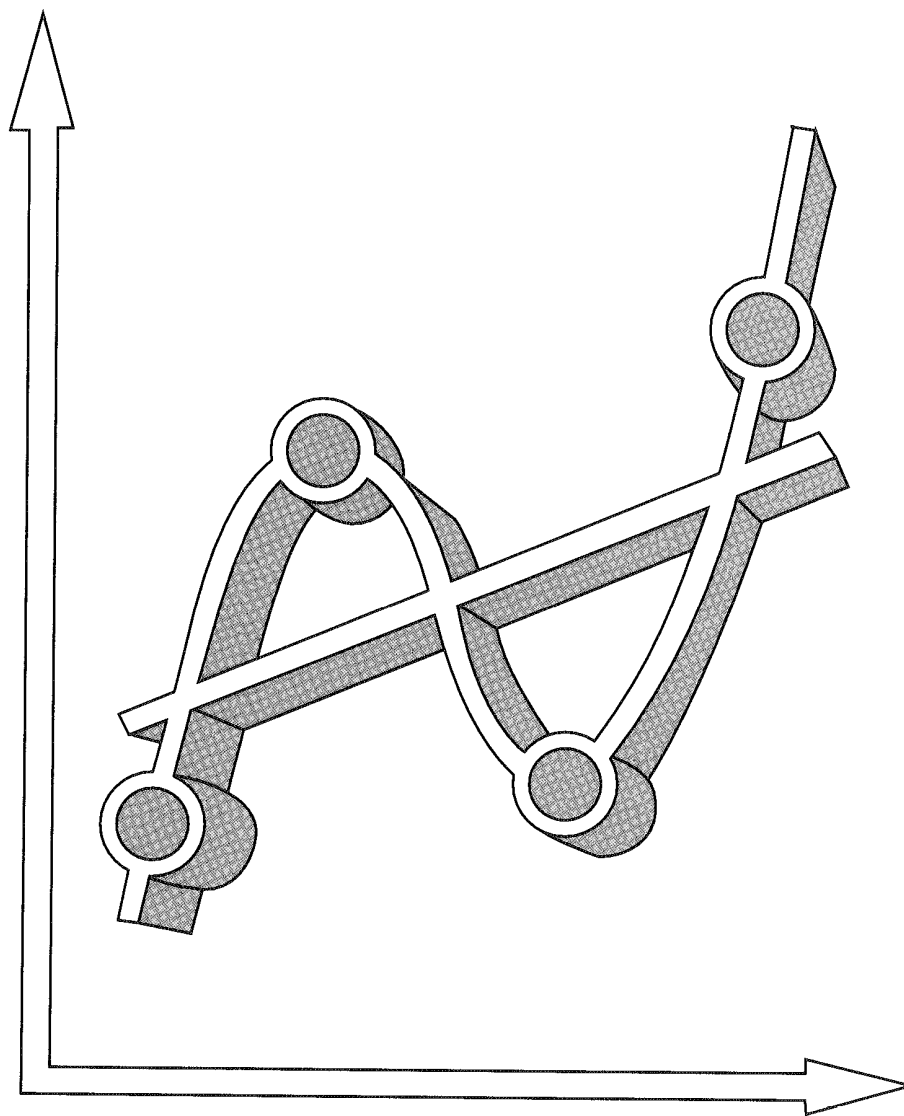


# PART FIVE

---



# CURVE FITTING

---

## PT5.1 MOTIVATION

---

Data is often given for discrete values along a continuum. However, you may require estimates at points between the discrete values. The present part of this book describes techniques to fit curves to such data to obtain intermediate estimates. In addition, you may require a simplified version of a complicated function. One way to do this is to compute values of the function at a number of discrete values along the range of interest. Then, a simpler function may be derived to fit these values. Both of these applications are known as *curve fitting*.

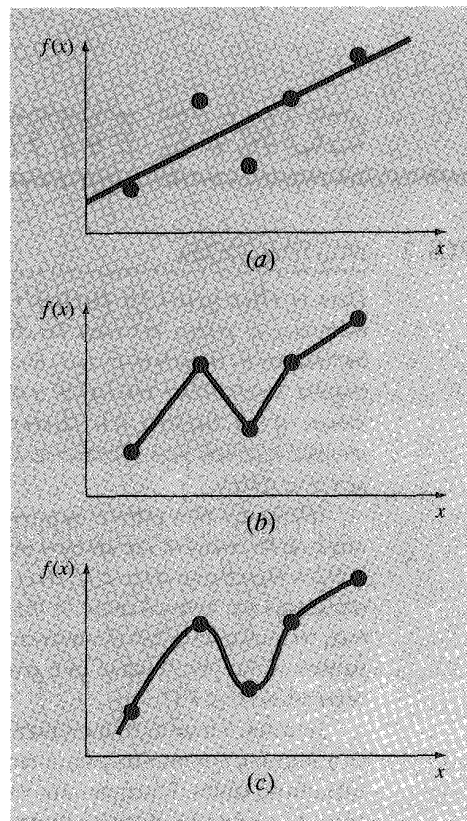
There are two general approaches for curve fitting that are distinguished from each other on the basis of the amount of error associated with the data. First, where the data exhibits a significant degree of error or “noise,” the strategy is to derive a single curve that represents the general trend of the data. Because any individual data point may be incorrect, we make no effort to intersect every point. Rather, the curve is designed to follow the pattern of the points taken as a group. One approach of this nature is called *least-squares regression* (Fig. PT5.1a).

Second, where the data is known to be very precise, the basic approach is to fit a curve or a series of curves that pass directly through each of the points. Such data usually originates from tables. Examples are values for the density of water or for the heat capacity of gases as a function of temperature. The estimation of values between well-known discrete points is called *interpolation* (Fig. PT5.1b and c).

### PT5.1.1 Noncomputer Methods for Curve Fitting

The simplest method for fitting a curve to data is to plot the points and then sketch a line that visually conforms to the data. Although this is a valid option when quick estimates are required, the results are dependent on the subjective viewpoint of the person sketching the curve.

For example, Fig. PT5.1 shows sketches developed from the same set of data by three engineers. The first did not attempt to connect the points, but rather, characterized the general upward trend of the data with a straight line (Fig. PT5.1a). The second engineer used straight-line segments or linear interpolation to connect the points (Fig. PT5.1b). This is a very common practice in engineering. If the values are truly close to being linear or are spaced closely, such an approximation provides estimates that are adequate for many engineering calculations. However, where the underlying relationship is highly curvilinear or the data is widely spaced, significant errors can be introduced by such linear interpolation. The third engineer used curves to try to capture the meanderings suggested by the data (Fig. PT5.1c). A fourth or fifth engineer would likely develop alternative fits. Obviously,

**FIGURE PT5.1**

Three attempts to fit a “best” curve through five data points. (a) Least-squares regression, (b) linear interpolation, and (c) curvilinear interpolation.

our goal here is to develop systematic and objective methods for the purpose of deriving such curves.

### PT5.1.2 Curve Fitting and Engineering Practice

Your first exposure to curve fitting may have been to determine intermediate values from tabulated data—for instance, from interest tables for engineering economics or from steam tables for thermodynamics. Throughout the remainder of your career, you will have frequent occasion to estimate intermediate values from such tables.

Although many of the widely used engineering properties have been tabulated, there are a great many more that are not available in this convenient form. Special cases and new problem contexts often require that you measure your own data and develop your own predictive relationships. Two types of applications are generally encountered when fitting experimental data: trend analysis and hypothesis testing.

Trend analysis represents the process of using the pattern of the data to make predictions. For cases where the data is measured with high precision, you might utilize interpolating polynomials. Imprecise data is often analyzed with least-squares regression.

*Trend analysis* may be used to predict or forecast values of the dependent variable. This can involve extrapolation beyond the limits of the observed data or interpolation within the range of the data. All fields of engineering commonly involve problems of this type.

A second engineering application of experimental curve fitting is *hypothesis testing*. Here, an existing mathematical model is compared with measured data. If the model coefficients are unknown, it may be necessary to determine values that best fit the observed data. On the other hand, if estimates of the model coefficients are already available, it may be appropriate to compare predicted values of the model with observed values to test the adequacy of the model. Often, alternative models are compared and the “best” selected on the basis of empirical observations.

In addition to the above engineering applications, curve fitting is important in other numerical methods such as integration and the approximate solution of differential equations. Finally, curve-fitting techniques can be used to derive simple functions to approximate complicated functions.

## PT5.2 MATHEMATICAL BACKGROUND

The prerequisite mathematical background for interpolation is found in the material on Taylor series expansions and finite divided differences introduced in Chap. 4. Least-squares regression requires additional information from the field of statistics. If you are familiar with the concepts of the mean, standard deviation, residual sum of the squares, normal distribution, and confidence intervals, feel free to skip the following pages and proceed directly to PT5.3. If you are unfamiliar with these concepts or are in need of a review, the following material is designed as a brief introduction to these topics.

### PT5.2.1 Simple Statistics

Suppose that in the course of an engineering study, several measurements were made of a particular quantity. For example, Table PT5.1 contains 24 readings of the coefficient of thermal expansion of a structural steel. Taken at face value, the data provides a limited amount of information—that is, that the values range from a minimum of 6.395 to a maximum of 6.775. Additional insight can be gained by summarizing the data in one or more well-chosen statistics that convey as much information as possible about specific characteristics of the data set. These descriptive statistics are most often selected to represent

**TABLE PT5.1** Measurements of the coefficient of thermal expansion of structural steel  
[ $\times 10^{-6}$  in/(in  $\cdot$  °F)]

6.495	6.595	6.615	6.635	6.485	6.555
6.665	6.505	6.435	6.625	6.715	6.655
6.755	6.625	6.715	6.575	6.655	6.605
6.565	6.515	6.555	6.395	6.775	6.685

(1) the location of the center of the distribution of the data and (2) the degree of spread of the data set.

The most common location statistic is the arithmetic mean. The *arithmetic mean* ( $\bar{y}$ ) of a sample is defined as the sum of the individual data points ( $y_i$ ) divided by the number of points ( $n$ ), or

$$\bar{y} = \frac{\sum y_i}{n} \quad (\text{PT5.1})$$

where the summation (and all the succeeding summations in this introduction) is from  $i = 1$  through  $n$ .

The most common measure of spread for a sample is the *standard deviation* ( $s_y$ ) about the mean,

$$s_y = \sqrt{\frac{S_t}{n-1}} \quad (\text{PT5.2})$$

where  $S_t$  is the total sum of the squares of the residuals between the data points and the mean, or

$$S_t = \sum (y_i - \bar{y})^2 \quad (\text{PT5.3})$$

Thus, if the individual measurements are spread out widely around the mean,  $S_t$  (and, consequently,  $s_y$ ) will be large. If they are grouped tightly, the standard deviation will be small. The spread can also be represented by the square of the standard deviation, which is called the *variance*:

$$s_y^2 = \frac{S_t}{n-1} \quad (\text{PT5.4})$$

Note that the denominator in both Eqs. (PT5.2) and (PT5.4) is  $n - 1$ . The quantity  $n - 1$  is referred to as the degrees of freedom. Hence  $S_t$  and  $s_y$  are said to be based on  $n - 1$  *degrees of freedom*. This nomenclature derives from the fact that the sum of the quantities upon which  $S_t$  is based (that is,  $\bar{y} - y_1, \bar{y} - y_2, \dots, \bar{y} - y_n$ ) is zero. Consequently, if  $\bar{y}$  is known and  $n - 1$  of the values are specified, the remaining value is fixed. Thus, only  $n - 1$  of the values are said to be freely determined. Another justification for dividing by  $n - 1$  is the fact that there is no such thing as the spread of a single data point. For the case where  $n = 1$ , Eqs. (PT5.2) and (PT5.4) yield a meaningless result of infinity.

It should be noted that an alternative, more convenient formula is available to compute the standard deviation,

$$s_y^2 = \frac{\sum y_i^2 - (\sum y_i)^2 / n}{n-1}$$

This version does not require precomputation of  $\bar{y}$  and yields an identical result as Eq. (PT5.4).

A final statistic that has utility in quantifying the spread of data is the *coefficient of variation* (*c.v.*). This statistic is the ratio of the standard deviation to the mean. As such, it provides a normalized measure of the spread. It is often multiplied by 100 so that it can be expressed in the form of a percent:

$$\text{c.v.} = \frac{s_y}{\bar{y}} 100\% \quad (\text{PT5.5})$$

Notice that the coefficient of variation is similar in spirit to the percent relative error ( $\epsilon_r$ ) discussed in Sec. 3.3. That is, it is the ratio of a measure of error ( $s_y$ ) to an estimate of the true value ( $\bar{y}$ ).

### EXAMPLE PT5.1 Simple Statistics of a Sample

**Problem Statement.** Compute the mean, variance, standard deviation, and coefficient of variation for the data in Table PT5.1.

**TABLE PT5.2** Computations for statistics for the readings of the coefficient of thermal expansion. The frequencies and bounds are developed to construct the histogram in Fig. PT5.2.

<i>i</i>	$y_i$	$(y_i - \bar{y})^2$	Frequency	Interval	
				Lower Bound	Upper Bound
1	6.395	0.042025	1	6.36	6.40
2	6.435	0.027225	1	6.40	6.44
3	6.485	0.013225	4	6.48	6.52
4	6.495	0.011025			
5	6.505	0.009025			
6	6.515	0.007225			
7	6.555	0.002025	2	6.52	6.56
8	6.555	0.002025			
9	6.565	0.001225			
10	6.575	0.000625	3	6.56	6.60
11	6.595	0.000025			
12	6.605	0.000025			
13	6.615	0.000225	5	6.60	6.64
14	6.625	0.000625			
15	6.625	0.000625			
16	6.635	0.001225			
17	6.655	0.003025			
18	6.655	0.003025	3	6.64	6.68
19	6.665	0.004225			
20	6.685	0.007225			
21	6.715	0.013225	3	6.68	6.72
22	6.715	0.013225			
23	6.755	0.024025	1	6.72	6.76
24	6.775	0.030625	1	6.76	6.80
$\Sigma$	158.4	0.217000			

**Solution.** The data is added (Table PT5.2) and the results are used to compute [Eq. (PT5.1)]

$$\bar{y} = \frac{158.4}{24} = 6.6$$

As in Table PT5.2, the sum of the squares of the residuals is 0.217000, which can be used to compute the standard deviation [Eq. (PT5.2)]:

$$s_y = \sqrt{\frac{0.217000}{24 - 1}} = 0.097133$$

the variance [Eq. (PT5.4)]:

$$s_y^2 = 0.009435$$

and the coefficient of variation [Eq. (PT5.5)]:

$$\text{c.v.} = \frac{0.097133}{6.6} 100\% = 1.47\%$$

### PT5.2.2 The Normal Distribution

Another characteristic that bears on the present discussion is the *data distribution*—that is, the shape with which the data is spread around the mean. A *histogram* provides a simple visual representation of the distribution. As in Table PT5.2, the histogram is constructed by sorting the measurements into intervals. The units of measurement are plotted on the abscissa and the frequency of occurrence of each interval is plotted on the ordinate. Thus, five of the measurements fall between 6.60 and 6.64. As in Fig. PT5.2, the histogram suggests that most of the data is grouped close to the mean value of 6.6.

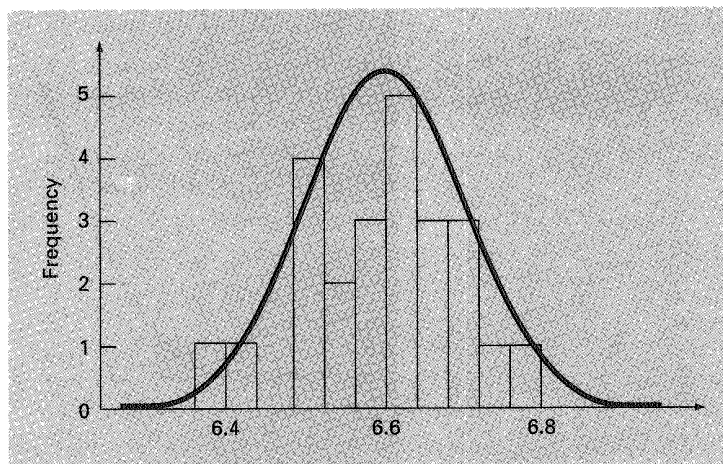
If we have a very large set of data, the histogram often can be approximated by a smooth curve. The symmetric, bell-shaped curve superimposed on Fig. PT5.2 is one such characteristic shape—the *normal distribution*. Given enough additional measurements, the histogram for this particular case could eventually approach the normal distribution.

The concepts of the mean, standard deviation, residual sum of the squares, and normal distribution all have great relevance to engineering practice. A very simple example is their use to quantify the confidence that can be ascribed to a particular measurement. If a quantity is normally distributed, the range defined by  $\bar{y} - s_y$  to  $\bar{y} + s_y$  will encompass approximately 68 percent of the total measurements. Similarly, the range defined by  $\bar{y} - 2s_y$  to  $\bar{y} + 2s_y$  will encompass approximately 95 percent.

For example, for the data in Table PT5.1 ( $\bar{y} = 6.6$  and  $s_y = 0.097133$ ), we can make the statement that approximately 95 percent of the readings should fall between 6.405734 and 6.794266. If someone told us that they had measured a value of 7.35, we would suspect that the measurement might be erroneous. The following section elaborates on such evaluations.

### PT5.2.3 Estimation of Confidence Intervals

As should be clear from the previous sections, one of the primary aims of statistics is to estimate the properties of a *population* based on a limited *sample* drawn from that population.

**FIGURE PT5.2**

A histogram used to depict the distribution of data. As the number of data points increases, the histogram could approach the smooth, bell-shaped curve called the normal distribution.

Clearly, it is impossible to measure the coefficient of thermal expansion for every piece of structural steel that has ever been produced. Consequently, as in Tables PT5.1 and PT5.2, we can randomly make a number of measurements and, on the basis of the sample, attempt to characterize the properties of the entire population.

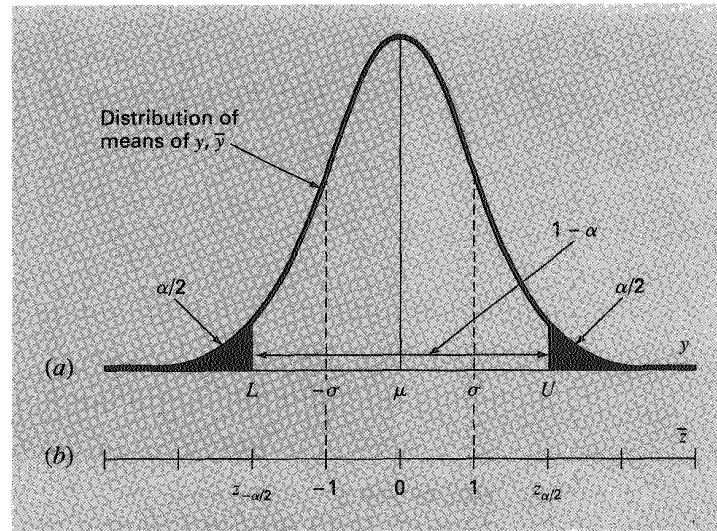
Because we “infer” properties of the unknown population from a limited sample, the endeavor is called *statistical inference*. Because the results are often reported as estimates of the population parameters, the process is also referred to as *estimation*.

We have already shown how we estimate the central tendency (sample mean,  $\bar{y}$ ) and spread (sample standard deviation and variance) of a limited sample. Now, we will briefly describe how we can attach probabilistic statements to the quality of these estimates. In particular, we will discuss how we can define a confidence interval around our estimate of the mean. We have chosen this particular topic because of its direct relevance to the regression models we will be describing in Chap. 17.

Note that in the following discussion, the nomenclature  $\bar{y}$  and  $s_y$  refer to the sample mean and standard deviation. The nomenclature  $\mu$  and  $\sigma$  refer to the population mean and standard deviation. The former are sometimes referred to as the “estimated” mean and standard deviation, whereas the latter are sometimes called the “true” mean and standard deviation.

An *interval estimator* gives the range of values within which the parameter is expected to lie with a given probability. Such intervals are described as being one-sided or two-sided. As the name implies, a *one-sided interval* expresses our confidence that the parameter estimate is less than or greater than the true value. In contrast, the *two-sided interval* deals with the more general proposition that the estimate agrees with the truth with no consideration to the sign of the discrepancy. Because it is more general, we will focus on the two-sided interval.



**FIGURE PT5.3**

A two-sided confidence interval. The abscissa scale in (a) is written in the natural units of the random variable  $y$ . (b) is a normalized version of the abscissa that places the origin at the location of the mean and scales the axis so that the standard deviation corresponds to a unit value.

A two-sided interval can be described by the statement

$$P\{L \leq \mu \leq U\} = 1 - \alpha$$

which reads, “the probability that the true mean of  $y$ ,  $\mu$ , falls within the bound from  $L$  to  $U$  is  $1 - \alpha$ .” The quantity  $\alpha$  is called the *significance level*. So the problem of defining a confidence interval reduces to estimating  $L$  and  $U$ . Although it is not absolutely necessary, it is customary to view the two-sided interval with the  $\alpha$  probability distributed evenly as  $\alpha/2$  in each tail of the distribution, as in Fig. PT5.3.

If the true variance of the distribution of  $y$ ,  $\sigma^2$ , is known (which is not usually the case), statistical theory states that the sample mean  $\bar{y}$  comes from a normal distribution with mean  $\mu$  and variance  $\sigma^2/n$  (Box PT5.1). In the case illustrated in Fig. PT5.3, we really do not know  $\mu$ . Therefore, we do not know where the normal curve is exactly located with respect to  $\bar{y}$ . To circumvent this dilemma, we compute a new quantity, the *standard normal estimate*

$$\bar{z} = \frac{\bar{y} - \mu}{\sigma/\sqrt{n}} \quad (\text{PT5.6})$$

which represents the normalized distance between  $\bar{y}$  and  $\mu$ . According to statistical theory, this quantity should be normally distributed with a mean of 0 and a variance of  $\sigma^2/n$ . Furthermore, the probability that  $\bar{z}$  would fall within the unshaded region of Fig. PT5.3

**Box PT5.1** A Little Statistics

Most engineers take several courses to become proficient at statistics. Because you may not have taken such a course yet, we'd like to mention a few ideas that might make this present section more coherent.

As we have stated, the “game” of inferential statistics assumes that the random variable you are sampling,  $y$ , has a true mean ( $\mu$ ) and variance ( $\sigma^2$ ). Further, in the present discussion, we also assume that it has a particular distribution: the normal distribution. The variance of this normal distribution has a finite value that specifies the “spread” of the normal distribution. If the variance is large, the distribution is broad. Conversely, if the variance is small, the distribution is narrow. Thus, the true variance quantifies the intrinsic uncertainty of the random variable.

In the game of statistics, we take a limited number of measurements of this quantity called a sample. From this sample, we can compute an estimated mean ( $\bar{y}$ ) and variance ( $s_y^2$ ). The more measurements we take, the better the estimates approximate the true values. That is, as  $n \rightarrow \infty$ ,  $\bar{y} \rightarrow \mu$  and  $s_y^2 \rightarrow \sigma^2$ .

Suppose that we take  $n$  samples and compute an estimated mean  $\bar{y}_1$ . Then, we take another  $n$  samples and compute another,  $\bar{y}_2$ . We can keep repeating this process until we have generated a sample of means:  $\bar{y}_1, \bar{y}_2, \bar{y}_3, \dots, \bar{y}_m$ , where  $m$  is large. We can then develop a histogram of these means and determine a “distribution of the means,” as well as a “mean of the means” and a “standard deviation of the means.” Now the question arises: does this new distribution of means and its statistics behave in a predictable fashion?

There is an extremely important theorem known as the *Central Limit Theorem* that speaks directly to this question. It can be stated as

*Let  $y_1, y_2, \dots, y_n$  be a random sample of size  $n$  from a distribution with mean  $\mu$  and variance  $\sigma^2$ . Then, for large  $n$ ,  $\bar{y}$  is approximately normal with mean  $\mu$  and variance  $\sigma^2/n$ . Furthermore, for large  $n$ , the random variable  $(\bar{y} - \mu)/(\sigma/\sqrt{n})$  is approximately standard normal.*

Thus, the theorem states the remarkable result that the distribution of means will always be normally distributed regardless of the underlying distribution of the random variables! It also yields the expected result that given a sufficiently large sample, the mean of the means should converge on the true population mean  $\mu$ .

Further, the theorem says that as the sample size gets larger, the variance of the means should approach zero. This makes sense, because if  $n$  is small, our individual estimates of the mean should be poor and the variance of the means should be large. As  $n$  increases, our estimates of the mean will improve and hence their spread should shrink. The Central Limit Theorem neatly defines exactly how this shrinkage relates to both the true variance and the sample size, i.e., as  $\sigma^2/n$ .

Finally, the theorem states the important result that we have given as Eq. (PT5.6). As is shown in this section, this result is the basis for constructing confidence intervals for the mean.

should be  $1 - \alpha$ . Therefore, the statement can be made that

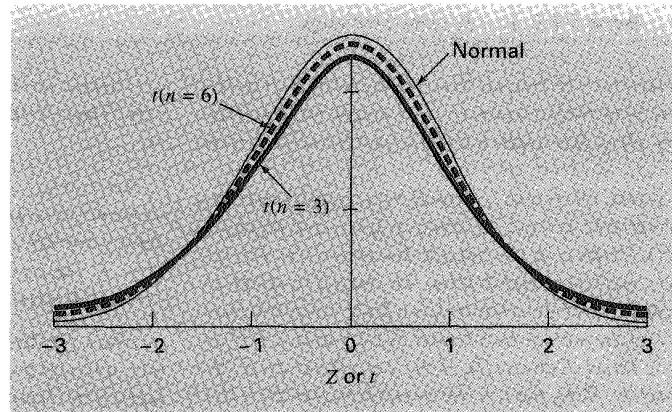
$$\frac{\bar{y} - \mu}{\sigma/\sqrt{n}} < -z_{\alpha/2} \quad \text{or} \quad \frac{\bar{y} - \mu}{\sigma/\sqrt{n}} > z_{\alpha/2}$$

with a probability of  $\alpha$ .

The quantity  $z_{\alpha/2}$  is a standard normal random variable. This is the distance measured along the normalized axis above and below the mean that encompasses  $1 - \alpha$  probability (Fig. PT5.3b). Values of  $z_{\alpha/2}$  are tabulated in statistics books (e.g., Milton and Arnold 1995). They can also be calculated using functions on software packages and libraries like Excel and IMSL. As an example, for  $\alpha = 0.05$  (in other words, defining an interval encompassing 95%),  $z_{\alpha/2}$  is equal to about 1.96. This means that an interval around the mean of width  $\pm 1.96$  times the standard deviation will encompass approximately 95% of the distribution.

These results can be rearranged to yield

$$L \leq \mu \leq U$$

**FIGURE PT5.4**

Comparison of the normal distribution with the  $t$  distribution for  $n = 3$  and  $n = 6$ . Notice how the  $t$  distribution is generally flatter.

with a probability of  $1 - \alpha$ , where

$$L = \bar{y} - \frac{\sigma}{\sqrt{n}} z_{\alpha/2} \quad U = \bar{y} + \frac{\sigma}{\sqrt{n}} z_{\alpha/2} \quad (\text{PT5.7})$$

Now, although the foregoing provides an estimate of  $L$  and  $U$ , it is based on knowledge of the true variance  $\sigma$ . For our case, we know only the estimated variance  $s_y$ . A straightforward alternative would be to develop a version of Eq. (PT5.6) based on  $s_y$ ,

$$t = \frac{\bar{y} - \mu}{s_y / \sqrt{n}} \quad (\text{PT5.8})$$

Even when we sample from a normal distribution, this fraction will not be normally distributed, particularly when  $n$  is small. It was found by W.S. Gossett that the random variable defined by Eq. (PT5.8) follows the so-called Student- $t$ , or simply,  $t$  distribution. For this case,

$$L = \bar{y} - \frac{s_y}{\sqrt{n}} t_{\alpha/2, n-1} \quad U = \bar{y} + \frac{s_y}{\sqrt{n}} t_{\alpha/2, n-1} \quad (\text{PT5.9})$$

where  $t_{\alpha/2, n-1}$  is the standard random variable for the  $t$  distribution for a probability of  $\alpha/2$ . As was the case for  $z_{\alpha/2}$ , values are tabulated in statistics books, and can also be calculated using software packages and libraries. For example, if  $\alpha = 0.05$  and  $n = 20$ ,  $t_{\alpha/2, n-1} = 2.086$ .

The  $t$  distribution can be thought of as a modification of the normal distribution that accounts for the fact that we have an imperfect estimate of the standard deviation. When  $n$  is small, it tends to be flatter than the normal (see Fig. PT5.4). Therefore, for small

numbers of measurements, it yields wider and hence more conservative confidence intervals. As  $n$  grows larger, the  $t$  distribution converges on the normal.

### EXAMPLE PT5.2 Confidence Interval on the Mean

**Problem Statement.** Determine the mean and the corresponding 95% confidence interval for the data from Table PT5.1. Perform three estimates based on (a) the first 8, (b) the first 16, and (c) all 24 measurements.

**Solution.** (a) The mean and standard deviation for the first 8 points is

$$\bar{y} = \frac{52.72}{8} = 6.59 \quad s_y = \sqrt{\frac{347.4814 - (52.72)^2/8}{8-1}} = 0.089921$$

The appropriate  $t$  statistic can be calculated as

$$t_{0.05/2, 8-1} = t_{0.025, 7} = 2.364623$$

which can be used to compute the interval

$$L = 6.59 - \frac{0.089921}{\sqrt{8}} 2.364623 = 6.5148$$

$$U = 6.59 + \frac{0.089921}{\sqrt{8}} 2.364623 = 6.6652$$

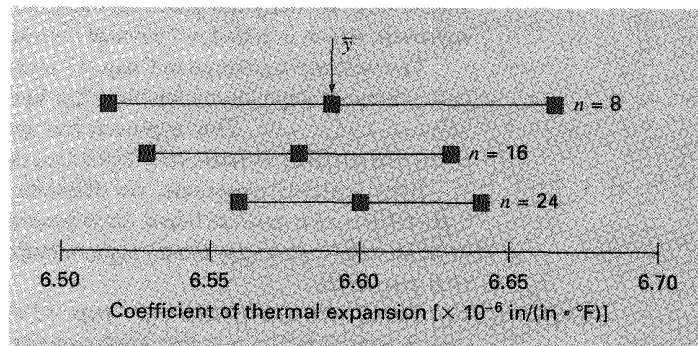
or

$$6.5148 \leq \mu \leq 6.6652$$

Thus, based on the first eight measurements, we conclude that there is a 95% probability that the true mean falls within the range 6.5148 to 6.6652.

**FIGURE PT5.5**

Estimates of the mean and 95% confidence intervals for different numbers of sample size.



The two other cases for (b) 16 points and (c) 24 points can be calculated in a similar fashion and the results tabulated along with case (a) as

$n$	$\bar{y}$	$s_y$	$t_{\alpha/2, n-1}$	$L$	$U$
8	6.5900	0.089921	2.364623	6.5148	6.6652
16	6.5794	0.095845	2.131451	6.5283	6.6304
24	6.6000	0.097133	2.068655	6.5590	6.6410

These results, which are also summarized in Fig. PT5.5, indicate the expected outcome that the confidence interval becomes more narrow as  $n$  increases. Thus, the more measurements we take, our estimate of the true value becomes more refined.

The above is just one simple example of how statistics can be used to make judgments regarding uncertain data. These concepts will also have direct relevance to our discussion of regression models. You can consult any basic statistics book (for example, Milton and Arnold, 1995) to obtain additional information on the subject.

### PT5.3 ORIENTATION

Before we proceed to numerical methods for curve fitting, some orientation might be helpful. The following is intended as an overview of the material discussed in Part Five. In addition, we have formulated some objectives to help focus your efforts when studying the material.

#### PT5.3.1 Scope and Preview

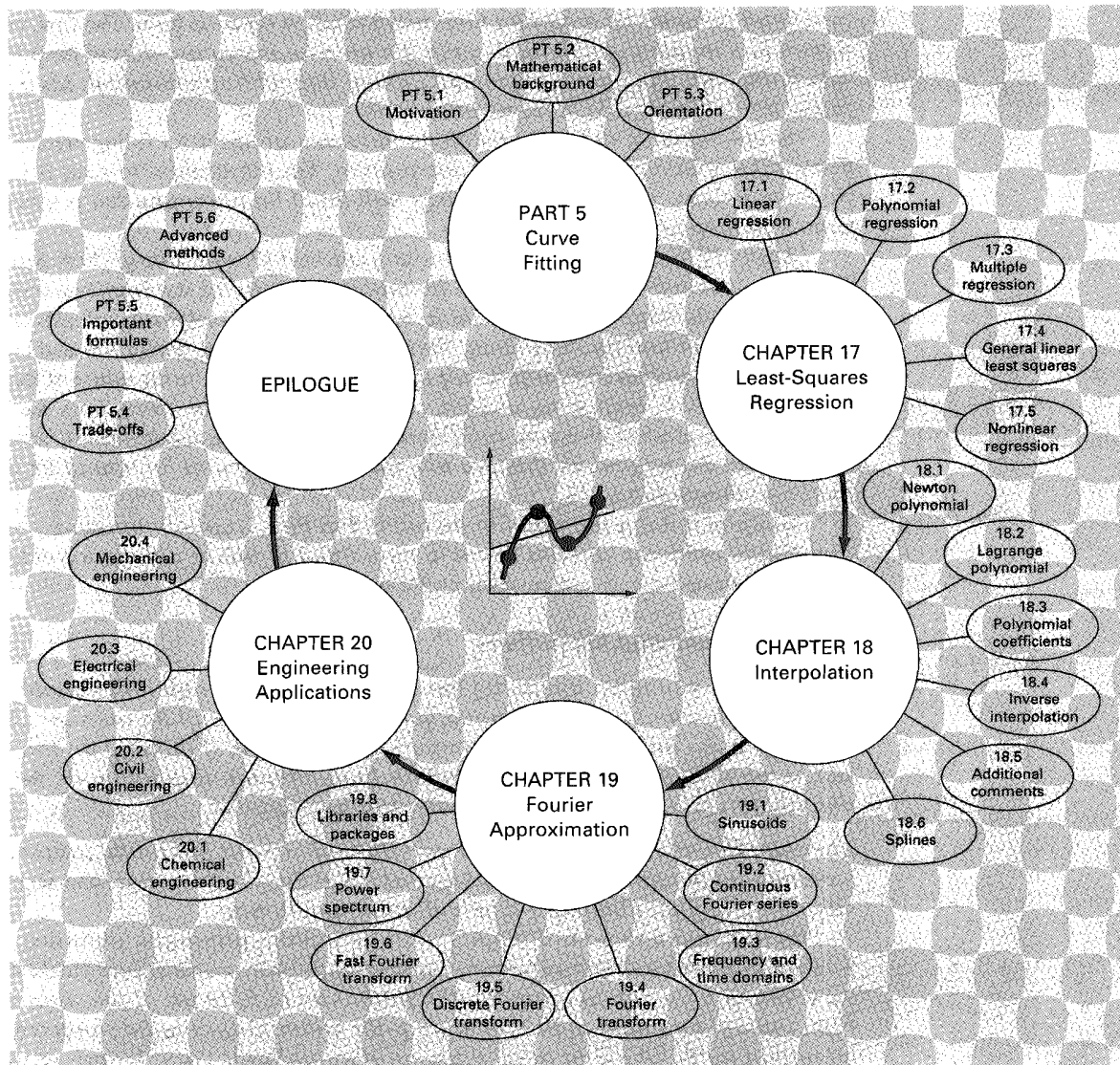
Figure PT5.5 provides a visual overview of the material to be covered in Part Five. *Chapter 17* is devoted to *least-squares regression*. We will first learn how to fit the “best” straight line through a set of uncertain data points. This technique is called *linear regression*. Besides discussing how to calculate the slope and intercept of this straight line, we also present quantitative and visual methods for evaluating the validity of the results.

In addition to fitting a straight line, we also present a general technique for fitting a “best” polynomial. Thus, you will learn to derive a parabolic, cubic, or higher-order polynomial that optimally fits uncertain data. Linear regression is a subset of this more general approach, which is called *polynomial regression*.

