "sinusoidal grating" and is used in experiments on the human visual system. There are a number of ways to display a matrix as

an image in MATLAB. In this example, we use MATLAB's `pcolor` and smooth the resulting display through interpolation using the MATLAB routine `shading interp`. This will generate a smooth image with color variation reflecting the sine wave's value. Images in this book are limited to black and white or levels of gray so we convert the color scale to a "grayscale" image using the command `colormap (bone)`.[9] The complete program becomes:

```
% Example 1.2 Construct the image of a sine wave grating having 4 cycles
%
N = 100;          % Number of pixels per line and number of lines
x = (0:N-1)/N;      % Spatial vector
y = sin(8*pi*x);  % Four cycle sine wave
for k = 1:N
  I(k,:) = y;     % Duplicate 100 times
end
pcolor(I);        % Display image
shading interp;   % Use interpolation
colormap(bone);   % Use a grayscale color map
```

Result: The image produced by this program is shown in Figure 1.8. The sinusoidal variation in gray levels is compromised somewhat in the printing process.

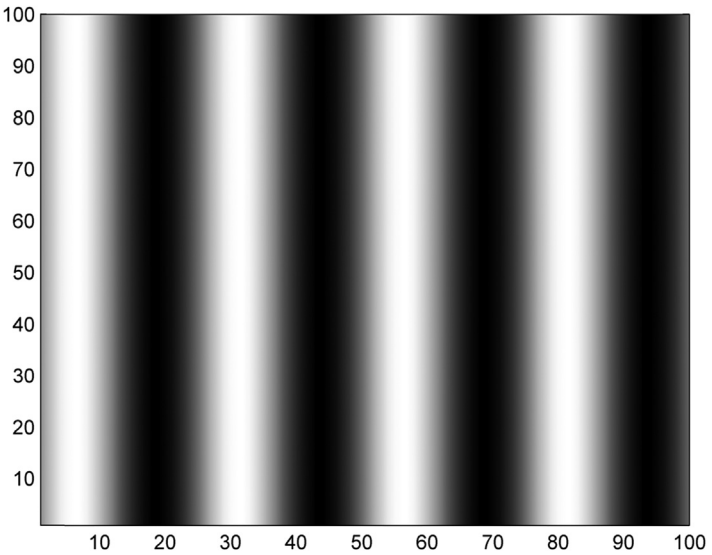


FIGURE 1.8 A sinusoidal grating generated in Example 1.3.

[8]As mentioned in the Preface, MATLAB code is distinguished in this book by a different `typeface`.
[9]Other MATLAB colormaps can be used in problem solutions to get some fun, near psychedelic effects! Try `colormap(jet)`, for example.

The grating in Figure 1.8 has a sinusoidal variation in intensity across the image. Since this is an image and not a time plot, the horizontal axis is position, not time, and the variation is a function of spatial position. The frequency of a sine wave that varies with position is called "spatial frequency" and is in units of cycles/distance. If, in this book, Figure 1.8 is printed to have a width of 4 inches, then the horizontal frequency of the sine wave would be 1.0 cycle/in. There is no variation vertically so the vertical frequency of this image is 0.0 cycles/in. One of the problems generates an image that has a sinusoidal variation in both dimensions.

Just about any MATLAB operation can be applied to an image, at least when it is represented as a standard MATLAB variable; conceptually the image is just another matrix to MATLAB. The next example shows some basic mathematical and threshold operations applied to an magnetic resonance (MR) image of the brain.

EXAMPLE 1.3

This example implements some standard MATLAB operations to a matrix that is actually an MR image of the brain. This image is stored as variable `I` in file `brain.mat`. We lighten the image by adding a constant to the entire array and enhance the contrast by multiplying the matrix by a constant. Finally, we threshold the image and make the darker regions white and all others as black (inverted with respect to the normal MATLAB coding of dark and light).

Solution: The file is loaded in the usual way using `load brain.mat`. The image is scaled to be between 0 (black) and 1.0 (white), a convention used by MATLAB for scaling image data. Since `pcolor` normalizes the data each time it is called, we will fix the display range to be between 0 and 1 using the `caxis` routine: `caxis([0 1])`. After displaying the original image as in the preceding example, we will lighten the image by increasing all the pixel values in the image by a value of 0.3 (remember 1.0 is white) and display. We then multiply the original image by 1.75 to increase its contrast and display. To apply the threshold operation, we examine every pixel and generate a new image that contains a 1.0 (i.e., white) in the corresponding pixel if the original pixel is below 0.25 and a 0 (i.e., black) otherwise. This image is then displayed.

```
% Example 1.3  Example to apply some mathematical and threshold operations
%  to an MR image % of the brain.
%
load brain;         % Load image
subplot(2,2,1);     % Display the images 2 by 2
pcolor(I);          % Display original image
shading interp;     % Use interpolation
colormap(bone);     % Grayscale color map
caxis([0 1]);       % Fix pcolor scale
title('Orignal Image');
%
subplot(2,2,2);
I1 = I + .3;        % Brighten the image by adding 0.3
  .......displayed and titled as above.......
%
subplot(2,2,3);
I1 = 1.75*I;       % Increase image contrast by multiplying by 1.75
  .......displayed and titled as above.......
%
```

```
subplot(2,2,4);
[r, c] = size(I);        % Get image dimensions
thresh = 0.25;           % Set threshold
for k = 1:r
  for j = 1:c
    if I(k,j) < thresh   % Test each pixel separately
      I1(k,j) = 1;       % If low make corresponding pixel white (1)
    else
      I1(k,j) = 0;       % Otherwise make it black (0)
    end
  end
end
  .....displayed and titled as above.....
```

Results: The four images produced by this program are shown in Figure 1.9. The changes produced by each of these operations are clearly seen. Many other operations could be performed on this, or any other image, using standard MATLAB routines. A few are explored in the problems and others in Chapter 11.

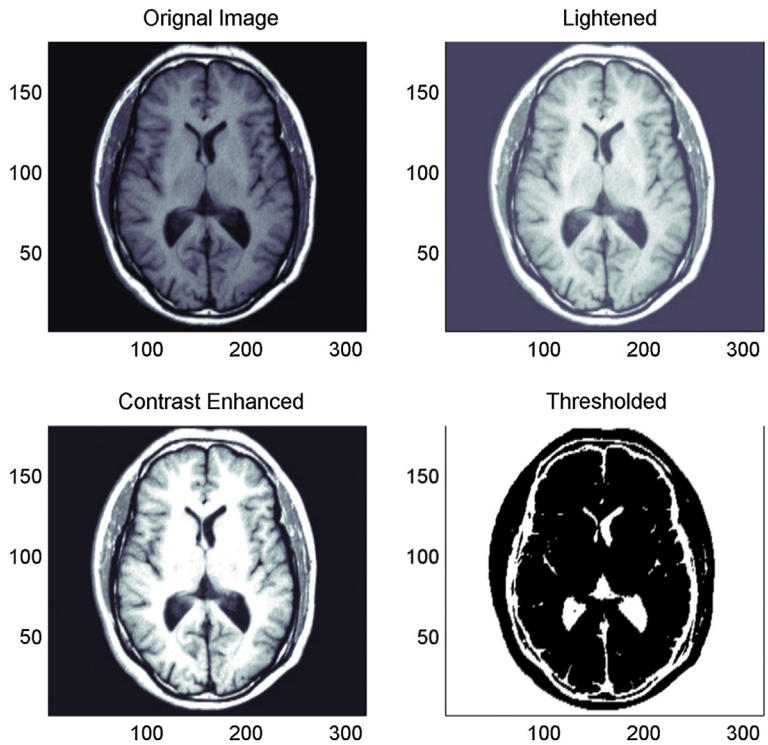


FIGURE 1.9 Images of the brain produced in Example 1.3. The MR image (upper left) has been brightened (upper right), contrast enhanced (lower left), and thresholded (lower right). In the thresholded image, the darker regions are shown as white and all other regions as black.

1.3 NOISE

All signals carry information as variations in energy (see Table 1.1). However, the reverse is not necessarily true: variations in energy do not always carry information, or at least not the information we want. We call these useless variations noise. When we make a real-world measurement, we usually get a combination of signal and noise. As bioengineers, we put a lot of effort into separating these components.

Where there is signal there is noise. Noise can originate in a variety of sources, including the biological system being measured, the measurement process, or the measurement environment. Random processes are known as "stochastic processes" and the resulting signal is said to be "stochastic" or can show stochastic behavior. When referring to signals, the terms "noise" or "stochastic behavior" are pretty much synonymous. The term "drift" also describes a noise, but the word implies that the stochastic behavior occurs at lower frequencies.

The extent of noise in a signal can vary widely and in the next chapter, we quantify the amount of noise in a signal as a ratio of the signal over noise in equivalent units. Occasionally the randomness will be at such a low level that it is of little concern, but more often it is this randomness that limits the amount of information we can get out of the signal. This is especially true for physiological measurements since they are subject to many potential sources of noise. Noise often limits the diagnostic value of a medical instrument and many devices employ advanced signal conditioning circuitry and signal processing algorithms in efforts (not always successful) to compensate.

1.3.1 Biological Noise Sources

Noise or stochastic behavior in biomedical measurements has four possible origins: (1) physiological indeterminacy; (2) environmental noise or interference; (3) measurement or transducer artifact; and (4) electronic noise.

Dealing with physiological indeterminacy is tough because the information you desire is based on measurements that are also influenced by other biological processes. For example, an assessment of respiratory function based on the measurement of blood pO_2 could be confused by other physiological mechanisms that alter blood pO_2, such as heart rate. Trying to determine pressure inside the heart from measurements on peripheral vessels will be affected in an unpredictable way by characteristics of the blood vessels and tissues. Physiological indeterminacy can be a very difficult problem to solve, sometimes requiring a total redesign of the approach.

Environmental noise can come from sources external or internal to the body. A classic example is the measurement of the fetal ECG signal where the desired signal is corrupted by the mother's ECG. Since it is not possible to describe the specific characteristics of environmental noise, typical noise reduction approaches such as filtering (described in Chapter 9) are not usually successful. Sometimes environmental noise can be reduced using "adaptive filters" or "noise cancellation" techniques that adjust their filtering properties based on the current situation.

Measurement artifact is produced when the biotransducer responds to energy modalities other than those desired. For example, the ECG recordings that reflect cardiac electrical

activity are made using electrodes placed on the skin. These electrodes are also sensitive to movement, so-called motion artifact, where the electrodes respond to mechanical movement as well as to the electrical activity of the heart. This artifact is not usually a problem when the ECG is recorded in the doctor's office, but it can be if the recording is made during a patient's normal daily living, as in a 24-hour "Holter" recording.[10] Measurement artifacts can sometimes be successfully addressed by modifications in transducer design. For example, aerospace research has led to the development of electrodes that are quite insensitive to motion artifact.

Electronic noise is due to stochastic processes within the transducer elements or associated electronics. Unlike other sources of variability, the characteristics of these processes are pretty well known. Appropriate transducer and electronic design can often reduce this noise. Since electronic noise is well defined and can be addressed through appropriate design decisions, the next section examines these noise sources in more detail.

The various sources of noise or variability along with their causes and possible remedies are presented in Table 1.4. Note that in three of four instances, appropriate transducer design may aid in the reduction of the variability or noise, another example of the important role the transducer plays in the overall performance of a medical device.