The next topic covered in Chap. 17 is *multiple linear regression*. It is designed for the case where the dependent variable  $y$  is a linear function of two or more independent variables  $x_1, x_2, \dots, x_m$ . This approach has special utility for evaluating experimental data where the variable of interest is dependent on a number of different factors.

After multiple regression, we illustrate how polynomial and multiple regression are both subsets of a *general linear least-squares model*. Among other things, this will allow us to introduce a concise matrix representation of regression and discuss its general statistical properties.

Finally, the last sections of Chap. 17 are devoted to *nonlinear regression*. This approach is designed to compute a least-squares fit of a nonlinear equation to data.

**FIGURE PT5.6**

Schematic of the organization of the material in Part Five: Curve Fitting.

In *Chap. 18*, the alternative curve-fitting technique called *interpolation* is described. As discussed previously, interpolation is used for estimating intermediate values between precise data points. In *Chap. 18*, polynomials are derived for this purpose. We introduce the basic concept of polynomial interpolation by using straight lines and parabolas to connect points. Then, we develop a generalized procedure for fitting an  $n$ th-order polynomial. Two

formats are presented for expressing these polynomials in equation form. The first, called *Newton's interpolating polynomial*, is preferable when the appropriate order of the polynomial is unknown. The second, called the *Lagrange interpolating polynomial*, has advantages when the proper order is known beforehand.

The last section of Chap. 18 presents an alternative technique for fitting precise data points. This technique, called *spline interpolation*, fits polynomials to data but in a piecewise fashion. As such, it is particularly well-suited for fitting data that is generally smooth but exhibits abrupt local changes.

*Chapter 19* deals with the Fourier transform approach to curve fitting where periodic functions are fit to data. Our emphasis in this section will be on the *fast Fourier transform*. At the end of this chapter, we also include an overview of several software packages and libraries that can be used for curve fitting. These are Mathcad, Excel, MATLAB, and IMSL.

*Chapter 20* is devoted to engineering applications that illustrate the utility of the numerical methods in engineering problem contexts. Examples are drawn from the four major specialty areas of chemical, civil, electrical, and mechanical engineering. In addition, some of the applications illustrate how software packages can be applied for engineering problem solving.

Finally, an epilogue is included at the end of Part Five. It contains a summary of the important formulas and concepts related to curve fitting as well as a discussion of trade-offs among the techniques and suggestions for future study.

### PT5.3.2 Goals and Objectives

**Study Objectives.** After completing Part Five, you should have greatly enhanced your capability to fit curves to data. In general, you should have mastered the techniques, have learned to assess the reliability of the answers, and be capable of choosing the preferred method (or methods) for any particular problem. In addition to these general goals, the specific concepts in Table PT5.3 should be assimilated and mastered.

**Computer Objectives.** You have been provided with software and simple computer algorithms to implement the techniques discussed in Part Five. You may also have access to software packages and libraries. All have utility as learning tools.

The Numerical Methods TOOLKIT software includes polynomial regression. The graphics associated with this software will enable you to easily visualize your problem and the associated mathematical operations. The graphics are a critical part of your assessment of the validity of a regression fit. They also provide guidance regarding the proper order of polynomial regression and the potential dangers of extrapolation. The software is very easy to apply to solve practical problems and can be used to check the results of any computer programs you may develop yourself.

In addition, pseudocode algorithms are provided for most of the other methods in Part Five. This information will allow you to expand your software library to include techniques beyond polynomial regression. For example, you may find it useful from a professional viewpoint to have software to implement multiple linear regression, Newton's interpolating polynomial, cubic spline interpolation, and the fast Fourier transform.

Finally, one of your most important goals should be to master several of the general-purpose software packages that are widely available. In particular, you should become adept at using these tools to implement numerical methods for engineering problem solving.

**TABLE PT5.3** Specific study objectives for Part Five.

1. Understand the fundamental difference between regression and interpolation and realize why confusing the two could lead to serious problems
2. Understand the derivation of linear least-squares regression and be able to assess the reliability of the fit using graphical and quantitative assessments
3. Know how to linearize data by transformation
4. Understand situations where polynomial, multiple, and nonlinear regression are appropriate
5. Be able to recognize general linear models, understand the general matrix formulation of linear least squares, and know how to compute confidence intervals for parameters
6. Understand that there is one and only one polynomial of degree  $n$  or less that passes exactly through  $n + 1$  points
7. Know how to derive the first-order Newton's interpolating polynomial
8. Understand the analogy between Newton's polynomial and the Taylor series expansion and how it relates to the truncation error
9. Recognize that the Newton and Lagrange equations are merely different formulations of the same interpolating polynomial and understand their respective advantages and disadvantages
10. Realize that more accurate results are generally obtained if data used for interpolation is centered around and close to the unknown point
11. Realize that data points do not have to be equally spaced nor in any particular order for either the Newton or Lagrange polynomials
12. Know why equispaced interpolation formulas have utility
13. Recognize the liabilities and risks associated with extrapolation
14. Understand why spline functions have utility for data with local areas of abrupt change
15. Recognize how the Fourier series is used to fit data with periodic functions
16. Understand the difference between the frequency and time domains



# CHAPTER 17

## Least-Squares Regression

Where substantial error is associated with data, polynomial interpolation is inappropriate and may yield unsatisfactory results when used to predict intermediate values. Experimental data is often of this type. For example, Fig. 17.1a shows seven experimentally derived data points exhibiting significant variability. Visual inspection of the data suggests a positive relationship between  $y$  and  $x$ . That is, the overall trend indicates that higher values of  $y$  are associated with higher values of  $x$ . Now, if a sixth-order interpolating polynomial is fitted to this data (Fig. 17.1b), it will pass exactly through all of the points. However, because of the variability in the data, the curve oscillates widely in the interval between the points. In particular, the interpolated values at  $x = 1.5$  and  $x = 6.5$  appear to be well beyond the range suggested by the data.

A more appropriate strategy for such cases is to derive an approximating function that fits the shape or general trend of the data without necessarily matching the individual points. Figure 17.1c illustrates how a straight line can be used to generally characterize the trend of the data without passing through any particular point.

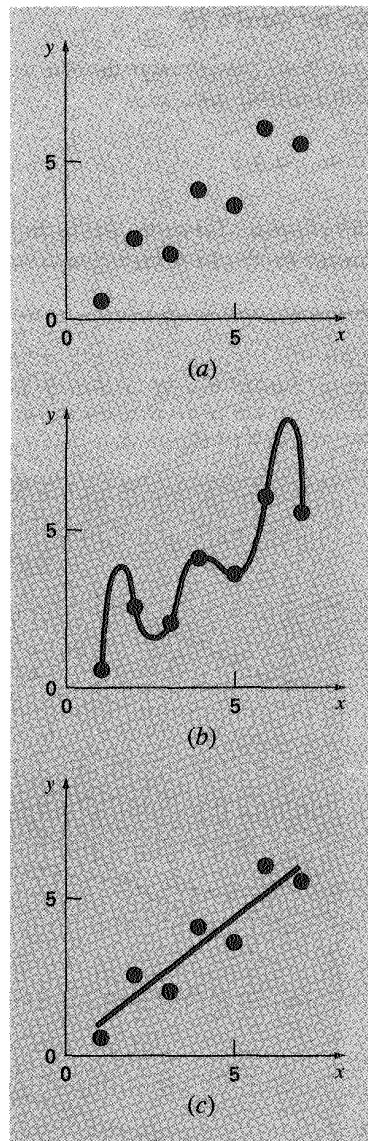
One way to determine the line in Fig. 17.1c is to visually inspect the plotted data and then sketch a “best” line through the points. Although such “eyeball” approaches have commonsense appeal and are valid for “back-of-the-envelope” calculations, they are deficient because they are arbitrary. That is, unless the points define a perfect straight line (in which case, interpolation would be appropriate), different analysts would draw different lines.

To remove this subjectivity, some criterion must be devised to establish a basis for the fit. One way to do this is to derive a curve that minimizes the discrepancy between the data points and the curve. A technique for accomplishing this objective, called *least-squares regression*, will be discussed in the present chapter.

### 17.1 LINEAR REGRESSION

The simplest example of a least-squares approximation is fitting a straight line to a set of paired observations:  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ . The mathematical expression for the straight line is

$$y = a_0 + a_1x + e \quad (17.1)$$

**FIGURE 17.1**

(a) Data exhibiting significant error. (b) Polynomial fit oscillating beyond the range of the data. (c) More satisfactory result using the least-squares fit.

where  $a_0$  and  $a_1$  are coefficients representing the intercept and the slope, respectively, and  $e$  is the error, or residual, between the model and the observations, which can be represented by rearranging Eq. (17.1) as

$$e = y - a_0 - a_1x$$

Thus, the error, or *residual*, is the discrepancy between the true value of  $y$  and the approximate value,  $a_0 + a_1x$ , predicted by the linear equation.

### 17.1.1 Criteria for a "Best" Fit

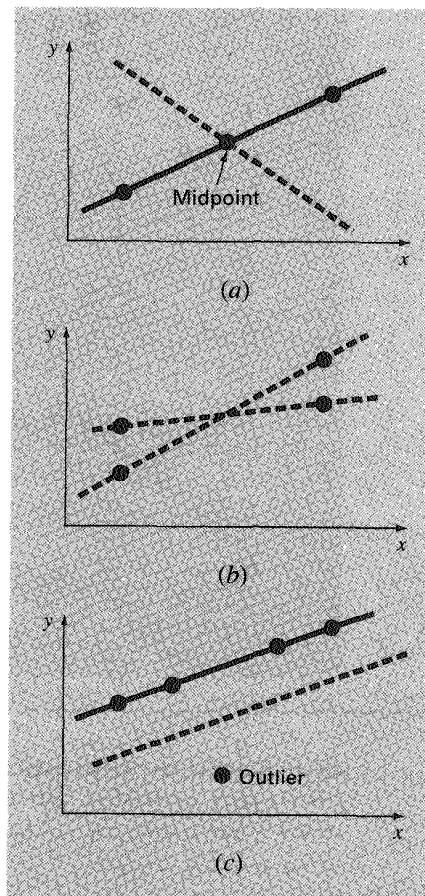
One strategy for fitting a "best?" line through the data would be to minimize the sum of the residual errors for all the available data, as in

$$\sum_{i=1}^n e_i = \sum_{i=1}^n (y_i - a_0 - a_1 x_i) \quad (17.2)$$

where  $n$  = total number of points. However, this is an inadequate criterion, as illustrated by Fig. 17.2a which depicts the fit of a straight line to two points. Obviously, the best fit is

#### FIGURE 17.2

Examples of some criteria for "best fit" that are inadequate for regression: (a) minimizes the sum of the residuals, (b) minimizes the sum of the absolute values of the residuals, and (c) minimizes the maximum error of any individual point.



the line connecting the points. However, any straight line passing through the midpoint of the connecting line (except a perfectly vertical line) results in a minimum value of Eq. (17.2) equal to zero because the errors cancel.

Therefore, another logical criterion might be to minimize the sum of the absolute values of the discrepancies, as in

$$\sum_{i=1}^n |e_i| = \sum_{i=1}^n |y_i - a_0 - a_1 x_i|$$

Figure 17.2*b* demonstrates why this criterion is also inadequate. For the four points shown, any straight line falling within the dashed lines will minimize the absolute value of the sum. Thus, this criterion also does not yield a unique best fit.

A third strategy for fitting a best line is the *minimax* criterion. In this technique, the line is chosen that minimizes the maximum distance that an individual point falls from the line. As depicted in Fig. 17.2*c*, this strategy is ill-suited for regression because it gives undue influence to an outlier, that is, a single point with a large error. It should be noted that the minimax principle is sometimes well-suited for fitting a simple function to a complicated function (Carnahan, Luther, and Wilkes, 1969).

A strategy that overcomes the shortcomings of the aforementioned approaches is to minimize the sum of the squares of the residuals between the measured  $y$  and the  $y$  calculated with the linear model

$$S_r = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_{i,\text{measured}} - y_{i,\text{model}})^2 = \sum_{i=1}^n (y_i - a_0 - a_1 x_i)^2 \quad (17.3)$$

This criterion has a number of advantages, including the fact that it yields a unique line for a given set of data. Before discussing these properties, we will present a technique for determining the values of  $a_0$  and  $a_1$  that minimize Eq. (17.3).

### 17.1.2 Least-Squares Fit of a Straight Line

To determine values for  $a_0$  and  $a_1$ , Eq. (17.3) is differentiated with respect to each coefficient:

$$\begin{aligned} \frac{\partial S_r}{\partial a_0} &= -2 \sum (y_i - a_0 - a_1 x_i) \\ \frac{\partial S_r}{\partial a_1} &= -2 \sum [(y_i - a_0 - a_1 x_i) x_i] \end{aligned}$$

Note that we have simplified the summation symbols; unless otherwise indicated, all summations are from  $i = 1$  to  $n$ . Setting these derivatives equal to zero will result in a minimum  $S_r$ . If this is done, the equations can be expressed as

$$\begin{aligned} 0 &= \sum y_i - \sum a_0 - \sum a_1 x_i \\ 0 &= \sum y_i x_i - \sum a_0 x_i - \sum a_1 x_i^2 \end{aligned}$$

Now, realizing that  $\Sigma a_0 = na_0$ , we can express the equations as a set of two simultaneous linear equations with two unknowns ( $a_0$  and  $a_1$ ):

$$na_0 + \left(\sum x_i\right) a_1 = \sum y_i \quad (17.4)$$

$$\left(\sum x_i\right) a_0 + \left(\sum x_i^2\right) a_1 = \sum x_i y_i \quad (17.5)$$

These are called the *normal equations*. They can be solved simultaneously

$$a_1 = \frac{n \Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n \Sigma x_i^2 - (\Sigma x_i)^2} \quad (17.6)$$

This result can then be used in conjunction with Eq. (17.4) to solve for

$$a_0 = \bar{y} - a_1 \bar{x} \quad (17.7)$$

where  $\bar{y}$  and  $\bar{x}$  are the means of  $y$  and  $x$ , respectively.

#### EXAMPLE 17.1 Linear Regression

**Problem Statement.** Fit a straight line to the  $x$  and  $y$  values in the first two columns of Table 17.1.

**Solution.** The following quantities can be computed:

$$n = 7 \quad \sum x_i y_i = 119.5 \quad \sum x_i^2 = 140$$

$$\sum x_i = 28 \quad \bar{x} = \frac{28}{7} = 4$$

$$\sum y_i = 24 \quad \bar{y} = \frac{24}{7} = 3.428571$$

Using Eqs. (17.6) and (17.7),

$$a_1 = \frac{7(119.5) - 28(24)}{7(140) - (28)^2} = 0.8392857$$

$$a_0 = 3.428571 - 0.8392857(4) = 0.07142857$$

**TABLE 17.1** Computations for an error analysis of the linear fit.

$x_i$	$y_i$	$(y_i - \bar{y})^2$	$(y_i - a_0 - a_1 x_i)^2$
1	0.5	8.5765	0.1687
2	2.5	0.8622	0.5625
3	2.0	2.0408	0.3473
4	4.0	0.3265	0.3265
5	3.5	0.0051	0.5896
6	6.0	6.6122	0.7972
7	5.5	4.2908	0.1993
$\Sigma$	24.0	22.7143	2.9911

Therefore, the least-squares fit is

$$y = 0.07142857 + 0.8392857x$$

The line, along with the data, is shown in Fig. 17.1c.

### 17.1.3 Quantification of Error of Linear Regression

Any line other than the one computed in Example 17.1 results in a larger sum of the squares of the residuals. Thus, the line is unique and in terms of our chosen criterion is a “best” line through the points. A number of additional properties of this fit can be elucidated by examining more closely the way in which residuals were computed. Recall that the sum of the squares is defined as [Eq. (17.3)]

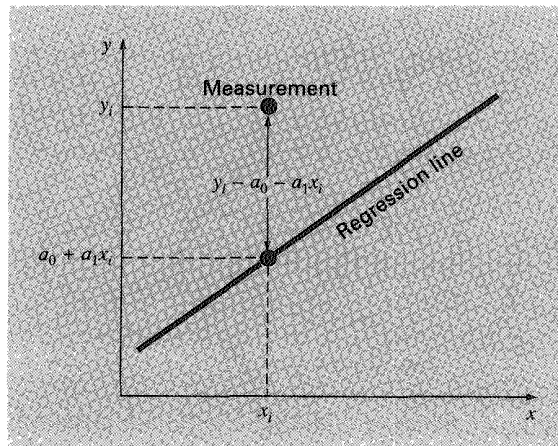
$$S_r = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - a_0 - a_1 x_i)^2 \quad (17.8)$$

Notice the similarity between Eqs. (PT5.3) and (17.8). In the former case, the square of the residual represented the square of the discrepancy between the data and a single estimate of the measure of central tendency—the mean. In Eq. (17.8), the square of the residual represents the square of the vertical distance between the data and another measure of central tendency—the straight line (Fig. 17.3).

The analogy can be extended further for cases where (1) the spread of the points around the line is of similar magnitude along the entire range of the data and (2) the distribution of these points about the line is normal. It can be demonstrated that if these criteria are met, least-squares regression will provide the best (that is, the most likely) estimates of  $a_0$  and  $a_1$  (Draper and Smith, 1981). This is called the *maximum likelihood principle* in

**FIGURE 17.3**

The residual in linear regression represents the vertical distance between a data point and the straight line.



statistics. In addition, if these criteria are met, a “standard deviation” for the regression line can be determined as [compare with Eq. (PT5.2)]

$$s_{y/x} = \sqrt{\frac{S_r}{n-2}} \quad (17.9)$$

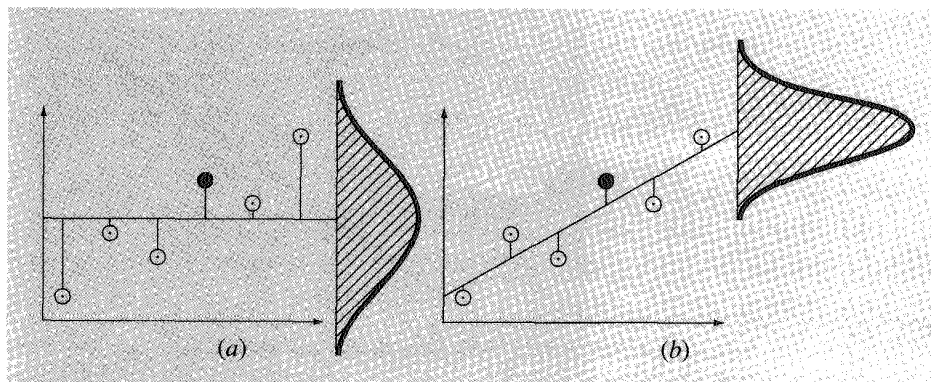
where  $s_{y/x}$  is called the *standard error of the estimate*. The subscript notation “y/x” designates that the error is for a predicted value of y corresponding to a particular value of x. Also, notice that we now divide by  $n - 2$  because two data-derived estimates— $a_0$  and  $a_1$ —were used to compute  $S_r$ ; thus, we have lost two degrees of freedom. As with our discussion of the standard deviation in PT5.2.1, another justification for dividing by  $n - 2$  is that there is no such thing as the “spread of data” around a straight line connecting two points. Thus, for the case where  $n = 2$ , Eq. (17.9) yields a meaningless result of infinity.

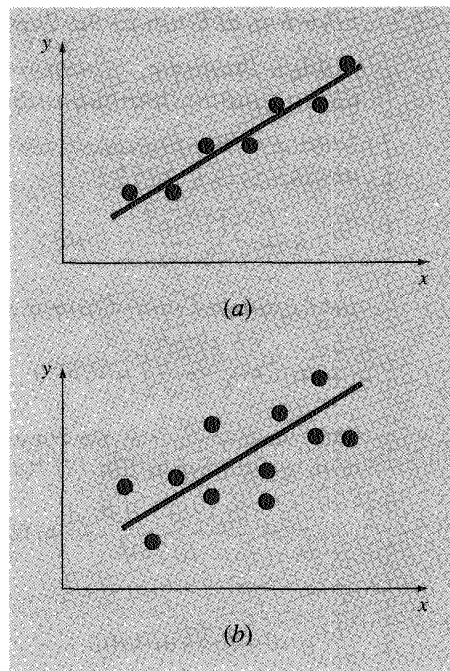
Just as was the case with the standard deviation, the standard error of the estimate quantifies the spread of the data. However,  $s_{y/x}$  quantifies the spread *around the regression line* as shown in Fig. 17.4b in contrast to the original standard deviation  $s_y$  that quantified the spread *around the mean* (Fig. 17.4a).

The above concepts can be used to quantify the “goodness” of our fit. This is particularly useful for comparison of several regressions (Fig. 17.5). To do this, we return to the original data and determine the *total sum of the squares* around the mean for the dependent variable (in our case, y). As was the case for Eq. (PT5.3), this quantity is designated  $S_r$ . This is the magnitude of the residual error associated with the dependent variable prior to regression. After performing the regression, we can compute  $S_r$ , the sum of the squares of the residuals around the regression line. This characterizes the residual error that remains after the regression. It is, therefore, sometimes called the unexplained sum of the squares. The

#### FIGURE 17.4

Regression data showing (a) the spread of the data around the mean of the dependent variable and (b) the spread of the data around the best-fit line. The reduction in the spread in going from (a) to (b), as indicated by the bell-shaped curves at the right, represents the improvement due to linear regression.



**FIGURE 17.5**

Examples of linear regression with (a) small and (b) large residual errors.

difference between the two quantities,  $S_r - S_f$ , quantifies the improvement or error reduction due to describing the data in terms of a straight line rather than as an average value. Because the magnitude of this quantity is scale-dependent, the difference is normalized to  $S_f$  to yield

$$r^2 = \frac{S_f - S_r}{S_f} \quad (17.10)$$

where  $r^2$  is called the *coefficient of determination* and  $r$  is the *correlation coefficient* ( $= \sqrt{r^2}$ ). For a perfect fit,  $S_r = 0$  and  $r = r^2 = 1$ , signifying that the line explains 100 percent of the variability of the data. For  $r = r^2 = 0$ ,  $S_r = S_f$  and the fit represents no improvement. An alternative formulation for  $r$  that is more convenient for computer implementation is

$$r = \frac{n \sum x_i y_i - (\sum x_i)(\sum y_i)}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}} \quad (17.11)$$



## EXAMPLE 17.2 Estimation of Errors for the Linear Least-Squares Fit

**Problem Statement.** Compute the total standard deviation, the standard error of the estimate, and the correlation coefficient for the data in Example 17.1.

**Solution.** The summations are performed and presented in Table 17.1. The standard deviation is [Eq. (PT5.2)]

$$s_y = \sqrt{\frac{22.7143}{7-1}} = 1.9457$$

and the standard error of the estimate is [Eq. (17.9)]

$$s_{y/x} = \sqrt{\frac{2.9911}{7-2}} = 0.7735$$

Thus, because  $s_{y/x} < s_y$ , the linear regression model has merit. The extent of the improvement is quantified by [Eq. (17.10)]

$$r^2 = \frac{22.7143 - 2.9911}{22.7143} = 0.868$$

or

$$r = \sqrt{0.868} = 0.932$$

These results indicate that 86.8 percent of the original uncertainty has been explained by the linear model.

Before proceeding to the computer program for linear regression, a word of caution is in order. Although the correlation coefficient provides a handy measure of goodness-of-fit, you should be careful not to ascribe more meaning to it than is warranted. Just because  $r$  is “close” to 1 does not mean that the fit is necessarily “good.” For example, it is possible to obtain a relatively high value of  $r$  when the underlying relationship between  $y$  and  $x$  is not even linear. Draper and Smith (1981) provide guidance and additional material regarding assessment of results for linear regression. In addition, at the minimum, you should *always* inspect a plot of the data along with your regression curve. As described in the next section, the Numerical Methods TOOLKIT software includes such a capability.

#### 17.1.4 Computer Program for Linear Regression

It is a relatively trivial matter to develop a pseudocode for linear regression (Fig. 17.6). As mentioned above, a plotting option is critical to the effective use and interpretation of regression and is included in the supplementary Numerical Methods TOOLKIT software. In addition, popular software packages like Excel and Mathcad can implement regression and have plotting capabilities. If your computer language has plotting capabilities, we recommend that you expand your program to include a plot of  $y$  versus  $x$  showing both the data and the regression line. The inclusion of the capability will greatly enhance the utility of the program in problem-solving contexts.

```

SUB Regres(x, y, n, a1, a0, syx, r2)

  sumx = 0: sumxy = 0: st = 0
  sumy = 0: sumx2 = 0: sr = 0
  DO i = 1, n
    sumx = sumx + xi
    sumy = sumy + yi
    sumxy = sumxy + xi*yi
    sumx2 = sumx2 + xi*xi
  END DO
  xm = sumx/n
  ym = sumy/n
  a1 = (n*sumxy - sumx*sumy)/(n*sumx2 - sumx*sumx)
  a0 = ym - a1*xm
  DO i = 1, n
    st = st + (yi - ym)2
    sr = sr + (yi - a1*xi - a0)2
  END DO
  syx = (sr/(n - 2))0.5
  r2 = (st - sr)/st

END Regres

```

**FIGURE 17.6**  
Algorithm for linear regression.

### EXAMPLE 17.3 Linear Regression Using the Computer

**Problem Statement.** A user-friendly computer program to implement linear regression is contained in the Numerical Methods TOOLKIT software package associated with this text. We can use this software to solve a hypothesis-testing problem associated with the falling parachutist discussed in Chap. 1. A theoretical mathematical model for the velocity of the parachutist was given as the following [Eq. (1.10)]:

$$v(t) = \frac{gm}{c} (1 - e^{(-c/m)t})$$

where  $v$  = velocity (m/s),  $g$  = gravitational constant (9.8 m/s<sup>2</sup>),  $m$  = mass of the parachutist equal to 68.1 kg, and  $c$  = drag coefficient of 12.5 kg/s. The model predicts the velocity of the parachutist as a function of time, as described in Example 1.1. A plot of the velocity variation was developed in Example 2.1.

An alternative empirical model for the velocity of the parachutist is given by

$$v(t) = \frac{gm}{c} \left( \frac{t}{3.75 + t} \right) \quad (\text{E17.3.1})$$

**TABLE 17.2** Measured and calculated velocities for the falling parachutist.

Time, s	Measured $v$ , m/s (a)	Model-calculated $v$ , m/s [Eq. (1.10)] (b)	Model-calculated $v$ , m/s [Eq. (E17.3.1)] (c)
1	10.00	8.953	11.240
2	16.30	16.405	18.570
3	23.00	22.607	23.729
4	27.50	27.769	27.556
5	31.00	32.065	30.509
6	35.60	35.641	32.855
7	39.00	38.617	34.766
8	41.50	41.095	36.351
9	42.90	43.156	37.687
10	45.00	44.872	38.829
11	46.00	46.301	39.816
12	45.50	47.490	40.678
13	46.00	48.479	41.437
14	49.00	49.303	42.110
15	50.00	49.988	42.712

Suppose that you would like to test and compare the adequacy of these two mathematical models. This might be accomplished by measuring the actual velocity of the parachutist at known values of time and comparing these results with the predicted velocities according to each model.

Such an experimental-data-collection program was implemented, and the results are listed in column (a) of Table 17.2. Computed velocities for each model are listed in columns (b) and (c).

**Solution.** The adequacy of the models can be tested by plotting the model-calculated velocity versus the measured velocity. Linear regression can be used to calculate the slope and the intercept of the plot. This line will have a slope of 1, an intercept of 0, and an  $r^2 = 1$  if the model matches the data perfectly. A significant deviation from these values can be used as an indication of the inadequacy of the model.

Figure 17.7a and b are plots of the line and data for the regressions of columns (b) and (c), respectively, versus column (a). For the first model [Eq. (1.10) as depicted in Fig. 17.7a],

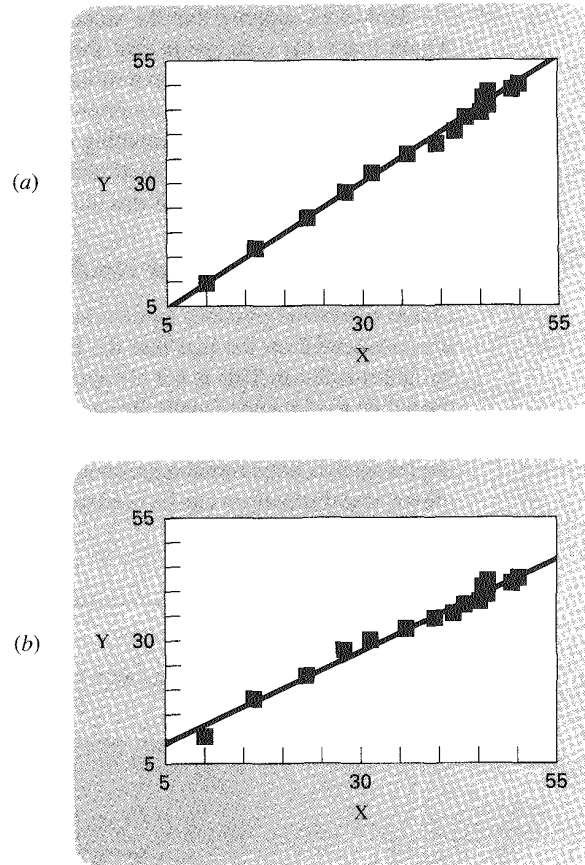
$$v_{\text{model}} = -0.859 + 1.032v_{\text{measure}}$$

and for the second model [Eq. (E17.3.1) as depicted in Fig. 17.7b],

$$v_{\text{model}} = 5.776 + 0.752v_{\text{measure}}$$

These plots indicate that the linear regression between the data and each of the models is highly significant. Both models match the data with a correlation coefficient of greater than 0.99.

However, the model described by Eq. (1.10) conforms to our hypothesis test criteria much better than that described by Eq. (E17.3.1) because the slope and intercept are more

**FIGURE 17.7**

(a) Results using linear regression to compare predictions computed with the theoretical model [Eq. (1.10)] versus measured values. (b) Results using linear regression to compare predictions computed with the empirical model [Eq. (E17.3.1)] versus measured values.

nearly equal to 1 and 0. Thus, although each plot is well described by a straight line, Eq. (1.10) appears to be a better model than Eq. (E17.3.1).

Model testing and selection are common and extremely important activities performed in all fields of engineering. The background material provided in this chapter, together with your software, should allow you to address many practical problems of this type.

There is one shortcoming with the analysis in Example 17.3. The example was unambiguous because the empirical model [Eq. (E17.3.1)] was clearly inferior to Eq. (1.10). Thus, the slope and intercept for the former were so much closer to the desired result of 1 and 0, that it was obvious which model was superior.

However, suppose that the slope were 0.85 and the intercept were 2. Obviously this would make the conclusion that the slope and intercept were 1 and 0 open to debate. Clearly, rather than relying on a subjective judgment, it would be preferable to base such a conclusion on a quantitative criterion.

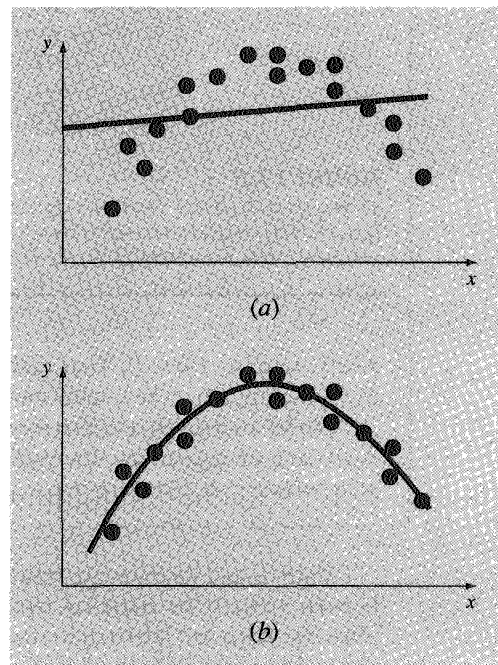
This can be done by computing confidence intervals for the model parameters in the same way that we developed confidence intervals for the mean in Sec. PT5.2.3. We will return to this topic at the end of this chapter.

### 17.1.5 Linearization of Nonlinear Relationships

Linear regression provides a powerful technique for fitting a “best” line to data. However, it is predicated on the fact that the relationship between the dependent and independent variables is linear. This is not always the case, and the first step in any regression analysis should be to plot and visually inspect the data to ascertain whether a linear model applies. For example, Fig. 17.8 shows some data that is obviously curvilinear. In some cases, techniques such as polynomial regression, which is described in Sec. 17.2, are appropriate. For others, transformations can be used to express the data in a form that is compatible with linear regression.

**FIGURE 17.8**

(a) Data that is ill-suited for linear least-squares regression. (b) Indication that a parabola is preferable.



One example is the *exponential model*

$$y = a_1 e^{b_1 x} \quad (17.12)$$

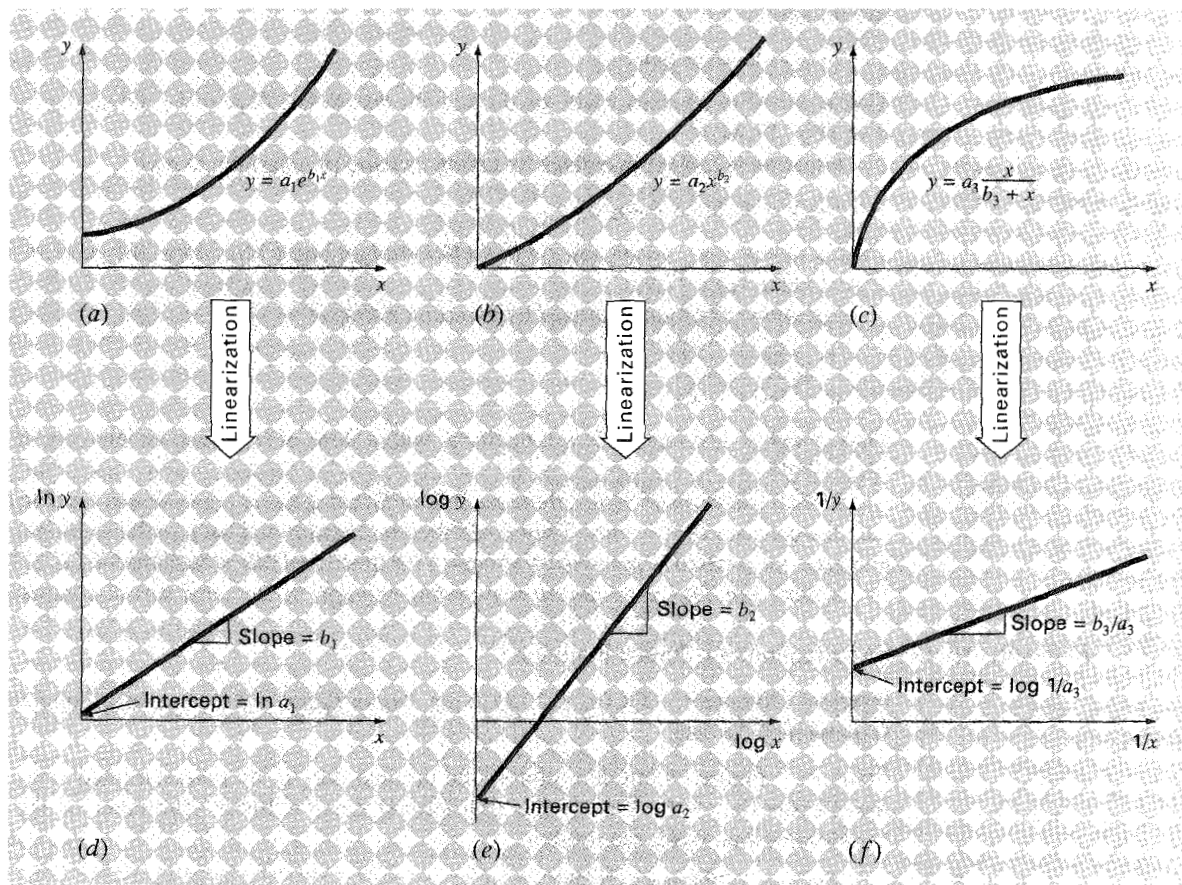
where  $a_1$  and  $b_1$  are constants. This model is used in many fields of engineering to characterize quantities that increase (positive  $b_1$ ) or decrease (negative  $b_1$ ) at a rate that is directly proportional to their own magnitude. For example, population growth or radioactive decay can exhibit such behavior. As depicted in Fig. 17.9a, the equation represents a nonlinear relationship (for  $b_1 \neq 0$ ) between  $y$  and  $x$ .

Another example of a nonlinear model is the simple power equation

$$y = a_2 x^{b_2} \quad (17.13)$$

**FIGURE 17.9**

(a) The exponential equation, (b) the power equation, and (c) the saturation-growth-rate equation. Parts (d), (e), and (f) are linearized versions of these equations that result from simple transformations.



where  $a_2$  and  $b_2$  are constant coefficients. This model has wide applicability in all fields of engineering. As depicted in Fig. 17.9*b*, the equation (for  $b_2 \neq 0$  or 1) is nonlinear.

A third example of a nonlinear model is the saturation-growth-rate equation [recall Eq. (E17.3.1)]

$$y = a_3 \frac{x}{b_3 + x} \quad (17.14)$$

where  $a_3$  and  $b_3$  are constant coefficients. This model, which is particularly well-suited for characterizing population growth rate under limiting conditions, also represents a nonlinear relationship between  $y$  and  $x$  (Fig. 17.9*c*) that levels off, or “saturates,” as  $x$  increases.

Nonlinear regression techniques are available to fit these equations to experimental data directly. (Note that we will discuss nonlinear regression in Sec. 17.5.) However, a simpler alternative is to use mathematical manipulations to transform the equations into a linear form. Then, simple linear regression can be employed to fit the equations to data.

For example, Eq. (17.12) can be linearized by taking its natural logarithm to yield

$$\ln y = \ln a_1 + b_1 x \ln e$$

But because  $\ln e = 1$ ,

$$\ln y = \ln a_1 + b_1 x \quad (17.15)$$

Thus a plot of  $\ln y$  versus  $x$  will yield a straight line with a slope of  $b_1$  and an intercept of  $\ln a_1$  (Fig. 17.9*d*).

Equation (17.14) is linearized by taking its base-10 logarithm to give

$$\log y = b_2 \log x + \log a_2 \quad (17.16)$$

Thus, a plot of  $\log y$  versus  $\log x$  will yield a straight line with a slope of  $b_2$  and an intercept of  $\log a_2$  (Fig. 17.9*e*).

Equation (17.14) is linearized by inverting it to give

$$\frac{1}{y} = \frac{b_3}{a_3} \frac{1}{x} + \frac{1}{a_3} \quad (17.17)$$

Thus, a plot of  $1/y$  versus  $1/x$  will be linear, with a slope of  $b_3/a_3$  and an intercept of  $1/a_3$  (Fig. 17.9*f*).

In their transformed forms, these models are fit using linear regression in order to evaluate the constant coefficients. They could then be transformed back to their original state and used for predictive purposes. Example 17.4 illustrates this procedure for Eq. (17.13). In addition, Sec. 20.1 provides an engineering example of the same sort of computation.

#### EXAMPLE 17.4 Linearization of a Power Equation

**Problem Statement.** Fit Eq. (17.13) to the data in Table 17.3 using a logarithmic transformation of the data.

**Solution.** Figure 17.10*a* is a plot of the original data in its untransformed state. Figure 17.10*b* shows the plot of the transformed data. A linear regression of the log-transformed data yields the result

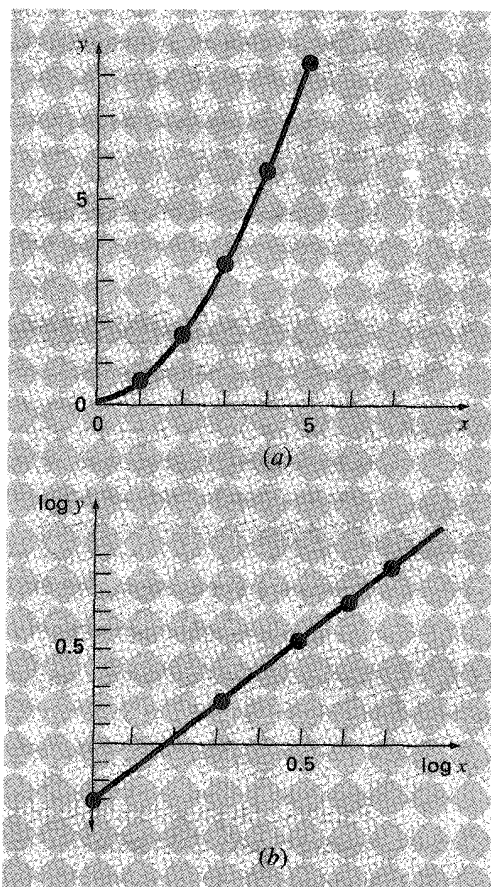
$$\log y = 1.75 \log x - 0.300$$

**TABLE 17.3** Data to be fit to the power equation.

$x$	$y$	$\log x$	$\log y$
1	0.5	0	-0.301
2	1.7	0.301	0.226
3	3.4	0.477	0.534
4	5.7	0.602	0.753
5	8.4	0.699	0.922

**FIGURE 17.10**

(a) Plot of untransformed data with the power equation that fits the data. (b) Plot of transformed data used to determine the coefficients of the power equation.





Thus, the intercept,  $\log a_2$ , equals  $-0.300$ , and therefore, by taking the antilogarithm,  $a_2 = 10^{-0.3} = 0.5$ . The slope is  $b_2 = 1.75$ . Consequently, the power equation is

$$y = 0.5x^{1.75}$$

This curve, as plotted in Fig. 17.10a, indicates a good fit.

### 17.1.6 General Comments on Linear Regression

Before proceeding to curvilinear and multiple linear regression, we must emphasize the introductory nature of the foregoing material on linear regression. We have focused on the simple derivation and practical use of equations to fit data. You should be cognizant of the fact that there are theoretical aspects of regression that are of practical importance but are beyond the scope of this book. For example, some statistical assumptions that are inherent in the linear least-squares procedures are

1. Each  $x$  has a fixed value; it is not random and is known without error.
2. The  $y$  values are independent random variables and all have the same variance.
3. The  $y$  values for a given  $x$  must be normally distributed.

Such assumptions are relevant to the proper derivation and use of regression. For example, the first assumption means that (1) the  $x$  values must be error-free and (2) the regression of  $y$  versus  $x$  is not the same as  $x$  versus  $y$  (try Prob. 17.4 at the end of the chapter). You are urged to consult other references such as Draper and Smith (1981) to appreciate aspects and nuances of regression that are beyond the scope of this book.

## 17.2 POLYNOMIAL REGRESSION

In Sec. 17.1, a procedure was developed to derive the equation of a straight line using the least-squares criterion. Some engineering data, although exhibiting a marked pattern such as seen in Fig. 17.8, is poorly represented by a straight line. For these cases, a curve would be better suited to fit the data. As discussed in the previous section, one method to accomplish this objective is to use transformations. Another alternative is to fit polynomials to the data using *polynomial regression*.

The least-squares procedure can be readily extended to fit the data to a higher-order polynomial. For example, suppose that we fit a second-order polynomial or quadratic:

$$y = a_0 + a_1x + a_2x^2 + e$$

For this case the sum of the squares of the residuals is [compare with Eq. (17.3)]

$$S_r = \sum_{i=1}^n (y_i - a_0 - a_1x_i - a_2x_i^2)^2 \quad (17.18)$$

Following the procedure of the previous section, we take the derivative of Eq. (17.18) with respect to each of the unknown coefficients of the polynomial, as in

$$\frac{\partial S_r}{\partial a_0} = -2 \sum (y_i - a_0 - a_1x_i - a_2x_i^2)$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum x_i (y_i - a_0 - a_1 x_i - a_2 x_i^2)$$

$$\frac{\partial S_r}{\partial a_2} = -2 \sum x_i^2 (y_i - a_0 - a_1 x_i - a_2 x_i^2)$$

These equations can be set equal to zero and rearranged to develop the following set of normal equations:

$$\begin{aligned} (n)a_0 + \left(\sum x_i\right)a_1 + \left(\sum x_i^2\right)a_2 &= \sum y_i \\ \left(\sum x_i\right)a_0 + \left(\sum x_i^2\right)a_1 + \left(\sum x_i^3\right)a_2 &= \sum x_i y_i \\ \left(\sum x_i^2\right)a_0 + \left(\sum x_i^3\right)a_1 + \left(\sum x_i^4\right)a_2 &= \sum x_i^2 y_i \end{aligned} \quad (17.19)$$

where all summations are from  $i = 1$  through  $n$ . Note that the above three equations are linear and have three unknowns:  $a_0$ ,  $a_1$ , and  $a_2$ . The coefficients of the unknowns can be calculated directly from the observed data.

For this case, we see that the problem of determining a least-squares second-order polynomial is equivalent to solving a system of three simultaneous linear equations. Techniques to solve such equations were discussed in Part Three.

The two-dimensional case can be easily extended to an  $m$ th-order polynomial as

$$y = a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m + e$$

The foregoing analysis can be easily extended to this more general case. Thus, we can recognize that determining the coefficients of an  $m$ th-order polynomial is equivalent to solving a system of  $m + 1$  simultaneous linear equations. For this case, the standard error is formulated as

$$s_{y/x} = \sqrt{\frac{S_r}{n - (m + 1)}} \quad (17.20)$$

This quantity is divided by  $n - (m + 1)$  because  $(m + 1)$  data-derived coefficients— $a_0$ ,  $a_1$ , . . . ,  $a_m$ —were used to compute  $S_r$ ; thus, we have lost  $m + 1$  degrees of freedom. In addition to the standard error, a coefficient of determination can also be computed for polynomial regression with Eq. (17.10).

### EXAMPLE 17.5 Polynomial Regression

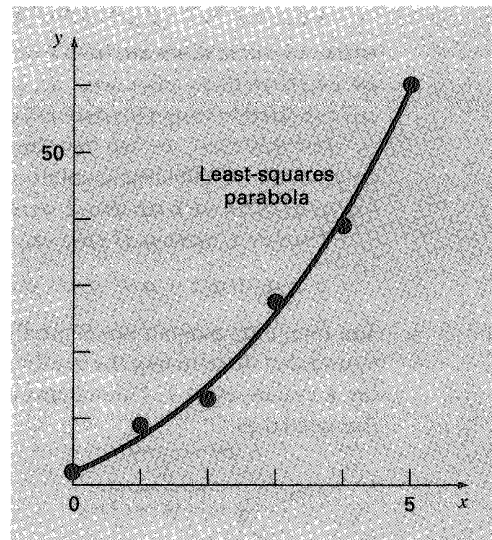
**Problem Statement.** Fit a second-order polynomial to the data in the first two columns of Table 17.4.

**Solution.** From the given data,

$$\begin{array}{lll} m = 2 & \sum x_i = 15 & \sum x_i^4 = 979 \\ n = 6 & \sum y_i = 152.6 & \sum x_i y_i = 585.6 \\ \bar{x} = 2.5 & \sum x_i^2 = 55 & \sum x_i^2 y_i = 2488.8 \\ \bar{y} = 25.433 & \sum x_i^3 = 225 & \end{array}$$

**TABLE 17.4** Computations for an error analysis of the quadratic least-squares fit.

$x_i$	$y_i$	$(y_i - \bar{y})^2$	$(y_i - a_0 - a_1x_i - a_2x_i^2)$
0	2.1	544.44	0.14332
1	7.7	314.47	1.00286
2	13.6	140.03	1.08158
3	27.2	3.12	0.80491
4	40.9	239.22	0.61951
5	61.1	1272.11	0.09439
$\Sigma$	152.6	2513.39	3.74657

**FIGURE 17.11**

Fit of a second-order polynomial.

Therefore, the simultaneous linear equations are

$$\begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 225 & 979 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 152.6 \\ 585.6 \\ 2488.8 \end{Bmatrix}$$

Solving these equations through a technique such as Gauss elimination gives  $a_0 = 2.47857$ ,  $a_1 = 2.35929$ , and  $a_2 = 1.86071$ . Therefore, the least-squares quadratic equation for this case is

$$y = 2.47857 + 2.35929x + 1.86071x^2$$

The standard error of the estimate based on the regression polynomial is [Eq. (17.20)]

$$s_{y/x} = \sqrt{\frac{3.74657}{6-3}} = 1.12$$

The coefficient of determination is

$$r^2 = \frac{2513.39 - 3.74657}{2513.39} = 0.99851$$

and the correlation coefficient is  $r = 0.99925$ .

These results indicate that 99.851 percent of the original uncertainty has been explained by the model. This result supports the conclusion that the quadratic equation represents an excellent fit, as is also evident from Fig. 17.11.

### 17.2.1 Algorithm for Polynomial Regression

An algorithm for polynomial regression is delineated in Fig. 17.12. Note that the primary task is the generation of the coefficients of the normal equations [Eq. (17.19)]. (Pseudocode for accomplishing this is presented in Fig. 17.13.) Then, techniques from Part Three can be applied to solve these simultaneous equations for the coefficients.

A potential problem associated with implementing polynomial regression on the computer is that the normal equations are sometimes ill-conditioned. This is particularly true for higher-order versions. For these cases, the computed coefficients may be highly susceptible to round-off error, and consequently, the results can be inaccurate. Among other things, this problem is related to the structure of the normal equations and to the fact that for higher-order polynomials the normal equations can have very large and very small coefficients. This is because the coefficients are summations of the data raised to powers.

Although the strategies for mitigating round-off error discussed in Part Three, such as pivoting, can help to partially remedy this problem, a simpler alternative is to use a computer with higher precision. Fortunately, most practical problems are limited to lower-order polynomials for which round-off is usually negligible. In situations where higher-order versions are required, other alternatives are available for certain types of data. However, these techniques (such as orthogonal polynomials) are beyond the scope of this book. The reader should consult texts on regression, such as Draper and Smith (1981), for additional information regarding the problem and possible alternatives.

#### FIGURE 17.12

Algorithm for implementation of polynomial and multiple linear regression.

- Step 1:** Input order of polynomial to be fit,  $m$ .
- Step 2:** Input number of data points,  $n$ .
- Step 3:** If  $n < m + 1$ , print out an error message that regression is impossible and terminate the process. If  $n \geq m + 1$ , continue.
- Step 4:** Compute the elements of the normal equation in the form of an augmented matrix.
- Step 5:** Solve the augmented matrix for the coefficients  $a_0, a_1, a_2, \dots, a_m$ , using an elimination method.
- Step 6:** Print out the coefficients.

```

DO i = 1, order + 1
  DO j = 1, i
    k = i + j - 2
    sum = 0
    DO l = 1, n
      sum = sum + xlk
    END DO
    ai,j = sum
    aj,i = sum
  END DO
  sum = 0
  DO l = 1, n
    sum = sum + yl · xli-1
  END DO
  ai, order+2 = sum
END DO

```

**FIGURE 17.13**

Pseudocode to assemble the elements of the normal equations for polynomial regression.

### EXAMPLE 17.6 Polynomial Regression Using the Computer

**Problem Statement.** A user-friendly computer program to implement polynomial regression is contained in the Numerical Methods TOOLKIT software associated with the text. We can use this software to fit polynomials to the following data:

x	2	4	5	6	6	7	9	1	0.5	7.5
y	6	2	3	7	8	8	1	5	3	7

**Solution.** Press the Fit Data with Curve button on the TOOLKIT main menu to obtain a blank screen similar to Fig. 17.14. This screen contains spaces for the input and output information needed to fit data with an  $m$ th-order least-squares regression polynomial.

The first step is to click the Input X vs Y Values table and enter up to 100 pairs of values for X and Y. Next you might decide to plot the data alone before making a decision concerning the order of the polynomial. This is done using a procedure similar to that described in Example 2.1. Inspection of the data shows two peaks and suggests that a polynomial of at least order 4 would be appropriate. For our example, we will first try a fifth-order polynomial. Simply enter a value of 5 for the order of the polynomial and the plot parameters in the Input Parameters table and click the red Calc and Plot buttons (in the process changing the buttons to black) to produce Fig. 17.14. The issue of determining the best order can be explored by examining how the standard error varies as a func-

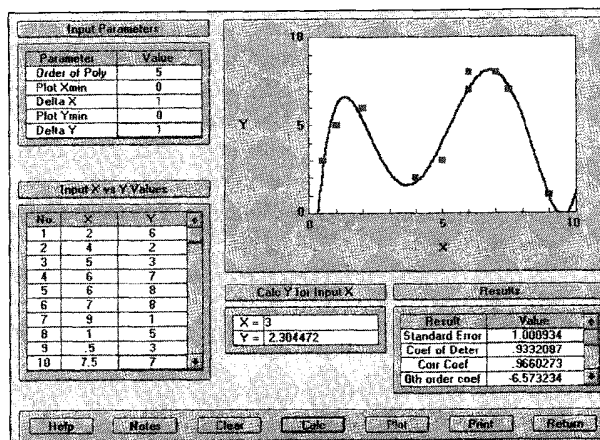


FIGURE 17.14

Screen from Numerical Methods TOOLKIT for fifth-order polynomial regression.

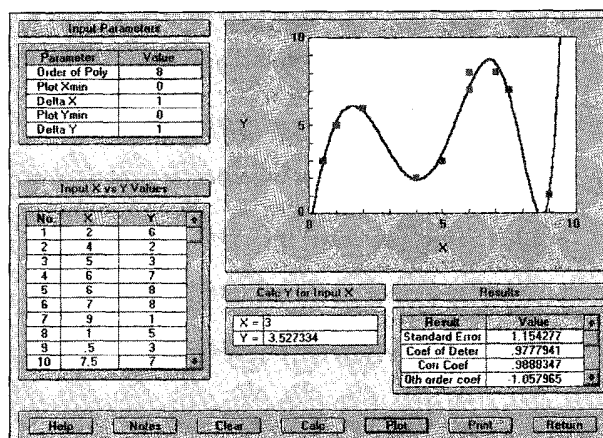


FIGURE 17.15

Plot for eighth-order polynomial regression.

tion of the order of the regression. The results for various order regression fits is tabulated below:

Order	1	2	3	4	5	6	7	8
Standard Error	2.71	2.60	2.34	1.38	1.00	1.12	1.17	1.15

Note that the standard error drops dramatically from order 3 to 4 and reaches a minimum for the order-5 polynomial. This suggests that not much is gained by expanding the computational effort to perform higher than fifth-order regression.

Figure 17.15 shows the plots for the eighth-order case. For this case, overshoot begins to become a problem in a manner similar to higher-order interpolation (we'll discuss this phenomenon in detail in the next chapter). Figure 17.15 shows that the eighth-order polynomial produces negative Y values for X values between 8 and 9. Also observe from both Figs. 17.14 and 17.15 that while the regression curves follow the trend of the data, it is highly inappropriate to extrapolate for values Y beyond the range of the data for X.

Interpolation can be performed by entering a value for X in the Calc Y for Input X table. For example at  $X = 3$ ,  $Y = 2.304472$  as calculated from the fifth-order polynomial (Fig. 17.14).

Finally, take a look at the Results table on the lower right. The first three results are statistical summaries of the regression: Standard Error, Coefficient of Determination, and Correlation Coefficient. Note how these values change for different orders of regression. The scroll bar on the Results table is used to observe the actual coefficients of the regression polynomial. Again, these values change with different orders.

### 17.3 MULTIPLE LINEAR REGRESSION

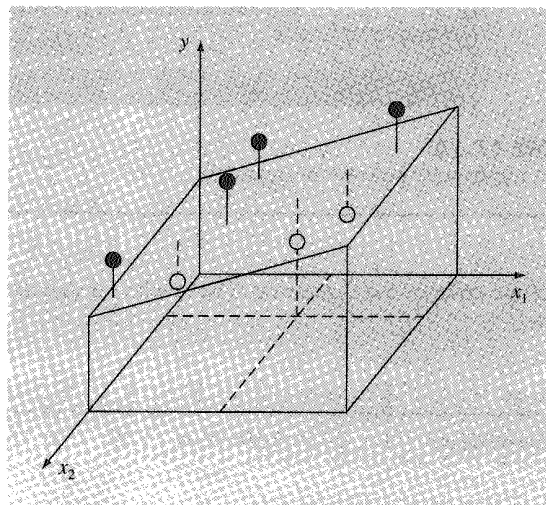
A useful extension of linear regression is the case where  $y$  is a linear function of two or more independent variables. For example,  $y$  might be a linear function of  $x_1$  and  $x_2$ , as in

$$y = a_0 + a_1x_1 + a_2x_2 + e$$

Such an equation is particularly useful when fitting experimental data where the variable being studied is often a function of two other variables. For this two-dimensional case, the regression "line" becomes a "plane" (Fig. 17.16).

**FIGURE 17.16**

Graphical depiction of multiple linear regression where  $y$  is a linear function of  $x_1$  and  $x_2$ .



As with the previous cases, the “best” values of the coefficients are determined by setting up the sum of the squares of the residuals,

$$S_r = \sum_{i=1}^n (y_i - a_0 - a_1x_{1i} - a_2x_{2i})^2 \quad (17.21)$$

and differentiating with respect to each of the unknown coefficients,

$$\begin{aligned} \frac{\partial S_r}{\partial a_0} &= -2 \sum (y_i - a_0 - a_1x_{1i} - a_2x_{2i}) \\ \frac{\partial S_r}{\partial a_1} &= -2 \sum x_{1i} (y_i - a_0 - a_1x_{1i} - a_2x_{2i}) \\ \frac{\partial S_r}{\partial a_2} &= -2 \sum x_{2i} (y_i - a_0 - a_1x_{1i} - a_2x_{2i}) \end{aligned}$$

The coefficients yielding the minimum sum of the squares of the residuals are obtained by setting the partial derivatives equal to zero and expressing the result in matrix form as

$$\begin{bmatrix} n & \Sigma x_{1i} & \Sigma x_{2i} \\ \Sigma x_{1i} & \Sigma x_{1i}^2 & \Sigma x_{1i}x_{2i} \\ \Sigma x_{2i} & \Sigma x_{1i}x_{2i} & \Sigma x_{2i}^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \Sigma y_i \\ \Sigma x_{1i}y_i \\ \Sigma x_{2i}y_i \end{bmatrix} \quad (17.22)$$

#### EXAMPLE 17.7 Multiple Linear Regression

**Problem Statement.** The following data was calculated from the equation  $y = 5 + 4x_1 - 3x_2$ :

$x_1$	$x_2$	$y$
0	0	5
2	1	10
2.5	2	9
1	3	0
4	6	3
7	2	27

Use multiple linear regression to fit this data.

**Solution.** The summations required to develop Eq. (17.22) are computed in Table 17.5. The result is

$$\begin{bmatrix} 6 & 16.5 & 14 \\ 16.5 & 76.25 & 48 \\ 14 & 48 & 54 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 54 \\ 243.5 \\ 100 \end{bmatrix}$$

which can be solved using a method such as Gauss elimination for

$$a_0 = 5 \quad a_1 = 4 \quad a_2 = -3$$

which is consistent with the original equation from which the data was derived.



**TABLE 17.5** Computations required to develop the normal equations for Example 17.7.

$y$	$x_1$	$x_2$	$x_1^2$	$x_2^2$	$x_1x_2$	$x_1y$	$x_2y$	
5	0	0	0	0	0	0	0	
10	2	1	4	1	2	20	10	
9	2.5	2	6.25	4	5	22.5	18	
0	1	3	1	9	3	0	0	
3	4	6	16	36	24	12	18	
27	7	2	49	4	14	189	54	
$\Sigma$	54	16.5	14	76.25	54	48	243.5	100

The foregoing two-dimensional case can be easily extended to  $m$  dimensions, as in

$$y = a_0 + a_1x_1 + a_2x_2 + \cdots + a_mx_m + e$$

where the standard error is formulated as

$$s_{y/x} = \sqrt{\frac{S_r}{n - (m + 1)}}$$

and the coefficient of determination is computed as in Eq. (17.10). An algorithm to set up the normal equations is listed in Fig. 17.17.

Although there may be certain cases where a variable is linearly related to two or more other variables, multiple linear regression has additional utility in the derivation of power equations of the general form

$$y = a_0x_1^{a_1}x_2^{a_2}\cdots x_m^{a_m}$$

**FIGURE 17.17**

Pseudocode to assemble the elements of the normal equations for multiple regression. Note that aside from storing the independent variables in  $x_{1,i}$ ,  $x_{2,i}$ , etc., 1's must be stored in  $x_{0,i}$  for this algorithm to work.

```

DO i = 1, order + 1
  DO j = 1, i
    sum = 0
    DO l = 1, n
      sum = sum + xi-1,l · xj-1,l
    END DO
    ai,j = sum
    aj,i = sum
  END DO
  sum = 0
  DO l = 1, n
    sum = sum + yl · xi-1,l
  END DO
  ai, order+2 = sum
END DO

```

Such equations are extremely useful when fitting experimental data. To use multiple linear regression, the equation is transformed by taking its logarithm to yield

$$\log y = \log a_0 + a_1 \log x_1 + a_2 \log x_2 + \cdots + a_m \log x_m$$

This transformation is similar in spirit to the one used in Sec. 17.1.5 and Example 17.4 to fit a power equation when  $y$  was a function of a single variable  $x$ . Section 20.4 provides an example of such an application for two independent variables.

## 17.4 GENERAL LINEAR LEAST SQUARES

To this point, we have focused on the mechanics of obtaining least-squares fits of some simple functions to data. Before turning to nonlinear regression, there are several issues that we would like to discuss to enrich your understanding of the preceding material.

### 17.4.1 General Matrix Formulation for Linear Least Squares

In the preceding pages, we have introduced three types of regression: simple linear, polynomial, and multiple linear. In fact, all three belong to the following general linear least-squares model:

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + \cdots + a_m z_m + e \quad (17.23)$$

where  $z_0, z_1, \dots, z_m$  are  $m + 1$  different functions. It can easily be seen how simple and multiple linear regression fall within this model—that is,  $z_0 = 1, z_1 = x_1, z_2 = x_2, \dots, z_m = x_m$ . Further, polynomial regression is also included if the  $z$ 's are simple monomials as in  $z_0 = x^0 = 1, z_1 = x, z_2 = x^2, \dots, z_m = x^m$ .

Note that the terminology “linear” refers only to the model’s dependence on its parameters—that is, the  $a$ 's. As in the case of polynomial regression, the functions themselves can be highly nonlinear. For example, the  $z$ 's can be sinusoids, as in

$$y = a_0 + a_1 \cos(\omega t) + a_2 \sin(\omega t)$$

Such a format is the basis of Fourier analysis described in Chap. 19.

On the other hand, a simple looking model like

$$f(x) = a_0 (1 - e^{-a_1 x})$$

is truly nonlinear because it cannot be manipulated into the format of Eq. (17.23). We will turn to such models at the end of this chapter.

For the time being, Eq. (17.23) can be expressed in matrix notation as

$$\{Y\} = [Z]\{A\} + \{E\} \quad (17.24)$$

where  $[Z]$  is a matrix of the calculated values of the  $z$  functions at the measured values of the independent variables,

$$[Z] = \begin{bmatrix} z_{01} & z_{11} & \cdots & z_{m1} \\ z_{02} & z_{12} & \cdots & z_{m2} \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ z_{0n} & z_{1n} & \cdots & z_{mn} \end{bmatrix}$$

where  $m$  is the number of variables in the model and  $n$  is the number of data points. Because  $n \geq m + 1$ , you should recognize that most of the time  $[Z]$  is not a square matrix.

The column vector  $\{Y\}$  contains the observed values of the dependent variable

$$\{Y\}^T = [y_1 \quad y_2 \quad \cdots \quad y_n]$$

The column vector  $\{A\}$  contains the unknown coefficients

$$\{A\}^T = [a_0 \quad a_1 \quad \cdots \quad a_m]$$

and the column vector  $\{E\}$  contains the residuals

$$\{E\}^T = [e_1 \quad e_2 \quad \cdots \quad e_n]$$

As was done throughout this chapter, the sum of the squares of the residuals for this model can be defined as

$$S_r = \sum_{i=1}^n \left( y_i - \sum_{j=0}^m a_j z_{ji} \right)^2$$

This quantity can be minimized by taking its partial derivative with respect to each of the coefficients and setting the resulting equation equal to zero. The outcome of this process is the normal equations that can be expressed concisely in matrix form as

$$[Z]^T [Z] \{A\} = [Z]^T \{Y\} \quad (17.25)$$

It can be shown that Eq. (17.25) is, in fact, equivalent to the normal equations developed previously for simple linear, polynomial, and multiple linear regression.

Our primary motivation for the foregoing has been to illustrate the unity among the three approaches and to show how they can all be expressed simply in the same matrix notation. It also sets the stage for the next section where we will gain some insights into the preferred strategies for solving Eq. (17.25). The matrix notation will also have relevance when we turn to nonlinear regression in the last section of this chapter.

### 17.4.2 Solution Techniques

In previous discussions in this chapter, we have glossed over the issue of the specific numerical techniques to solve the normal equations. Now that we have established the unity among the various models, we can explore this question in more detail.

First, it should be clear that Gauss-Seidel cannot be employed because the normal equations are not diagonally dominant. We are thus left with the elimination methods. For the present purposes, we can divide these techniques into three categories: (1) *LU* decomposition methods including Gauss elimination, (2) Cholesky's method, and (3) matrix inversion approaches. There are obviously overlaps involved in this breakdown. For example, Cholesky's method is, in fact, an *LU* decomposition, and all the approaches can be formulated so that they can generate the matrix inverse. However, this breakdown has merit in that each category offers benefits regarding the solution of the normal equations.

***LU* Decomposition.** If you are merely interested in applying a least-squares fit for the case where the appropriate model is known a priori, any of the *LU* decomposition approaches described in Chap. 9 is perfectly acceptable. In fact, the non-*LU*-decomposition formulation of Gauss elimination can also be employed. It is a relatively straightforward

programming task to incorporate any of these into an algorithm for linear least squares. In fact, if a modular approach has been followed, it is almost trivial.

**Cholesky's Method.** Cholesky's decomposition algorithm has several advantages with regard to the solution of the general linear regression problem. First, it is expressly designed for solving symmetric matrices like the normal equations. Thus, it is fast and requires less storage space to solve such systems. Second, it is ideally suited for cases where the order of the model [that is, the value of  $m$  in Eq. (17.23)] is not known beforehand (see Ralston and Rabinowitz, 1978). A case in point would be polynomial regression. For this case, we might not know a priori whether a linear, quadratic, cubic, or higher-order polynomial is the "best" model to describe our data. Because of the way in which both the normal equations are constructed and the Cholesky algorithm proceeds (Fig. 11.3), we can develop successively higher-order models in an extremely efficient manner. At each step we could examine the residual sum of the squares error (and a plot!) to examine whether the inclusion of higher-order terms significantly improves the fit.

The analogous situation for multiple linear regression occurs when independent variables are added to the model one at a time. Suppose that the dependent variable of interest is a function of a number of independent variables: say, temperature, moisture content, pressure, etc. We could first perform a linear regression with temperature and compute a residual error. Next, we could include moisture content by performing a two-variable multiple regression and see whether the additional variable results in an improved fit. Cholesky's method makes this process efficient because the decomposition of the linear model would merely be supplemented to incorporate a new variable.

**Matrix Inverse Approaches.** From Eq. (PT3.6), recall that the matrix inverse can be employed to solve Eq. (17.25), as in

$$\{A\} = [Z]^T [Z]^{-1} \{Z\}^T \{Y\} \quad (17.26)$$

Each of the elimination methods can be used to determine the inverse and, thus, can be used to implement Eq. (17.26). However, as we have learned in Part Three, this is an inefficient approach for solving a set of simultaneous equations. Thus, if we were merely interested in solving for the regression coefficients, it is preferable to employ an  $LU$  decomposition approach without inversion. However, from a statistical perspective, there are a number of reasons why we might be interested in obtaining the inverse and examining its coefficients. These reasons will be discussed next.

### 17.4.3 Statistical Aspects of Least-Squares Theory

In Sec. PT5.2.1, we reviewed a number of descriptive statistics that can be used to describe a sample. These included the arithmetic mean, the standard deviation, and the variance.

Aside from yielding a solution for the regression coefficients, the matrix formulation of Eq. (17.26) provides estimates of their statistics. It can be shown (Draper and Smith, 1981) that the diagonal and off-diagonal terms of the matrix  $[Z]^T [Z]^{-1}$  give, respectively, the variances and the covariances<sup>1</sup> of the  $a$ 's. If the diagonal elements of

<sup>1</sup>The covariance is a statistic that measures the dependency of one variable on another. Thus,  $\text{cov}(x, y)$  indicates the dependency of  $x$  and  $y$ . For example,  $\text{cov}(x, y) = 0$  would indicate that  $x$  and  $y$  are totally independent.

$[[Z]^T[Z]]^{-1}$  are designated as  $z_{ii}^{-1}$ , then

$$\text{var}(a_{i-1}) = z_{ii}^{-1} s_{y/x}^2 \quad (17.27)$$

and

$$\text{cov}(a_{i-1}, a_j) = z_{i-1,j}^{-1} s_{y/x}^2 \quad (17.28)$$

These statistics have a number of important applications. For our present purposes, we will illustrate how they can be used to develop confidence intervals for the intercept and slope.

Using an approach similar to that in Sec. PT5.2.3, it can be shown that lower and upper bounds on the intercept can be formulated as (see Milton and Arnold 1995 for details)

$$L = a_0 - t_{\alpha/2, n-2} s(a_0) \quad U = a_0 + t_{\alpha/2, n-2} s(a_0) \quad (17.29)$$

where  $s(a_j)$  = the standard error of coefficient  $a_j = \sqrt{\text{var}(a_j)}$ . In a similar manner, lower and upper bounds on the slope can be formulated as

$$L = a_1 - t_{\alpha/2, n-2} s(a_1) \quad U = a_1 + t_{\alpha/2, n-2} s(a_1) \quad (17.30)$$

The following example illustrates how these intervals can be used to make quantitative inferences related to linear regression.

#### EXAMPLE 17.8 Confidence Intervals for Linear Regression

**Problem Statement.** In Example 17.3, we used regression to develop the following relationship between measurements and model predictions:

$$y = -0.859 + 1.032x$$

where  $y$  = the model predictions and  $x$  = the measurements. We concluded that there was a good agreement between the two because the intercept was approximately equal to 0 and the slope approximately equal to 1. Recompute the regression but use the matrix approach to estimate standard errors for the parameters. Then employ these errors to develop confidence intervals and use these to make a probabilistic statement regarding the goodness of fit.

**Solution.** The data can be written in matrix format for simple linear regression as:

$$[Z] = \begin{bmatrix} 1 & 10 \\ 1 & 16.3 \\ 1 & 23 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1 & 50 \end{bmatrix} \quad \{Y\} = \begin{bmatrix} 8.953 \\ 16.405 \\ 22.607 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ 49.988 \end{bmatrix}$$

Matrix transposition and multiplication can then be used to generate the normal equations as

$$[[Z]^T[Z]] \{A\} = \{[Z]^T\{Y\}\}$$

$$\begin{bmatrix} 15 & 548.3 \\ 548.3 & 22191.21 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} 552.741 \\ 22421.43 \end{Bmatrix}$$

Matrix inversion can be used to obtain the slope and intercept as

$$\begin{aligned} \{A\} &= [Z]^T [Z]^{-1} \{[Z]^T \{Y\}\} \\ &= \begin{bmatrix} 0.688414 & -0.01701 \\ -0.01701 & 0.000465 \end{bmatrix} \begin{Bmatrix} 552.741 \\ 22421.43 \end{Bmatrix} = \begin{Bmatrix} -0.85872 \\ 1.031592 \end{Bmatrix} \end{aligned}$$

Thus, the intercept and the slope are determined as  $a_0 = -0.85872$  and  $a_1 = 1.031592$ , respectively. These values in turn can be used to compute the standard error of the estimate as  $s_{y/x} = 0.863403$ . This value can be used along with the diagonal elements of the matrix inverse to calculate the standard errors of the coefficients,

$$s(a_0) = \sqrt{z_{11}^{-1} s_{y/x}^2} = \sqrt{0.688414(0.863403)^2} = 0.716372$$

$$s(a_1) = \sqrt{z_{22}^{-1} s_{y/x}^2} = \sqrt{0.000465(0.863403)^2} = 0.018625$$

The statistic,  $t_{\alpha/2, n-1}$  needed for a 95% confidence interval with  $n - 2 = 15 - 2 = 13$  degrees of freedom can be determined from a statistics table or using software. We used an Excel function, `TINV`, to come up with the proper value, as in

$$= \text{TINV}(0.05, 13)$$

which yielded a value of 2.160368. Equations (17.29) and (17.30) can then be used to compute the confidence intervals as

$$\begin{aligned} a_0 &= -0.85872 \pm 2.160368(0.716372) \\ &= -0.85872 \pm 1.547627 = [-2.40634, 0.688912] \end{aligned}$$

$$\begin{aligned} a_1 &= 1.031592 \pm 2.160368(0.018625) \\ &= 1.031592 \pm 0.040237 = [0.991355, 1.071828] \end{aligned}$$

Notice that the desired values (0 for intercept and slope and 1 for the intercept) fall within the intervals. On the basis of this analysis we could make the following statement regarding the slope: We have strong grounds for believing that the slope of the true regression line lies within the interval from 0.991355 to 1.071828. Because 1 falls within this interval, we also have strong grounds for believing that the result supports the agreement between the measurements and the model. Because zero falls within the intercept interval, a similar statement can be made regarding the intercept.