1.3.2 Noise Properties: Additive Gaussian Noise

Noise is a common enemy and since it is useful to know your enemy, we should try to understand its properties. Like signals, noise can have different properties, but one set of properties, additive Gaussian noise, is of particular interest to biomedical engineers. The properties of Gaussian noise are well known: it adds a random variable to your signal and that variable has a Gaussian or normal distribution. In Chapter 2, we develop a simple signal processing technique that can reduce additive Gaussian noise in some situations.

TABLE 1.4 Sources of Variability

Source	Cause	Potential Remedy
Physiological indeterminacy	Measurement indirectly related to variable of interest	Modify overall approach
Environmental (internal or external)	Other sources of similar energy	Apply noise cancellation
		Alter transducer design
Artifact	Transducer responds to other energy sources	Alter transducer design
Electronic	Thermal or shot noise	Alter transducer or electronic design

[10]A Holter monitor is a battery-operated portable device that continuously measures and records the ECG for 24–48 h. The patient maintains his or her daily routine and keeps a diary of any cardiac abnormalities. Patient-reported symptoms are then compared with the cardiac electrical activity at the time of the event.

Since noise is random, a time function or time plot is not particularly useful. It is more common to discuss other properties of noise such as its probability distribution, its variance, or its frequency characteristics as described in Chapter 2. Although noise can take on a variety of different probability distributions, the "Central Limit Theorem" implies that most noise will have a Gaussian or normal distribution.[11] The Central Limit Theorem states that when noise is generated by a large number of independent sources, it will have a Gaussian probability distribution regardless of the probability distribution of the individual sources.

Figure 1.10 provides a dramatic demonstration of the Central Limit Theorem at work. In Figure 1.10A, the distribution of 20,000 uniformly distributed random numbers between

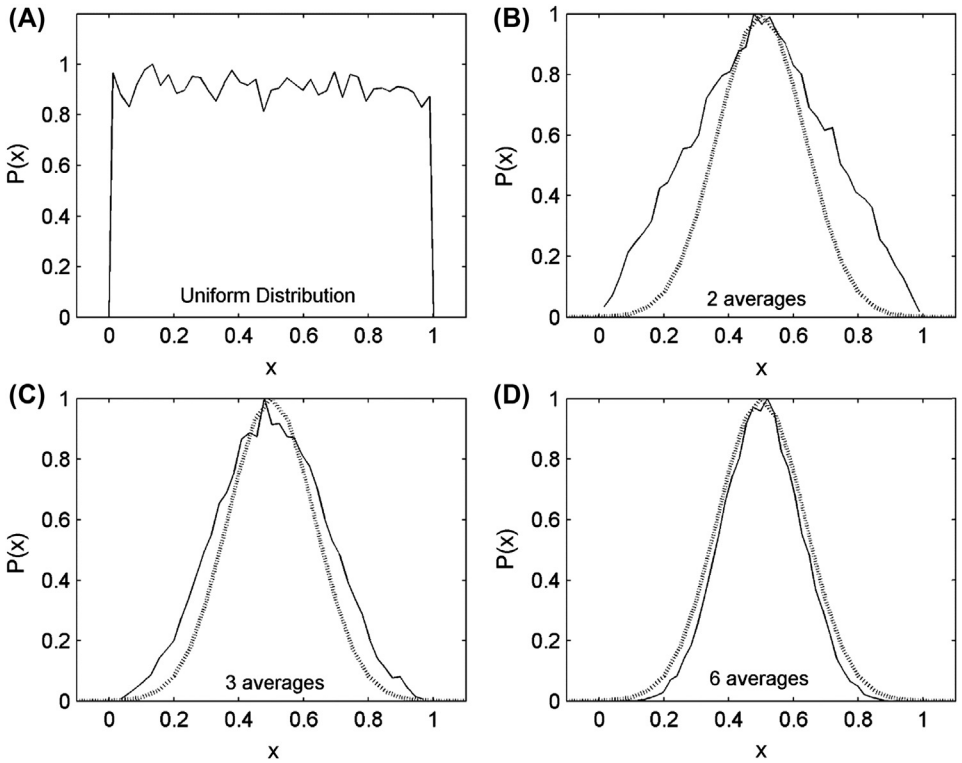


FIGURE 1.10 (A) The distribution of 20,000 uniformly distributed random numbers. The distribution function is approximately flat between 0 and 1. (B) The distribution of the same size data set where each number is the average of just two of the uniformly distributed numbers. The dashed line is the Gaussian distribution for comparison. (C) Each number in the data set is the average of three uniformly distributed numbers. (D) When only six uniformly distributed numbers are averaged to produce a number in the data set, the distribution becomes very close to Gaussian. (Note: this averaging trick is used by some computer algorithms to produce a Gaussian distribution from more easily generated uniformly distributed pseudo-random numbers.)

[11]Both terms are commonly used. We use the term "Gaussian" to avoid the value judgment implied by the word "normal"!

0 and +1 is shown (generated using MATLAB's `rand` function). A uniformly distributed variable has an equal probability of any value between 0 and 1, so its distribution is approximately flat between these limits. But now if we produce a new data set that is the average of just two uniformly distributed random numbers, we get a distribution that begins to look Gaussian, Figure 1.10B (the dashed line is a Gaussian distribution for comparison). When the data set is produced from the average of only six uniformly distributed random numbers, the resulting distribution is very nearly Gaussian, Figure 1.10D.

The probability distribution of a Gaussianly distributed variable, x, with zero mean is specified in the well-known equation:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/2\sigma^2} \tag{1.5}$$

This is the distribution produced by the MATLAB function `randn`. (Note: The MATLAB routine `rand` produces uniformly distributed data with a mean of 0.5, whereas `randn` produces Gaussianly distributed data with a mean of 0.0. The Gaussian distributions shown in Figure 1.10 do not have zero mean because they were generated by averaging uniform data with a mean of 0.5.)

EXAMPLE 1.4

Use a large data set generated by `randn` to determine if these data have a Gaussian probability distribution. Also estimate the probability distribution of the data produced by `rand`.

Solution: A straightforward way to estimate the distribution function of an existing data set is to construct a "histogram" of the data set. A histogram is done by dividing a data set into a number of ranges, or "bins." For example, if the values of a data set ranged between 0 and 99, you could divide this range into five bins, each having a range of values: 0–19, 20–39, etc., Figure 1.11. Next, you

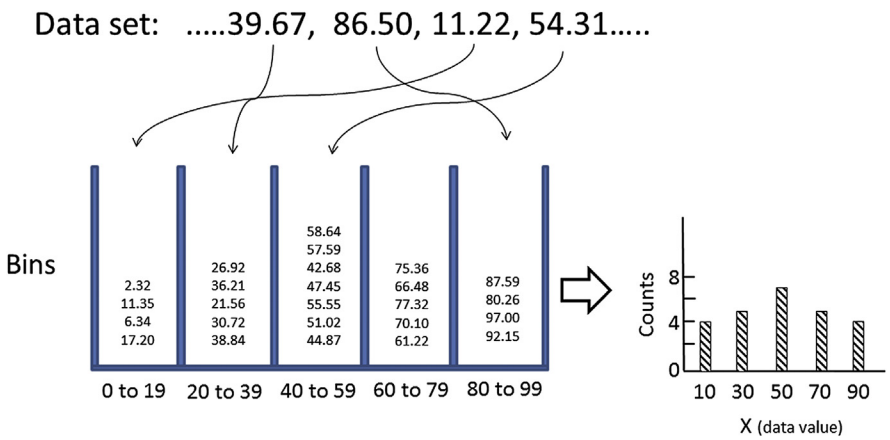


FIGURE 1.11 Construction of a histogram. Data are binned into five set ranges (left). After all the data in the set are binned, the number of data points in each bin is counted and plotted against the range value(s) (right).

count the number of specific values in your data set that fall within each of five ranges, Figure 1.11 (left). Then plot the number of counts in each range against the range, or the middle value of the range, Figure 1.11 (right), to produce the histogram. In other words, the histogram is a plot of the number of data points that fall within a given range (i.e., fall within a given bin) against the range, or mean value, of that range. You usually decide how many bins to use. More bins means smaller ranges for each bin, which produces higher resolution, but the bins will also have fewer counts, and if the data set is small, the histogram will be noisier. Usually the mean value of a range is used and is plotted on the horizontal axis with counts on the vertical axis.[12] Bar-type plots are commonly used for plotting histograms.

Construction of histograms is such an important operation that MATLAB has a routine to calculate and plot them. As is typical of MATLAB routines, `hist` has a number of options. The most useful calling structure for this example is:

```
[ht,xout] = hist(x,nu_bins);    % Calculate the histogram of data in x
```

where `x` is the data set and `nu_bins` is the number of bins desired. The outputs are the histogram, `ht`, and an array, `xout`, useful in plotting the histogram. This array gives the range of each bin as the mean of its range. So the statement `plot(nu_bins, ht)` plots the histogram and correctly scales the horizontal axis plotting (although as mentioned, it is more common to use a bar type plot as in the example below). Arrays that make plotting and scaling the axes easier (usually the horizontal axis) are often found as outputs in MATLAB routines and you benefit from using them.

This example first constructs a large (20,000-point) data set of Gaussianly distributed random numbers using `randn`, then uses `hist` to calculate the histogram and plot the results. This procedure is then repeated (but not shown) using `rand` to produce a uniformly distributed data set.

```
% Example 1.4 Evaluation of the distribution of data produced by MATLAB's
% rand and randn functions.
%
N = 20000;                      % Number of data points
nu_bins = 40;                   % Number of bins
y = randn(1,N);                 % Generate random Gaussian noise
[ht,xout] = hist(y,nu_bins);    % Calculate histogram
ht = ht/max(ht);                % Normalize histogram to 1.0
bar(xout, ht, 'c');             % Plot as bar graph (use color)
  ....... Label axes and title .......
  .......Repeat for rand ........
```

Results: The bar graphs produced by this example are shown in Figure 1.12. Note the approximately Gaussian distribution for the `randn` function, Figure 1.12A, and close to flat for the `rand` function, Figure 1.12B. One of the problems explores the distributions obtained using different data lengths.

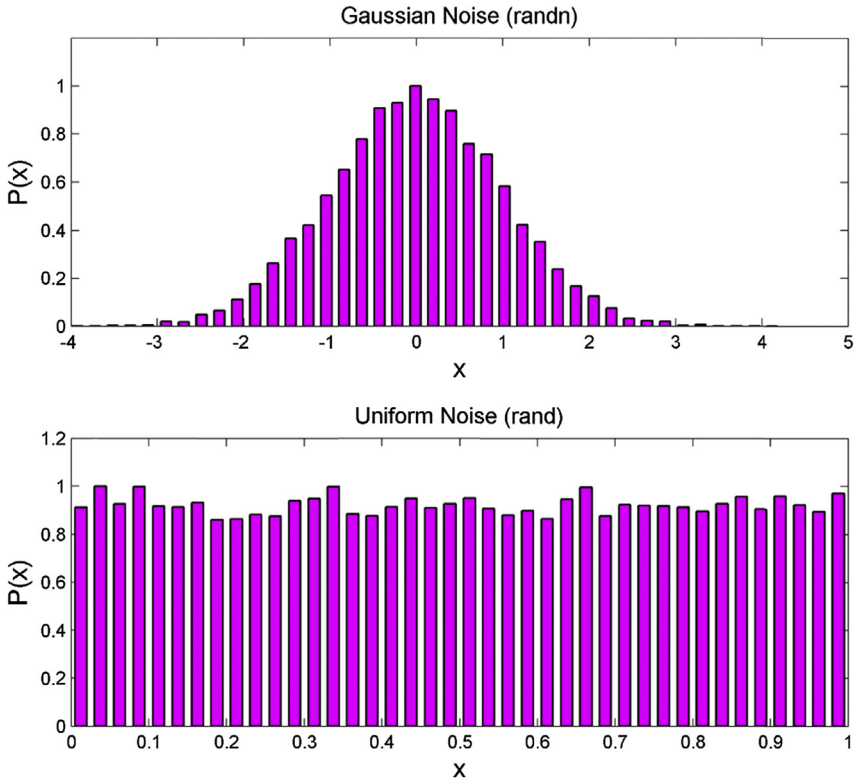


FIGURE 1.12 (A) The distribution of a 20,000-point data set produced by the MATLAB random number routine `randn`. As is seen here, the distribution is quite close to the theoretical Gaussian distribution. (B) The distribution of a 20,000-point data set produced by `rand`.

[12]The vertical axis of a histogram may also be labeled "Number" (or *N*), "Occurrences," "Frequency of occurrence," or shortened to "Frequency." The latter term should not be confused with "frequency" when describing cycles per second as is the most common use of the word.

1.3.2.1 Electronic Noise

Any biomedical engineer involved with devices will eventually encounter the additive Gaussian noise that is inherent in electronics. This noise falls into two, source-dependent, classes: "thermal" or "Johnson" noise, and "shot" noise. The former is produced primarily in resistors or resistance materials, whereas the latter is related to voltage barriers associated with semiconductors.

The good thing about electronic noise is we know a lot about it: not only where it comes from, but also how much noise to expect from a given component. We also know its frequency characteristics: its value at different frequencies. Both Johnson and shot noise contain energy over a broad range of frequencies, often extending from DC (i.e., 0 Hz) to 10^{12}–10^{13} Hz. White light contains energy at all frequencies, or at least at all frequencies

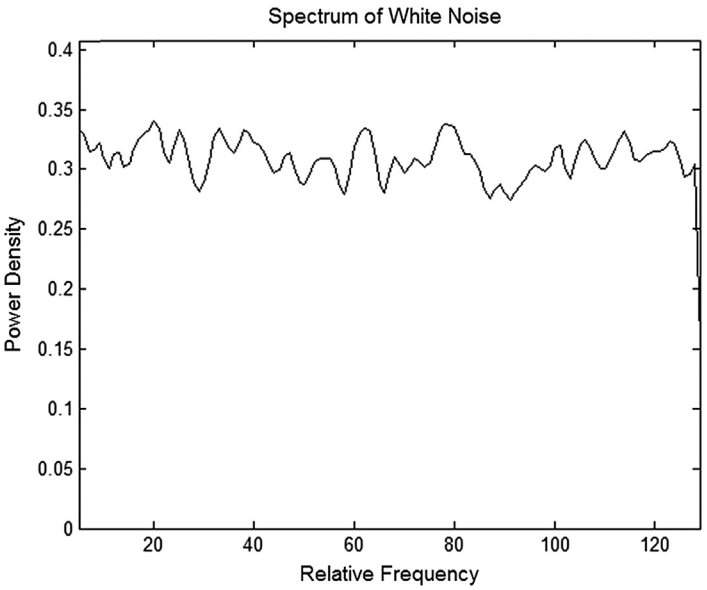


FIGURE 1.13 A plot of the energy in white noise as a function of frequency. The noise has a fairly flat spectral characteristic: it has nearly the same energy over the frequencies plotted. This equal-energy characteristic gives rise to the term "white noise." Techniques for producing a general signal's spectral plot are covered in Chapter 3.

we can see, so such broad-spectrum noise is also referred to as "white noise" since it contains energy at all frequencies, or all the frequencies we ever worry about. Figure 1.13 shows a plot of the energy in a simulated white noise waveform (actually, an array of random numbers) plotted against frequency. Again, this is a plot of energy against frequency, so it is called a spectrum.

1.3.2.1.1 JOHNSON NOISE

Johnson or thermal noise is produced by resistance sources and the amount of noise generated is related to the resistance and to the temperature:

$$V_J = \sqrt{4kT\,R\,BW} \text{ volts} \tag{1.6}$$

where R is the resistance in ohms, T is the temperature in degrees Kelvin, and k is Boltzman's constant ($k = 1.38 \times 10^{-23}$ J/K). A temperature of 310 K is often used as room temperature, in which case $4kT = 1.7 \times 10^{-20}$ J. Here BW is the range of frequencies that is included in the signal. This range of frequencies is termed "bandwidth" and is better defined in Chapter 4. This frequency range is usually determined by the characteristics in the measurement system, often the filters used in the system. Since noise is spread over all frequencies, the greater the signal bandwidth, the greater the noise in any given situation.

If noise current is of interest, the equation for Johnson noise current can be obtained from Equation 1.6 in conjunction with Ohm's law:

$$I_J = \sqrt{4kT\,BW/R} \text{ amps} \tag{1.7}$$

In practice, there will be limits imposed on the frequencies present within any waveform (including noise waveforms), and these limits are used to determine bandwidth. In the problem set at the end of the chapter, bandwidth, BW, is explicitly stated. It is common to specify BW as a frequency range in units of hertz, which are actually units of inverse seconds: i.e., $\text{Hz} = 1/\text{s}$.[13]

Since bandwidth is not always known in advance, it is common to describe a relative noise; specifically, the noise that would occur if the bandwidth were 1.0 Hz. Such relative noise specification can be identified by the unusual units required: volts$/\sqrt{\text{Hz}}$ or amps$/\sqrt{\text{Hz}}$.

1.3.2.1.2 SHOT NOISE

Shot noise is defined as current noise and is proportional to the baseline current through a semiconductor junction:

$$I_s = \sqrt{2qI_d\,BW}\ \text{amps} \tag{1.8}$$

where q is the charge on an electron (1.662×10^{-19} coul.), and I_d is the baseline semiconductor current. In photodetectors, the baseline current that generates shot noise is termed the "dark current," and this was the motivation for the letter "d" in the current symbol, I_d, in Equation 1.8. As with Johnson noise, the noise is spread across all frequencies, so the bandwidth, BW, must be specified to obtain a specific value or else a relative noise can be specified in amps$/\sqrt{\text{Hz}}$.

When multiple noise sources are present, as is often the case, their voltage or current contributions to the total noise add as the square root of the sum of the squares, assuming that the individual noise sources are independent. For voltages:

$$V_T = \sqrt{V_1^2 + V_2^2 + V_3^2 + \ldots V_N^2} \tag{1.9}$$

A similar equation applies to current noise.

EXAMPLE 1.5

A 20 mA current flows through both a diode (i.e., a semiconductor) and a 200-Ω resistor, Figure 1.14. What is the net current noise, i_n? Assume a bandwidth of 1 MHz (i.e., 1×10^6 Hz).

Solution. Find the noise contributed by the diode using Equation 1.7 and the noise contributed by the resistor using Equation 1.6 and then combine them using Equation 1.9.

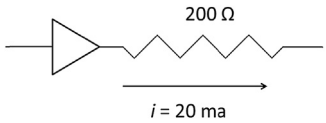


FIGURE 1.14 Simple electric circuit consisting of a resistor and diode used in Example 1.5.

[13]In fact, frequencies and bandwidths were given in units of "cycles per second" until the 1970s when it was decided that an inverse definition, a "per" definition, was not appropriate for such an important unit.

$$i_{nd} = \sqrt{2qI_dBW} = \sqrt{2(1.66 \times 10^{-19})(20 \times 10^{-3})10^6} = 8.15 \times 10^{-8} \text{ amps}$$
$$i_{nR} = \sqrt{4kTBW/R} = \sqrt{1.7 \times 10^{-20}\left(10^6/200\right)} = 9.22 \times 10^{-9} \text{ amps}$$
$$i_{nT} = \sqrt{i_{nd}^2 + i_{nR}^2} = \sqrt{6.64 \times 10^{-15} + 8.46 \times 10^{-17}} = 8.20 \times 10^{-8} \text{ amps}$$

Note that most of the current noise is coming from the diode, so the addition of the resistor's current noise does not contribute much to the noise current. Also the arithmetic in this example could be simplified by calculating the square of the noise current (i.e., not taking the square roots) and using those values to get the total noise.

1.4 BIOLOGICAL SYSTEMS

"Consider the source" is a good rule for signal processors, since signal features often reflect their source. Biological systems, biosystems, produce or modify biosignals, and the properties of these systems transfer to their signals. Many important system (and signal) behaviors are dichotomous; that is, they are mutually exclusive: linear versus nonlinear, time invariant versus nonstationary, and deterministic or stochastic. But a system can exhibit both behaviors in different mixes. A major conundrum in biosignal analysis is that, although most biological systems are to some degree nonlinear, nonstationary, and stochastic, our most powerful analytical tools apply only to systems that are the opposite: deterministic, linear, and time invariant. For this reason, we often analyze biosignals, and the underlying biosystems, as if they were deterministic, linear, time-invariant systems. Such systems are given the shorthand term "LTI" systems. There is some exciting ongoing work in the analysis of nonlinear signals; at least we have some methods to determine if a signal contains nonlinearity, but unfortunately, these nonlinear tests are not always definitive and are often fooled by noise.

1.4.1 Deterministic Versus Stochastic Signals and Systems

Deterministic systems and signals can be described by a system of differential equations, at least in principle. These equations may be very complicated and they may not always be known. A deterministic signal has no randomness in its behavior, and it is possible to predict the future behavior of the signal from past values. The noise-free sine wave given by Equation 1.14 is an example of a deterministic signal. We can predict with perfect accuracy the value of a sine wave at any time knowing only the frequency, amplitude, and phase of the sine wave.

Stochastic behavior (a.k.a., noise) is inherently random: it cannot be described by a set of equations. In the next chapter we describe some general properties of a stochastic signal such as the mean, variance, and distribution function. Again, whether or not a system or signal is to be considered deterministic or stochastic depends on how much noise is in the signal. In practice, even signals produced by deterministic systems are corrupted by some noise so their future values may not be exactly predictable. Many signal processing techniques have been specifically designed to mitigate the influence of stochastic behavior.

1.4.1.1 Chaotic Signals and Systems

The signal from a chaotic system appears to combine properties from both random and deterministic signals but in fact chaotic processes are deterministic, not stochastic. There are three criteria used to define a chaotic system: (1) steady-state values are not fixed and do not repeat: they are not periodic; (2) system behavior is very sensitive to initial conditions; and (3) the system behavior is not simply a response to a random stimulus. There are methods to test if an apparently random signal is actually chaotic, many based on the first two criteria. Unfortunately, these tests are often less than definitive, especially if noise is intermixed with the chaotic signal.

A popular example of a chaotic system comes from the study of the variability in species populations. An equation that predicts animal populations fairly well is a simple quadratic equation called the *logistic difference* equation or *logistic map*, since it maps population values at generation n to values at the next generation, $n + 1$. In other words, the equation relates the population $x[n + 1]$ to the previous generation's population $x[n]$. The logistic equation is given as:

$$x[n] = rx[n](1 - x[n]) \tag{1.10}$$

where $x[n]$ is the population at generation n (normalized to 1.0), $x[n + 1]$ is the population of the next generation, and r is the so-called driving parameter, which is related to growth. For small values of r, less than 3.0, Equation 1.14 produces smooth monotonic changes from one generation to the next, which eventually converge to a stable value. Above 3.0, the population alternates between two levels and, as r increases further, it alternates between 4, then 8, then 16 different population values: a behavior known as period doubling. Above a value of 3.57, the generation-to-generation fluctuation becomes apparently random and is shown in Figure 1.15. Although this fluctuation appears to be random, it is not since it is still completely determined by Equation 1.10. This behavior is explored in a problem at the end of the chapter.