The foregoing is a limited introduction to the rich topic of statistical inference and its relationship to regression. There are many subtleties that are beyond the scope of this book. Our primary motivation has been to illustrate the power of the matrix approach to general linear least squares. You should consult some of the excellent books on the subject (e.g., Draper and Smith 1981) for additional information. In addition, it should be noted that software packages and libraries can generate least-squares regression fits along with information relevant to inferential statistics. We will explore some of these capabilities when we describe these packages at the end of Chap. 19.

## 17.5 NONLINEAR REGRESSION

There are many cases in engineering where nonlinear models must be fit to data. In the present context, these models are defined as those that have a nonlinear dependence on their parameters. For example,

$$f(x) = a_0 (1 - e^{-a_1 x}) + e \quad (17.31)$$

This equation cannot be manipulated so that it conforms to the general form of Eq. (17.23).

As with linear least squares, nonlinear regression is based on determining the values of the parameters that minimize the sum of the squares of the residuals. However, for the nonlinear case, the solution must proceed in an iterative fashion.

The *Gauss-Newton method* is one algorithm for minimizing the sum of the squares of the residuals between data and nonlinear equations. The key concept underlying the technique is that a Taylor series expansion is used to express the original nonlinear equation in an approximate, linear form. Then, least-squares theory can be used to obtain new estimates of the parameters that move in the direction of minimizing the residual.

To illustrate how this is done, first the relationship between the nonlinear equation and the data can be expressed generally as

$$y_i = f(x_i; a_0, a_1, \dots, a_m) + e_i$$

where  $y_i$  = a measured value of the dependent variable,  $f(x_i; a_0, a_1, \dots, a_m)$  = the equation that is a function of the independent variable  $x_i$  and a nonlinear function of the parameters  $a_0, a_1, \dots, a_m$ , and  $e_i$  = a random error. For convenience, this model can be expressed in abbreviated form by omitting the parameters,

$$y_i = f(x_i) + e_i \quad (17.32)$$

The nonlinear model can be expanded in a Taylor series around the parameter values and curtailed after the first derivatives. For example, for a two-parameter case,

$$f(x_i)_{j+1} = f(x_i)_j + \frac{\partial f(x_i)_j}{\partial a_0} \Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1} \Delta a_1 \quad (17.33)$$

where  $j$  = the initial guess,  $j + 1$  = the prediction,  $\Delta a_0 = a_{0,j+1} - a_{0,j}$ , and  $\Delta a_1 = a_{1,j+1} - a_{1,j}$ . Thus, we have linearized the original model with respect to the parameters. Equation (17.33) can be substituted into Eq. (17.32) to yield

$$y_i - f(x_i)_j = \frac{\partial f(x_i)_j}{\partial a_0} \Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1} \Delta a_1 + e_i$$

or in matrix form [compare with Eq. (17.24)],

$$\{D\} = [Z_j] \{\Delta A\} + \{E\} \quad (17.34)$$

where  $[Z_j]$  is the matrix of partial derivatives of the function evaluated at the initial guess  $j$ ,

$$[Z_j] = \begin{bmatrix} \partial f_1 / \partial a_0 & \partial f_1 / \partial a_1 \\ \partial f_2 / \partial a_0 & \partial f_2 / \partial a_1 \\ \vdots & \vdots \\ \partial f_n / \partial a_0 & \partial f_n / \partial a_1 \end{bmatrix}$$

where  $n$  = the number of data points and  $\partial f_i / \partial a_k$  = the partial derivative of the function with respect to the  $k$ th parameter evaluated at the  $i$ th data point. The vector  $\{D\}$  contains the differences between the measurements and the function values,

$$\{D\} = \begin{Bmatrix} y_1 - f(x_1) \\ y_2 - f(x_2) \\ \vdots \\ y_n - f(x_n) \end{Bmatrix}$$

and the vector  $\{\Delta A\}$  contains the changes in the parameter values,

$$\{\Delta A\} = \begin{Bmatrix} \Delta a_0 \\ \Delta a_1 \\ \vdots \\ \Delta a_m \end{Bmatrix}$$

Applying linear least-squares theory to Eq. (17.34) results in the following normal equations [recall Eq. (17.25)]:

$$[Z_j]^T [Z_j] \{\Delta A\} = \{[Z_j]^T \{D\}\} \quad (17.35)$$

Thus, the approach consists of solving Eq. (17.35) for  $\{\Delta A\}$  which can be employed to compute improved values for the parameters, as in

$$a_{0,j+1} = a_{0,j} + \Delta a_0$$

and

$$a_{1,j+1} = a_{1,j} + \Delta a_1$$

This procedure is repeated until the solution converges—that is, until

$$|\varepsilon_a|_k = \left| \frac{a_{k,j+1} - a_{k,j}}{a_{k,j+1}} \right| 100\% \quad (17.36)$$

falls below an acceptable stopping criterion.

### EXAMPLE 17.9 Gauss-Newton Method

**Problem Statement.** Fit the function  $f(x; a_0, a_1) = a_0(1 - e^{-a_1 x})$  to the data:

$x$	0.25	0.75	1.25	1.75	2.25
$y$	0.28	0.57	0.68	0.74	0.79

Use initial guesses of  $a_0 = 1.0$  and  $a_1 = 1.0$  for the parameters. Note that for these guesses the initial sum of the squares of the residuals is 0.0248.

**Solution.** The partial derivatives of the function with respect to the parameters are

$$\frac{\partial f}{\partial a_0} = 1 - e^{-a_1 x} \quad (\text{E17.9.1})$$



and

$$\frac{\partial f}{\partial a_1} = a_0 x e^{-a_1 x} \quad (\text{E17.9.2})$$

Equations (E17.9.1) and (E17.9.2) can be used to evaluate the matrix

$$[Z_0] = \begin{bmatrix} 0.2212 & 0.1947 \\ 0.5276 & 0.3543 \\ 0.7135 & 0.3581 \\ 0.8262 & 0.3041 \\ 0.8946 & 0.2371 \end{bmatrix}$$

This matrix multiplied by its transpose results in

$$[Z_0]^T [Z_0] = \begin{bmatrix} 2.3193 & 0.9489 \\ 0.9489 & 0.4404 \end{bmatrix}$$

which in turn can be inverted to yield

$$[[Z_0]^T [Z_0]]^{-1} = \begin{bmatrix} 3.6397 & -7.8421 \\ -7.8421 & 19.1678 \end{bmatrix}$$

The vector  $\{D\}$  consists of the differences between the measurements and the model predictions,

$$\{D\} = \begin{Bmatrix} 0.28 - 0.2212 \\ 0.57 - 0.5276 \\ 0.68 - 0.7135 \\ 0.74 - 0.8262 \\ 0.79 - 0.8946 \end{Bmatrix} = \begin{Bmatrix} 0.0588 \\ 0.0424 \\ -0.0335 \\ -0.0862 \\ -0.1046 \end{Bmatrix}$$

It is multiplied by  $[Z_0]^T$  to give

$$[Z_0]^T \{D\} = \begin{bmatrix} -0.1533 \\ -0.0365 \end{bmatrix}$$

The vector  $\{\Delta A\}$  is then calculated by solving Eq. (17.35) for

$$\Delta A = \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix}$$

which can be added to the initial parameter guesses to yield

$$\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix} + \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix} = \begin{Bmatrix} 0.7286 \\ 1.5019 \end{Bmatrix}$$

Thus, the improved estimates of the parameters are  $a_0 = 0.7286$  and  $a_1 = 1.5019$ . The new parameters result in a sum of the squares of the residuals equal to 0.0242. Equation (17.36) can be used to compute  $\varepsilon_0$  and  $\varepsilon_1$  equal to 37 and 33 percent, respectively. The computation would then be repeated until these values fell below the prescribed stopping criterion. The final result is  $a_0 = 0.79186$  and  $a_1 = 1.6751$ . These coefficients give a sum of the squares of the residuals of 0.000662.

A potential problem with the Gauss-Newton method as developed to this point is that the partial derivatives of the function may be difficult to evaluate. Consequently, many computer programs use difference equations to approximate the partial derivatives. One method is

$$\frac{\partial f_i}{\partial a_k} \cong \frac{f(x_i; a_0, \dots, a_k + \delta a_k, \dots, a_m) - f(x_i; a_0, \dots, a_k, \dots, a_m)}{\delta a_k} \quad (17.37)$$

where  $\delta$  = a small fractional perturbation.

The Gauss-Newton method has a number of other possible shortcomings:

1. It may converge slowly.
2. It may oscillate widely, i.e., continually change directions.
3. It may not converge at all.

Modifications of the method (Booth and Peterson, 1958; Hartley, 1961) have been developed to remedy the shortcomings.

In addition, although there are several approaches expressly designed for regression, a more general approach is to use nonlinear optimization routines as described in Part Four. To do this, a guess for the parameters is made, and the sum of the squares of the residuals is computed. For example, for Eq. (17.31) it would be computed as

$$S_r = \sum_{i=1}^n [y_i - a_0(1 - e^{-a_1 x_i})]^2 \quad (17.38)$$

Then, the parameters would be adjusted systematically to minimize  $S_r$  using search techniques of the type described previously in Chap. 14. We will illustrate how this is done when we describe software applications at the end of Chap. 19.

## PROBLEMS

### 17.1 Given the data

0.90	1.42	1.30	1.55	1.63
1.32	1.35	1.47	1.95	1.66
1.96	1.47	1.92	1.35	1.05
1.85	1.74	1.65	1.78	1.71
2.29	1.82	2.06	2.14	1.27

determine (a) the mean, (b) the standard deviation, (c) the variance, (d) the coefficient of variation, and (e) the 95% confidence interval for the mean.

17.2 Construct a histogram for the data in Prob. 17.1. Use a range of 0.6 to 2.4 with intervals of 0.2.

### 17.3 Given the data

15	6	18	21	26	28	32
39	22	28	24	27	27	33
2	12	17	34	29	31	38
45	36	41	37	43	38	26

determine (a) the mean, (b) the standard deviation, (c) the variance, (d) the coefficient of variation, and (e) the 90% confidence interval for the mean. (f) Construct a histogram. Use a range from 0 to 55 with increments of 5. (g) Assuming that the distribution is normal and that your estimate of the standard deviation is valid, compute the range (that is, the lower and the upper values) that encompasses 68% of the readings. Determine whether this is a valid estimate for the data in this problem.

17.4 Use least-squares regression to fit a straight line to

x	1	3	5	7	10	12	13	16	18	20
y	4	5	6	5	8	7	6	9	12	11

Along with the slope and intercept, compute the standard error of the estimate and the correlation coefficient. Plot the data and the regression line. Then repeat the problem, but regress  $x$  versus  $y$ —that is, switch the variables. Interpret your results.

17.5 Use least-squares regression to fit a straight line to

x	5	6	10	14	16	20	22	28	28	36	38
y	30	22	28	14	22	16	8	8	14	0	4

Along with the slope and the intercept, compute the standard error of the estimate and the correlation coefficient. Plot the data and the regression line. If someone made an additional measurement of  $x = 5$ ,  $y = 5$ , would you suspect, based on a visual assessment and the standard error, that the measurement was valid or faulty? Justify your conclusion.

**17.6** Use least-squares regression to fit a straight line to

x	2	3	4	7	8	9	5	5
y	9	6	5	10	9	11	2	3

(a) Along with the slope and intercept, compute the standard error of the estimate and the correlation coefficient. Plot the data and the straight line. Assess the fit.

(b) Recompute (a), but use polynomial regression to fit a parabola to the data. Compare the results with those of (a).

**17.7** Fit a saturation-growth-rate model to

x	0.75	2	2.5	4	6	8	8.5
y	0.8	1.3	1.2	1.6	1.7	1.8	1.7

Plot the data and the equation. Find the standard error.

**17.8** Fit a power equation to the data from Prob. 17.7. Plot the data and the equation, and find the standard error.

**17.9** Fit a parabola to the data from Prob. 17.7. Plot the data and the equation, and find the standard error.

**17.10** Fit a power equation to

x	2.5	3.5	5	6	7.5	10	12.5	15	17.5	20
y	7	5.5	3.9	3.6	3.1	2.8	2.6	2.4	2.3	2.3

Plot  $y$  versus  $x$  along with the power equation.

**17.11** Fit an exponential model to

x	0.4	0.8	1.2	1.6	2.0	2.3
y	750	1000	1400	2000	2700	3750

Plot the data and the equation on both standard and semi-logarithmic graph paper. Discuss your results.

**17.12** Fit a parabola to the data in Prob. 17.11. Plot the data and the equation.

**17.13** Given the data

x	5	10	15	20	25	30	35	40	45	50
y	16	25	32	33	38	36	39	40	42	42

use least-squares regression to fit (a) a straight line, (b) a power equation, (c) a saturation-growth-rate equation, and (d) a parabola. Plot the data along with all the curves. Is any one of the curves superior? If so, justify.

**17.14** Use multiple linear regression to fit

$x_1$	0	1	1	2	2	3	3	4	4
$x_2$	0	1	2	1	2	1	2	1	2
y	15	18	12.8	25.7	20.6	35.0	29.8	45.5	40.3

Compute the coefficients, the standard error of the estimate, and the correlation coefficient.

**17.15** Use multiple linear regression to fit

$x_1$	0	0	1	2	0	1	2	2	1
$x_2$	0	2	2	4	4	6	6	2	1
y	15	19	12	11	24	22	15	5	19

Compute the coefficients, the standard error of the estimate, and the correlation coefficient.

**17.16** Use nonlinear regression to fit a parabola to the following data,

x	0.075	0.5	1	1.2	1.7	2.0	2.3
y	600	800	1200	1400	2050	2650	3750

**17.17** Use nonlinear regression to fit a saturation-growth-rate equation to the data in Prob. 17.13.

**17.18** Recompute the regression fits from Probs. (a) 17.4, and (b) 17.13, using the matrix approach. Estimate the standard errors and develop 90% confidence intervals for the slope and the intercept.

**17.19** Develop, debug, and test a subprogram in either a high-level language or macro language of your choice to implement linear regression. Among other things: (a) Add statements to document the code, and (b) determine the standard error and the coefficient of determination.

**17.20** Use the Numerical Methods TOOLKIT software to solve Probs. (a) 17.4, (b) 17.5, (c) 17.6, (d) 17.9, and (e) 17.12.

# CHAPTER 18

## Interpolation

You will frequently have occasion to estimate intermediate values between precise data points. The most common method used for this purpose is polynomial interpolation. Recall that the general formula for an  $n$ th-order polynomial is

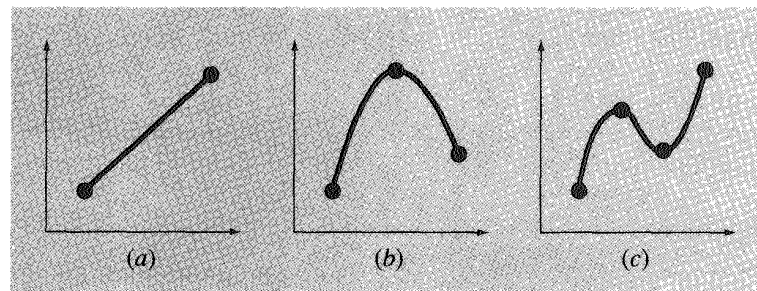
$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n \quad (18.1)$$

For  $n + 1$  data points, there is one and only one polynomial of order  $n$  that passes through all the points. For example, there is only one straight line (that is, a first-order polynomial) that connects two points (Fig. 18.1a). Similarly, only one parabola connects a set of three points (Fig. 18.1b). *Polynomial interpolation* consists of determining the unique  $n$ th-order polynomial that fits  $n + 1$  data points. This polynomial then provides a formula to compute intermediate values.

Although there is one and only one  $n$ th-order polynomial that fits  $n + 1$  points, there are a variety of mathematical formats in which this polynomial can be expressed. In this chapter, we will describe two alternatives that are well-suited for computer implementation: the Newton and the Lagrange polynomials.

**FIGURE 18.1**

Examples of interpolating polynomials: (a) first-order (linear) connecting two points, (b) second-order (quadratic or parabolic) connecting three points, and (c) third-order (cubic) connecting four points.



## 18.1 NEWTON'S DIVIDED-DIFFERENCE INTERPOLATING POLYNOMIALS

As stated above, there are a variety of alternative forms for expressing an interpolating polynomial. *Newton's divided-difference interpolating polynomial* is among the most popular and useful forms. Before presenting the general equation, we will introduce the first- and second-order versions because of their simple visual interpretation.

### 18.1.1 Linear Interpolation

The simplest form of interpolation is to connect two data points with a straight line. This technique, called *linear interpolation*, is depicted graphically in Fig. 18.2. Using similar triangles,

$$\frac{f_1(x) - f(x_0)}{x - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

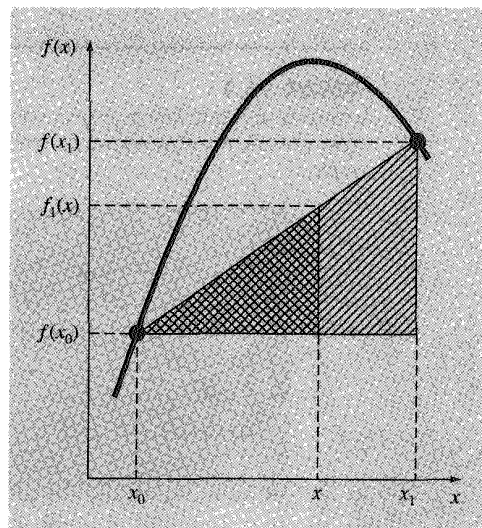
which can be rearranged to yield

$$f_1(x) = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0} (x - x_0) \quad (18.2)$$

which is a *linear-interpolation formula*. The notation  $f_1(x)$  designates that this is a first-order interpolating polynomial. Notice that besides representing the slope of the line connecting the points, the term  $[f(x_1) - f(x_0)]/(x_1 - x_0)$  is a finite-divided-difference

**FIGURE 18.2**

Graphical depiction of linear interpolation. The shaded areas indicate the similar triangles used to derive the linear-interpolation formula [Eq. (18.2)].



approximation of the first derivative [recall Eq. (4.17)]. In general, the smaller the interval between the data points, the better the approximation. This is due to the fact that, as the interval decreases, a continuous function will be better approximated by a straight line. This characteristic is demonstrated in the following example.

### EXAMPLE 18.1 Linear Interpolation

**Problem Statement.** Estimate the natural logarithm of 2 using linear interpolation. First, perform the computation by interpolating between  $\ln 1 = 0$  and  $\ln 6 = 1.791759$ . Then, repeat the procedure, but use a smaller interval from  $\ln 1$  to  $\ln 4$  (1.386294). Note that the true value of  $\ln 2$  is 0.6931472.

**Solution.** We use Eq. (18.2) and a linear interpolation for  $\ln(2)$  from  $x_0 = 1$  to  $x_1 = 6$  to give

$$f_1(2) = 0 + \frac{1.791759 - 0}{6 - 1} (2 - 1) = 0.3583519$$

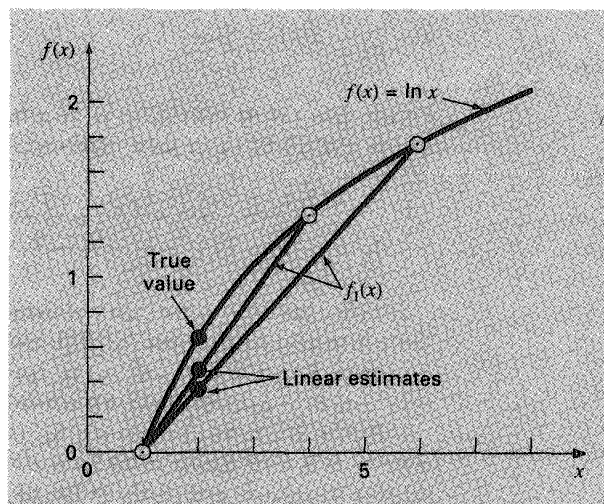
which represents an error of  $\varepsilon_t = 48.3\%$ . Using the smaller interval from  $x_0 = 1$  to  $x_1 = 4$  yields

$$f_1(2) = 0 + \frac{1.386294 - 0}{4 - 1} (2 - 1) = 0.4620981$$

Thus, using the shorter interval reduces the percent relative error to  $\varepsilon_t = 33.3\%$ . Both interpolations are shown in Fig. 18.3, along with the true function.

**FIGURE 18.3**

Two linear interpolations to estimate  $\ln 2$ . Note how the smaller interval provides a better estimate.



### 18.1.2 Quadratic Interpolation

The error in Example 18.1 resulted from our approximating a curve with a straight line. Consequently, a strategy for improving the estimate is to introduce some curvature into the line connecting the points. If three data points are available, this can be accomplished with a second-order polynomial (also called a quadratic polynomial or a *parabola*). A particularly convenient form for this purpose is

$$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) \quad (18.3)$$

Note that although Eq. (18.3) might seem to differ from the general polynomial [Eq. (18.1)], the two equations are equivalent. This can be shown by multiplying the terms in Eq. (18.3) to yield

$$f_2(x) = b_0 + b_1x - b_1x_0 + b_2x^2 + b_2x_0x_1 - b_2xx_0 - b_2xx_1$$

or, collecting terms,

$$f_2(x) = a_0 + a_1x + a_2x^2$$

where

$$a_0 = b_0 - b_1x_0 + b_2x_0x_1$$

$$a_1 = b_1 - b_2x_0 - b_2x_1$$

$$a_2 = b_2$$

Thus, Eqs. (18.1) and (18.3) are alternative, equivalent formulations of the unique second-order polynomial joining the three points.

A simple procedure can be used to determine the values of the coefficients. For  $b_0$ , Eq. (18.3) with  $x = x_0$  can be used to compute

$$b_0 = f(x_0) \quad (18.4)$$

Equation (18.4) can be substituted into Eq. (18.3), which can be evaluated at  $x = x_1$  for

$$b_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \quad (18.5)$$

Finally, Eqs. (18.4) and (18.5) can be substituted into Eq. (18.3), which can be evaluated at  $x = x_2$  and solved (after some algebraic manipulations) for

$$b_2 = \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0} \quad (18.6)$$

Notice that, as was the case with linear interpolation,  $b_1$  still represents the slope of the line connecting points  $x_0$  and  $x_1$ . Thus, the first two terms of Eq. (18.3) are equivalent to linear interpolation from  $x_0$  to  $x_1$ , as specified previously in Eq. (18.2). The last term,  $b_2(x - x_0)(x - x_1)$ , introduces the second-order curvature into the formula.

Before illustrating how to use Eq. (18.3), we should examine the form of the coefficient  $b_2$ . It is very similar to the finite-divided-difference approximation of the second derivative introduced previously in Eq. (4.24). Thus, Eq. (18.3) is beginning to manifest a structure that is very similar to the Taylor series expansion. This observation will be

explored further when we relate Newton's interpolating polynomials to the Taylor series in Sec. 18.1.4. But first, we will do an example that shows how Eq. (18.3) is used to interpolate among three points.

### EXAMPLE 18.2 Quadratic Interpolation

**Problem Statement.** Fit a second-order polynomial to the three points used in Example 18.1:

$$\begin{aligned}x_0 &= 1 & f(x_0) &= 0 \\x_1 &= 4 & f(x_1) &= 1.386294 \\x_2 &= 6 & f(x_2) &= 1.791759\end{aligned}$$

Use the polynomial to evaluate  $\ln 2$ .

**Solution.** Applying Eq. (18.4) yields

$$b_0 = 0$$

Equation (18.5) yields

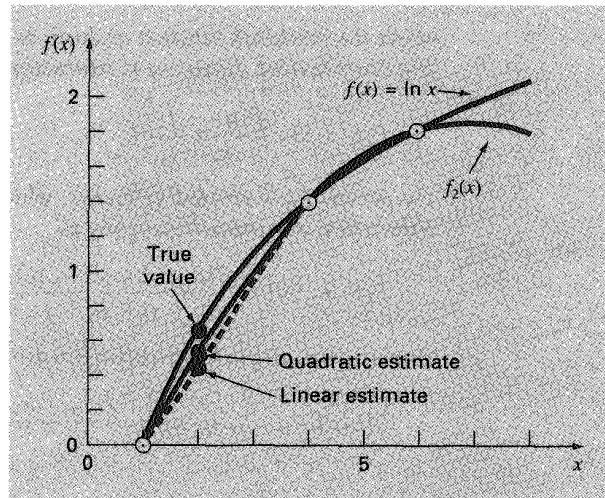
$$b_1 = \frac{1.386294 - 0}{4 - 1} = 0.4620981$$

and Eq. (18.6) gives

$$b_2 = \frac{\frac{1.791759 - 1.386294}{6 - 4} - 0.4620981}{6 - 1} = -0.0518731$$

**FIGURE 18.4**

The use of quadratic interpolation to estimate  $\ln 2$ . The linear interpolation from  $x = 1$  to 4 is also included for comparison.





Substituting these values into Eq. (18.3) yields the quadratic formula

$$f_2(x) = 0 + 0.4620981(x - 1) - 0.0518731(x - 1)(x - 4)$$

which can be evaluated at  $x = 2$  for

$$f_2(2) = 0.5658444$$

which represents a relative error of  $\varepsilon_t = 18.4\%$ . Thus, the curvature introduced by the quadratic formula (Fig. 18.4) improves the interpolation compared with the result obtained using straight lines in Example 18.1 and Fig. 18.3.

### 18.1.3 General Form of Newton's Interpolating Polynomials

The preceding analysis can be generalized to fit an  $n$ th-order polynomial to  $n + 1$  data points. The  $n$ th-order polynomial is

$$f_n(x) = b_0 + b_1(x - x_0) + \cdots + b_n(x - x_0)(x - x_1) \cdots (x - x_{n-1}) \quad (18.7)$$

As was done previously with the linear and quadratic interpolations, data points can be used to evaluate the coefficients  $b_0, b_1, \dots, b_n$ . For an  $n$ th-order polynomial,  $n + 1$  data points are required:  $[x_0, f(x_0)], [x_1, f(x_1)], \dots, [x_n, f(x_n)]$ . We use these data points and the following equations to evaluate the coefficients:

$$b_0 = f(x_0) \quad (18.8)$$

$$b_1 = f[x_1, x_0] \quad (18.9)$$

$$b_2 = f[x_2, x_1, x_0] \quad (18.10)$$

.

.

.

$$b_n = f[x_n, x_{n-1}, \dots, x_1, x_0] \quad (18.11)$$

where the bracketed function evaluations are finite divided differences. For example, the first finite divided difference is represented generally as

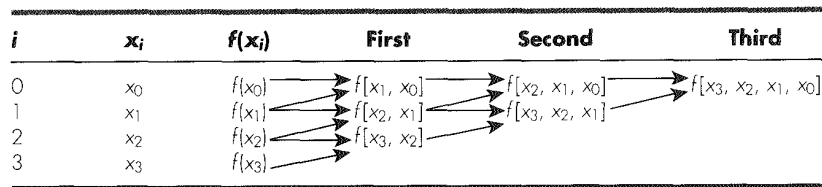
$$f[x_i, x_j] = \frac{f(x_i) - f(x_j)}{x_i - x_j} \quad (18.12)$$

The *second finite divided difference*, which represents the difference of two first divided differences, is expressed generally as

$$f[x_i, x_j, x_k] = \frac{f[x_i, x_j] - f[x_j, x_k]}{x_i - x_k} \quad (18.13)$$

Similarly, the  *$n$ th finite divided difference* is

$$f[x_n, x_{n-1}, \dots, x_1, x_0] = \frac{f[x_n, x_{n-1}, \dots, x_1] - f[x_{n-1}, x_{n-2}, \dots, x_0]}{x_n - x_0} \quad (18.14)$$

**FIGURE 18.5**

Graphical depiction of the recursive nature of finite divided differences.

These differences can be used to evaluate the coefficients in Eqs. (18.8) through (18.11), which can then be substituted into Eq. (18.7) to yield the interpolating polynomial

$$f_n(x) = f(x_0) + (x - x_0)f[x_1, x_0] + (x - x_0)(x - x_1)f[x_2, x_1, x_0] + \cdots + (x - x_0)(x - x_1) \cdots (x - x_{n-1})f[x_n, x_{n-1}, \dots, x_0] \quad (18.15)$$

which is called *Newton's divided-difference interpolating polynomial*. It should be noted that it is not necessary that the data points used in Eq. (18.15) be equally spaced or that the abscissa values necessarily be in ascending order, as illustrated in the following example. Also, notice how Eqs. (18.12) through (18.14) are recursive—that is, higher-order differences are computed by taking differences of lower-order differences (Fig. 18.5). This property will be exploited when we develop an efficient computer program in Sec. 18.1.5 to implement the method.

### EXAMPLE 18.3 Newton's Divided-Difference Interpolating Polynomials

**Problem Statement.** In Example 18.2, data points at  $x_0 = 1$ ,  $x_1 = 4$ , and  $x_2 = 6$  were used to estimate  $\ln 2$  with a parabola. Now, adding a fourth point [ $x_3 = 5$ ;  $f(x_3) = 1.609438$ ], estimate  $\ln 2$  with a third-order Newton's interpolating polynomial.

**Solution.** The third-order polynomial, Eq. (18.7) with  $n = 3$ , is

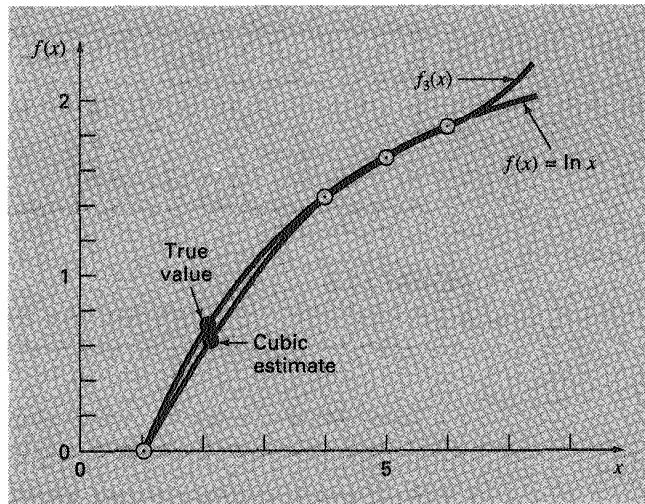
$$f_3(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1) + b_3(x - x_0)(x - x_1)(x - x_2)$$

The first divided differences for the problem are [Eq. (18.12)]

$$f[x_1, x_0] = \frac{1.386294 - 0}{4 - 0} = 0.4620981$$

$$f[x_2, x_1] = \frac{1.791759 - 1.386294}{6 - 4} = 0.2027326$$

$$f[x_3, x_2] = \frac{1.609438 - 1.386294}{5 - 6} = 0.1823216$$

**FIGURE 18.6**

The use of cubic interpolation to estimate  $\ln 2$ .

The second divided differences are [Eq. (18.13)]

$$f[x_2, x_1, x_0] = \frac{0.2027326 - 0.4620981}{6 - 1} = -0.05187311$$

$$f[x_3, x_2, x_1] = \frac{0.1823216 - 0.2027326}{5 - 4} = -0.02041100$$

The third divided difference is [Eq. (18.14) with  $n = 3$ ]

$$f[x_3, x_2, x_1, x_0] = \frac{-0.02041100 - (-0.05187311)}{5 - 1} = 0.007865529$$

The results for  $f[x_1, x_0]$ ,  $f[x_2, x_1, x_0]$ , and  $f[x_3, x_2, x_1, x_0]$  represent the coefficients  $b_1$ ,  $b_2$ , and  $b_3$  of Eq. (18.7). Along with  $b_0 = f(x_0) = 0.0$ , Eq. (18.7) is

$$f_3(x) = 0 + 0.4620981(x - 1) - 0.05187311(x - 1)(x - 4) + 0.007865529(x - 1)(x - 4)(x - 6)$$

which can be used to evaluate  $f_3(2) = 0.6287686$ , which represents a relative error of  $\varepsilon_t = 9.3\%$ . The complete cubic polynomial is shown in Fig. 18.6.

### 18.1.4 Errors of Newton's Interpolating Polynomials

Notice that the structure of Eq. (18.15) is similar to the Taylor series expansion in the sense that terms are added sequentially to capture the higher-order behavior of the underlying function. These terms are finite divided differences and, thus, represent approximations of

the higher-order derivatives. Consequently, as with the Taylor series, if the true underlying function is an  $n$ th-order polynomial, the  $n$ th-order interpolating polynomial based on  $n + 1$  data points will yield exact results.

Also, as was the case with the Taylor series, a formulation for the truncation error can be obtained. Recall from Eq. (4.6) that the truncation error for the Taylor series could be expressed generally as

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x_{i+1} - x_i)^{n+1} \quad (4.6)$$

where  $\xi$  is somewhere in the interval  $x_i$  to  $x_{i+1}$ . For an  $n$ th-order interpolating polynomial, an analogous relationship for the error is

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!} (x - x_0)(x - x_1) \cdots (x - x_n) \quad (18.16)$$

where  $\xi$  is somewhere in the interval containing the unknown and the data. For this formula to be of use, the function in question must be known and differentiable. This is not usually the case. Fortunately, an alternative formulation is available that does not require prior knowledge of the function. Rather, it uses a finite divided difference to approximate the  $(n + 1)$ th derivative,

$$R_n = f[x, x_n, x_{n-1}, \dots, x_0](x - x_0)(x - x_1) \cdots (x - x_n) \quad (18.17)$$

where  $f[x, x_n, x_{n-1}, \dots, x_0]$  is the  $(n + 1)$ th finite divided difference. Because Eq. (18.17) contains the unknown  $f(x)$ , it cannot be solved for the error. However, if an additional data point  $f(x_{n+1})$  is available, Eq. (18.17) can be used to estimate the error, as in

$$R_n \cong f[x_{n+1}, x_n, x_{n-1}, \dots, x_0](x - x_0)(x - x_1) \cdots (x - x_n) \quad (18.18)$$

#### EXAMPLE 18.4 Error Estimation for Newton's Polynomial

**Problem Statement.** Use Eq. (18.18) to estimate the error for the second-order polynomial interpolation of Example 18.2. Use the additional data point  $f(x_3) = f(5) = 1.609438$  to obtain your results.

**Solution.** Recall that in Example 18.2, the second-order interpolating polynomial provided an estimate of  $f_2(2) = 0.5658444$ , which represents an error of  $0.6931472 - 0.5658444 = 0.1273028$ . If we had not known the true value, as is most usually the case, Eq. (18.18), along with the additional value at  $x_3$ , could have been used to estimate the error, as in

$$R_2 = f[x_3, x_2, x_1, x_0](x - x_0)(x - x_1)(x - x_2)$$

or

$$R_2 = 0.007865529(x - 1)(x - 4)(x - 6)$$

where the value for the third-order finite divided difference is as computed previously in Example 18.3. This relationship can be evaluated at  $x = 2$  for

$$R_2 = 0.007865529(2 - 1)(2 - 4)(2 - 6) = 0.0629242$$

which is of the same order of magnitude as the true error.

From the previous example and from Eq. (18.18), it should be clear that the error estimate for the  $n$ th-order polynomial is equivalent to the difference between the  $(n + 1)$ th order and the  $n$ th-order prediction. That is,

$$R_n = f_{n+1}(x) - f_n(x) \quad (18.19)$$

In other words, the increment that is added to the  $n$ th-order case to create the  $(n + 1)$ th-order case [that is, Eq. (18.18)] is interpreted as an estimate of the  $n$ th-order error. This can be clearly seen by rearranging Eq. (18.19) to give

$$f_{n+1}(x) = f_n(x) + R_n$$

The validity of this approach is predicated on the fact that the series is strongly convergent. For such a situation, the  $(n + 1)$ th-order prediction should be much closer to the true value than the  $n$ th-order prediction. Consequently, Eq. (18.19) conforms to our standard definition of error as representing the difference between the truth and an approximation. However, note that whereas all other error estimates for iterative approaches introduced up to this point have been determined as a present prediction minus a previous one, Eq. (18.19) represents a future prediction minus a present one. This means that for a series that is converging rapidly, the error estimate of Eq. (18.19) could be less than the true error. This would represent a highly unattractive quality if the error estimate were being employed as a stopping criterion. However, as will be described in the following section, higher-order interpolating polynomials are highly sensitive to data errors—that is, they are very ill-conditioned. When employed for interpolation, they often yield predictions that diverge significantly from the true value. By “looking ahead” to sense errors, Eq. (18.19) is more sensitive to such divergence. As such, it is more valuable for the sort of exploratory data analysis for which Newton’s polynomial is best-suited.

### 18.1.5 Computer Algorithm for Newton’s Interpolating Polynomial

Three properties make Newton’s interpolating polynomials extremely attractive for computer applications:

1. As in Eq. (18.7), higher-order versions can be developed sequentially by adding a single term to the next lower-order equation. This facilitates the evaluation of several different-order versions in the same program. Such a capability is especially valuable when the order of the polynomial is not known a priori. By adding new terms sequentially, we can determine when a point of diminishing returns is reached—that is, when addition of higher-order terms no longer significantly improves the estimate or in certain situations actually detracts from it. The error equations discussed below in (3) are useful in devising an objective criterion for identifying this point of diminishing terms.
2. The finite divided differences that constitute the coefficients of the polynomial [Eqs. (18.8) through (18.11)] can be computed efficiently. That is, as in Eq. (18.14) and Fig. 18.5, lower-order differences are used to compute higher-order differences. By utilizing this previously determined information, the coefficients can be computed efficiently. The algorithm in Fig. 18.7 contains such a scheme.
3. The error estimate [Eq. (18.18)] can be very simply incorporated into a computer algorithm because of the sequential way in which the prediction is built.

```

SUBROUTINE NewtInt (x, y, n, xi, yint, ea)
  LOCAL fddn,n
  DO i = 0, n
    fddi,0 = yi
  END DO
  DO j = 1, n
    DO i = 0, n - j
      fddi,j = (fddi+1,j-1 - fddi,j-1) / (xi+j - xi)
    END DO
  END DO
  xterm = 1
  yint0 = fdd0,0
  DO order = 1, n
    xterm = xterm * (xi - xorder-1)
    yint2 = yintorder-1 + fdd0,order * xterm
    Eaorder-1 = yint2 - yintorder-1
    yintorder = yint2
  END order
END NewtInt

```

**FIGURE 18.7**

An algorithm for Newton's interpolating polynomial written in pseudocode.

All the above characteristics can be exploited and incorporated into a general algorithm for implementing Newton's polynomial (Fig. 18.7). Note that the algorithm consists of two parts: the first determines the coefficients from Eq. (18.7); the second determines the predictions and their associated error. The utility of this algorithm is demonstrated in the following example.

#### EXAMPLE 18.5 Error Estimates to Determine the Appropriate Order of Interpolation

**Problem Statement.** After incorporating the error [Eq. (18.18)], utilize the computer algorithm given in Fig. 18.7 and the following information to evaluate  $f(x) = \ln x$  at  $x = 2$ :

<b>x</b>	<b>f(x) = ln x</b>
0	1
4	1.3862944
6	1.7917595
5	1.6094379
3	1.0986123
1.5	0.4054641
2.5	0.9162907
3.5	1.2527630

Solution. The results of employing the algorithm in Fig. 18.7 to obtain a solution are shown in Fig. 18.8. The error estimates, along with the true error (based on the fact that  $\ln 2 = 0.6931472$ ), are depicted in Fig. 18.9. Note that the estimated error and the true error are similar and that their agreement improves as the order increases. From these results, it can be concluded that the fifth-order version yields a good estimate and that higher-order terms do not significantly enhance the prediction.

This exercise also illustrates the importance of the positioning and ordering of the points. For example, up through the third-order estimate, the rate of improvement is slow because the points that are added (at  $x = 4, 6,$  and  $5$ ) are distant and on one side of the point in question at  $x = 2$ . The fourth-order estimate shows a somewhat greater improvement because the new point at  $x = 3$  is closer to the unknown. However, the most dramatic decrease in the error is associated with the inclusion of the fifth-order term using the data point at  $x = 1.5$ . Not only is this point close to the unknown but it is also positioned on the opposite side from most of the other points. As a consequence, the error is reduced almost an order of magnitude.

The significance of the position and sequence of the data can also be illustrated by using the same data to obtain an estimate for  $\ln 2$ , but considering the points in a different sequence. Figure 18.9 shows results for the case of reversing the order of the original data, that is,  $x_0 = 3.5, x_1 = 2.5, x_3 = 1.5$ , and so forth. Because the initial points for this case are closer to and spaced on either side of  $\ln 2$ , the error decreases much more rapidly than for the original situation. By the second-order term, the error has been reduced to less than  $\epsilon_t = 2\%$ . Other combinations could be employed to obtain different rates of convergence.

**FIGURE 18.8**

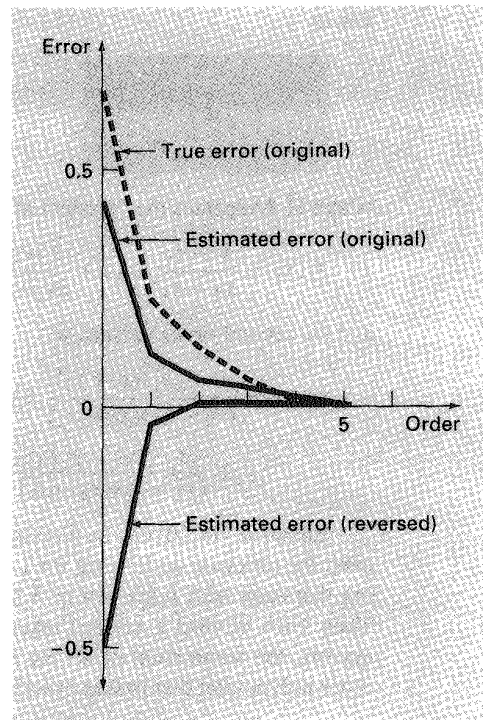
The output of a program, based on the algorithm from Fig. 18.7, to evaluate  $\ln 2$ .

```

NUMBER OF POINTS? 8
X( 0 ), y( 0 ) = ? 1,0
X( 1 ), y( 1 ) = ? 4,1.3862944
X( 2 ), y( 2 ) = ? 6,1.7917595
X( 3 ), y( 3 ) = ? 5,1.6094379
X( 4 ), y( 4 ) = ? 3,1.0986123
X( 5 ), y( 5 ) = ? 1.5,0.40546411
X( 6 ), y( 6 ) = ? 2.5,0.91629073
X( 7 ), y( 7 ) = ? 3.5,1.2527630

INTERPOLATION AT X = 2
ORDER      F(X)      ERROR
0          0.000000  0.462098
1          0.462098  0.103746
2          0.565844  0.062924
3          0.628769  0.046953
4          0.675722  0.021792
5          0.697514  -0.003616
6          0.693898  -0.000459
7          0.693439

```

**FIGURE 18.9**

Percent relative errors for the prediction of  $\ln 2$  as a function of the order of the interpolating polynomial.

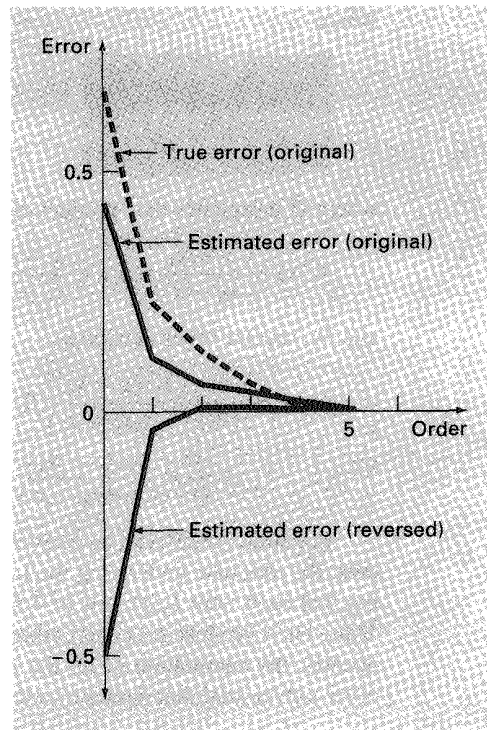
The foregoing example illustrates the importance of the choice of base points. As should be intuitively obvious, the points should be centered around and as close as possible to the unknown. This observation is also supported by direct examination of the error equation [Eq. (18.17)]. If we assume that the finite divided difference does not vary markedly along the range of the data, the error is proportional to the product:  $(x - x_0)(x - x_1) \cdots (x - x_n)$ . Obviously, the closer the base points are to  $x$ , the smaller the magnitude of this product.

## 18.2 LAGRANGE INTERPOLATING POLYNOMIALS

The *Lagrange interpolating polynomial* is simply a reformulation of the Newton polynomial that avoids the computation of divided differences. It can be represented concisely as

$$f_n(x) = \sum_{i=0}^n L_i(x) f(x_i) \quad (18.20)$$



**FIGURE 18.9**

Percent relative errors for the prediction of  $\ln 2$  as a function of the order of the interpolating polynomial.

The foregoing example illustrates the importance of the choice of base points. As should be intuitively obvious, the points should be centered around and as close as possible to the unknown. This observation is also supported by direct examination of the error equation [Eq. (18.17)]. If we assume that the finite divided difference does not vary markedly along the range of the data, the error is proportional to the product:  $(x - x_0)(x - x_1) \cdots (x - x_n)$ . Obviously, the closer the base points are to  $x$ , the smaller the magnitude of this product.

## 18.2 LAGRANGE INTERPOLATING POLYNOMIALS

The *Lagrange interpolating polynomial* is simply a reformulation of the Newton polynomial that avoids the computation of divided differences. It can be represented concisely as

$$f_n(x) = \sum_{i=0}^n L_i(x) f(x_i) \quad (18.20)$$

where

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j} \quad (18.21)$$

where  $\Pi$  designates the “product of.” For example, the linear version ( $n = 1$ ) is

$$f_1(x) = \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1) \quad (18.22)$$

and the second-order version is

$$\begin{aligned} f_2(x) = & \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f(x_0) + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} f(x_1) \\ & + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f(x_2) \end{aligned} \quad (18.23)$$

Equation (18.20) can be derived directly from Newton’s polynomial (Box 18.1). However, the rationale underlying the Lagrange formulation can be grasped directly by realizing that each term  $L_i(x)$  will be 1 at  $x = x_i$  and 0 at all other sample points (Fig. 18.10). Thus, each product  $L_i(x)f(x_i)$  takes on the value of  $f(x_i)$  at the sample point  $x_i$ . Consequently, the summation of all the products designated by Eq. (18.20) is the unique  $n$ th-order polynomial that passes exactly through all  $n + 1$  data points.

#### EXAMPLE 18.6 Lagrange Interpolating Polynomials

**Problem Statement.** Use a Lagrange interpolating polynomial of the first and second order to evaluate  $\ln 2$  on the basis of the data given in Example 18.2:

$$\begin{aligned} x_0 = 1 & \quad f(x_0) = 0 \\ x_1 = 4 & \quad f(x_1) = 1.386294 \\ x_2 = 6 & \quad f(x_2) = 1.791760 \end{aligned}$$

**Solution.** The first-order polynomial [Eq. (18.22)] can be used to obtain the estimate at  $x = 2$ ,

$$f_1(2) = \frac{2 - 4}{1 - 4} 0 + \frac{2 - 1}{4 - 1} 1.386294 = 0.4620981$$

In a similar fashion, the second-order polynomial is developed as [Eq. (18.23)]

$$\begin{aligned} f_2(2) = & \frac{(2 - 4)(2 - 6)}{(1 - 4)(1 - 6)} 0 + \frac{(2 - 1)(2 - 6)}{(4 - 1)(4 - 6)} 1.386294 \\ & + \frac{(2 - 1)(2 - 4)}{(6 - 1)(6 - 4)} 1.791760 = 0.5658444 \end{aligned}$$

As expected, both these results agree with those previously obtained using Newton’s interpolating polynomial.

### Box 18.1 Derivation of the Lagrange Form Directly from Newton's Interpolating Polynomial

The Lagrange interpolating polynomial can be derived directly from Newton's formulation. We will do this for the first-order case only [Eq. (18.2)]. To derive the Lagrange form, we reformulate the divided differences. For example, the first divided difference,

$$f[x_1, x_0] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \quad (\text{B18.1.1})$$

can be reformulated as

$$f[x_1, x_0] = \frac{f(x_1)}{x_1 - x_0} + \frac{f(x_0)}{x_0 - x_1} \quad (\text{B18.1.2})$$

which is referred to as the *symmetric form*. Substituting Eq. (B18.1.2) into Eq. (18.2) yields

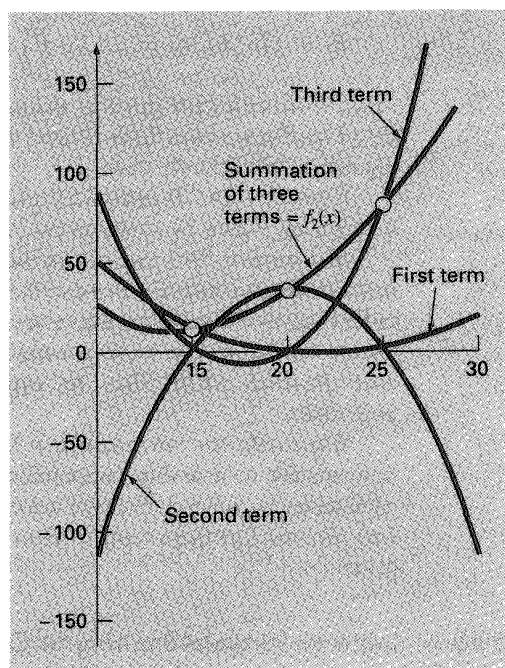
$$f_1(x) = f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1) + \frac{x - x_0}{x_0 - x_1} f(x_0)$$

Finally, grouping similar terms and simplifying yields the Lagrange form,

$$f_1(x) = \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1)$$

**FIGURE 18.10**

A visual depiction of the rationale behind the Lagrange polynomial. This figure shows a second-order case. Each of the three terms in Eq. (18.23) passes through one of the data points and is zero at the other two. The summation of the three terms must, therefore, be the unique second-order polynomial  $f_2(x)$  that passes exactly through the three points.



```

FUNCTION Lagrng(x, y, n, x)
  sum = 0
  DO i = 0, n
    product = y_i
    DO j = 0, n
      IF i ≠ j THEN
        product = product*(x - x_j)/(x_i - x_j)
      ENDIF
    END DO
    sum = sum + product
  END DO
  Lagrng = sum
END Lagrng

```

**FIGURE 18.11**

Pseudocode to implement Lagrange interpolation. This algorithm is set up to compute a single  $n$ th-order prediction, where  $n + 1$  is the number of data points.

Note that, as with Newton's method, the Lagrange version has an estimated error of [Eq. (18.17)]

$$R_n = f[x, x_n, x_{n-1}, \dots, x_0] \prod_{i=0}^n (x - x_i)$$

Thus, if an additional point is available at  $x = x_{n+1}$ , an error estimate can be obtained. However, because the finite divided differences are not employed as part of the Lagrange algorithm, this is rarely done.

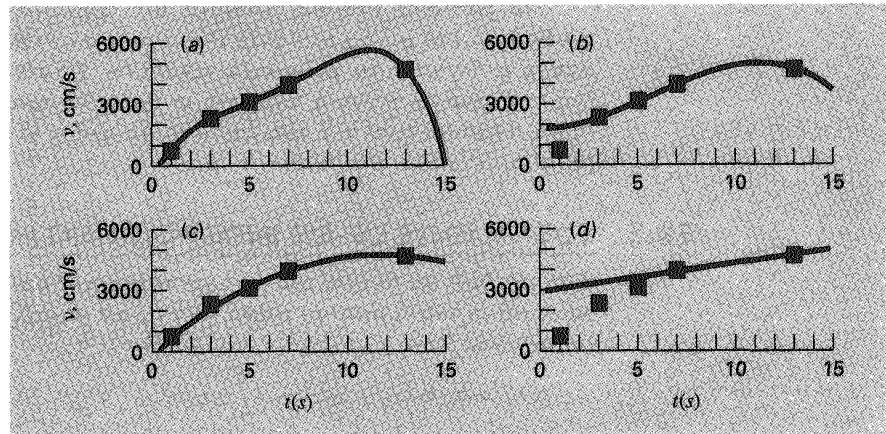
Equations (18.20) and (18.21) can be very simply programmed for implementation on a computer. Figure 18.11 shows pseudocode that can be employed for this purpose.

In summary, for cases where the order of the polynomial is unknown, the Newton method has advantages because of the insight it provides into the behavior of the different-order formulas. In addition, the error estimate represented by Eq. (18.18) can usually be integrated easily into the Newton computation because the estimate employs a finite difference (Example 18.5). Thus, for exploratory computations, Newton's method is often preferable.

When only one interpolation is to be performed, the Lagrange and Newton formulations require comparable computational effort. However, the Lagrange version is somewhat easier to program. Because it does not require computation and storage of divided differences, the Lagrange form is often used when the order of the polynomial is known a priori.

#### EXAMPLE 18.7 Lagrange Interpolation Using the Computer

**Problem Statement.** We can use the algorithm from Fig. 18.11 to study a trend analysis problem associated with our now-familiar falling parachutist. Assume that we have



**FIGURE 18.12**  
Plots showing (a) fourth-order, (b) third-order, (c) second-order, and (d) first-order interpolations.

developed instrumentation to measure the velocity of the parachutist. The measured data obtained for a particular test case is

Time, s	Measured Velocity $v$ , cm/s
1	800
3	2310
5	3090
7	3940
13	4755

Our problem is to estimate the velocity of the parachutist at  $t = 10$  s to fill in the large gap in the measurements between  $t = 7$  and  $t = 13$  s. We are aware that the behavior of interpolating polynomials can be unexpected. Therefore, we will construct polynomials of orders 4, 3, 2, and 1 and compare the results.

**Solution.** The Lagrange algorithm can be used to construct fourth-, third-, second-, and first-order interpolating polynomials.

The fourth-order polynomial and the input data can be plotted as shown in Fig. 18.12a. It is evident from this plot that the estimated value of  $y$  at  $x = 10$  is higher than the overall trend of the data.

Figure 18.12b through d shows plots of the results of the computations for third-, second-, and first-order interpolating polynomials. It is noted that the lower the order, the lower the estimated value of the velocity at  $t = 10$  s. The plots of the interpolating polynomials indicate that the higher-order polynomials tend to overshoot the trend of the data. This suggests that the first- or second-order versions are most appropriate for this particular trend analysis. It should be remembered, however, that because we are dealing with uncertain data, regression would actually be more appropriate.

The preceding example illustrates that higher-order polynomials tend to be ill-conditioned; that is, they tend to be highly sensitive to round-off error. The same problem applies to higher-order polynomial regression. Double-precision arithmetic sometimes helps mitigate the problem. However, as the order increases, there will come a point at which round-off error will interfere with the ability to interpolate using the simple approaches covered to this point.

### 18.3 COEFFICIENTS OF AN INTERPOLATING POLYNOMIAL

Although both the Newton and the Lagrange polynomials are well-suited for determining intermediate values between points, they do not provide a convenient polynomial of the conventional form

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n \quad (18.24)$$

A straightforward method for computing the coefficients of this polynomial is based on the fact that  $n + 1$  data points are required to determine the  $n + 1$  coefficients. Thus, simultaneous linear algebraic equations can be used to calculate the  $a$ 's. For example, suppose that you desired to compute the coefficients of the parabola

$$f(x) = a_0 + a_1x + a_2x^2 \quad (18.25)$$

Three data points are required:  $[x_0, f(x_0)]$ ,  $[x_1, f(x_1)]$ , and  $[x_2, f(x_2)]$ . Each can be substituted into Eq. (18.25) to give

$$\begin{aligned} f(x_0) &= a_0 + a_1x_0 + a_2x_0^2 \\ f(x_1) &= a_0 + a_1x_1 + a_2x_1^2 \\ f(x_2) &= a_0 + a_1x_2 + a_2x_2^2 \end{aligned} \quad (18.26)$$

Thus, for this case, the  $x$ 's are the knowns and the  $a$ 's are the unknowns. Because there are the same number of equations as unknowns, Eq. (18.26) could be solved by an elimination method from Part Three.

It should be noted that the foregoing approach is not the most efficient method that is available to determine the coefficients of an interpolating polynomial. Press et al. (1986) provide a discussion and computer codes for more efficient approaches. Whatever technique is employed, a word of caution is in order. Systems such as Eq. (18.26) are notoriously ill-conditioned. Whether they are solved with an elimination method or with a more efficient algorithm, the resulting coefficients can be highly inaccurate, particularly for large  $n$ . When used for a subsequent interpolation, they often yield erroneous results.

In summary, if you are interested in determining an intermediate point, employ Newton or Lagrange interpolation. If you must determine an equation of the form of Eq. (18.24), limit yourself to lower-order polynomials and check your results carefully.

### 18.4 INVERSE INTERPOLATION

As the nomenclature implies, the  $f(x)$  and  $x$  values in most interpolation contexts are the dependent and independent variables, respectively. As a consequence, the values of the  $x$ 's are typically uniformly spaced. A simple example is a table of values derived for the

function  $f(x) = 1/x$ ,

$x$	1	2	3	4	5	6	7
$f(x)$	1	0.5	0.3333	0.25	0.2	0.1667	0.1429

Now suppose that you must use the same data, but you are given a value for  $f(x)$  and must determine the corresponding value of  $x$ . For instance, for the data above, suppose that you were asked to determine the value of  $x$  that corresponded to  $f(x) = 0.3$ . For this case, because the function is available and easy to manipulate, the correct answer can be determined directly as  $x = 1/0.3 = 3.3333$ .

Such a problem is called *inverse interpolation*. For a more complicated case, you might be tempted to switch the  $f(x)$  and  $x$  values [that is, merely plot  $x$  versus  $f(x)$ ] and use an approach like Lagrange interpolation to determine the result. Unfortunately, when you reverse the variables, there is no guarantee that the values along the new abscissa [the  $f(x)$ 's] will be evenly spaced. In fact, in many cases, the values will be “telescoped.” That is, they will have the appearance of a logarithmic scale with some adjacent points bunched together and others spread out widely. For example, for  $f(x) = 1/x$  the result is

$f(x)$	0.1429	0.1667	0.2	0.25	0.3333	0.5	1
$x$	7	6	5	4	3	2	1

Such nonuniform spacing on the abscissa often leads to oscillations in the resulting interpolating polynomial. This can occur even for lower-order polynomials.

An alternative strategy is to fit an  $n$ th-order interpolating polynomial,  $f_n(x)$ , to the original data [i.e., with  $f(x)$  versus  $x$ ]. In most cases, because the  $x$ 's are evenly spaced, this polynomial will not be ill-conditioned. The answer to your problem then amounts to finding the value of  $x$  that makes this polynomial equal to the given  $f(x)$ . Thus, the interpolation problem reduces to a roots problem!

For example, for the problem outlined above, a simple approach would be to fit a quadratic polynomial to the three points: (2, 0.5), (3, 0.3333) and (4, 0.25). The result would be

$$f_2(x) = 1.08333 - 0.375x + 0.041667x^2$$

The answer to the inverse interpolation problem of finding the  $x$  corresponding to  $f(x) = 0.3$  would therefore involve determining the root of

$$0.3 = 1.08333 - 0.375x + 0.041667x^2$$

For this simple case, the quadratic formula can be used to calculate

$$x = \frac{0.375 \pm \sqrt{(-0.375)^2 - 4(0.041667)(0.78333)}}{2(0.041667)} = \frac{5.704158}{3.295842}$$

Thus, the second root, 3.296, is a good approximation of the true value of 3.333. If additional accuracy were desired, a third- or fourth-order polynomial along with one of the root location methods from Part Two could be employed.

## 18.5 ADDITIONAL COMMENTS

Before proceeding to the next section, we must mention two additional topics: interpolation with equally spaced data and extrapolation.

### Box 18.2 Interpolation with Equally Spaced Data

If data is equally spaced and in ascending order, then the independent variable assumes values of

$$\begin{aligned}x_1 &= x_0 + h \\x_2 &= x_0 + 2h \\&\vdots \\&\vdots \\x_n &= x_0 + nh\end{aligned}$$

where  $h$  is the interval, or step size, between the data. On this basis, the finite divided differences can be expressed in concise form. For example, the second forward divided difference is

$$f[x_0, x_1, x_2] = \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

which can be expressed as

$$f[x_0, x_1, x_2] = \frac{f(x_2) - 2f(x_1) + f(x_0)}{2h^2} \quad (\text{B18.2.1})$$

because  $x_1 - x_0 = x_2 - x_1 = (x_2 - x_0)/2 = h$ . Now recall that the second forward difference is equal to [numerator of Eq. (4.24)]

$$\Delta^2 f(x_0) = f(x_2) - 2f(x_1) + f(x_0)$$

Therefore, Eq. (B18.2.1) can be represented as

$$f[x_0, x_1, x_2] = \frac{\Delta^2 f(x_0)}{2!h^2}$$

or, in general,

$$f[x_0, x_1, \dots, x_n] = \frac{\Delta^n f(x_0)}{n!h^n} \quad (\text{B18.2.2})$$

Using Eq. (B18.2.2), we can express Newton's interpolating polynomial [Eq. (18.15)] for the case of equispaced data as

$$\begin{aligned}f_n(x) &= f(x_0) + \frac{\Delta f(x_0)}{h}(x - x_0) \\&+ \frac{\Delta^2 f(x_0)}{2!h^2}(x - x_0)(x - x_0 - h) \\&+ \dots + \frac{\Delta^n f(x_0)}{n!h^n}(x - x_0)(x - x_0 - h) \\&\dots [x - x_0 - (n - 1)h] + R_n\end{aligned} \quad (\text{B18.2.3})$$

where the remainder is the same as Eq. (18.16). This equation is known as *Newton's formula*, or the *Newton-Gregory forward formula*. It can be simplified further by defining a new quantity,  $\alpha$ :

$$\alpha = \frac{x - x_0}{h}$$

This definition can be used to develop the following simplified expressions for the terms in Eq. (B18.2.3):

$$\begin{aligned}x - x_0 &= \alpha h \\x - x_0 - h &= \alpha h - h = h(\alpha - 1) \\&\vdots \\&\vdots \\x - x_0 - (n - 1)h &= \alpha h - (n - 1)h = h(\alpha - n + 1)\end{aligned}$$

which can be substituted into Eq. (B18.2.3) to give

$$\begin{aligned}f_n(x) &= f(x_0) + \Delta f(x_0)\alpha + \frac{\Delta^2 f(x_0)}{2!}\alpha(\alpha - 1) \\&+ \dots + \frac{\Delta^n f(x_0)}{n!}\alpha(\alpha - 1)\dots(\alpha - n + 1) + R_n\end{aligned} \quad (\text{B18.2.4})$$

where

$$R_n = \frac{f^{(n+1)}(\xi)}{(n + 1)!}h^{n+1}\alpha(\alpha - 1)(\alpha - 2)\dots(\alpha - n)$$

This concise notation will have utility in our derivation and error analyses of the integration formulas in Chap. 21.

In addition to the forward formula, backward and central Newton-Gregory formulas are also available. Carnahan, Luther, and Wilkes (1969) can be consulted for further information regarding interpolation for equally spaced data.



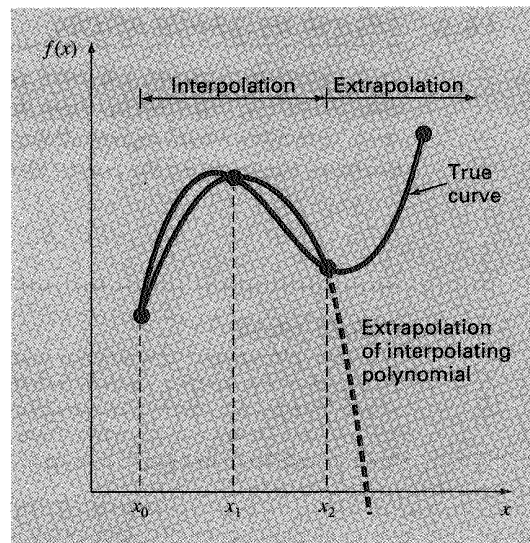
**FIGURE 18.13**

Illustration of the possible divergence of an extrapolated prediction. The extrapolation is based on fitting a parabola through the first three known points.

Because both the Newton and Lagrange polynomials are compatible with arbitrarily spaced data, you might wonder why we address the special case of equally spaced data (Box 18.2). Prior to the advent of digital computers, these techniques had great utility for interpolation from tables with equally spaced arguments. In fact, a computational framework known as a divided-difference table was developed to facilitate the implementation of these techniques. (Figure 18.5 is an example of such a table.)

However, because the formulas are subsets of the computer-compatible Newton and Lagrange schemes and because many tabular functions are available as library subroutines, the need for the equispaced versions has waned. In spite of this, we have included them at this point because of their relevance to later parts of this book. In particular, they are needed to derive numerical integration formulas that typically employ equispaced data (Chap. 21). Because the numerical integration formulas have relevance to the solution of ordinary differential equations, the material in Box 18.2 also has significance to Part Seven.

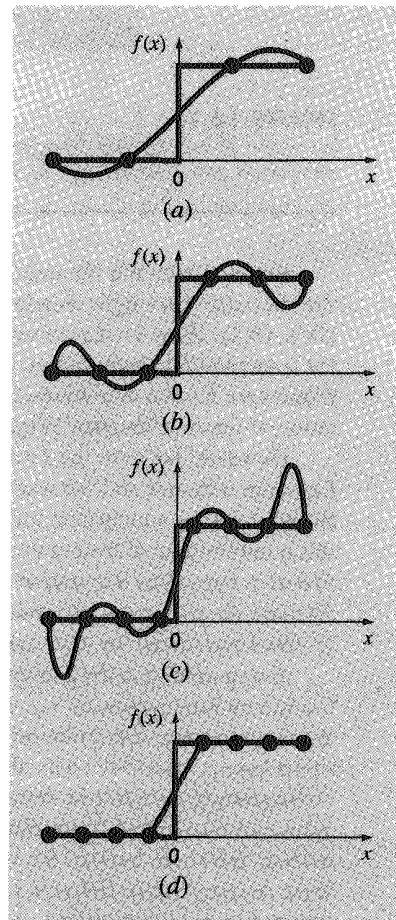
*Extrapolation* is the process of estimating a value of  $f(x)$  that lies outside the range of the known base points,  $x_0, x_1, \dots, x_n$  (Fig. 18.13). In a previous section, we mentioned that the most accurate interpolation is usually obtained when the unknown lies near the center of the base points. Obviously, this is violated when the unknown lies outside the range, and consequently, the error in extrapolation can be very large. As depicted in Fig. 18.13, the open-ended nature of extrapolation represents a step into the unknown because the process extends the curve beyond the known region. As such, the true curve could easily diverge from the prediction. Extreme care should, therefore, be exercised whenever a case arises where one must extrapolate.

## 18.6 SPLINE INTERPOLATION

In the previous sections,  $n$ th-order polynomials were used to interpolate between  $n + 1$  data points. For example, for eight points, we can derive a perfect seventh-order polynomial. This curve would capture all the meanderings (at least up to and including seventh derivatives) suggested by the points. However, there are cases where these functions can lead

**FIGURE 18.14**

A visual representation of a situation where the splines are superior to higher-order interpolating polynomials. The function to be fit undergoes an abrupt increase at  $x = 0$ . Parts (a) through (c) indicate that the abrupt change induces oscillations in interpolating polynomials. In contrast, because it is limited to third-order curves with smooth transitions, the cubic spline (d) provides a much more acceptable approximation.



to erroneous results because of round-off error and overshoot. An alternative approach is to apply lower-order polynomials to subsets of data points. Such connecting polynomials are called *spline functions*.

For example, third-order curves employed to connect each pair of data points are called *cubic splines*. These functions can be constructed so that the connections between adjacent cubic equations are visually smooth. On the surface, it would seem that the third-order approximation of the splines would be inferior to the seventh-order expression. You might wonder why a spline would ever be preferable.

Figure 18.14 illustrates a situation where a spline performs better than a higher-order polynomial. This is the case where a function is generally smooth but undergoes an abrupt change somewhere along the region of interest. The step increase depicted in Fig. 18.14 is an extreme example of such a change and serves to illustrate the point.

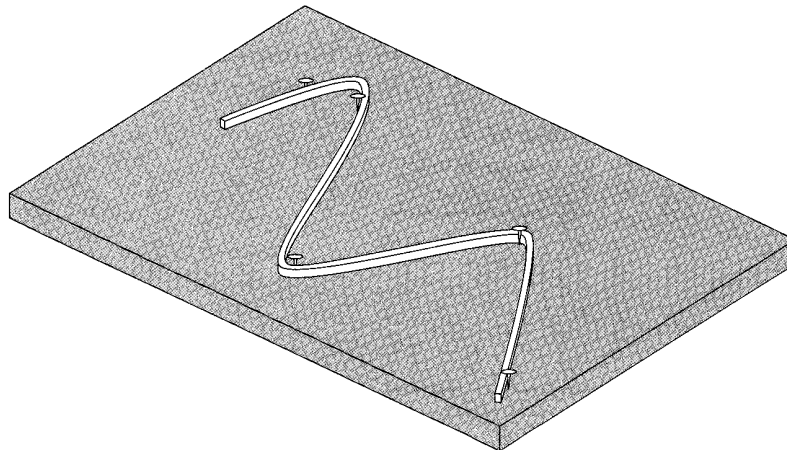
Figure 18.14*a* through *c* illustrates how higher-order polynomials tend to swing through wild oscillations in the vicinity of an abrupt change. In contrast, the spline also connects the points, but because it is limited to third-order changes, the oscillations are kept to a minimum. As such, the spline usually provides a superior approximation of the behavior of functions that have local, abrupt changes.

The concept of the spline originated from the drafting technique of using a thin, flexible strip (called a *spline*) to draw smooth curves through a set of points. The process is depicted in Fig. 18.15 for a series of five pins (data points). In this technique, the drafter places paper over a wooden board and hammers nails or pins into the paper (and board) at the location of the data points. A smooth cubic curve results from interweaving the strip between the pins. Hence, the name “cubic spline” has been adopted for polynomials of this type.

In this section, simple linear functions will first be used to introduce some basic concepts and problems associated with spline interpolation. Then we derive an algorithm for fitting quadratic splines to data. Finally, we present material on the cubic spline, which is the most common and useful version in engineering practice.

**FIGURE 18.15**

The drafting technique of using a spline to draw smooth curves through a series of points. Notice how, at the end points, the spline straightens out. This is called a “natural” spline.