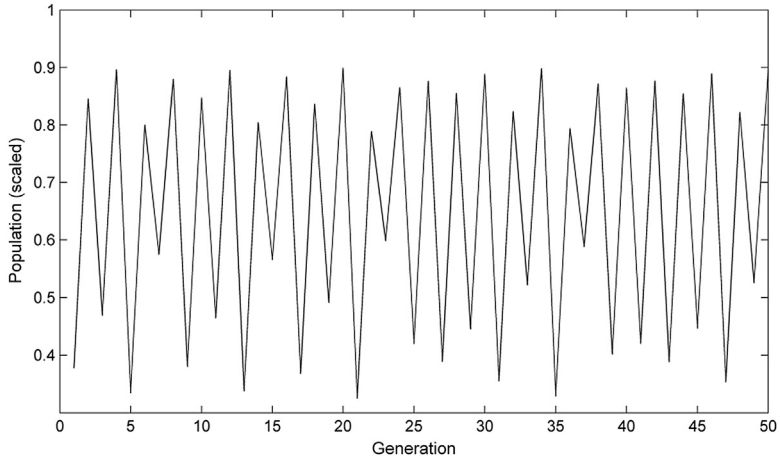


FIGURE 1.15 Value of the logistic equation, Equation 1.10 as a function of n where $r = 3.6$. The chaotic signal appears to be random, but is deterministic as it is generated by an equation, a rather simple one.

Chaos analysis is being applied in a number of different areas. In addition to ecology and epidemiology, chaos analysis has been applied to epileptic seizures, cardiac rhythms, patterns of nervous activity, fluctuation in leukocytes of patients with leukemia, and basic neural function. For example, it has been suggested that the basic operations of the nervous system employ chaos. In the cardiovascular system, chaotic heart rhythms are considered positive and some disease processes have been shown to reduce this chaos. Finally, studies are finding that some of the noise associated with biological systems may actually be chaotic behavior.

1.4.1.2 Fractal Signals and Images

Fractal signals are deterministic signals that look the same, or at least similar, at every magnification level, or "scale." Such signals are said to be "self-similar," "scale-invariant," or "scale free" because the pattern is much the same no matter what scale is used to view the pattern. One classic example is the outline of a coast. From high above the earth, coastlines have random-looking patterns, but as you get closer, and closer, they still have a similar random-looking pattern. Although fractal signals are only occasionally found in biology, there are many examples of fractal images. Figure 1.16 shows a fractal branching pattern. For each descending generation, a line branches into two shorter, diverging lines, and this progression continues through 16 generations. Two levels of increasing magnification show similar patterns, the hallmark of a fractal image. The bronchi of the lungs and coronary vascularization exhibit similar fractal patterns. Fractals, like chaos, demonstrate how complex structures can emerge from a simple strategy.

In fractal signals, small segments are similar to larger segments scaled-up in amplitude. The relationship between the change in timescale and change in amplitude scale often follows a power relationship. Finding the value for this scaling can be useful in research, and changes

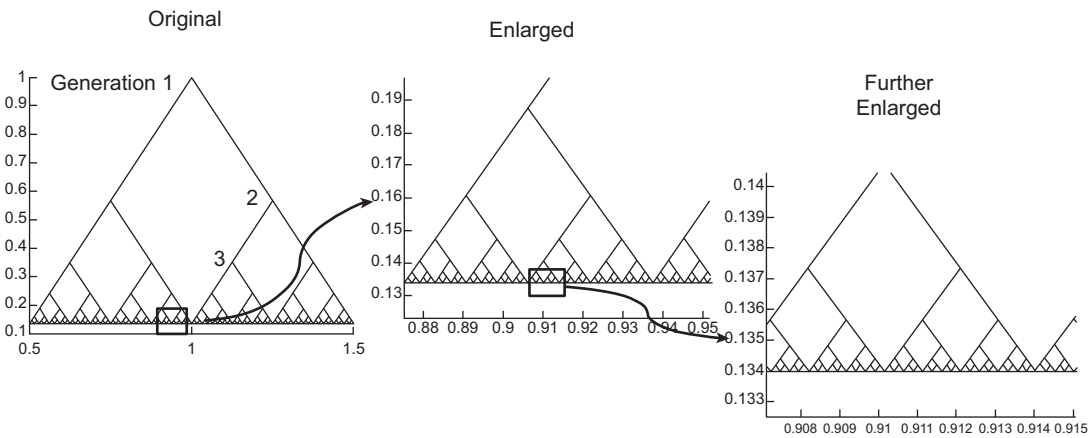


FIGURE 1.16 A fractal pattern generated by a simple strategy where each line branches into two divergent lines as it descends. The original pattern and two magnifications are shown. The similarity at each scale is emblematic of fractals.

in this relationship may indicate the onset or prognosis of disease. This promise motivates some major ongoing research to increase our understanding of these intriguing signal properties. Methods that search for fractal properties in signals are described in Chapter 10.

1.4.2 Deterministic Signal Properties: Periodic, Aperiodic, and Transient

Many of the signals discussed in this text can be divided into three general classes: "periodic," "aperiodic," and step-like or "transient."[14] Periodic signals repeat exactly after a fixed period of time known as the "period," T. The formal mathematical definition of a periodic signal is:

$$x(t) = x(t+T) \tag{1.11}$$

Frequency can be expressed in either radians or hertz (the units formerly known as "cycles per second") and the two are related by 2π:

$$\omega_p = 2\pi f_p \tag{1.12}$$

Both forms of frequency are used in the text and you should be comfortable with both, although frequency in hertz is most common in engineering. Frequency is the inverse of the period, T:

$$f_p = \frac{1}{T} \tag{1.13}$$

The sine wave is an example of a periodic signal, one that repeats in an identical manner following some time period T, or frequency ω_p, or frequency f_p:

$$x(t) = A\sin(\omega_p t) = A\sin(2\pi f_p t) = A\sin\left(\frac{2\pi t}{T}\right) \tag{1.14}$$

The sine wave is rather boring: if you have seen one cycle you have seen them all. Worse, the boredom goes on forever since the sine wave of Equation 1.14 is infinite in duration. Since the signal is completely defined by A (amplitude) and f_p (frequency), if neither changes over time, this signal does not convey any information. These limitations notwithstanding, sine waves (and cosine waves) are the foundation of many signal analysis techniques. Some reasons why sine waves are so important in signal processing are given in Chapter 3, but part of their importance stems from their simplicity.

Other common periodic signals include the square wave, pulse train, and sawtooth as shown in Figure 1.17. Since all periods of a periodic signal are identical, only one period is

[14]All time-varying signals could be called "transient" in the sense that they are always changing. However, in describing signals, the term transient is often reserved for signals that change from one level to another and never return to their initial value.

required to completely define a periodic signal, and any operations on such a signal need only be performed over one cycle. For example, to find the average value of a periodic signal, it is only necessary to determine the average over one period.

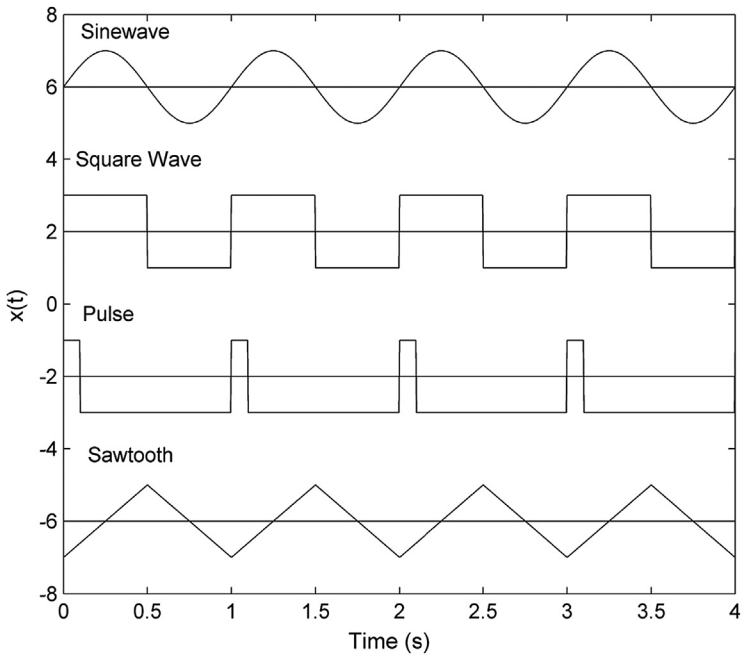


FIGURE 1.17 Four different periodic signals having the same, 1 s, period: sine wave, square wave, pulse train, and sawtooth.

EXAMPLE 1.6

Use MATLAB to construct a sinusoidal signal having an amplitude of 5.5 and a frequency of 10 Hz. Construct the sine wave in a 1000-point array. Plot two cycles of this signal as a function of time. Assume a sampling frequency of $f_s = 1000$ Hz.

Solution: This example is a simplification of Example 1.4, but it uses a formalism better adapted to signals as opposed to images. Throughout this text, we encounter the need to construct waveforms to illustrate various operations and principles; waveforms having specific properties such as shape or frequency. Given this ongoing need, let us establish a general approach that we can fall back on to make the coding of constructed signals easier.

Since we are working in the digital domain, we use a MATLAB array to hold the sine wave. Usually, the first step is to determine an appropriate length for the array: `N`, the number of data points in the array. In this problem, the array length is assigned as 1000 points, a value we often use as a sort of default. Before constructing the sine wave, we often construct a time vector[15] of the same length. To construct this time vector, we use a variation of Equation 1.3:

$$t = n/f_s$$

where n is a MATLAB variable going from 1 to N (or 0 to N − 1). This gives the MATLAB code lines:

```
fs = 1000;          % Define the sample frequency
N = 1000;           % Define the total number of points (given)
t = (0-N-1)/fs;     % Construct the time vector[16]
```

Next we construct the sine wave in a MATLAB vector, let us call it x, using Equation 1.8 (the middle version on the right hand side):

```
A = 5.5;              % Sine wave amplitude (given)
fp = 10;              % Sine wave frequency (given)
x = A*sin(2*pi*f*t);  % Construct the sine wave
```

Since we are requested to plot just two cycles of the sine wave, we need to limit the number of points plotted. Note that from rearranging Equation 1.7 one cycle is: $T = 1/f_p = 1/10 = 0.1$ s; so two cycles is $2T = 0.2$ s. The easiest way to do this conceptually is to plot the entire vector then as MATLAB's xlim to limit the time axis to the first 2 s:

```
plot(t,x);        % Plot sine wave as a function of time
xlim([0 0.2]);    % Limit the plot to the first 0.2 sec
```

We also add labels to the two axes. Given how easy this is to do in MATLAB, it should be a part of any graphical output, as it adds significant clarity to graphs. Putting this all together gives the solution code:

```
% Example 1.6  Plot two cycles of a 10 Hz sine wave
%
fs = 1000;                % Define the sample frequency
N = 1000;                 % Define the total number of points (given)
t = (0-N-1)/fs;           % Construct the time vector
A = 5.5;                  % Sine wave amplitude (given)
fp = 10;                  % Sine wave frequency (given)
x = A*sin(2*pi*fp*t);     % Define the sine wave
plot(t,x,'k');            % Plot sine wave
xlim([0 0.2]);            % Limit plot to first 0.2 sec
xlabel('Time (sec)');     % Label time axis
ylabel('Amplitude');      % Label y axis
```