### 18.6.1 Linear Splines

The simplest connection between two points is a straight line. The first-order splines for a group of ordered data points can be defined as a set of linear functions,

$$\begin{aligned} f(x) &= f(x_0) + m_0(x - x_0) & x_0 \leq x \leq x_1 \\ f(x) &= f(x_1) + m_1(x - x_1) & x_1 \leq x \leq x_2 \\ &\vdots \\ &\vdots \\ f(x) &= f(x_{n-1}) + m_{n-1}(x - x_{n-1}) & x_{n-1} \leq x \leq x_n \end{aligned}$$

where  $m_i$  is the slope of the straight line connecting the points:

$$m_i = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \quad (18.27)$$

These equations can be used to evaluate the function at any point between  $x_0$  and  $x_n$  by first locating the interval within which the point lies. Then the appropriate equation is used to determine the function value within the interval. The method is obviously identical to linear interpolation.

#### EXAMPLE 18.8 First-Order Splines

**Problem Statement.** Fit the data in Table 18.1 with first-order splines. Evaluate the function at  $x = 5$ .

**Solution.** The data can be used to determine the slopes between points. For example, for the interval  $x = 4.5$  to  $x = 7$  the slope can be computed using Eq. (18.27):

$$m = \frac{2.5 - 1}{7 - 4.5} = 0.60$$

The slopes for the other intervals can be computed, and the resulting first-order splines are plotted in Fig. 18.16a. The value at  $x = 5$  is 1.3.

**TABLE 18.1**

Data to be fit with spline functions.

$x$	$f(x)$
3.0	2.5
4.5	1.0
7.0	2.5
9.0	0.5

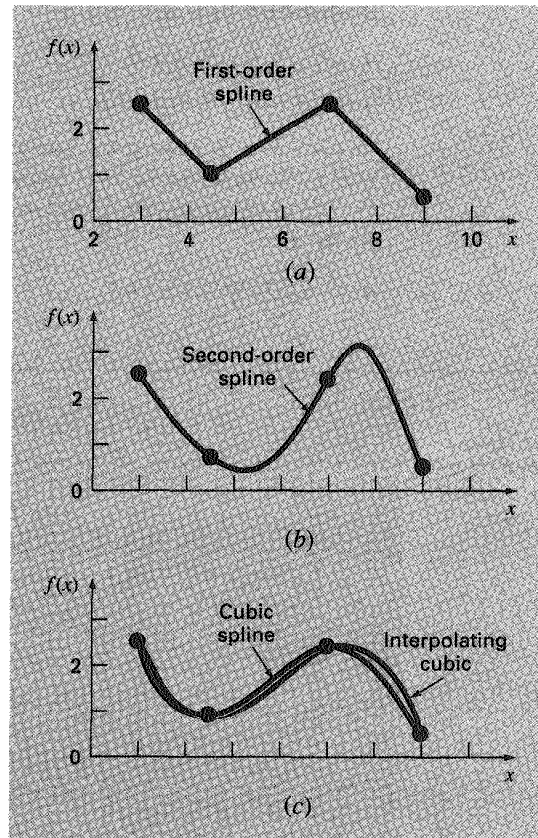
Visual inspection of Fig. 18.16a indicates that the primary disadvantage of first-order splines is that they are not smooth. In essence, at the data points where two splines meet (called a *knot*), the slope changes abruptly. In formal terms, the first derivative of the function is discontinuous at these points. This deficiency is overcome by using higher-order polynomial splines that ensure smoothness at the knots by equating derivatives at these points, as discussed in the next section.

### 18.6.2 Quadratic Splines

To ensure that the  $m$ th derivatives are continuous at the knots, a spline of at least  $m + 1$  order must be used. Third-order polynomials or cubic splines that ensure continuous first and second derivatives are most frequently used in practice. Although third and higher

**FIGURE 18.16**

Spline fits of a set of four points. (a) Linear spline, (b) quadratic spline, and (c) cubic spline, with a cubic interpolating polynomial also plotted.



derivatives could be discontinuous when using cubic splines, they usually cannot be detected visually and consequently are ignored.

Because the derivation of cubic splines is somewhat involved, we have chosen to include them in a subsequent section. We have decided to first illustrate the concept of spline interpolation using second-order polynomials. These “quadratic splines” have continuous first derivatives at the knots. Although quadratic splines do not ensure equal second derivatives at the knots, they serve nicely to demonstrate the general procedure for developing higher-order splines.

The objective in quadratic splines is to derive a second-order polynomial for each interval between data points. The polynomial for each interval can be represented generally as

$$f_i(x) = a_i x^2 + b_i x + c_i \quad (18.28)$$

Figure 18.17 has been included to help clarify the notation. For  $n + 1$  data points ( $i = 0, 1, 2, \dots, n$ ), there are  $n$  intervals and, consequently,  $3n$  unknown constants (the  $a$ 's,  $b$ 's, and  $c$ 's) to evaluate. Therefore,  $3n$  equations or conditions are required to evaluate the unknowns. These are:

1. The function values of adjacent polynomials must be equal at the interior knots. This condition can be represented as

$$a_{i-1}x_{i-1}^2 + b_{i-1}x_{i-1} + c_{i-1} = f(x_{i-1}) \quad (18.29)$$

$$a_i x_{i-1}^2 + b_i x_{i-1} + c_i = f(x_{i-1}) \quad (18.30)$$

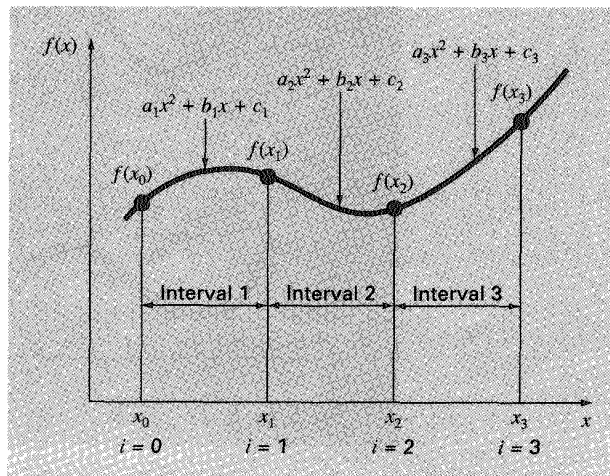
for  $i = 2$  to  $n$ . Because only interior knots are used, Eqs. (18.29) and (18.30) each provide  $n - 1$  for a total of  $2n - 2$  conditions.

2. The first and last functions must pass through the end points. This adds two additional equations:

$$a_1 x_0^2 + b_1 x_0 + c_1 = f(x_0) \quad (18.31)$$

**FIGURE 18.17**

Notation used to derive quadratic splines. Notice that there are  $n$  intervals and  $n + 1$  data points. The example shown is for  $n = 3$ .



$$a_n x_n^2 + b_n x_n + c_n = f(x_n) \quad (18.32)$$

for a total of  $2n - 2 + 2 = 2n$  conditions.

3. *The first derivatives at the interior knots must be equal.* The first derivative of Eq. (18.28) is

$$f'(x) = 2ax + b$$

Therefore, the condition can be represented generally as

$$2a_{i-1}x_{i-1} + b_{i-1} = 2a_i x_{i-1} + b_i \quad (18.33)$$

for  $i = 2$  to  $n$ . This provides another  $n - 1$  conditions for a total of  $2n + n - 1 = 3n - 1$ . Because we have  $3n$  unknowns, we are one condition short. Unless we have some additional information regarding the functions or their derivatives, we must make an arbitrary choice to successfully compute the constants. Although there are a number of different choices that can be made, we select the following:

4. *Assume that the second derivative is zero at the first point.* Because the second derivative of Eq. (18.28) is  $2a_i$ , this condition can be expressed mathematically as

$$a_1 = 0 \quad (18.34)$$

The visual interpretation of this condition is that the first two points will be connected by a straight line.

#### EXAMPLE 18.9 Quadratic Splines

**Problem Statement.** Fit quadratic splines to the same data used in Example 18.8 (Table 18.1). Use the results to estimate the value at  $x = 5$ .

**Solution.** For the present problem, we have four data points and  $n = 3$  intervals. Therefore,  $3(3) = 9$  unknowns must be determined. Equations (18.29) and (18.30) yield  $2(3) - 2 = 4$  conditions:

$$\begin{aligned} 20.25a_1 + 4.5b_1 + c_1 &= 1.0 \\ 20.25a_2 + 4.5b_2 + c_2 &= 1.0 \\ 49a_2 + 7b_2 + c_2 &= 2.5 \\ 49a_3 + 7b_3 + c_3 &= 2.5 \end{aligned}$$

Passing the first and last functions through the initial and final values adds 2 more [Eq. (18.31)]:

$$9a_1 + 3b_1 + c_1 = 2.5$$

and [Eq. (18.32)]

$$81a_3 + 9b_3 + c_3 = 0.5$$

Continuity of derivatives creates an additional  $3 - 1 = 2$  [Eq. (18.33)]:

$$\begin{aligned} 9a_1 + b_1 &= 9a_2 + b_2 \\ 14a_2 + b_2 &= 14a_3 + b_3 \end{aligned}$$

Finally, Eq. (18.34) specifies that  $a_1 = 0$ . Because this equation specifies  $a_1$  exactly, the problem reduces to solving eight simultaneous equations. These conditions can be expressed in matrix form as

$$\begin{bmatrix} 4.5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 20.25 & 4.5 & 1 & 0 & 0 & 0 \\ 0 & 0 & 49 & 7 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 49 & 7 & 1 \\ 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 81 & 9 & 1 \\ 1 & 0 & -9 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 14 & 1 & 0 & -14 & -1 & 0 \end{bmatrix} \begin{Bmatrix} b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \\ c_3 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 1 \\ 2.5 \\ 2.5 \\ 2.5 \\ 0.5 \\ 0 \\ 0 \end{Bmatrix}$$

These equations can be solved using techniques from Part Three, with the results:

$$\begin{aligned} a_1 &= 0 & b_1 &= -1 & c_1 &= 5.5 \\ a_2 &= 0.64 & b_2 &= -6.76 & c_2 &= 18.46 \\ a_3 &= -1.6 & b_3 &= 24.6 & c_3 &= -91.3 \end{aligned}$$

which can be substituted into the original quadratic equations to develop the following relationships for each interval:

$$\begin{aligned} f_1(x) &= -x + 5.5 & 3.0 \leq x \leq 4.5 \\ f_2(x) &= 0.64x^2 - 6.76x + 18.46 & 4.5 \leq x \leq 7.0 \\ f_3(x) &= -1.6x^2 + 24.6x - 91.3 & 7.0 \leq x \leq 9.0 \end{aligned}$$

When we use  $f_2$ , the prediction for  $x = 5$  is, therefore,

$$f_2(5) = 0.64(5)^2 - 6.76(5) + 18.46 = 0.66$$

The total spline fit is depicted in Fig. 18.16*b*. Notice that there are two shortcomings that detract from the fit: (1) the straight line connecting the first two points and (2) the spline for the last interval seems to swing too high. The cubic splines in the next section do not exhibit these shortcomings and, as a consequence, are better methods for spline interpolation.

### 18.6.3 Cubic Splines

The objective in cubic splines is to derive a third-order polynomial for each interval between knots, as in

$$f_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \quad (18.35)$$

Thus, for  $n + 1$  data points ( $i = 0, 1, 2, \dots, n$ ), there are  $n$  intervals and, consequently,  $4n$  unknown constants to evaluate. Just as for quadratic splines,  $4n$  conditions are required to evaluate the unknowns. These are:

1. The function values must be equal at the interior knots ( $2n - 2$  conditions).
2. The first and last functions must pass through the end points (2 conditions).



3. The first derivatives at the interior knots must be equal ( $n - 1$  conditions).
4. The second derivatives at the interior knots must be equal ( $n - 1$  conditions).
5. The second derivatives at the end knots are zero (2 conditions).

The visual interpretation of condition 5 is that the function becomes a straight line at the end knots. Specification of such an end condition leads to what is termed a “natural” spline. It is given this name because the drafting spline naturally behaves in this fashion (Fig. 18.15). If the value of the second derivative at the end knots is nonzero (that is, there is some curvature), this information can be used alternatively to supply the two final conditions.

The above five types of conditions provide the total of  $4n$  equations required to solve for the  $4n$  coefficients. Whereas it is certainly possible to develop cubic splines in this fashion, we will present an alternative technique that requires the solution of only  $n - 1$  equations. Although the derivation of this method (Box 18.3) is somewhat less straightforward than that for quadratic splines, the gain in efficiency is well worth the effort.

### Box 18.3 Derivation of Cubic Splines

The first step in the derivation (Cheney and Kincaid, 1985) is based on the observation that because each pair of knots is connected by a cubic, the second derivative within each interval is a straight line. Equation (18.35) can be differentiated twice to verify this observation. On this basis, the second derivatives can be represented by a first-order Lagrange interpolating polynomial [Eq. (18.22)]:

$$f_i''(x) = f_i''(x_{i-1}) \frac{x - x_i}{x_{i-1} - x_i} + f_i''(x_i) \frac{x - x_{i-1}}{x_i - x_{i-1}} \quad (\text{B18.3.1})$$

where  $f_i''(x)$  is the value of the second derivative at any point  $x$  within the  $i$ th interval. Thus, this equation is a straight line connecting the second derivative at the first knot  $f''(x_{i-1})$  with the second derivative at the second knot  $f''(x_i)$ .

Next, Eq. (B18.3.1) can be integrated twice to yield an expression for  $f_i(x)$ . However, this expression will contain two unknown constants of integration. These constants can be evaluated by invoking the function-equality conditions— $f(x)$  must equal  $f(x_{i-1})$  at  $x_{i-1}$  and  $f(x)$  must equal  $f(x_i)$  at  $x_i$ . By performing these evaluations, the following cubic equation results:

$$\begin{aligned} f_i(x) &= \frac{f_i''(x_{i-1})}{6(x_i - x_{i-1})} (x_i - x)^3 + \frac{f_i''(x_i)}{6(x_i - x_{i-1})} (x - x_{i-1})^3 \\ &+ \left[ \frac{f(x_{i-1})}{x_i - x_{i-1}} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6} \right] (x_i - x) \\ &+ \left[ \frac{f(x_i)}{x_i - x_{i-1}} - \frac{f''(x_i)(x_i - x_{i-1})}{6} \right] (x - x_{i-1}) \end{aligned} \quad (\text{B18.3.2})$$

Now, admittedly, this relationship is a much more complicated expression for the cubic spline for the  $i$ th interval than, say,

Eq. (18.35). However, notice that it contains only two unknown “coefficients,” the second derivatives at the beginning and the end of the interval— $f''(x_{i-1})$  and  $f''(x_i)$ . Thus, if we can determine the proper second derivative at each knot, Eq. (B18.3.2) is a third-order polynomial that can be used to interpolate within the interval.

The second derivatives can be evaluated by invoking the condition that the first derivatives at the knots must be continuous:

$$f'_{i-1}(x_i) = f'_i(x_i) \quad (\text{B18.3.3})$$

Equation (B18.3.2) can be differentiated to give an expression for the first derivative. If this is done for both the  $(i - 1)$ th and the  $i$ th intervals and the two results are set equal according to Eq. (B18.3.3), the following relationship results:

$$\begin{aligned} (x_i - x_{i-1})f''(x_{i-1}) + 2(x_{i+1} - x_{i-1})f''(x_i) \\ + (x_{i+1} - x_i)f''(x_{i+1}) \\ = \frac{6}{x_{i+1} - x_i} [f(x_{i+1}) - f(x_i)] \\ + \frac{6}{x_i - x_{i-1}} [f(x_{i-1}) - f(x_i)] \end{aligned} \quad (\text{B18.3.4})$$

If Eq. (B18.3.4) is written for all interior knots,  $n - 1$  simultaneous equations result with  $n + 1$  unknown second derivatives. However, because this is a natural cubic spline, the second derivatives at the end knots are zero and the problem reduces to  $n - 1$  equations with  $n - 1$  unknowns. In addition, notice that the system of equations will be tridiagonal. Thus, not only have we reduced the number of equations but we have also cast them in a form that is extremely easy to solve (recall Sec. 11.1.1).

The derivation from Box 18.3 results in the following cubic equation for each interval:

$$\begin{aligned}
 f_i(x) = & \frac{f_i''(x_{i-1})}{6(x_i - x_{i-1})}(x_i - x)^3 + \frac{f_i''(x_i)}{6(x_i - x_{i-1})}(x - x_{i-1})^3 \\
 & + \left[ \frac{f(x_{i-1})}{x_i - x_{i-1}} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6} \right] (x_i - x) \\
 & + \left[ \frac{f(x_i)}{x_i - x_{i-1}} - \frac{f''(x_i)(x_i - x_{i-1})}{6} \right] (x - x_{i-1})
 \end{aligned} \tag{18.36}$$

This equation contains only two unknowns—the second derivatives at the end of each interval. These unknowns can be evaluated using the following equation:

$$\begin{aligned}
 (x_i - x_{i-1})f''(x_{i-1}) + 2(x_{i+1} - x_{i-1})f''(x_i) + (x_{i+1} - x_i)f''(x_{i+1}) \\
 = \frac{6}{x_{i+1} - x_i}[f(x_{i+1}) - f(x_i)] + \frac{6}{x_i - x_{i-1}}[f(x_{i-1}) - f(x_i)]
 \end{aligned} \tag{18.37}$$

If this equation is written for all the interior knots,  $n - 1$  simultaneous equations result with  $n - 1$  unknowns. (Remember, the second derivatives at the end knots are zero.) The application of these equations is illustrated in the following example.

#### EXAMPLE 18.10 Cubic Splines

**Problem Statement.** Fit cubic splines to the same data used in Examples 18.8 and 18.9 (Table 18.1). Utilize the results to estimate the value at  $x = 5$ .

**Solution.** The first step is to employ Eq. (18.37) to generate the set of simultaneous equations that will be utilized to determine the second derivatives at the knots. For example, for the first interior knot, the following data is used:

$$\begin{aligned}
 x_0 = 3 & & f(x_0) = 2.5 \\
 x_1 = 4.5 & & f(x_1) = 1 \\
 x_2 = 7 & & f(x_2) = 2.5
 \end{aligned}$$

These values can be substituted into Eq. (18.37) to yield

$$\begin{aligned}
 (4.5 - 3)f''(3) + 2(7 - 3)f''(4.5) + (7 - 4.5)f''(7) \\
 = \frac{6}{7 - 4.5}(2.5 - 1) + \frac{6}{4.5 - 3}(2.5 - 1)
 \end{aligned}$$

Because of the natural spline condition,  $f''(3) = 0$ , and the equation reduces to

$$8f''(4.5) + 2.5f''(7) = 9.6$$

In a similar fashion, Eq. (18.37) can be applied to the second interior point to give

$$2.5f''(4.5) + 9f''(7) = -9.6$$

These two equations can be solved simultaneously for

$$\begin{aligned}
 f''(4.5) &= 1.67909 \\
 f''(7) &= -1.53308
 \end{aligned}$$

These values can then be substituted into Eq. (18.36), along with values for the  $x$ 's and the  $f(x)$ 's, to yield

$$f_1(x) = \frac{1.67909}{6(4.5 - 3)}(x - 3)^3 + \frac{2.5}{4.5 - 3}(4.5 - x) + \left[ \frac{1}{4.5 - 3} - \frac{1.67909(4.5 - 3)}{6} \right](x - 3)$$

or

$$f_1(x) = 0.186566(x - 3)^3 + 1.666667(4.5 - x) + 0.246894(x - 3)$$

This equation is the cubic spline for the first interval. Similar substitutions can be made to develop the equations for the second and third intervals:

$$f_2(x) = 0.111939(7 - x)^3 - 0.102205(x - 4.5)^3 - 0.299621(7 - x) + 1.638783(x - 4.5)$$

and

$$f_3(x) = -0.127757(9 - x)^3 + 1.761027(9 - x) + 0.25(x - 7)$$

The three equations can then be employed to compute values within each interval. For example, the value at  $x = 5$ , which falls within the second interval, is calculated as

$$f_2(5) = 0.111939(7 - 5)^3 - 0.102205(5 - 4.5)^3 - 0.299621(7 - 5) + 1.638783(5 - 4.5) = 1.102886$$

Other values are computed and the results are plotted in Fig. 18.16c.

The results of Examples 18.8 through 18.10 are summarized in Fig. 18.16. Notice the progressive improvement of the fit as we move from linear to quadratic to cubic splines. We have also superimposed a cubic interpolating polynomial on Fig. 18.16c. Although the cubic spline consists of a series of third-order curves, the resulting fit differs from that obtained using the third-order polynomial. This is due to the fact that the natural spline requires zero second derivatives at the end knots, whereas the cubic polynomial has no such constraint.

### 18.6.4 Computer Algorithm for Cubic Splines

The method for calculating cubic splines outlined in the previous section is ideal for computer implementation. Recall that, by some clever manipulations, the method reduced to solving  $n - 1$  simultaneous equations. An added benefit of the derivation is that, as specified by Eq. (18.37), the system of equations is tridiagonal. As described in Sec. 11.1, algorithms are available to solve such systems in an extremely efficient manner. Figure 18.18 outlines a computational framework that incorporates these features.

Note that the routine in Fig. 18.18 returns a single interpolated value,  $y_u$ , for a given value of the dependent variable,  $x_u$ . This is but one way in which spline interpolation can be implemented. For example, you might want to determine the coefficients once, and then

```

SUBROUTINE Spline (x,y,n,xu,yu,dy,d2y)
LOCAL en, fn, gn, rn, d2xn
CALL Tridiag(x,y,n,e,f,g,r)
CALL Decomp(e,f,g,n-1)
CALL Subst(e,f,g,r,n-1,d2x)
CALL Interpol(x,y,n,d2x,xu,yu,dy,d2y)
END Spline

SUBROUTINE Tridiag (x, y, n, e, f, g, r)
f1 = 2 * (x2-x0)
g1 = (x2-x1)
r1 = 6 / (x2-x1) * (y2-y1)
r1 = r1 + 6 / (x1-x0) * (y0-y1)
DO i = 2, n - 2
    ei = (xi - xi-1)
    fi = 2 * (xi+1 - xi-1)
    gi = (xi+1 - xi)
    ri = 6 / (xi+1 - xi) * (yi+1 - yi)
    ri = ri + 6 / (xi - xi-1) * (yi-1 - yi)
END DO
en-1 = (xn-1 - xn-2)
fn-1 = 2 * (xn - xn-2)
rn-1 = 6 / (xn - xn-1) * (yn - yn-1)
rn-1 = rn-1 + 6 / (xn-1 - xn-2) * (yn-2 - yn-1)
END Tridiag

```

```

SUBROUTINE Interpol (x,y,n,d2x,xu,yu,dy,d2y)
flag = 0
i = 1
DO
    IF xu ≥ xi-1 AND xu ≤ xi THEN
        c1 = d2xi-1 / 6 / (xi - xi-1)
        c2 = d2xi / 6 / (xi - xi-1)
        c3 = (yi-1 / (xi - xi-1) - d2xi-1 * (xi - xi-1) / 6)
        c4 = (yi / (xi - xi-1) - d2xi * (xi - xi-1) / 6)
        t1 = c1 * (xi - xu)3
        t2 = c2 * (xu - xi-1)3
        t3 = c3 * (xi - xu)
        t4 = c4 * (xu - xi-1)
        yu = t1 + t2 + t3 + t4
        t1 = -3 * c1 * (xi - xu)2
        t2 = 3 * c2 * (xu - xi-1)2
        t3 = -c3
        t4 = c4
        dy = t1 + t2 + t3 + t4
        t1 = 6 * c1 * (xi - xu)
        t2 = 6 * c2 * (xu - xi-1)
        d2y = t1 + t2
        flag = 1
    ELSE
        i = i + 1
    END IF
    IF i = n + 1 OR flag = 1 EXIT
END DO
IF flag = 0 THEN
    PRINT "outside range"
    pause
END IF
END Interpol

```

**FIGURE 18.18**

Algorithm for cubic spline interpolation.

perform many interpolations. In addition, the routine returns both the first ( $dy$ ) and second ( $dy2$ ) derivative at  $x_u$ . Although it is not necessary to compute these quantities, they prove useful in many applications of spline interpolation.

## PROBLEMS

**18.1** Estimate the logarithm of 5 to the base 10 ( $\log 5$ ) using linear interpolation.

- (a) Interpolate between  $\log 4 = 0.60206$  and  $\log 6 = 0.7781513$ .  
 (b) Interpolate between  $\log 4.5 = 0.6532125$  and  $\log 5.5 = 0.7403627$ .

For each of the interpolations, compute the percent relative error based on the true value.

**18.2** Fit a second-order Newton's interpolating polynomial to estimate  $\log 5$  using the data from Prob. 18.1. Compute the true percent relative error.

**18.3** Fit a third-order Newton's interpolating polynomial to estimate  $\log 5$  using the data from Prob. 18.1.

**18.4** Given the data

$x$	1	2	2.5	3	4	5
$f(x)$	1	5	7	8	2	1

- (a) Calculate  $f(3.4)$  using Newton's interpolating polynomials of order 1 through 3. Choose the sequence of the points for your estimates to attain the best possible accuracy.  
 (b) Utilize Eq. (18.18) to estimate the error for each prediction.

**18.5** Given the data

$x$	1	2	3	5	6
$f(x)$	4.75	4	5.25	19.75	36

Calculate  $f(4)$  using Newton's interpolating polynomials of order 1 through 4. Choose your base points to attain good accuracy. What do your results indicate regarding the order of the polynomial used to generate the data in the table?

**18.6** Repeat Probs. 18.1 through 18.3 using the Lagrange polynomial.

**18.7** Repeat Prob. 18.5 using the Lagrange polynomial of order 1 through 3.

**18.8** Employ inverse interpolation using a cubic interpolating polynomial and bisection to determine the value of  $x$  that corresponds to  $f(x) = 0.3$  for the following tabulated data,

$x$	1	2	3	4	5	6	7
$f(x)$	1	0.5	0.3333	0.25	0.2	0.1667	0.1429

**18.9** Employ inverse interpolation to determine the value of  $x$  that corresponds to  $f(x) = 0.93$  for the following tabulated data,

$x$	0	1	2	3	4	5
$f(x)$	0	0.5	0.8	0.9	0.941176	0.961538

Note that the values in the table were generated with the function  $f(x) = x^2/(1 + x^2)$ .

- (a) Determine the correct value analytically.  
 (b) Use quadratic interpolation and the quadratic formula to determine the value numerically.  
 (c) Use cubic interpolation and bisection to determine the value numerically.

**18.10** Develop quadratic splines for the first 5 data points in Prob. 18.4 and predict  $f(3.4)$  and  $f(2.2)$ .

**18.11** Develop cubic splines for the data in Prob. 18.5 and (a) predict  $f(4)$  and  $f(2.5)$  and (b) verify that  $f_2(3)$  and  $f_3(3) = 5.25$ .

**18.12** Determine the coefficients of the parabola that passes through the last three points in Prob. 18.4.

**18.13** Determine the coefficients of the cubic equation that passes through the first four points in Prob. 18.5.

**18.14** Develop, debug, and test a subprogram in either a high-level language or macro language of your choice to implement Newton's interpolating polynomial based on Fig. 18.7.

**18.15** Test the program you developed in Prob. 18.14 by duplicating the computation from Example 18.5.

**18.16** Use the program you developed in Prob. 18.14 to solve Probs. 18.1 through 18.3.

**18.17** Use the program you developed in Prob. 18.14 to solve Probs. 18.4 and 18.5. In Prob. 18.4, utilize all the data to develop first- through fifth-order polynomials. For both problems, plot the estimated error versus order.

**18.18** Develop, debug, and test a subprogram in either a high-level language or macro language of your choice to implement Lagrange interpolation. Base it on the pseudocode from Fig. 18.11. Test it by duplicating Example 18.7.

**18.19** A useful application of Lagrange interpolation is called a *table look-up*. As the name implies, this involves "looking-up" an intermediate value from a table. To develop such an algorithm, the

table of  $x$  and  $f(x)$  values are first stored in a pair of one-dimensional arrays. These values are then passed to a function along with the  $x$  value you wish to evaluate. The function then performs two tasks. First, it loops down through the table until it finds the interval within which the unknown lies. Then it applies a technique like Lagrange interpolation to determine the proper  $f(x)$  value. Develop such a function using a cubic Lagrange polynomial to perform the interpolation. For intermediate intervals, this is a nice choice because the unknown will be located in the interval in the middle of the four points necessary to generate the cubic. Be careful at the first and last intervals where this is not the case. Also have your code detect when

the user requests a value outside the range of  $x$ 's. For such cases, the function should display an error message. Test your program for  $f(x) = \ln x$  using data from  $x = 0, 1, 2, \dots, 10$ .

**18.20** Develop, debug, and test a subprogram in either a high-level language or macro language of your choice to implement cubic spline interpolation based on Fig. 18.18. Test the program by duplicating Example 18.10.

**18.21** Use the software developed in Prob. 18.20 to fit cubic splines through the data in Probs. 18.4 and 18.5. For both cases, predict  $f(2.25)$ .

# CHAPTER 19

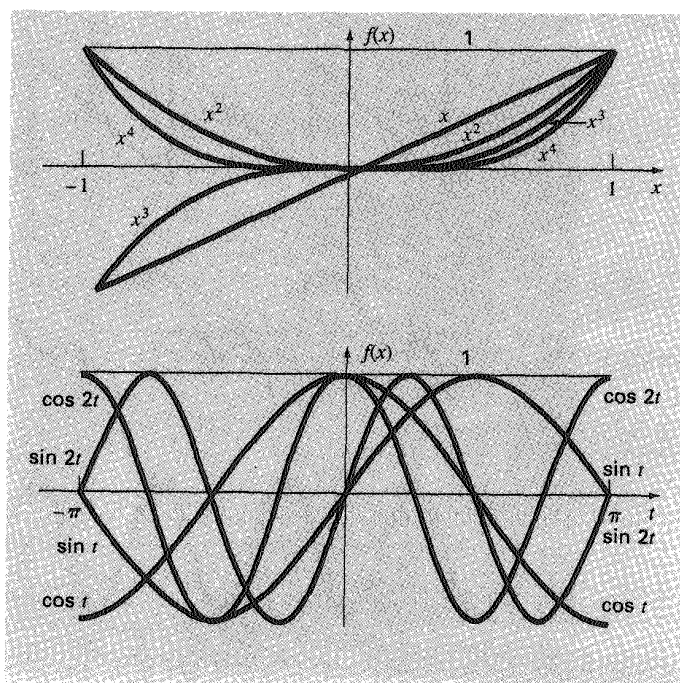
## Fourier Approximation

To this point, our presentation of interpolation has emphasized standard polynomials—that is, linear combinations of the monomials  $1, x, x^2, \dots, x^m$  (Fig. 19.1a). We now turn to another class of functions that has immense importance in engineering. These are the trigonometric functions  $1, \cos x, \cos 2x, \dots, \cos nx, \sin x, \sin 2x, \dots, \sin nx$  (Fig. 19.1b).

Engineers often deal with systems that oscillate or vibrate. As might be expected, trigonometric functions play a fundamental role in modeling such problem contexts.

**FIGURE 19.1**

The first five (a) monomials and (b) trigonometric functions. Note that for the intervals shown, both types of function range in value between  $-1$  and  $1$ . However, notice that the peak values for the monomials all occur at the extremes whereas for the trigonometric functions the peaks are more uniformly distributed across the interval.



*Fourier approximation* represents a systematic framework for using trigonometric series for this purpose.

One of the hallmarks of a Fourier analysis is that it deals with both the time and the frequency domains. Because some engineers are not comfortable with the latter, we have devoted a large fraction of the subsequent material to a general overview of Fourier approximation. An important aspect of this overview will be to familiarize you with the frequency domain. This orientation is then followed by an introduction to numerical methods for computing discrete Fourier transforms.

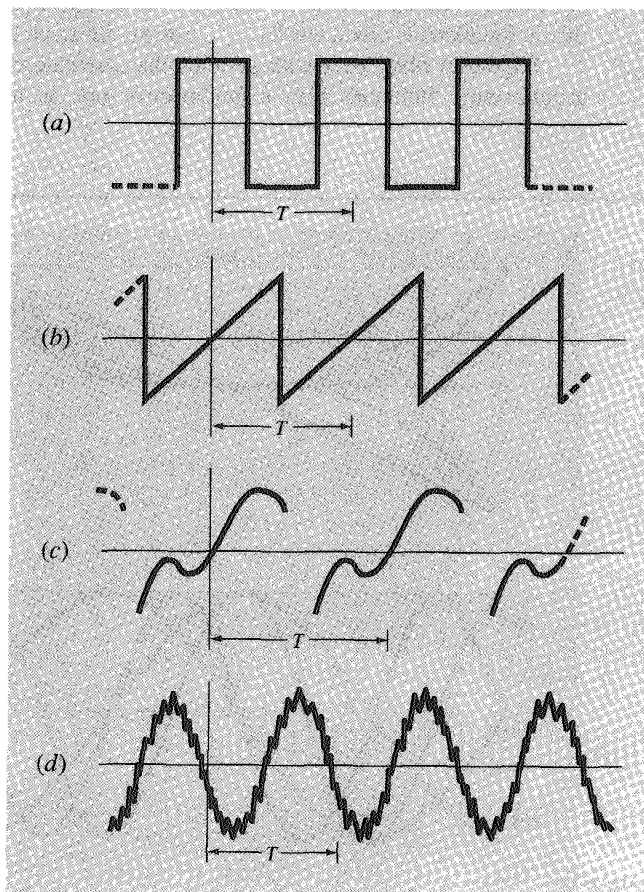
## 19.1 CURVE FITTING WITH SINUSOIDAL FUNCTIONS

A periodic function  $f(t)$  is one for which

$$f(t) = f(t + T) \quad (19.1)$$

**FIGURE 19.2**

Aside from trigonometric functions such as sines and cosines, periodic functions include waveforms such as (a) the square wave and (b) the sawtooth wave. Beyond these idealized forms, periodic signals in nature can be (c) nonideal and (d) contaminated by noise. The trigonometric functions can be used to represent and to analyze all these cases.





where  $T$  is a constant called the *period* that is the smallest value for which Eq. (19.1) holds. Common examples include waveforms such as square and sawtooth waves (Fig. 19.2). The most fundamental are sinusoidal functions.

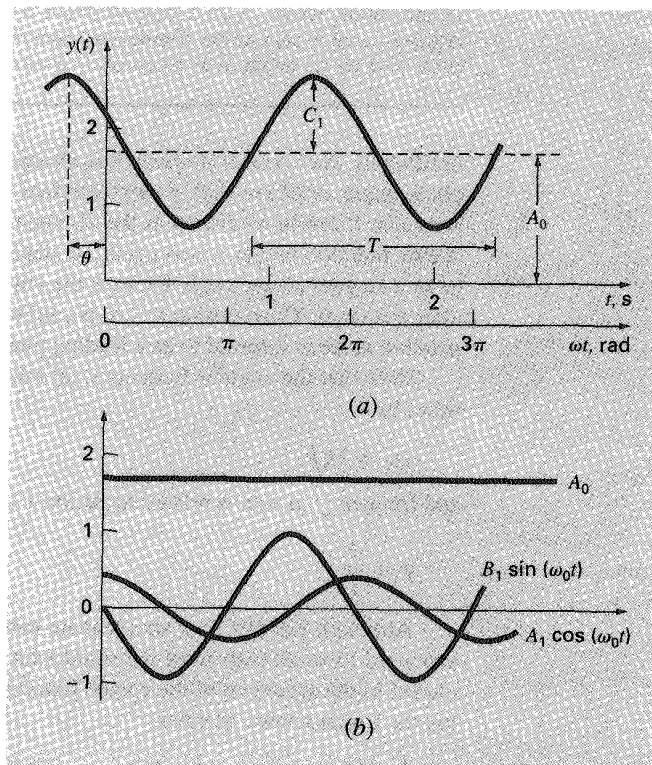
In the present discussion, we will use the term *sinusoid* to represent any waveform that can be described as a sine or cosine. There is no clear-cut convention for choosing either function, and in any case, the results will be identical. For this chapter we will use the cosine, which is expressed generally as

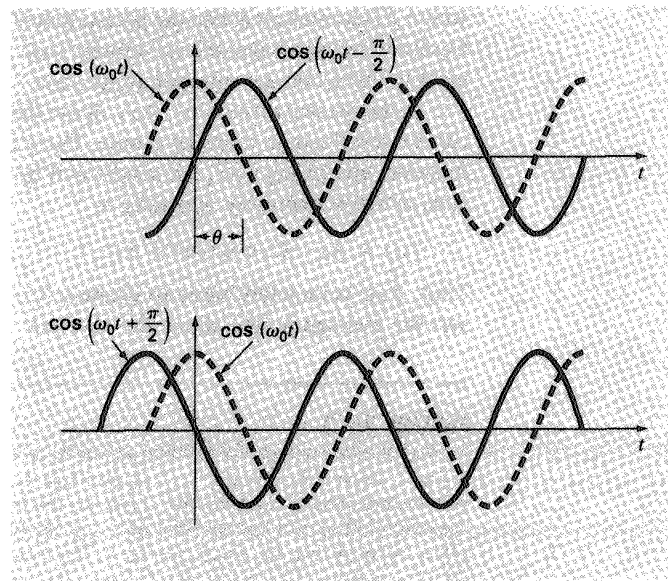
$$f(t) = A_0 + C_1 \cos(\omega_0 t + \theta) \quad (19.2)$$

Thus, four parameters serve to characterize the sinusoid (Fig. 19.3). The *mean value*  $A_0$  sets the average height above the abscissa. The *amplitude*  $C_1$  specifies the height of the

**FIGURE 19.3**

(a) A plot of the sinusoidal function  $y(t) = A_0 + C_1 \cos(\omega_0 t + \theta)$ . For this case,  $A_0 = 1.7$ ,  $C_1 = 1$ ,  $\omega_0 = 2\pi/T = 2\pi/(1.5 \text{ s})$ , and  $\theta = \pi/3$  radians  $= 1.0472$  ( $\approx 0.25 \text{ s}$ ). Other parameters used to describe the curve are the frequency  $f = \omega_0/(2\pi)$ , which for this case is 1 cycle/(1.5 s) and the period  $T = 1.5 \text{ s}$ . (b) An alternative expression of the same curve is  $y(t) = A_0 + A_1 \cos(\omega_0 t) + B_1 \sin(\omega_0 t)$ . The three components of this function are depicted in (b), where  $A_1 = 0.5$  and  $B_1 = -0.866$ . The summation of the three curves in (b) yields the single curve in (a).