Results/comments: The output of this program is shown in Figure 1.18 to be a 10 Hz sine wave having an amplitude of 5.5. Since we are not told what the sine wave represents (e.g., voltage, current, other?), the y-axis units are arbitrary. Note that once we take the trouble of setting up the

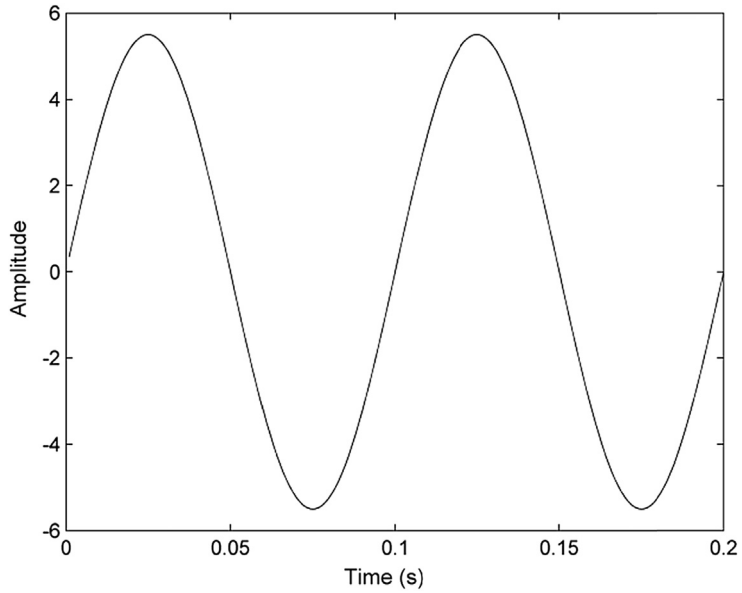


FIGURE 1.18 The 10 Hz sine wave produced by the MATLAB code in Example 1.2.

appropriate time vector, `t`, the code that produces the sine wave is the same as the equation for a sine wave given by Equation 1.14. We frequently find this to be the case when implementing waveform equations in MATLAB.

[15]As mentioned, we often refer to arrays as "vectors." The motivation for using this term is given in the next chapter, but for now assume the two words are synonymous.
[16]In this book, we use the typeface shown here for MATLAB variables and code.

Aperiodic signals exist over some finite time frame, but are zero before and after that time frame. Figure 1.19 shows a few examples of aperiodic signals. The assumption is that these signals are zero outside the period shown. A mathematically convenient way to view aperiodic signals is as periodic signals where the period goes to infinity. Operations on aperiodic signals need only be applied to their finite, nonzero segments.

The third class of signals, transient signals, includes signals that change, often dramatically, but never return to an initial level. A few examples are given in Figure 1.20, including the commonly used step signal. These signals do not end at the plot, but continue to infinity. These signals are the hardest to treat mathematically. They stretch to infinity, but are not periodic, so any mathematical operation must be carried out from zero to infinity. For example, it is impossible to calculate an average value for any of the signals in Figure 1.20 since these signals are nonzero to infinity, so their average values must also infinite. Moreover, it is not possible to represent such signals in a computer since they would require an infinite memory. However, methods for dealing with these signals exist and are discussed in Chapter 7.

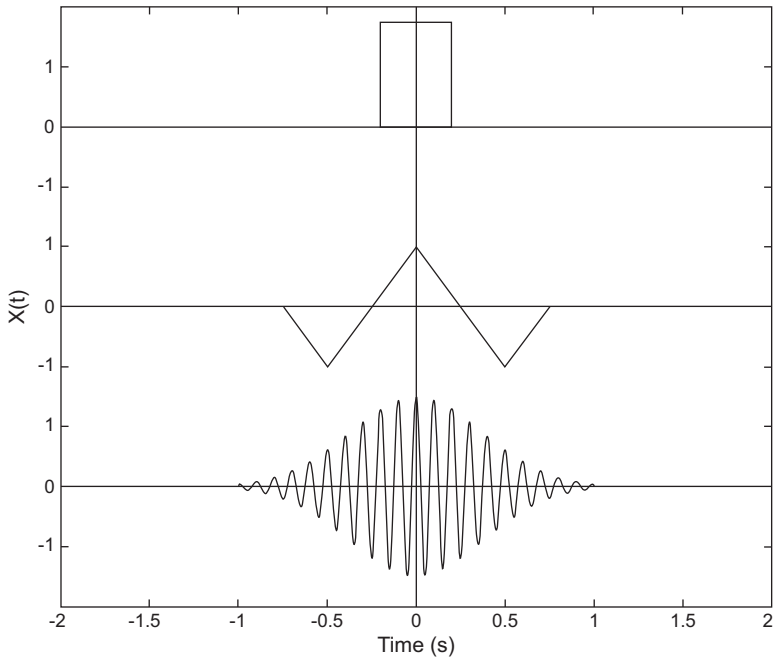


FIGURE 1.19 Three examples of aperiodic signals. It is common to show these signals centered on $t = 0$, especially when they are symmetric.

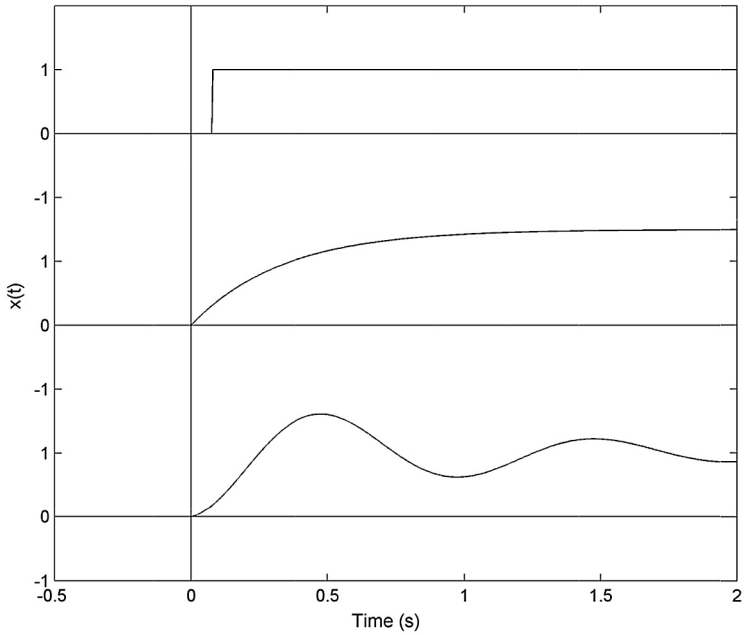


FIGURE 1.20 Three examples of transient or step-like signals. These signals do not end at the end of the plot but continue to infinity. The step signal is common in signal processing.

1.4.2.1 *Time-Invariant Versus Nonstationary Signals and Systems*

The name defines it, time invariant: signals and systems that do no change over time. Time-invariant systems have fixed equations that are not functions of time. Time-invariant systems produce time-invariant signals. Time-invariant signals can change with time (most do), but they are called time invariant because they can be described by equations that do not change over time. In other words, a time-invariant signal can be "time varying" (i.e., a function of time) as long as the equation defining that signal does not change. For example, the sine wave defined by Equation 1.14 is time varying, but since Equation 1.14 is fixed, it is a time-invariant signal.

The mathematical definition of a time-invariant function, f, is given as:

$$\begin{aligned} &y = f(x) \quad \text{where } f \text{ is a linear function, then :} \\ &y(t-T) = f(x(t-T)) \end{aligned} \tag{1.15}$$

Nonstationarity is associated with systems or signals that change their basic statistical properties with time; if these signals have defining equations, those change over time. If signals or systems are nonstationary, then their statistical properties may be the same or they may change over time. Statistical properties of a signal that vary over time could include the mean or average value, the range of variability (variance), or the correlations between the signal and different delayed versions of the signal.

It would be nice if biological systems did not change over time, yet many do. As the systems change so do the signals they produce. The EEG signal, the electrical activity of the brain, is a classic example of a nonstationary signal. Its statistical properties are dependent on internal stated of the brain. Figure 1.21 shows a segment of an EEG signal where

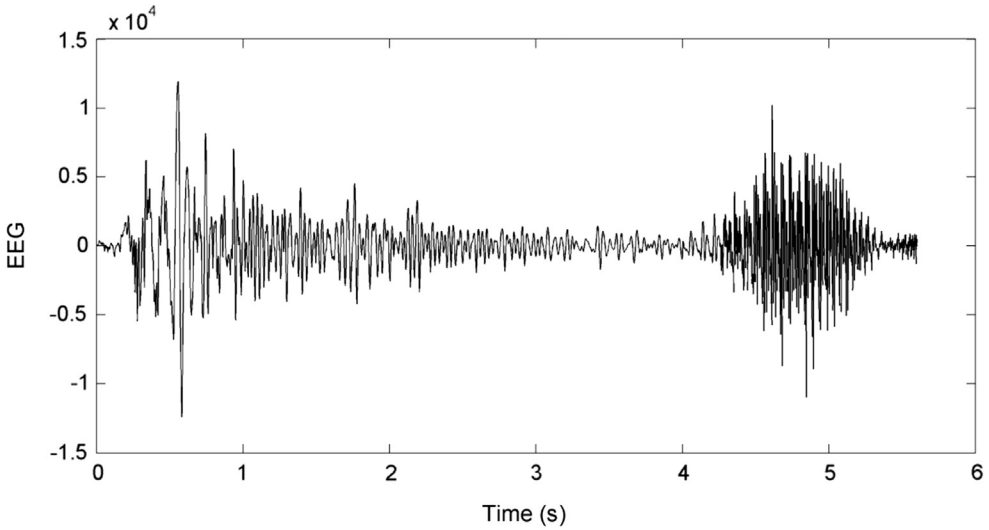


FIGURE 1.21 A segment of an EEG signal, the electrical activity of the brain. The basic characteristics appear to change over the course of the signal, an indication of nonstationarity.

nonstationary behavior can be directly observed: the basic nature of the signal appears to be different over different segments of the signal. In Chapter 10, techniques for testing signals for nonstationarity are described and applied to signals such as the EEG signal in Figure 1.21.

Nonstationarities present a moving target to any analysis approach. If dealt with at all, it is usually on an ad hoc basis, depending strongly on the type of nonstationarity. Most methods to deal with nonstationary signals are quite advanced and depend on the specific nonstationarity, but some approaches are described in Chapter 10. For example, if the mean or average value of the signal is varying, it may be possible to compensate for, or eliminate, this variation. Taking the derivative, which removes the mean along with any variation in mean, is one approach. Another is to estimate the variation in signal mean and subtract it out. Such an approach is called "detrending" and an example is presented in Chapter 10. Another popular approach that can be applied to a wide range of nonstationary behavior is simply to limit the signal analyses to time segments of the signal where it can be considered stationary.

1.4.3 Causal Versus Noncausal Signals and Systems

Causality is a fundamental Newtonian concept that responses are caused by stimuli. In systems, output signals are caused by input signals. In other words, responses cannot come before the thing they are responding to. Stated mathematically, the output of a system at $t = t_0$ can only be due to an input signal, $x(t)$, at $t \leq t_0$. This seems rather self-evident and unimportant, as all physical systems must be causal. But if the data are stored in a computer, it is possible to perform operations that include data points from both the past and future. Consider an operation on a MATLAB array; assume the following statement is in a `for` loop: `y(k) = x(k-4) - x(k+4);` A new array, `y`, is being generated from points before and after the equivalent point (`k`) in the new array. If the data were a time series, then this operation would be equivalent to the time series equation:

$$y(t) = x(t-4) - \mathrm{x}(t+4) \tag{1.16}$$

The output signal is based, in part on a signal from the future, $x(t+4)$. So this is a "noncausal" system. Again, noncausal systems can only be implemented using signals that have already occurred, for example, those stored in a computer. Physical systems must be causal and computer-based systems that produce real-time outputs must also be causal.

Causal signals are presumed to begin at $t = 0$, so the mathematical definition for a casual signal is:

$$x(t) = 0 \quad \text{for } t < 0 \tag{1.17}$$

Noncausal signals, those that exist for $t < 0$, are frequently shown here. This is just a graphical convention; all real-world signals are causal.

1.4.4 Linear Versus Nonlinear Signals and Systems

The concept of linearity has a rigorous definition, but the basic idea is one of "proportionality of response." If you double the stimulus into a linear system, you get twice the response. One way of stating this proportionality property mathematically is:

$$\begin{aligned} y &= f(x) \quad \text{where } f \text{ is a linear function, then :} \\ ky &= f(kx) \quad \text{where } k \text{ is a constant} \end{aligned} \tag{1.18}$$

In other words, the output scales in proportion to the input.

Also, if f is a linear function:

$$f(x_1(t)) + f(x_2(t)) = f(x_1(t) + x_2(t)) \tag{1.19}$$

In addition, if: $y = f(x)$ and $z = \frac{df(x)}{dx}$, then $\frac{df(kx)}{dx} = k\left(\frac{df(x)}{dx}\right) = kz$.

Similarly, if: $y = f(x)$ and $z = \int f(x)dx$, then $\int f(kx)dx = k\int f(x)dx = kz$.

Derivation and integration are linear operations. When they are applied to linear functions, these operations preserve linearity. Recall that systems that are both linear and time invariant are referred to as linear time invariant, or LTI, systems.

In a nonlinear system, the concept of proportionality (Equation 1.18) does not hold, nor does the summation of two signals expressed on Equation 1.19. Consider the simple example of a nonlinear system described by:

$$y(t) = x^2(t) \tag{1.20}$$

If the input to this system is value A, then the output is A^2, whereas the output for an input of value B is B^2. If this system were linear, the output to the combined input of $A + B$ would be $A^2 + B^2$, but in this system it is: $(A + B)^2 = A^2 + 2AB + B^2$.

If you have a biological system that you can isolate, drive with a controlled stimulus, and measure the response, then you might be able to test for nonlinearity using Equation 1.18. For example, you could input signals having different amplitudes (k in Equation 1.18) and determine if the output amplitude follows proportionally. This is the approach used in the next example.

EXAMPLE 1.7

The MATLAB routine `unknown1.m` takes a single input argument and produces a single output; i.e., `out = unknown1(in)`. Consider this routine as representing an input−output system and design a test to see if it is linear.

Solution: It is not possible to test if a system is linear over all possible inputs. Indeed most real systems are nonlinear if the amplitude range is large enough. Moreover, some systems may be linear for some frequencies, but nonlinear for others. Here we test to see if the system is linear for a 10 Hz sine wave over a range of amplitudes of 1−100 (arbitrary units).

First, generate a 10 Hz sine wave using the same approach as in Example 1.6. To reuse some of the code in that example, we will use an array length of 1000 points and a sampling frequency of $f_s = 1000$ Hz. Again these numbers were chosen arbitrarily, but they are appropriate for representing a 10 Hz signal. We use this sine wave as the input to `unknown1.m` and then plot the peak values of the output as a function of the input. If the sine wave is linear, the output scales in proportion with the input and the plot should be a straight line as per Equation 1.18.

```
% Example 1.7 to test nonlinearity of 'system' unknown.m
% System will be evaluated using a 10 Hz sine wave over an
%  amplitude range of 1-100.
% Use a 1000 point array to store the signal and assume a sampling
%  frequency of 1000 Hz
%
fs = 1000;                    % Sampling frequency in Hz
N = 1000;                     % Array for sine wave
t = (1:N)/fs;                 % Generate a time array using Equation 1.3
fp = 10;                      % Sine wave frequency in Hz
sine wave = sin(2*pi*fp*t);   % Generate 10 Hz sine wave having
                              % an amplitude of 1 using Equation 1.8
for k = 1:100                 % Test 100 different amplitudes
  in = sine wave * k;         % Adjust sine wave amplitude
  out = unknown1(in);         % Input sine wave to unknown system
  input_ampl(k) = max(in);    % Save input amplitude
  out_ampl(k) = max(out);     % Get output/input peak values
end
plot(input_ampl,out_ampl);    % Plot results
       ..........labels.............
```

Discussion and Results: After defining the sample frequency, `fs`, data array length, `N`, and sine wave frequency, `fp`, we generate a time array, `t`, using Equation 1.3. This is our approach in all MATLAB problems that involve a time function. We then use Equation 1.8 to produce the sine wave array. Using a `for` loop, we multiply the sine wave by numbers 1 through 100 and take the result as input to the routine `unknown1.m`. For each input amplitude, we save the maximum input and output signal amplitude.

The input and output signals are shown in Figure 1.22A to be sinusoidal, but the phase of the input (solid line) and the output (dotted line) are different. A plot of the output maximum amplitude as a function of the input maximum value, Figure 1.22B, is a straight line. This indicates that the output scales in proportion with the input as described by Equation 1.18. So although the unknown system is still unknown, we can say that it is linear, at least over the range of inputs tested.

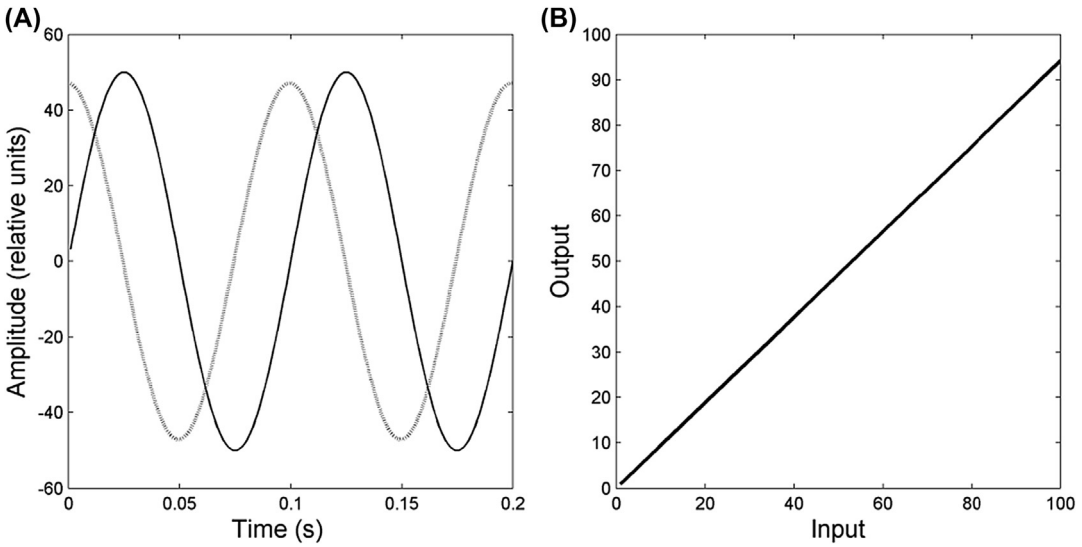


FIGURE 1.22 (A) A time plot showing the input signal to an unknown system (*solid line*) and the resulting output signal (*dashed line*). The input and output signals are 1 s long (1000 samples × 1/1000 samples per sec = 1 s), but only the first 0.2 s are shown for clarity. (B) A plot of the output signal's maximum amplitude versus the input signal's maximum amplitude. The resulting straight line indicates that the unknown process is linear as described by Equation 1.18.

If you have a biosignal, but do not have access to the system that generated that signal, testing for nonlinearity is more difficult. Signals from a nonlinear system may contain irregularities that appear more complicated than signals from linear systems and there are a range of tests that evaluate this complexity. Some of these tests are described in Chapter 10. Unfortunately, not all complex biosignals are from nonlinear systems and the development of definitive tests for nonlinearity is an active field of biomedical signal processing research.

1.4.5 Biosystems Modeling

Models are simplified representations of a system, simplified to better understand the function and/or operation of that system. Some might call a qualitative description of a biological system, such as found in basic physiology textbooks, a model, but as engineers, our models should be quantitative; that is, models that are supported by mathematical equations. It is usually possible to find solutions to these equations (sometimes with the help of a computer) and then we can check model predictions against the behavior of the real system.

Sometimes the model is a collection of equations, most often differential equations, but in a complex system these equations can be difficult to relate to actual biological system and its underlying components. Here we give examples of two modeling approaches that use elements as stand-ins for the differential equations: the "analog model" and the "systems model." These approaches differ in the type of elements used, but both can be represented graphically. In analog models, electrical or mechanical elements are used to represent a

biological feature; in system models, an element represents the stimulus–response behavior of a related biological component.[17] In both models, the elements are alternative representations of differential equations, but they add an identity and a visual component that augments, and we hope clarifies, the relationship between the model and the real-world system.

1.4.5.1 *Analog Models*

In analog models, individual electrical or mechanical elements represent a biological function. For example, an analog muscle model might use springs and friction elements to represent a muscle's elastic and viscous elements. The mechanical elements have the same defining differential equations as the biological mechanical properties they represent. When mechanical elements are used to represent biological mechanics, model forces and velocities represent biological forces and velocities.

A model of skeletal muscle that features three mechanical elements is shown in Figure 1.23. The model is based on pioneering work on isolated muscle preparations by A. V. Hill in the 1930s. The spring labeled PE is the parallel elastic element of muscle and the spring labeled SE is the series elasticity. The element labeled CE is the contractile element, which is responsible for generating the muscle force. In Chapter 12, we find that elastic elements functionally relate force to the integral of velocity. In the model here, both elastic elements are nonlinear, so the relationship between the force they produce and the velocity at which they are stretched is nonlinear. The contractile element is also nonlinear and generates a force that is dependent on both the velocity and initial length of the element. Simulations of this model have shown that it can represent many important contractile features of muscle.

Analog models can also use electrical elements as long as the analogous element has the same defining equations as the real-world element it purports to represent. When electrical elements are used, voltage is a stand-in for force (or pressure) and current represents velocity.[18] In Chapter 7, we note that a resistor imposes a proportional relationship between voltage and current where the constant of proportionality is the resistance in ohms (i.e.,

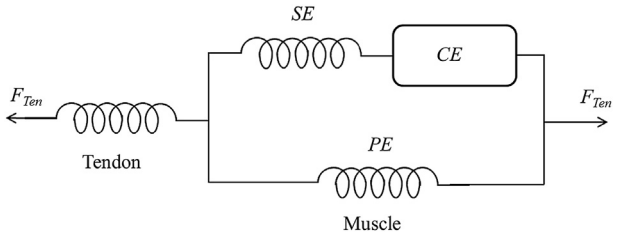


FIGURE 1.23 A three-element model of muscle based on early experimental work of A.V. Hill. Simulations of this model have shown that it can represent the important features of muscle contraction.