**FIGURE 19.4**

Graphical depictions of (a) a lagging phase angle and (b) a leading phase angle. Note that the lagging curve in (a) can be alternatively described as  $\cos(\omega_0 t + 3\pi/2)$ . In other words, if a curve lags by an angle of  $\alpha$ , it can also be represented as leading by  $2\pi - \alpha$ .

oscillation. The *angular frequency*  $\omega_0$  characterizes how often the cycles occur. Finally, the phase angle, or *phase shift*,  $\theta$  parameterizes the extent to which the sinusoid is shifted horizontally. It can be measured as the distance in radians from  $t = 0$  to the point at which the cosine function begins a new cycle. As depicted in Fig. 19.4a, a negative value is referred to as a *lagging phase angle* because the curve  $\cos(\omega_0 t - \theta)$  begins a new cycle  $\theta$  radians after  $\cos(\omega_0 t)$ . Thus,  $\cos(\omega_0 t - \theta)$  is said to lag  $\cos(\omega_0 t)$ . Conversely, as in Fig. 19.4b, a positive value is referred to as a *leading phase angle*.

Note that the angular frequency (in radians/time) is related to frequency  $f$  (in cycles/time) by

$$\omega_0 = 2\pi f \quad (19.3)$$

and frequency in turn is related to period  $T$  (in units of time) by

$$f = \frac{1}{T} \quad (19.4)$$

Although Eq. (19.2) is an adequate mathematical characterization of a sinusoid, it is awkward to work with from the standpoint of curve fitting because the phase shift is included in the argument of the cosine function. This deficiency can be overcome by invoking the trigonometric identity

$$C_1 \cos(\omega_0 t + \theta) = C_1 [\cos(\omega_0 t) \cos(\theta) - \sin(\omega_0 t) \sin(\theta)] \quad (19.5)$$

Substituting Eq. (19.5) into Eq. (19.2) and collecting terms gives (Fig. 19.3*b*)

$$f(t) = A_0 + A_1 \cos(\omega_0 t) + B_1 \sin(\omega_0 t) \quad (19.6)$$

where

$$A_1 = C_1 \cos(\theta) \quad B_1 = -C_1 \sin(\theta) \quad (19.7)$$

Dividing the two parts of Eq. (19.7) gives

$$\theta = \arctan\left(-\frac{B_1}{A_1}\right) \quad (19.8)$$

where, if  $A_1 < 0$ , add  $\pi$  to  $\theta$ . Squaring and summing Eq. (19.7) leads to

$$C_1 = \sqrt{A_1^2 + B_1^2} \quad (19.9)$$

Thus, Eq. (19.6) represents an alternative formulation of Eq. (19.2) that still requires four parameters but that is cast in the format of a general linear model [recall Eq. (17.23)]. As we will discuss in the next section, it can be simply applied as the basis for a least-squares fit.

Before proceeding to the next section, however, we should stress that we could have employed a sine rather than a cosine as our fundamental model of Eq. (19.2). For example,

$$f(t) = A_0 + C_1 \sin(\omega_0 t + \delta)$$

could have been used. Simple relationships can be applied to convert between the two forms

$$\sin(\omega_0 t + \delta) = \cos\left(\omega_0 t + \delta - \frac{\pi}{2}\right)$$

and

$$\cos(\omega_0 t + \theta) = \sin\left(\omega_0 t + \theta + \frac{\pi}{2}\right) \quad (19.10)$$

In other words,  $\theta = \delta - \pi/2$ . The only important consideration is that one or the other format should be used consistently. Thus, we will use the cosine version throughout our discussion.

### 19.1.1 Least-Squares Fit of a Sinusoid

Equation (19.6) can be thought of as a linear least-squares model

$$y = A_0 + A_1 \cos(\omega_0 t) + B_1 \sin(\omega_0 t) + e \quad (19.11)$$

which is just another example of the general model [recall Eq. (17.23)]

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + \cdots + a_m z_m + e \quad (17.23)$$

where  $z_0 = 1$ ,  $z_1 = \cos(\omega_0 t)$ ,  $z_2 = \sin(\omega_0 t)$ , and all other  $z$ 's = 0. Thus, our goal is to determine coefficient values that minimize

$$S_r = \sum_{i=1}^N \{y_i - [A_0 + A_1 \cos(\omega_0 t_i) + B_1 \sin(\omega_0 t_i)]\}^2$$

The normal equations to accomplish this minimization can be expressed in matrix form as [recall Eq. (17.25)]

$$\begin{bmatrix} N & \Sigma \cos(\omega_0 t) & \Sigma \sin(\omega_0 t) \\ \Sigma \cos(\omega_0 t) & \Sigma \cos^2(\omega_0 t) & \Sigma \cos(\omega_0 t) \sin(\omega_0 t) \\ \Sigma \sin(\omega_0 t) & \Sigma \cos(\omega_0 t) \sin(\omega_0 t) & \Sigma \sin^2(\omega_0 t) \end{bmatrix} \begin{Bmatrix} A_0 \\ A_1 \\ B_1 \end{Bmatrix} = \begin{Bmatrix} \Sigma y \\ \Sigma y \cos(\omega_0 t) \\ \Sigma y \sin(\omega_0 t) \end{Bmatrix} \quad (19.12)$$

These equations can be employed to solve for the unknown coefficients. However, rather than do this, we can examine the special case where there are  $N$  observations equispaced at intervals of  $\Delta t$  and with a total record length of  $T = (N - 1) \Delta t$ . For this situation, the following average values can be determined (see Prob. 19.3):

$$\begin{aligned} \frac{\Sigma \sin(\omega_0 t)}{N} &= 0 & \frac{\Sigma \cos(\omega_0 t)}{N} &= 0 \\ \frac{\Sigma \sin^2(\omega_0 t)}{N} &= \frac{1}{2} & \frac{\Sigma \cos^2(\omega_0 t)}{N} &= \frac{1}{2} \\ \frac{\Sigma \cos(\omega_0 t) \sin(\omega_0 t)}{N} &= 0 \end{aligned} \quad (19.13)$$

Thus, for equispaced points the normal equations become

$$\begin{bmatrix} N & 0 & 0 \\ 0 & N/2 & 0 \\ 0 & 0 & N/2 \end{bmatrix} \begin{Bmatrix} A_0 \\ A_1 \\ B_1 \end{Bmatrix} = \begin{Bmatrix} \Sigma y \\ \Sigma y \cos(\omega_0 t) \\ \Sigma y \sin(\omega_0 t) \end{Bmatrix}$$

The inverse of a diagonal matrix is merely another diagonal matrix whose elements are the reciprocals of the original. Thus, the coefficients can be determined as

$$\begin{Bmatrix} A_0 \\ A_1 \\ B_1 \end{Bmatrix} = \begin{bmatrix} 1/N & 0 & 0 \\ 0 & 2/N & 0 \\ 0 & 0 & 2/N \end{bmatrix} \begin{Bmatrix} \Sigma y \\ \Sigma y \cos(\omega_0 t) \\ \Sigma y \sin(\omega_0 t) \end{Bmatrix}$$

or

$$A_0 = \frac{\Sigma y}{N} \quad (19.14)$$

$$A_1 = \frac{2}{N} \Sigma y \cos(\omega_0 t) \quad (19.15)$$

$$B_1 = \frac{2}{N} \Sigma y \sin(\omega_0 t) \quad (19.16)$$

#### EXAMPLE 19.1 Least-Squares Fit of a Sinusoid

**Problem Statement.** The curve in Fig. 19.3 is described by  $y = 1.7 + \cos(4.189t + 1.0472)$ . Generate 10 discrete values for this curve at intervals of  $\Delta t = 0.15$  for the range

$t = 0$  to 1.35. Use this information to evaluate the coefficients of Eq. (19.11) by a least-squares fit.

**Solution.** The data required to evaluate the coefficients with  $\omega = 4.189$  are

$t$	$y$	$y \cos (\omega_0 t)$	$y \sin (\omega_0 t)$
0	2.200	2.200	0.000
0.15	1.595	1.291	0.938
0.30	1.031	0.319	0.980
0.45	0.722	-0.223	0.687
0.60	0.786	-0.636	0.462
0.75	1.200	-1.200	0.000
0.90	1.805	-1.460	-1.061
1.05	2.369	-0.732	-2.253
1.20	2.678	0.829	-2.547
1.35	2.614	2.114	-1.536
$\Sigma =$	17.000	2.502	-4.330

These results can be used to determine [Eqs. (19.14) through (19.16)]

$$A_0 = \frac{17.000}{10} = 1.7 \quad A_1 = \frac{2}{10} 2.502 = 0.500 \quad B_1 = \frac{2}{10} (-4.330) = -0.866$$

Thus, the least-squares fit is

$$y = 1.7 + 0.500 \cos (\omega_0 t) - 0.866 \sin (\omega_0 t)$$

The model can also be expressed in the format of Eq. (19.2) by calculating [Eq. (19.8)]

$$\theta = \arctan \left( -\frac{-0.866}{0.500} \right) = 1.0472$$

and [Eq. (19.9)]

$$C_1 = \sqrt{(0.5)^2 + (-0.866)^2} = 1.00$$

to give

$$y = 1.7 + \cos (\omega_0 t + 1.0472)$$

or alternatively, as a sine by using [Eq. (19.10)]

$$y = 1.7 + \sin (\omega_0 t + 2.618)$$

The foregoing analysis can be extended to the general model

$$f(t) = A_0 + A_1 \cos (\omega_0 t) + B_1 \sin (\omega_0 t) + A_2 \cos (2\omega_0 t) + B_2 \sin (2\omega_0 t) \\ + \cdots + A_m \cos (m\omega_0 t) + B_m \sin (m\omega_0 t)$$

where, for equally spaced data, the coefficients can be evaluated by

$$\left. \begin{aligned} A_0 &= \frac{\Sigma y}{N} \\ A_j &= \frac{2}{N} \Sigma y \cos(j\omega_0 t) \\ B_j &= \frac{2}{N} \Sigma y \sin(j\omega_0 t) \end{aligned} \right\} \quad j = 1, 2, \dots, m$$

Although these relationships can be used to fit data in the regression sense (that is,  $N > 2m + 1$ ), an alternative application is to employ them for interpolation or collocation—that is, to use them for the case where the number of unknowns,  $2m + 1$ , is equal to the number of data points,  $N$ . This is the approach used in the continuous Fourier series, as described next.

## 19.2 CONTINUOUS FOURIER SERIES

In the course of studying heat-flow problems, Fourier showed that an arbitrary periodic function can be represented by an infinite series of sinusoids of harmonically related frequencies. For a function with period  $T$ , a continuous Fourier series can be written<sup>1</sup>

$$f(t) = a_0 + a_1 \cos(\omega_0 t) + b_1 \sin(\omega_0 t) + a_2 \cos(2\omega_0 t) + b_2 \sin(2\omega_0 t) + \dots$$

or more concisely,

$$f(t) = a_0 + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] \quad (19.17)$$

where  $\omega_0 = 2\pi/T$  is called the *fundamental frequency* and its constant multiples  $2\omega_0, 3\omega_0$ , etc., are called *harmonics*. Thus, Eq. (19.17) expresses  $f(t)$  as a linear combination of the basis functions:  $1, \cos(\omega_0 t), \sin(\omega_0 t), \cos(2\omega_0 t), \sin(2\omega_0 t), \dots$

As described in Box 19.1, the coefficients of Eq. (19.17) can be computed via

$$a_k = \frac{2}{T} \int_0^T f(t) \cos(k\omega_0 t) dt \quad (19.18)$$

and

$$b_k = \frac{2}{T} \int_0^T f(t) \sin(k\omega_0 t) dt \quad (19.19)$$

for  $k = 1, 2, \dots$  and

$$a_0 = \frac{1}{T} \int_0^T f(t) dt \quad (19.20)$$

<sup>1</sup>The existence of the Fourier series is predicated on the Dirichlet conditions. These specify that the periodic function have a finite number of maxima and minima and that there be a finite number of jump discontinuities. In general, all physically derived periodic functions satisfy these conditions.

**Box 19.1** Determination of the Coefficients of the Continuous Fourier Series

As was done for the discrete data of Sec. 19.1.1, the following relationships can be established:

$$\int_0^T \sin(k\omega_0 t) dt = \int_0^T \cos(k\omega_0 t) dt = 0 \quad (\text{B19.1.1})$$

$$\int_0^T \cos(k\omega_0 t) \sin(g\omega_0 t) dt = 0 \quad (\text{B19.1.2})$$

$$\int_0^T \sin(k\omega_0 t) \sin(g\omega_0 t) dt = 0 \quad (\text{B19.1.3})$$

$$\int_0^T \cos(k\omega_0 t) \cos(g\omega_0 t) dt = 0 \quad (\text{B19.1.4})$$

$$\int_0^T \sin^2(k\omega_0 t) dt = \int_0^T \cos^2(k\omega_0 t) dt = \frac{T}{2} \quad (\text{B19.1.5})$$

To evaluate its coefficients, each side of Eq. (19.17) can be integrated to give

$$\int_0^T f(t) dt = \int_0^T a_0 dt + \int_0^T \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] dt$$

Because every term in the summation is of the form of Eq. (B19.1.1), the equation becomes

$$\int_0^T f(t) dt = a_0 T$$

which can be solved for

$$a_0 = \frac{\int_0^T f(t) dt}{T}$$

Thus,  $a_0$  is simply the average value of the function over the period.

To evaluate one of the cosine coefficients, for example,  $a_m$ , Eq. (19.17) can be multiplied by  $\cos(m\omega_0 t)$  and integrated to give

$$\begin{aligned} \int_0^T f(t) \cos(m\omega_0 t) dt &= \int_0^T a_0 \cos(m\omega_0 t) dt \\ &+ \int_0^T \sum_{k=1}^{\infty} a_k \cos(k\omega_0 t) \cos(m\omega_0 t) dt \\ &+ \int_0^T \sum_{k=1}^{\infty} b_k \sin(k\omega_0 t) \cos(m\omega_0 t) dt \end{aligned} \quad (\text{B19.1.6})$$

From Eqs. (B19.1.1), (B19.1.2), and (B19.1.4), we see that every term on the right-hand side is zero, with the exception of the case where  $k = m$ . This latter case can be evaluated by Eq. (B19.1.5) and, therefore, Eq. (B19.1.6) can be solved for  $a_m$ , or more generally [Eq. (19.18)],

$$a_k = \frac{2}{T} \int_0^T f(t) \cos(k\omega_0 t) dt$$

for  $k = 1, 2, \dots$

In a similar fashion Eq. (19.17) can be multiplied by  $\sin(m\omega_0 t)$ , integrated, and manipulated to yield Eq. (19.19).

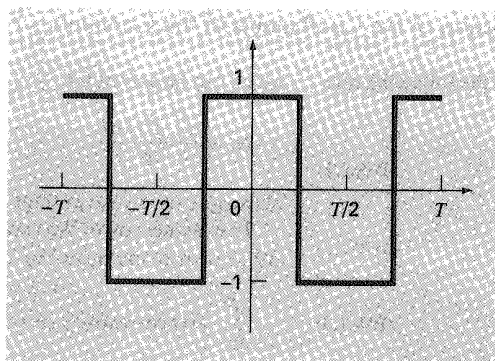
**EXAMPLE 19.2** Continuous Fourier Series Approximation

**Problem Statement.** Use the continuous Fourier series to approximate the square or rectangular wave function (Fig. 19.5)

$$f(t) = \begin{cases} -1 & -T/2 < t < -T/4 \\ 1 & -T/4 < t < T/4 \\ -1 & T/4 < t < T/2 \end{cases}$$

**Solution.** Because the average height of the wave is zero, a value of  $a_0 = 0$  can be obtained directly. The remaining coefficients can be evaluated as [Eq. (19.18)]

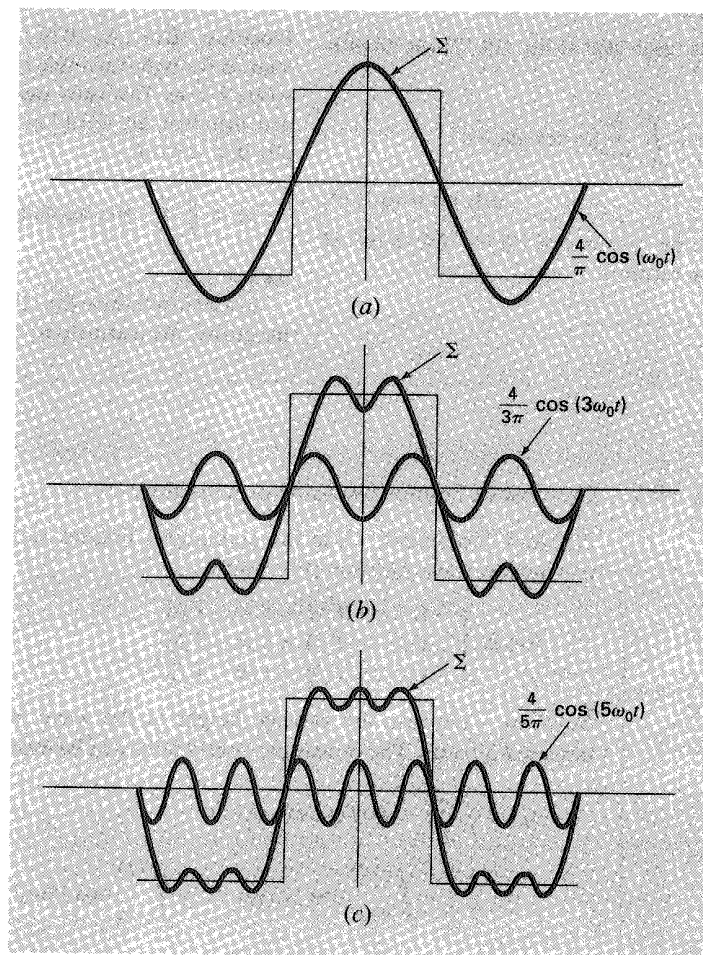
$$\begin{aligned} a_k &= \frac{2}{T} \int_{-T/2}^{T/2} f(t) \cos(k\omega_0 t) dt \\ &= \frac{2}{T} \left[ - \int_{-T/2}^{-T/4} \cos(k\omega_0 t) dt + \int_{-T/4}^{T/4} \cos(k\omega_0 t) dt - \int_{T/4}^{T/2} \cos(k\omega_0 t) dt \right] \end{aligned}$$

**FIGURE 19.5**

A square or rectangular waveform with a height of 2 and a period  $T = 2\pi/\omega_0$ .

**FIGURE 19.6**

The Fourier series approximation of the square wave from Fig. 19.5. The series of plots shows the summation up to and including the (a) first, (b) second, and (c) third terms. The individual terms that were added at each stage are also shown.





The integrals can be evaluated to give

$$a_k = \begin{cases} 4/(k\pi) & \text{for } k = 1, 5, 9, \dots \\ -4/(k\pi) & \text{for } k = 3, 7, 11, \dots \\ 0 & \text{for } k = \text{even integers} \end{cases}$$

Similarly, it can be determined that all the  $b$ 's = 0. Therefore, the Fourier series approximation is

$$f(t) = \frac{4}{\pi} \cos(\omega_0 t) - \frac{4}{3\pi} \cos(3\omega_0 t) + \frac{4}{5\pi} \cos(5\omega_0 t) - \frac{4}{7\pi} \cos(7\omega_0 t) + \dots$$

The results up to the first three terms are shown in Fig. 19.6.

It should be mentioned that the square wave in Fig. 19.5 is called an *even function* because  $f(t) = f(-t)$ . Another example of an even function is  $\cos(t)$ . It can be shown (Van Valkenburg, 1974) that the  $b$ 's in the Fourier series always equal zero for even functions. Note also that *odd functions* are those for which  $f(t) = -f(-t)$ . The function  $\sin(t)$  is an odd function. For this case, the  $a$ 's will equal zero.

Aside from the trigonometric format of Eq. (19.17), the Fourier series can be expressed in terms of exponential functions as (see Box 19.2 and App. A)

$$f(t) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{ik\omega_0 t} \quad (19.21)$$

where  $i = \sqrt{-1}$  and

$$\tilde{c}_k = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-ik\omega_0 t} dt \quad (19.22)$$

This alternative formulation will have utility throughout the remainder of the chapter.

## 19.3 FREQUENCY AND TIME DOMAINS

To this point, our discussion of Fourier approximation has been limited to the *time domain*. We have done this because most of us are fairly comfortable conceptualizing a function's behavior in this dimension. Although it is not as familiar, the *frequency domain* provides an alternative perspective for characterizing the behavior of oscillating functions.

Thus, just as amplitude can be plotted versus time, so also can it be plotted versus frequency. Both types of expression are depicted in Fig. 19.7a, where we have drawn a three-dimensional graph of a sinusoidal function,

$$f(t) = C_1 \cos\left(t + \frac{\pi}{2}\right)$$

In this plot, the magnitude or amplitude of the curve,  $f(t)$ , is the dependent variable and time  $t$  and frequency  $f = \omega_0/2\pi$  are the independent variables. Thus, the amplitude and the time axes form a time plane, and the amplitude and the frequency axes form a *frequency*

### Box 19.2 Complex Form of the Fourier Series

The trigonometric form of the continuous Fourier series is

$$f(t) = a_0 + \sum_{k=1}^{\infty} [a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t)] \quad (\text{B19.2.1})$$

From Euler's identity, the sine and cosine can be expressed in exponential form as

$$\sin x = \frac{e^{ix} - e^{-ix}}{2i} \quad (\text{B19.2.2})$$

$$\cos x = \frac{e^{ix} + e^{-ix}}{2} \quad (\text{B19.2.3})$$

which can be substituted into Eq. (B19.2.1) to give

$$f(t) = a_0 + \sum_{k=1}^{\infty} \left( e^{ik\omega_0 t} \frac{a_k - ib_k}{2} + e^{-ik\omega_0 t} \frac{a_k + ib_k}{2} \right) \quad (\text{B19.2.4})$$

because  $1/i = -i$ . We can define a number of constants

$$\begin{aligned} \tilde{c}_0 &= a_0 \\ \tilde{c}_k &= \frac{a_k - ib_k}{2} \\ \tilde{c}_{-k} &= \frac{a_{-k} - ib_{-k}}{2} = \frac{a_k + ib_k}{2} \end{aligned} \quad (\text{B19.2.5})$$

where, because of the odd and even properties of the sine and cosine,  $a_k = a_{-k}$  and  $b_k = -b_{-k}$ . Equation (B19.2.4) can therefore, be re-expressed as

$$f(t) = \tilde{c}_0 + \sum_{k=1}^{\infty} \tilde{c}_k e^{ik\omega_0 t} + \sum_{k=1}^{\infty} \tilde{c}_{-k} e^{-ik\omega_0 t}$$

or

$$f(t) = \sum_{k=0}^{\infty} \tilde{c}_k e^{ik\omega_0 t} + \sum_{k=1}^{\infty} \tilde{c}_{-k} e^{-ik\omega_0 t}$$

To simplify further, instead of summing the second series from 1 to  $\infty$ , perform the sum from  $-1$  to  $-\infty$ ,

$$f(t) = \sum_{k=0}^{\infty} \tilde{c}_k e^{ik\omega_0 t} + \sum_{k=-1}^{-\infty} \tilde{c}_{-k} e^{ik\omega_0 t}$$

or

$$f(t) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{ik\omega_0 t} \quad (\text{B19.2.6})$$

where the summation includes a term for  $k = 0$ .

To evaluate the  $\tilde{c}_k$ 's Eqs. (19.18) and (19.19) can be substituted into Eq. (B19.2.5) to yield

$$\tilde{c}_k = \frac{1}{T} \int_{-T/2}^{T/2} f(t) \cos(k\omega_0 t) dt - i \frac{1}{T} \int_{-T/2}^{T/2} f(t) \sin(k\omega_0 t) dt$$

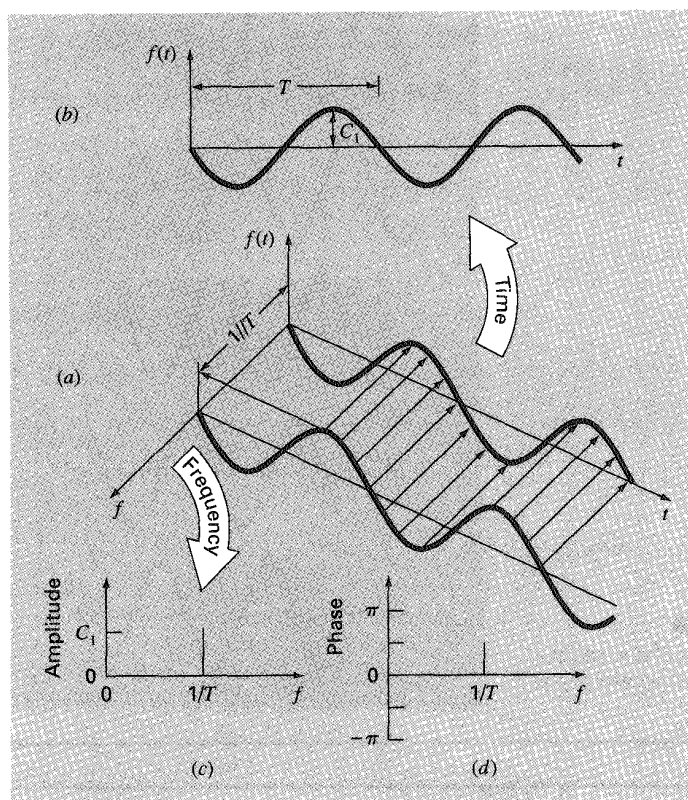
Employing Eqs. (B19.2.2) and (B19.2.3) and simplifying gives

$$\tilde{c}_k = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-ik\omega_0 t} dt \quad (\text{B19.2.7})$$

Therefore, Eqs. (B19.2.6) and (B19.2.7) are the complex versions of Eqs. (19.17) through (19.20). Note that App. A includes a summary of the interrelationships among all the formats of the Fourier series introduced in this chapter.

*plane*. The sinusoid can, therefore, be conceived of as existing a distance  $1/T$  out along the frequency axis and running parallel to the time axes. Consequently, when we speak about the behavior of the sinusoid in the time domain, we mean the projection of the curve onto the time plane (Fig. 19.7*b*). Similarly, the behavior in the frequency domain is merely its projection onto the frequency plane.

As in Fig. 19.7*c*, this projection is a measure of the sinusoid's maximum positive amplitude  $C_1$ . The full peak-to-peak swing is unnecessary because of the symmetry. Together with the location  $1/T$  along the frequency axis, Fig. 19.7*c* now defines the amplitude and frequency of the sinusoid. This is enough information to reproduce the shape and size of the curve in the time domain. However, one more parameter, namely, the phase angle, is required to position the curve relative to  $t = 0$ . Consequently, a phase diagram, as shown in Fig. 19.7*d*, must also be included. The phase angle is determined as the distance (in radians) from zero to the point at which the positive peak occurs. If the peak occurs after zero,

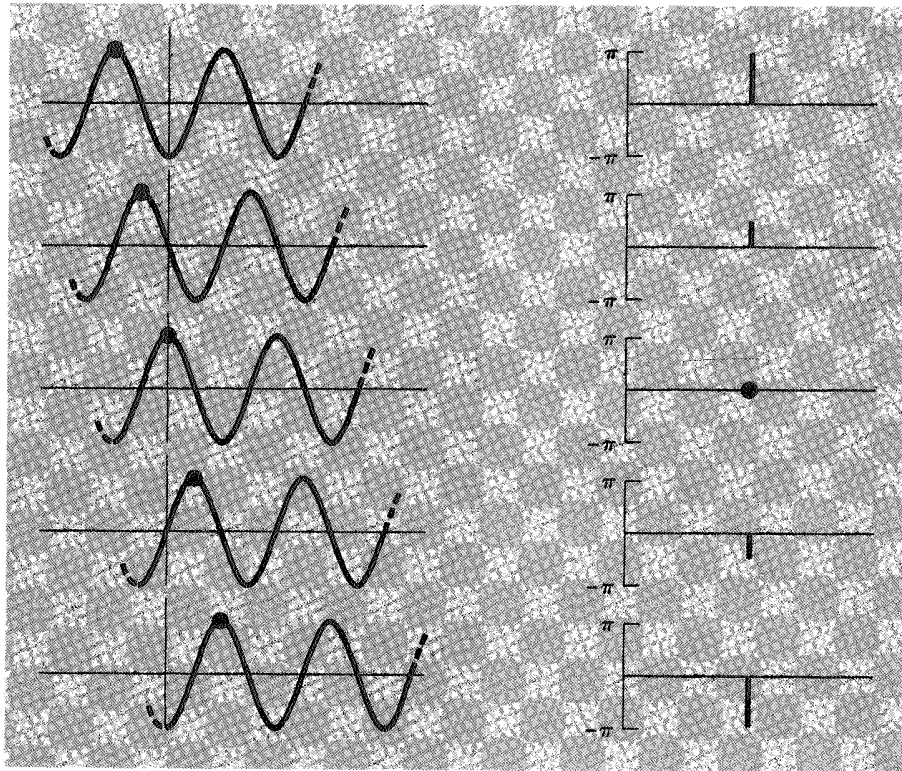
**FIGURE 19.7**

(a) A depiction of how a sinusoid can be portrayed in the time and the frequency domains. The time projection is reproduced in (b), whereas the amplitude-frequency projection is reproduced in (c). The phase-frequency projection is shown in (d).

it is said to be delayed (recall our discussion of lags and leads in Sec. 19.1), and by convention, the phase angle is given a negative sign. Conversely, a peak before zero is said to be advanced and the phase angle is positive. Thus, for Fig. 19.7, the peak leads zero and the phase angle is plotted as  $+\pi/2$ . Figure 19.8 depicts some other possibilities.

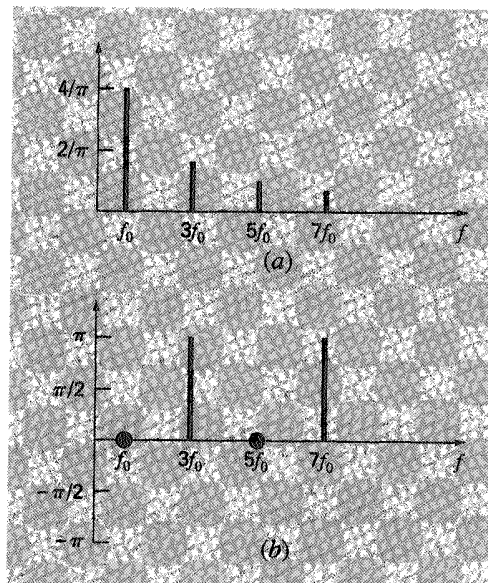
We can now see that Fig. 19.7c and d provides an alternative way to present or summarize the pertinent features of the sinusoid in Fig. 19.7a. They are referred to as *line spectra*. Admittedly, for a single sinusoid they are not very interesting. However, when applied to a more complicated situation, say, a Fourier series, their true power and value is revealed. For example, Fig. 19.9 shows the amplitude and phase line spectra for the square-wave function from Example 19.2.

Such spectra provide information that would not be apparent from the time domain. This can be seen by contrasting Figs. 19.6 and 19.9. Figure 19.6 presents two alternative time-domain perspectives. The first, the original square wave, tells us nothing about



**FIGURE 19.8**  
Various phases of a sinusoid showing the associated phase line spectra.

**FIGURE 19.9**  
(a) Amplitude and (b) phase line spectra for the square wave from Fig. 19.5.



the sinusoids that comprise it. The alternative is to display these sinusoids—that is,  $(4/\pi) \cos(\omega_0 t)$ ,  $-(4/3\pi) \cos(3\omega_0 t)$ ,  $(4/5\pi) \cos(5\omega_0 t)$ , etc. This alternative does not provide an adequate visualization of the structure of these harmonics. In contrast, Fig. 19.9a and b provides a graphic display of this structure. As such, the line spectra represent “fingerprints” that can help us to characterize and understand a complicated waveform. They are particularly valuable for nonidealized cases where they sometimes allow us to discern structure in otherwise obscure signals. In the next section, we will describe the Fourier transform that will allow us to extend such analyses to nonperiodic waveforms.

## 19.4 FOURIER INTEGRAL AND TRANSFORM

Although the Fourier series is a useful tool for investigating the spectrum of a periodic function, there are many waveforms that do not repeat themselves regularly. For example, a lightning bolt occurs only once (or at least it will be a long time until it occurs again), but it will cause interference with receivers operating on a broad range of frequencies—for example, TVs, radios, shortwave receivers, etc. Such evidence suggests that a nonrecurring signal such as that produced by lightning exhibits a continuous frequency spectrum. Because such phenomena are of great interest to engineers, an alternative to the Fourier series would be valuable for analyzing these aperiodic waveforms.