[17]"Analog" in this use describes the fact that the model elements are *analogous* to the biological function they represent. An analog model is conceptually continuous (like an analog signal), but so is a systems model. When simulated using a computer, both models are transported to the digital domain, but they are still conceptualized as continuous.

[18]This analogy is enhanced by the similarity between the major variables in the electrical and cardiovascular system. Voltage is a pressure that drives electrons and current is the flow of electrons. This dualism between mechanical and electrical variables is discussed in Chapter 12.

Ohms law: $V = RI$). When used to represent features of the cardiovascular system, a resistor describes relationships between cardiac pressures (or force) and blood flow.

Blood vessels present a resistance to flow and, if they were rigid, the relationship between pressure and flow could be completely described by an analogous resistance. However, blood vessels can expand, particularly larger vessels; they have what is known as "compliance" in addition to resistance. This feature can be represented by a capacitor since the pressure versus flow equation for mechanical compliance is similar to voltage versus current equation for a capacitor.

Combining resistance with compliance leads to the two-element Windkessel model shown in Figure 1.24A. This model was originally proposed by the German physiologist Otto Frank in 1899 to represent the load presented to the heart by the aorta and blood vessels. In modified form, it is still in use today. The purpose of this model is to quantify the mechanical load on the heart and from this representation predict the relationship between cardiac pressure and blood flow. The original model represented this load as two passive elements: a parallel resistor and a capacitor driven by pressure $P(t)$, the pressure generated by the heart, Figure 1.24A. The resistor, R, represents the combined resistance of all the blood vessels (in units of mm Hg s/cm^3), although the primary contribution comes from the smaller peripheral vessels. The capacitor, C, represents the net compliance of the blood vessels (in units of cm^3/mm Hg), which is mainly due to the aorta. Using a circuits simulation program, this model can be solved for any $P(t)$ to give blood flow from the heart $F(t)$ (in units of cm^3/s.).

The two-element Windkessel model was successful at modeling flow during diastole, but not systole, motivating the addition of other elements. Figure 1.24B shows a four-element Windkessel model. The series impedance, Z, represents what is known as the "characteristic impedance" of the aorta (impedance is discussed in Chapters 12–14). The parallel inductor, L, represents the inertia of the blood in the aorta. The four-element model represents a good approximation to the real system.

The Windkessel model is an example of a "lumped-parameter" model because it lumps features that are spread throughout the vascular system into single element. Frank's original motivation for this model was to predict cardiac output (i.e., flow) given the pressure waveform, and this it does quite well. It cannot tell you about the distribution of pressures and

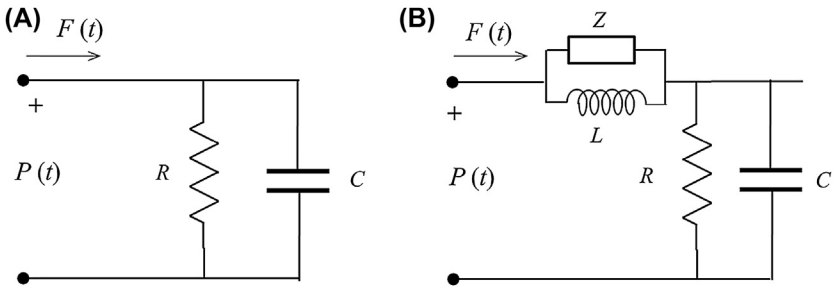


FIGURE 1.24 (A) A two-element Windkessel model of the load imposed on the heart by the blood vessels. This model can be used to predict aortic blood flow, $F(t)$, given the cardiac pressure, $P(t)$. (B) A more elaborate four-element Windkessel model that includes an additional resistor to model the peripheral aorta and an inductor to represent the inertia of the blood.

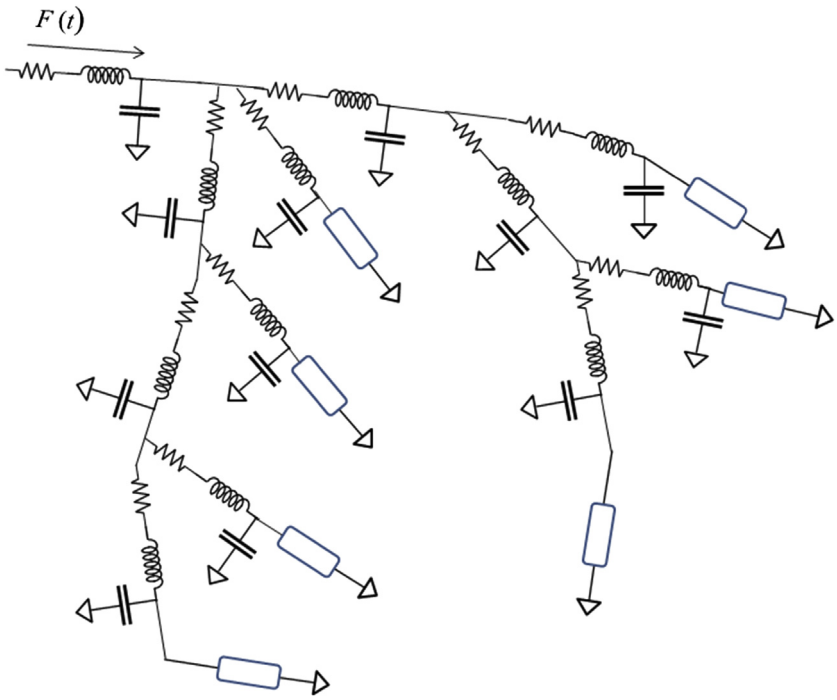


FIGURE 1.25 A distributed model of the coronary arterial system that can provide information on the distribution of pressure and flows in different arteries. This model was developed Wang, his mentor Welkowitz, and colleagues at Rutgers University. *Adapted from Wang et al., 1989. Med. Biol. Engr. Comp. 27:416.*

flows in the cardiovascular system. For that, a "distributed model" would be necessary. An example of a distributed model of the primary coronary arteries is shown in Figure 1.25. This model was used to predict coronary blood flow in the study of the sounds that might be produced by partially blocked arteries, a highly prevalent disorder known as coronary artery disease.

1.4.5.2 Systems Models (Transfer Function Models)

In a systems model, a biological process is represented by an element that mimics the stimulus–response behavior of that process. The stimulus is the input to a model element, whereas the response is the element's output. The elements are composed, internally, of differential equations and may include nonlinear terms. These equations are constructed to parody the stimulus–response behavior of the biological process. Because the elements describe how an input is transferred to the output, the elements are also known as "transfer functions." (The transfer function is introduced in Chapter 6.) Systems models are really good at representing the flow of information through a complex biological system, but they are less clear on how biological processes mediate a specific stimulus–response operation. The elements of a systems model describe what the process does, but not how.

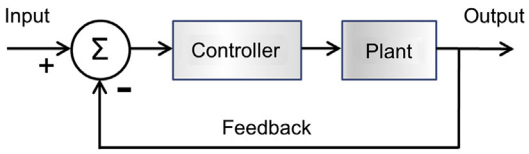


FIGURE 1.26 A common system model structure that includes a controller that governs an operation known as a plant (as in a chemical or manufacturing plant). A feedback pathway lets the controller know what the plant is doing. The controller then modifies its control signals to make the plant achieve the desired end as indicted by the input signal. This configuration is known as a "feedback control system."

Systems models date from the 1950s when control engineers turned their attention to biological systems. The classic structure of a systems model is shown in Figure 1.26. These models are divided into two major subdivisions: components that control an operation and components that implement that operation. The latter are referred to as the "plant," a term held over from the application of systems modeling to large manufacturing facilities such as a chemical plant. Logically, the collection of control components is called the "controller." A "feedback" pathway connects the output of the plant back to the controller. This feedback signal lets the controller know what the plant is doing so that it can adjust its control to achieve some desired end, or output. This desired end is indicated by the input signal.

A classic example of a feedback control system is a climate control system such as a home thermostat. The plant, the effector mechanism, is an air conditioner and/or furnace and the controller is a thermostat. The desired output is room temperature, say 70°. The feedback pathway sends information about the plant output, which is the room temperature, to the controller. The controller compares the plant output (room temperature) with a desired setting, to determine if it should activate the air conditioner or furnace and to what extent.

Systems models have been applied to a wide range of physiological mechanisms, including muscle-based control, fluid compartment models (see Chapter 10), drug uptake dynamics, and some endocrine systems. The first system models were of muscle-based control systems, including physiological motor control systems, the respiratory system, and the cardiovascular system. These systems fit the controller—plant paradigm quite well: nervous system components serve as the controller and muscle mechanisms are the plant. For motor control systems, the desired output is a movement; for respiratory and cardiovascular systems, the outputs are blood gas levels.

The control of human eye movements requires high precision and high speed. Eye movement positional errors are less than a degree and a typical movement takes less than 400 ms. (Small movements approach velocities of 600 deg/s!) This extraordinary performance attracted some of the earliest research in muscle control modeling. Figure 1.27 shows a model of the control of "vergence" eye movements where the eyes move in opposition to view targets at different depths. The plant in this model includes the dynamics of the extraocular muscles and mechanics of the eyeball. The input to the plant is a neural signal delivered by the oculomotor neurons in the brainstem and the output is the angle at which the two eyes converge. The plant component is actually a differential equation that represents this

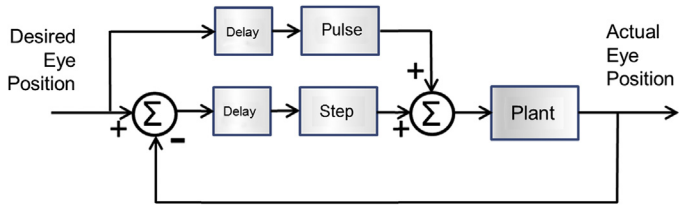


FIGURE 1.27 A systems model of the vergence eye movement control system; the ocular responses that drive the eye in opposition to fixate targets at different depths. A rapid response pulse control signal gets the eyes close to the desired position. It does not rely on feedback and is not subject to the longer delays involved with the feedback signal. Feedback is used to drive the step signal and is used to achieve high levels of positional accuracy, but more slowly. *Adapted from Lee et al., 2012. J. Eye Movement Res., 5:1.*

input–output relationship; that is, the relationship between nerve signal amplitude and the angular position of the eyes.

Although all system elements have a defining equation, they are not always indicated in the element graphic. Whether shown or not, the defining equation describes the element's dynamic properties: how its output varies over time to any given input. In the case of the eye movement plant, these properties have been determined with good accuracy for a range of eye movements, through neurophysiological and behavioral studies. The plant element, like the overall model, provides a tool for unifying experimental findings and for connecting theory and experiment.

The controller features two separate pathways that reflect the underlying neurophysiology. The upper pathway develops a pulse-like signal that originates from so-called burst cells in the brainstem. This neural signal starts the eyes moving rapidly, bringing them close to their final destination. The lower pathway produces a step-like signal originating from "tonic cells" found in the same region of the brainstem as the burst cells. This signal holds the eyes in a position at the final value, and by using the feedback signal, provides fine tuning that results in a very accurate final position.

The feedback signal is explicitly shown to project from the output position back to the input (through appropriate sensory mechanisms), and when compared with the desired output position, provides an error signal that implements the fine adjustment needed to achieve the high position accuracy of these movements. Since this error signal is obtained by comparing the desired position with the negative of the feedback signal, this configuration is sometimes referred to as a "negative feedback control system."

In theory, the vergence eye controller could rely only on the feedback signal to drive the eyes, but because of delays in neural processing, the resultant system would be either very slow or unstable. To solve this problem, the neural controller in the brain adds a pulse signal to increase the speed of the controlled movement. The model configuration of Figure 1.27 shows that the pulse signal is not influenced by the feedback signal. This rapid response pulse signal brings the eyes quickly, but only approximately, to the desired position. This component is not affected by the neural delays associated with feedback. The step signal does use feedback to achieve a high level of accuracy, but at a much slower pace. Finally, the plant consists of differential equations.

Again, each of the lines in Figure 1.27 represents a signal pathway with the direction of the signal indicated by the arrow. Each box represents a set of differential equations that describe how the input signal is transformed into an output signal. The behavior of this model can be evaluated through simulation: programming the operation of the model on a computer. As described in Chapter 9, MATLAB has an effective, easy-to-use simulation tool called Simulink.

In some systems, the desired response is a fixed constant so that the only signal to the controller is the feedback signal. An example is temperature regulation in the human body. The desired response is a fixed constant temperature (98.6°F). Although body temperature is maintained by a complex plant that involves a number of different mechanisms (peripheral blood flow, muscle activity), the controller resides in the brain and relies on core body temperature to develop the necessary neural control signals.

1.5 SUMMARY

It is a great time to be a biomedical engineer, but you need some specific tools to carry out your tasks. The primary purpose of this book is to develop two important biomedical skill sets: signal analysis and systems analysis. In addition, circuit analysis skills are presented in the final sections of the book. I cannot present all the many tools in these skill sets, but we do cover the most important and the most fundamental.

From traditional reductionist viewpoint, living things are described in terms of component systems. Traditional physiological systems, such as the cardiovascular, endocrine, and nervous systems, are large, complex, and composed of numerous subsystems. Biosignals provide communication between systems and subsystems and are our primary source of information on the behavior of these systems. Interpretation and transformation of signals is a major focus of this text.

Biosignals, like all signals, must be carried by some form of energy. Common biological energy sources include chemical, mechanical, and electrical energy (in the body, electrical signals are carried by ions, not electrons). Sometimes these signals can be obtained directly from their biological source as in cardiac auscultation (i.e., listening to the heart with a stethoscope). Other approaches look at how the physiological system interacts with an external energy source such as in ultrasound and CT scanning. Finally, as in MRI, external energy can be introduced into the body and this originally external energy creates the biological signal source.

Measuring a signal usually entails converting it to an electric signal to be compatible with computer processing. Signal conversion is achieved by a device known as an input transducer, and when the energy is from a living system, it is termed a biotransducer. Creation of a digital (or discrete-time) signal requires slicing a continuous signal into discrete levels of both amplitude and time and is accomplished with an electronic device known, logically, as an analog-to-digital converter. Amplitude slicing adds noise to the signal and the noise level is dependent on the size of the slice, but typical slices are so small that this noise is usually ignored. Time slicing produces a more complicated transformation that is fully explored in Chapter 4.

Signals have a number of important properties that we need to understand in order to work with them effectively. Many of the properties are shared by, and often caused by the systems that produce (or modify) those signals. Some signals and systems are stationary or time invariant: they follow the same basic patterns for all time; their properties, and sometimes the complete signal, can be described by equations. Signals that change their basic nature over time are called nonstationary signals. Sometimes we can adjust for these changes, but often we just try to pick out sections of the signal that are stationary. In Chapter 10, we present a few techniques for analyzing nonstationary signals.

Signals and systems can also be causal or noncausal. Noncausal systems produce signal outputs that respond to parts of the input signal that have not yet occurred; at least part of the system's response is driven by a signal from the future. Although such systems would be great for analyzing the stock market, they do not exist in the real world. You can construct a noncausal system in a computer if it is acting on prestored data.

Systems, and their resultant signals, can be linear or nonlinear. Linear systems respond proportionally: double the input and you get double the output. In Chapter 5, we discover that linearity allows us to use a powerful analysis tool called "superposition." Nonlinear systems are more difficult to analyze, although we find in Chapter 9 that simulation methods allow us to predict the response of systems that have well-defined nonlinearities. It can be very difficult just finding out if a signal has nonlinear properties; however, such properties can be of great diagnostic value. In Chapter 10, we examine a few tools recently developed to identify signal nonlinearity.

Signals can be deterministic or stochastic. Deterministic signals are totally predictable and can be represented by an equation. These are the signals we generally rely on throughout this book. Deterministic signals are necessarily time invariant (their statistical properties do not change with time) and can be divided into three classes: periodic signals that regularly repeat, aperiodic signals that occur only once for a finite time, and transient signals that make a step-like change and never return to their baseline or starting level. Stochastic signals are unpredictable, but if they are time invariant, some of their properties such as mean and variance can be uniquely defined.

By definition, a signal is what you want and noise is everything else; all signals contain some noise. Since we are generally not interested in a random signal, stochastic signals are usually regarded as noise. But a deterministic signals can also be noise if it is unwanted. Noise has many sources and only electronic noise can be evaluated based on known characteristics of the instrumentation. Since electronic noise exists at all frequencies, reducing the frequency ranges of a signal is bound to reduce noise. That provides the motivation for filtering a signal as detailed in Chapter 9.

Biosystems modeling is a powerful analytical tool for investigating living systems. Two very different models have been used to represent various physiological systems: analog models and system models. In analog models, analogous electrical or mechanical elements are used to represent specific biological processes (analog as used here refers to the analogy made between the living process and the element and has a different meaning than when used in analog signal). In systems models, each element represents a relationship between the input and the output signals of a process. This relationship is often called a transfer function since it effectively transfers the input to the output (see Chapter 6).

The strength of analog models is that they have a more intuitive relationship to the physiological process. The elasticity of muscle is represented by a spring, its mass by a mass element; there is a close relationship between the physical activity of the process and that of the element. In system models, no such relationship exists: a muscle is represented by its transfer function: a simple input–output relationship where nerve activation is the input and force, or movement, the output. The strength of systems models is the simplification provided by the transfer function and that information flow is shown explicitly. For this reason, systems models are favored when large biosystems are involved and where the influence of one process on another is of particular concern. Moreover, systems models can be easily simulated (as described in Chapter 10), which has led to a growing number of biological applications.

The goal of this book is to present the most important and fundamental of the many powerful signal and systems analysis tools available to biomedical engineers. It is a mixed blessing, you have a lot of options with which to attack life science problems, but you need to know a lot to use these tools effectively. Still, if you wanted it easy, you would have chosen something other than engineering!

PROBLEMS

1. The file on the CD `quantized.mat` contains a signal, `x`, and an amplitude sliced version of that signal, `y`, and time vector (for plotting) in seconds, `t`. This signal has been sliced into 16 different levels. In Chapter 3, we find that amplitude slicing is like adding noise to a signal. Plot the two signals superimposed. Find the effective "noise signal" by subtracting `y` from `x` and plot on the same graph. Now find the RMS value of this noise and compare it to the RMS value of the original signal. As with all graphs, label the axes. Also use the time vector to correctly scale the horizontal axis. (Hint: RMS means "root mean squared" the equation, and MATLAB code just follows the name (root mean squared); take the square root of the mean of the signal squared: `RMS_x = sqrt(mean(x.^2));`)
2. Image generation. Using the approach shown in Example 1.2, construct a sinusoidal grating having a spatial frequency of five cycles per horizontal distance. Transpose this image and display. To transpose, use the MATALAB® operator. You should have a sinusoidal grating with horizontal strips. Then multiply each row in the transposed image by the original sine wave and display. This should generate a checkerboard pattern.
3. Load the image of the brain found in `brain.mat`, display the original and apply several mathematical operations. (1) Invert the image: make black white and white black. (2) Apply a nonlinear transformation: make a new image that is the square root of the pixel value of the original. (Note: this requires only one line of MATLAB code.) (3) Create a doubly thresholded image. Set all values below 0.25 in the original image to zero (black), all values above 0.5 to 1.0 (white), and anything in between to 0.5 (gray). (In this exercise, it is easier not to use `caxis` to set the grayscale range. Just let the `pcolor` routine automatically adjust to the range of your transformed images.) These images should be plotted using the `bone` colormap to reproduce the grayscale image

with accuracy, but after plotting out the figure you could apply other colormaps such as `jet`, `hot`, or `hsv` for some interesting pseudocolor effects. Just type `colormap(hot)`, etc.

4. Repeat Example 1.4 using Gaussian arrays that are 100, 500, 1000, and 5000 points long. Plot the distribution functions generated by the MATLAB `hist` routine using a bin width of 40. Use subplot to combine the plots. (Hint: A `for` loop can significantly reduce the amount of code needed.)
5. Repeat Problem 4, with a single Gaussian array of 200 points. Construct four different histograms using 10, 20, 30, and 40 bins. Plot as in Problem 4.
6. Follow Example 1.6 to construct plots of a 2.5 Hz sine wave and a 1.5 Hz cosine wave. Make the peak amplitude of both 20. Use a 500-point array (`N` = 500) and make the sampling frequency 250 Hz. Plot the two waveforms in different colors superimposed and label both axes. Also plot a zero centerline.
7. Generate the sine and cosine wave used in Problem 6, following the procedure used in Example 1.6, but make the frequency of both 2.5 Hz. Use an array of 5000 points and an `fs` of 2500. Find the distribution of the sine wave (or cosine wave, they are the same) by taking the histogram. Use 40 bins. The result should be shaped like a "U" since the sine (or cosine) spends most of its time, so to speak, at the two extremes. Now combine the sine and cosine by simply adding them together and construct the histogram. Note that the distribution of the combined waveform is very different, approaching a Gaussian distribution. Another example of the Central Limit Theorem in action.
8. A resistor produces 10 μV noise (i.e., 10×10^{-6} volts noise) when the room temperature is 310 K and the bandwidth is 1 kHz (i.e., 1000 Hz). What current noise would be produced by this resistor?
9. The noise voltage out of a 1-MΩ (i.e.,10^6-ohm) resistor is measured using a digital voltmeter as 1.5 μV at a room temperature of 310 K. What is the effective bandwidth of the voltmeter?
10. A 3 mA current flows through both a diode (i.e., a semiconductor) and a 20,000-Ω (i.e. 20-kΩ) resistor. What is the net current noise, i_n? Assume a bandwidth of 1 kHz (i.e. 1×10^3 Hz). Which of the two components is responsible for producing the most noise?
11. Use MATLAB to evaluate the logistic equation (Equation 1.10) for four different values of r: 1.25, 2.25, 3.2, and 3.6. Evaluate the first 50 generations (use a `for` loop to increment n from 1 to 50) and start with an initial value for x of 0.02. Plot the population x as a function of generation, n. Use subplot to put the four plots together. Label the plots appropriately.
12. Construct the aperiodic function similar to that shown in the upper trace in Figure 1.19. The waveform should be 1.0 from −0.5 to +0.5 s and zero elsewhere. Use a 400-point array and set the sampling frequency so the time scale is from −1 to +1 s. Label and scale the time axis correctly. (Hint: Use Equation 1.3 to get f_s where n = `N` = 400 and t = total time = 2.0 s. You can construct the time vector for plotting as in Example 1.2, but will need to subtract to get it to begin at −1.0 s.)
13. Follow the procedure in Example 1.7 to determine if the unknown process "`unknown2.m`" is linear or over what range of inputs it could be considered approximately linear.