The *Fourier integral* is the primary tool available for this purpose. It can be derived from the exponential form of the Fourier series

$$f(t) = \sum_{k=-\infty}^{\infty} \tilde{c}_k e^{ik\omega_0 t} \quad (19.23)$$

where

$$\tilde{c}_k = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-ik\omega_0 t} dt \quad (19.24)$$

where  $\omega_0 = 2\pi/T$  and  $k = 0, 1, 2, \dots$

The transition from a periodic to a nonperiodic function can be effected by allowing the period to approach infinity. In other words, as  $T$  becomes infinite, the function never repeats itself and thus becomes aperiodic. If this is allowed to occur, it can be demonstrated (for example, Van Valkenburg, 1974; Hayt and Kemmerly, 1986) that the Fourier series reduces to

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(i\omega_0) e^{i\omega_0 t} d\omega_0 \quad (19.25)$$

and the coefficients become a continuous function of the frequency variable  $\omega$ , as in

$$F(i\omega_0) = \int_{-\infty}^{\infty} f(t) e^{-i\omega_0 t} dt \quad (19.26)$$

The function  $F(i\omega_0)$ , as defined by Eq. (19.26), is called the *Fourier integral* of  $f(t)$ . In addition, Eqs. (19.25) and (19.26) are collectively referred to as the *Fourier transform pair*. Thus, along with being called the Fourier integral,  $F(i\omega_0)$  is also called the *Fourier transform* of  $f(t)$ . In the same spirit,  $f(t)$ , as defined by Eq. (19.25), is referred to as the *inverse*

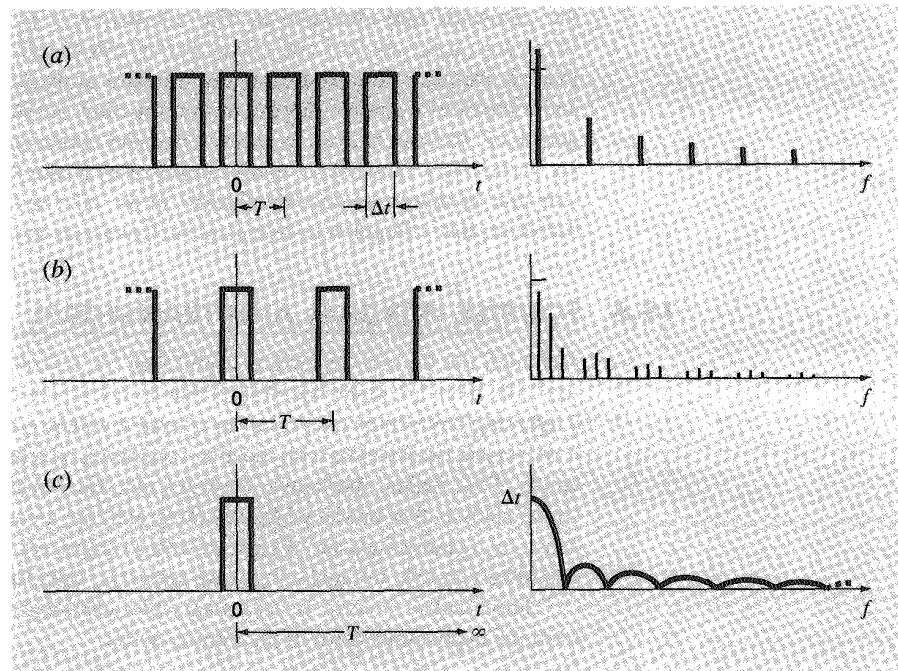
**FIGURE 19.10**

Illustration of how the discrete frequency spectrum of a Fourier series for a pulse train (a) approaches a continuous frequency spectrum of a Fourier integral (c) as the period is allowed to approach infinity.

*Fourier transform of  $F(i\omega_0)$ .* Thus, the pair allows us to transform back and forth between the time and the frequency domains for an aperiodic signal.

The distinction between the Fourier series and transform should now be quite clear. The major difference is that each applies to a different class of functions—the series to periodic and the transform to nonperiodic waveforms. Beyond this major distinction, the two approaches differ in how they move between the time and the frequency domains. The Fourier series converts a continuous, periodic time-domain function to frequency-domain magnitudes at discrete frequencies. In contrast, the Fourier transform converts a continuous time-domain function to a continuous frequency-domain function. Thus, the discrete frequency spectrum generated by the Fourier series is analogous to a continuous frequency spectrum generated by the Fourier transform.

The shift from a discrete to a continuous spectrum can be illustrated graphically. In Fig. 19.10a, we can see a pulse train of rectangular waves with pulse widths equal to one-half the period along with its associated discrete spectrum. This is the same function as was investigated previously in Example 19.2, with the exception that it is shifted vertically.

In Fig. 19.10*b*, a doubling of the pulse train's period has two effects on the spectrum. First, two additional frequency lines are added on either side of the original components. Second, the amplitudes of the components are reduced.

As the period is allowed to approach infinity, these effects continue as more and more spectral lines are packed together until the spacing between lines goes to zero. At the limit, the series converges on the continuous Fourier integral, depicted in Fig. 19.10*c*.

Now that we have introduced a way to analyze an aperiodic signal, we will take the final step in our development. In the next section, we will acknowledge the fact that a signal is rarely characterized as a continuous function of the sort needed to implement Eq. (19.26). Rather, the data is invariably in a discrete form. Thus, we will now show how to compute a Fourier transform for such discrete measurements.

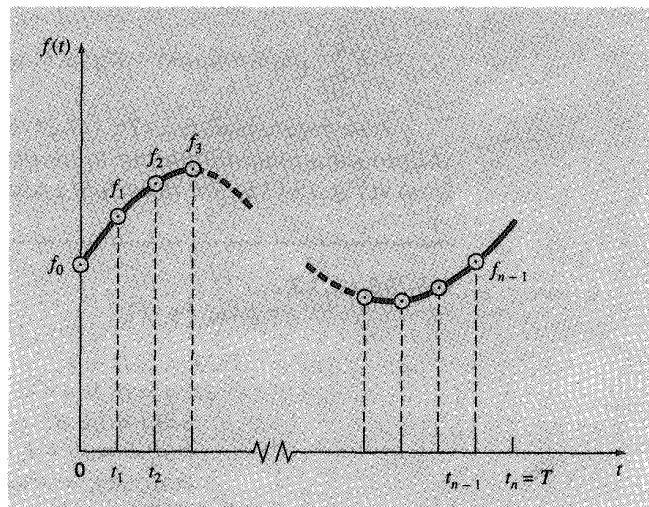
## 19.5 DISCRETE FOURIER TRANSFORM (DFT)

In engineering, functions are often represented by finite sets of discrete values. Additionally, data is often collected in or converted to such a discrete format. As depicted in Fig. 19.11, an interval from 0 to  $t$  can be divided into  $N$  equispaced subintervals with widths of  $\Delta t = T/N$ . The subscript  $n$  is employed to designate the discrete times at which samples are taken. Thus,  $f_n$  designates a value of the continuous function  $f(t)$  taken at  $t_n$ .

Note that the data points are specified at  $n = 0, 1, 2, \dots, N - 1$ . A value is not included at  $n = N$ . (See Ramirez, 1985, for the rationale for excluding  $f_N$ .)

**FIGURE 19.11**

The sampling points of the discrete Fourier series.



For the system in Fig. 19.11, a discrete Fourier transform can be written as

$$F_k = \sum_{n=0}^{N-1} f_n e^{-i\omega_0 n} \quad \text{for } k = 0 \text{ to } N - 1 \quad (19.27)$$

and the inverse Fourier transform as

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{i\omega_0 n} \quad \text{for } n = 0 \text{ to } N - 1 \quad (19.28)$$

where  $\omega_0 = 2\pi/N$ .

Equations (19.27) and (19.28) represent the discrete analogs of Eqs. (19.26) and (19.25), respectively. As such, they can be employed to compute both a direct and an inverse Fourier transform for discrete data. Although such calculations can be performed by hand, they are extremely arduous. As expressed by Eq. (19.27), the DFT requires  $N^2$  complex operations. Thus, we will now develop a computer algorithm to implement the DFT.

**Computer Algorithm for the DFT.** Note that the factor  $1/N$  in Eq. (19.28) is merely a scale factor that can be included in either Eq. (19.27) or (19.28), but not both. For our computer algorithm, we will shift it to Eq. (19.27) so that the first coefficient  $F_0$  (which is the analog of the continuous coefficient  $a_0$ ) is equal to the arithmetic mean of the samples. Also, to develop an algorithm that can be implemented in languages that do not accommodate complex variables, we can use Euler's identity,

$$e^{\pm ia} = \cos a \pm i \sin a$$

to re-express Eqs. (19.27) and (19.28) as

$$F_k = \frac{1}{N} \sum_{n=0}^{N-1} [f_n \cos(k\omega_0 n) - i f_n \sin(k\omega_0 n)] \quad (19.29)$$

and

$$f_n = \sum_{k=0}^{N-1} [F_k \cos(k\omega_0 n) + i F_k \sin(k\omega_0 n)] \quad (19.30)$$

Pseudocode to implement Eq. (19.29) is listed in Fig. 19.12. This algorithm can be developed into a computer program to compute the DFT. The output from such a program is listed in Fig. 19.13 for the analysis of a cosine function.

### FIGURE 19.12

Pseudocode for computing the DFT.

```

DO k = 0, N - 1
  DO n = 0, N - 1
    angle = k*omega_0*n
    real_k = real_k + f_n*cos(angle)/N
    imaginary_k = imaginary_k - f_n*sin(angle)/N
  END DO
END DO

```



INDEX	f(t)	REAL	IMAGINARY
0	1.000	0.0000	0.0000
1	0.707	0.0000	0.0000
2	-0.000	0.0000	-0.0000
3	-0.707	0.0000	-0.0000
4	-1.000	0.5000	0.0000
5	-0.707	0.0000	-0.0000
6	-0.000	0.0000	0.0000
7	0.707	0.0000	0.0000
8	1.000	-0.0000	0.0000
9	0.707	0.0000	0.0000
10	-0.000	-0.0000	-0.0000
11	-0.707	0.0000	0.0000
12	-1.000	-0.0000	0.0000
13	-0.707	0.0000	-0.0000
14	0.000	0.0000	0.0000
15	0.707	0.0000	-0.0000
16	1.000	-0.0000	0.0000
17	0.707	0.0000	0.0000
18	-0.000	-0.0000	0.0000
19	-0.707	-0.0000	-0.0000
20	-1.000	-0.0000	0.0000
21	-0.707	0.0000	0.0000
22	0.000	0.0000	0.0000
23	0.707	-0.0000	-0.0000
24	1.000	-0.0000	0.0000
25	0.707	-0.0000	0.0000
26	-0.000	-0.0000	0.0000
27	-0.707	-0.0000	-0.0000
28	-1.000	0.5000	-0.0000
29	-0.707	0.0000	-0.0000
30	0.000	0.0000	0.0000
31	0.707	0.0000	0.0000

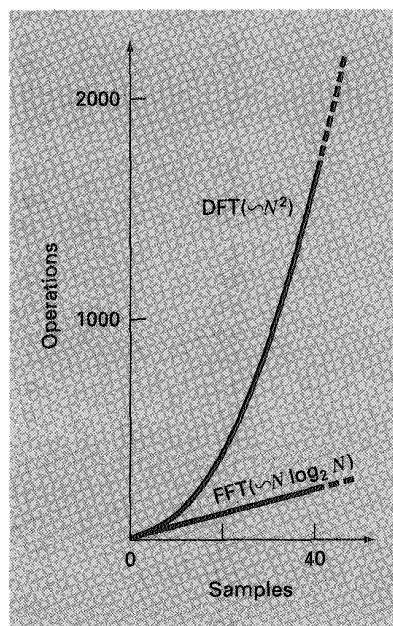
**FIGURE 19.13**

Output of a program based on the algorithm from Fig. 19.12 for the DFT of data generated by a cosine function  $f(t) = \cos [2\pi(12.5)t]$  at 32 points with  $\Delta t = 0.01$  s.

## 19.6 FAST FOURIER TRANSFORM (FFT)

Although the algorithm described in the previous section adequately calculates the DFT, it is computationally burdensome because  $N^2$  operations are required. Consequently, for data samples of even moderate size, the direct determination of the DFT can be extremely time-consuming.

The *fast Fourier transform*, or *FFT*, is an algorithm that has been developed to compute the DFT in an extremely economical fashion. Its speed stems from the fact that it utilizes the

**FIGURE 19.14**

Plot of number of operations vs. sample size for the standard DFT and the FFT.

results of previous computations to reduce the number of operations. In particular, it exploits the periodicity and symmetry of trigonometric functions to compute the transform with approximately  $N \log_2 N$  operations (Fig. 19.14). Thus, for  $N = 50$  samples, the FFT is about 10 times faster than the standard DFT. For  $N = 1000$ , it is about 100 times faster.

The first FFT algorithm was developed by Gauss in the early nineteenth century (Heideman et al., 1984). Other major contributions were made by Runge, Danielson, Lanczos, and others in the early twentieth century. However, because discrete transforms often took days to weeks to calculate by hand, they did not attract broad interest prior to the development of the modern digital computer.

In 1965, J. W. Cooley and J. W. Tukey published a key paper in which they outlined an algorithm for calculating the FFT. This scheme, which is similar to those of Gauss and other earlier investigators, is called the *Cooley-Tukey algorithm*. Today, there are a host of other approaches that are offshoots of this method.

The basic idea behind each of these algorithms is that a DFT of length  $N$  is decomposed, or “decimated,” into successively smaller DFTs. There are a variety of different ways to implement this principle. For example, the Cooley-Tukey algorithm is a member of what are called *decimation-in-time* techniques. In the present section, we will describe an alternative approach called the *Sande-Tukey algorithm*. This method is a member of another class of algorithms called *decimation-in-frequency* techniques. The distinction between the two classes will be discussed after we have elaborated on the method.

### 19.6.1 Sande-Tukey Algorithm

In the present case,  $N$  will be assumed to be an integral power of 2,

$$N = 2^M \quad (19.31)$$

where  $M$  is an integer. This constraint is introduced to simplify the resulting algorithm. Now, recall that the DFT can be generally represented as

$$F_k = \sum_{n=0}^{N-1} f_n e^{-i(2\pi/N)nk} \quad \text{for } k = 0 \text{ to } N - 1 \quad (19.32)$$

where  $2\pi/N = \omega_0$ . Equation (19.32) can also be expressed as

$$F_k = \sum_{n=0}^{N-1} f_n W^{nk}$$

where  $W$  is a complex-valued weighting function defined as

$$W = e^{-i(2\pi/N)} \quad (19.33)$$

Suppose now that we divide the sample in half and express Eq. (19.32) in terms of the first and last  $N/2$  points:

$$F_k = \sum_{n=0}^{(N/2)-1} f_n e^{-i(2\pi/N)kn} + \sum_{n=N/2}^{N-1} f_n e^{-i(2\pi/N)kn}$$

where  $k = 0, 1, 2, \dots, N - 1$ . A new variable,  $m = n - N/2$ , can be created so that the range of the second summation is consistent with the first,

$$F_k = \sum_{n=0}^{(N/2)-1} f_n e^{-i(2\pi/N)kn} + \sum_{m=0}^{(N/2)-1} f_{m+N/2} e^{-i(2\pi/N)k(m+N/2)}$$

or

$$F_k = \sum_{n=0}^{(N/2)-1} (f_n + e^{-i\pi k} f_{n+N/2}) e^{-i2\pi kn/N} \quad (19.34)$$

Next, recognize that the factor  $e^{-i\pi k} = (-1)^k$ . Thus, for even points it is equal to 1 and for odd points it is equal to  $-1$ . Therefore, the next step in the method is to separate Eq. (19.34) according to even values and odd values of  $k$ . For the even values,

$$F_{2k} = \sum_{n=0}^{(N/2)-1} (f_n + f_{n+N/2}) e^{-i2\pi(2k)n/N} = \sum_{n=0}^{(N/2)-1} (f_n + f_{n+N/2}) e^{-i2\pi kn/(N/2)}$$

and for the odd values,

$$\begin{aligned} F_{2k+1} &= \sum_{n=0}^{(N/2)-1} (f_n - f_{n+N/2}) e^{-i2\pi(2k+1)n/N} \\ &= \sum_{n=0}^{(N/2)-1} (f_n - f_{n+N/2}) e^{-i2\pi n/N} e^{-i2\pi kn/(N/2)} \end{aligned}$$

for  $k = 0, 1, 2, \dots, (N/2) - 1$ .

These equations can also be expressed in terms of Eq. (19.33). For the even values,

$$F_{2k} = \sum_{n=0}^{(N/2)-1} (f_n + f_{n+N/2}) W^{2kn}$$

and for the odd values,

$$F_{2k+1} = \sum_{n=0}^{(N/2)-1} (f_n - f_{n+N/2}) W^n W^{2kn}$$

Now, a key insight can be made. These even and odd expressions can be interpreted as being equal to the transforms of the  $(N/2)$ -length sequences

$$g_n = f_n + f_{n+N/2} \quad (19.35)$$

and

$$h_n = (f_n - f_{n+N/2}) W^n \quad \text{for } n = 0, 1, 2, \dots, (N/2) - 1 \quad (19.36)$$

Thus, it directly follows that

$$\left. \begin{array}{l} F_{2k} = G_k \\ F_{2k+1} = H_k \end{array} \right\} \quad \text{for } k = 0, 1, 2, \dots, (N/2) - 1$$

In other words, one  $N$ -point computation has been replaced by two  $(N/2)$ -point computations. Because each of the latter requires approximately  $(N/2)^2$  complex multiplications and additions, the approach produces a factor-of-2 savings—that is,  $N^2$  versus  $2(N/2)^2 = N^2/2$ .

The scheme is depicted in Fig. 19.15 for  $N = 8$ . The DFT is computed by first forming the sequence  $g^n$  and  $h^n$  and then computing the  $N/2$  DFTs to obtain the even- and odd-numbered transforms. The weights  $W^n$  are sometimes called *twiddle factors*.

Now it is clear that this “divide-and-conquer” approach can be repeated at the second stage. Thus, we can compute the  $(N/4)$ -point DFTs of the four  $N/4$  sequences composed of the first and last  $N/4$  points of Eqs. (19.35) and (19.36).

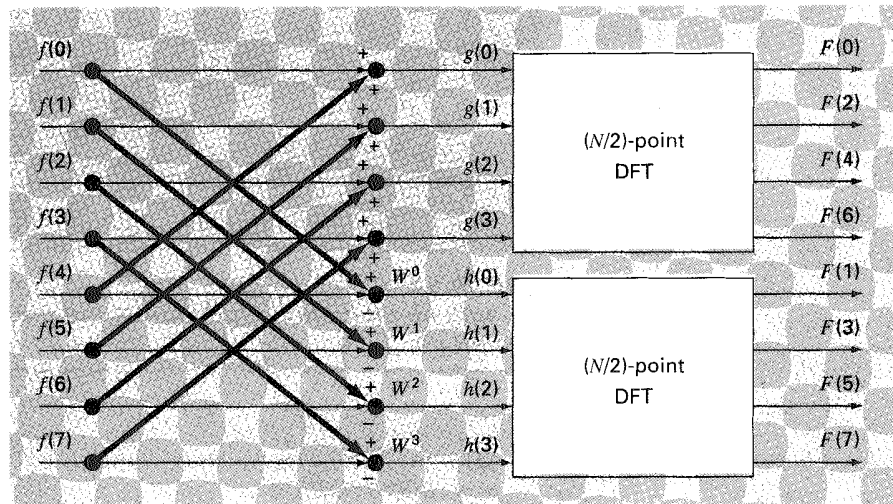
The strategy is continued to its inevitable conclusion when  $N/2$  two-point DFTs are computed (Fig. 19.16). The total number of calculations for the entire computation is on the order of  $N \log_2 N$ . The contrast between this level of effort and that of the standard DFT (Fig. 19.14) illustrates why the FFT is so important.

**Computer Algorithm.** It is a relatively straightforward proposition to express Fig. 19.16 as an algorithm. As was the case for the DFT algorithm of Fig. 19.12, we will use Euler’s identity,

$$e^{\pm ia} = \cos a \pm i \sin a$$

to allow the algorithm to be implemented in languages that do not explicitly accommodate complex variables.

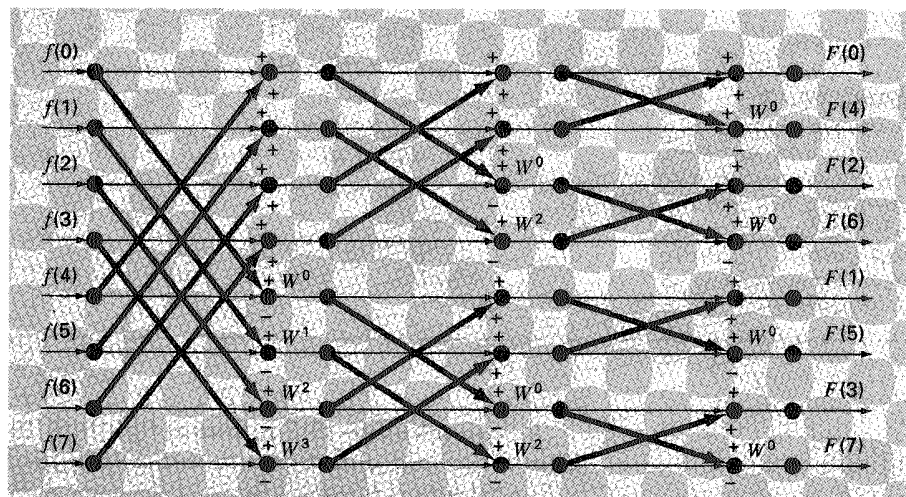
Close inspection of Fig. 19.16 indicates that its fundamental computational molecule is the so-called *butterfly network* depicted in Fig. 19.17a. Pseudocode to implement one of these molecules is shown in Fig. 19.17b.

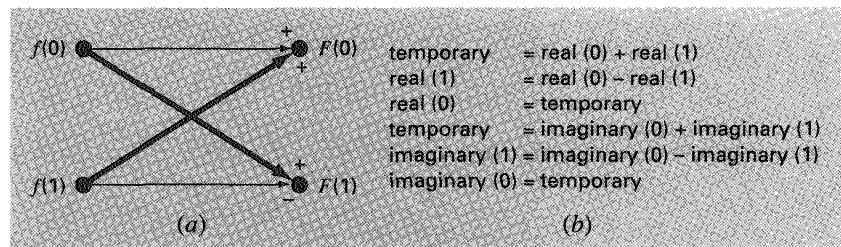
**FIGURE 19.15**

Flow graph of the first stage in a decimation-in-frequency decomposition of an  $N$ -point DFT into two  $(N/2)$ -point DFTs for  $N = 8$ .

**FIGURE 19.16**

Flow graph of the complete decimation-in-frequency decomposition of an eight-point DFT.



**FIGURE 19.17**

(a) A butterfly network that represents the fundamental computation of Fig. 19.16. (b) Pseudocode to implement (a).

```

(a)
m = LOG(N) / LOG(2)
N2 = N
DO k = 1, m
  N1 = N2
  N2 = N2 / 2
  angle = 0
  arg = 2π / N1
  DO j = 0, N2 - 1
    c = cos(angle)
    s = -sin(angle)
    DO i = j, N - 1, N1
      kk = i + N2
      xt = x(i) - x(kk)
      x(i) = x(i) + x(kk)
      yt = y(i) - y(kk)
      y(i) = y(i) + y(kk)
      x(kk) = xt * c - yt * s
      y(kk) = yt * c + xt * s
    END DO
    angle = (j + 1) * arg
  END DO
END DO

```

```

(b)
j = 0
DO i = 0, N - 2
  IF (i < j) THEN
    xt = xj
    xj = xi
    xi = xt
    yt = yj
    yj = yi
    yi = yt
  END IF
  k = N / 2
  DO
    IF (k ≥ j + 1) EXIT
    j = j - k
    k = k / 2
  END DO
  j = j + k
END DO
DO i = 0, N - 1
  x(i) = x(i) / N
  y(i) = y(i) / N
END DO

```

**FIGURE 19.18**

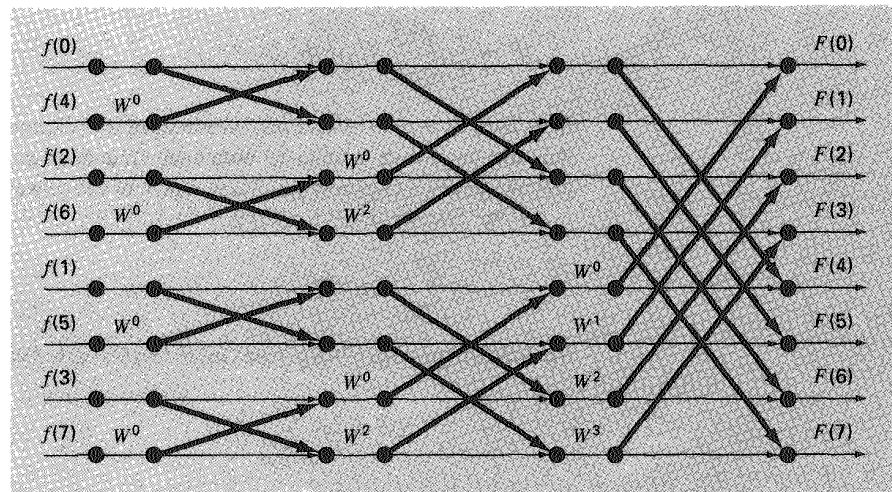
Pseudocode to implement a decimation-in-frequency FFT. Note that the pseudocode is composed of two parts: (a) the FFT itself and (b) a bit-reversal routine to unscramble the order of the resulting Fourier coefficients.

Pseudocode for the FFT is listed in Fig. 19.18. The first part consists essentially of three nested loops to implement the computation embodied in Fig. 19.16. Note that the real-valued data is originally stored in the array  $x$ . Also note that the outer loop steps through the  $M$  stages [recall Eq. (19.31)] of the flow graph.

After this first part is executed, the DFT will have been computed but in a scrambled order (see the right-hand side of Fig. 19.16). These Fourier coefficients can be unscrambled

Scrambled Order (Decimal)	Scrambled Order (Binary)	Bit-Reversed Order (Binary)	Final Result (Decimal)
$F(0)$	$F(000)$	$F(000)$	$F(0)$
$F(4)$	$F(100)$	$F(001)$	$F(1)$
$F(2)$	$F(010)$	$F(010)$	$F(2)$
$F(6)$	$F(110)$	$F(011)$	$F(3)$
$F(1)$	$F(001)$	$F(100)$	$F(4)$
$F(5)$	$F(101)$	$F(101)$	$F(5)$
$F(3)$	$F(011)$	$F(110)$	$F(6)$
$F(7)$	$F(111)$	$F(111)$	$F(7)$

**FIGURE 19.19**  
Depiction of the bit-reversal process.



**FIGURE 19.20**  
Flow graph of a decimation-in-time FFT of an eight-point DFT.

by a procedure called *bit reversal*. If the subscripts 0 through 7 are expressed in binary, the correct ordering can be obtained by reversing these bits (Fig. 19.19). The second part of the algorithm implements this procedure.

### 19.6.2 Cooley-Tukey Algorithm

Figure 19.20 shows a flow network to implement the Cooley-Tukey algorithm. For this case, the sample is initially divided into odd- and even-numbered points, and the final results are in correct order.

This approach is called a *decimation in time*. It is the reverse of the Sande-Tukey algorithm described in the previous section. Although the two classes of method differ in organization, they both exhibit the  $N \log_2 N$  operations that are the strength of the FFT approach.

## 19.7 THE POWER SPECTRUM

The FFT has many engineering applications, ranging from vibration analysis of structures and mechanisms to signal processing. As described previously, amplitude and phase spectra provide a means to discern the underlying structure of seemingly random signals. Similarly, a useful analysis called a power spectrum can be developed from the Fourier transform.

As the name implies, the power spectrum derives from the analysis of the power output of electrical systems. In mathematical terms, the power of a periodic signal in the time domain can be defined as

$$P = \frac{1}{T} \int_{-T/2}^{T/2} f^2(t) dt \quad (19.37)$$

Now another way to look at this information is to express it in the frequency domain by calculating the power associated with each frequency component. This information can be then displayed as a *power spectrum*, a plot of the power versus frequency.

If the Fourier series for  $f(t)$  is

$$f(t) = \sum_{k=-\infty}^{\infty} F_n e^{ik\omega_0 t} \quad (19.38)$$

the following relation holds (see Gabel and Roberts 1987) for details):

$$\frac{1}{T} \int_{-T/2}^{T/2} f^2(t) dt = \sum_{k=-\infty}^{\infty} |F_k|^2 \quad (19.39)$$

Thus, the power in  $f(t)$  can be determined by adding together the squares of the Fourier coefficients; that is, the powers associated with the individual frequency components.

Now, remember that in this representation, the single real harmonic consists of both frequency components at  $\pm k\omega_0$ . We also know that the positive and negative coefficients are equal. Therefore, the power in  $f_k(t)$ , the  $k$ th real harmonic of  $f(t)$ , is

$$p_k = 2 |F_k|^2 \quad (19.40)$$

The power spectrum is the plot of  $p_k$  as a function of frequency  $k\omega_0$ . We will devote Sec. 20.3 to an engineering application involving the FFT and the power spectrum generated with software packages.

**Additional Information.** The foregoing has been a brief introduction to Fourier approximation and the FFT. Additional information on the former can be found in Van Valkenburg



(1974), Chirlian (1969), and Hayt and Kemmerly (1986). References on the FFT include Davis and Rabinowitz (1975); Cooley, Lewis, and Welch (1977); and Brigham (1974). Nice introductions to both can be found in Ramirez (1985), Oppenheim and Schafer (1975), and Gabel and Roberts (1987).

## 19.8 CURVE FITTING WITH LIBRARIES AND PACKAGES

Software libraries and packages have great capabilities for curve fitting. In this section, we will give you a taste of some of the more useful ones.

### 19.8.1 Excel

In the present context, the most useful application of Excel is for regression analysis and, to a lesser extent, polynomial interpolation. Aside from a few built-in functions (see Table 19.1), there are two primary ways in which this capability can be implemented: the Trendline command and the Data Analysis Toolpack.

The Trendline Command (Insert Menu). This command allows a number of different trend models to be added to a chart. These models include linear, polynomial, logarithmic, exponential, power, and moving average fits. The following example illustrates how the Trendline command is invoked.

**TABLE 19.1** Excel built-in functions related to regression fits of data.

Function	Description
FORECAST	Returns a value along a linear trend
GROWTH	Returns values along an exponential trend
INTERCEPT	Returns the intercept of the linear regression line
LINEST	Returns the parameters of a linear trend
LOGEST	Returns the parameters of an exponential trend
SLOPE	Returns the slope of the linear regression line
TREND	Returns values along a linear trend

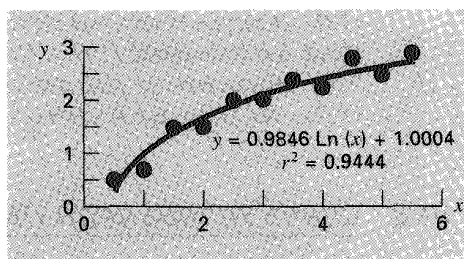
### EXAMPLE 19.3 Using Excel's Trendline Command

**Problem Statement.** You may have noticed that several of the fits available on Trendline were discussed previously in Chap. 17 (e.g., linear, polynomial, exponential, and power). An additional capability is the logarithmic model

$$y = a_0 + a_1 \log x$$

Fit the following data with this model using Excel's Trendline command:

x	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5	5.5
y	0.53	0.69	1.5	1.5	2	2.06	2.28	2.23	2.73	2.42	2.79

**FIGURE 19.21**

Fit of a logarithmic model to the data from Example 19.3.

**Solution.** To invoke the Trendline command, a chart relating a series of dependent and independent variables must be created. For the present case, we use the Excel Chart Wizard to create an XY-plot of the data.

Next, we can select the chart (by double clicking on it) and the series (by positioning the mouse arrow on one of the values and single clicking). The Insert and Trendline commands are then invoked with the mouse or by the key sequence

/ Insert Trendline

At this point, a dialogue box opens with two tabs: Options tab and the Type tab. The Options tab provides ways to customize the fit. The most important in the present context is to display both the equation and the value for the coefficient of determination ( $r^2$ ) on the chart. The primary choice on the Type tab is to specify the type of trendline. For the present case, select **Logarithmic**. The resulting fit along with  $r^2$  is displayed in Fig. 19.21.

The Trendline command provides a handy way to fit a number of commonly used models to data. In addition, its inclusion of the **Polynomial** option means that it can also be used for polynomial interpolation. However, the fact that its statistical content is limited to  $r^2$  means that it does not allow statistical inferences to be drawn regarding the model fit. The Data Analysis Toolpack described next provides a nice alternative where such inferences are necessary.

**The Data Analysis Toolpack.** This Excel Add-in Package contains a comprehensive capability for curve fitting with general linear least squares. As previously described in Sec. 17.4, such models are of the general form

$$y = a_0z_0 + a_1z_1 + a_2z_2 + \cdots + a_mz_m + e \quad (17.23)$$

where  $z_0, z_1, \dots, z_m$  are  $m + 1$  different functions. The next example illustrates how such models can be fit with Excel.

## EXAMPLE 19.4 Using Excel's Data Analysis Toolpack

**Problem Statement.** The following data was collected for the slope, hydraulic radius, and velocity of water flowing in a canal:

$S, \text{ m/m}$	0.0002	0.0002	0.0005	0.0005	0.001	0.001
$R, \text{ m}$	0.2	0.5	0.2	0.5	0.2	0.5
$U, \text{ m/s}$	0.25	0.5	0.4	0.75	0.5	1

There are theoretical reasons (recall Sec. 8.2) for believing that this data can be fit to a power model of the form

$$U = \alpha S^\sigma R^\rho$$

where  $\alpha$ ,  $\sigma$ , and  $\rho$  are empirically derived coefficients. There are theoretical reasons (again, see Sec. 8.2) for believing that  $\sigma$  and  $\rho$  should have values of approximately 0.5 and 0.667, respectively. Fit this data with Excel and evaluate whether your regression estimates contradict the expected values for the model coefficients.

**Solution.** The logarithm of the power model is first used to convert it to the linear format of Eq. (17.23),

$$U = \log \alpha + \sigma \log S + \rho \log R$$

An Excel spreadsheet can be developed with both the original data along with their common logarithms, as in the following:

	A	B	C	D	E	F
1	S	Rh	U	log(S)	log(Rh)	log(U)
2	0.0002	0.2	0.25	-3.69897	-0.69897	-0.60206
3	0.0002	0.5	0.5	-3.69897	-0.30103	-0.30103
4	0.0005	0.2	0.4	-3.30103	-0.69897	-0.39794
5	0.0005	0.5	0.75	-3.30103	-0.30103	-0.12494
6	0.001	0.2	0.5	-3	-0.69897	-0.30103
7	0.001	0.5	1	-3	-0.30103	0

= log(A2)

As shown, an efficient way to generate the logarithms is to type the formula to compute the first log(S). This formula can then be copied to the right and down to generate the other logarithms.

Because of its status as an "Add-In" on the version of Excel available at the time of this book's printing, the Data Analysis Toolpack must sometimes be loaded into Excel. To do this, merely use the mouse or the key sequence

**/Tools Add-Ins**

Then select **Analysis Toolpack** and OK. If the add-in is successful, the selection Data Analysis will be added to the Tools menu.

After selecting **Data Analysis** from the Tools menu, a Data Analysis menu will appear on the screen containing a large number of statistically oriented routines. Select **Regression** and a dialogue box will appear, prompting you for information on the regression. After

making sure that the default selection **New Worksheet Ply** is selected, fill in F2:F7 for the y range and D2:E7 for the x range, and select OK. The following worksheet will be created:

	A	B	C	D	E	F	G
1	SUMMARY OUTPUT						
2							
3	Regression Statistics						
4	Multiple R	0.998353					
5	R Square	0.996708					
6	Adjusted R Square	0.994513					
7	Standard Error	0.015559					
8	Observations	6					
9							
10	ANOVA						
11		df	SS	MS	F	Significance F	
12	Regression	2	0.219867	0.109933	454.1106	0.0001889	
13	Residual	3	0.000726	0.000242			
14	Total	5	0.220593				
15							
16		Coeffs	Std. Error	t Stat	P-value	Lower 95%	Upper 95%
17	Intercept	1.522452	0.075932	20.0501	0.0002712	1.2808009	1.7641028
18	X Variable 1	0.433137	0.022189	19.5203	0.0002937	0.3625211	0.5037521
19	X Variable 2	0.732993	0.031924	22.96038	0.000181	0.6313953	0.8345899

Thus, the resulting fit is

$$\log U = 1.522 + 0.433 \log S + 0.733 \log R$$

or by taking antilog,

$$U = 33.3S^{0.433}R^{0.733}$$

Notice that 95% confidence intervals are generated for the coefficients. Thus, there is a 95% probability that the true slope exponent falls between 0.363 and 0.504, and the true hydraulic radius coefficient falls between 0.631 and 0.835. Thus, the fit does not contradict the theoretical exponents.

Finally, it should be noted that the Excel Solver tool can be used to perform *nonlinear regression* by directly minimizing the sum of the squares of the residuals between a nonlinear model prediction and data. We devote Sec. 20.1 to an example of how this can be done.

### 19.8.2 Mathcad

Mathcad can perform a wide variety of statistical, curve fitting, and data-smoothing tasks. These include relatively simple jobs like plotting histograms and calculating population statistic summaries such as mean, median, variance, standard deviations, and correlation coefficients. In addition, Mathcad can predict intermediate values by connecting known data points with either straight lines (linear interpolation) using **linterp** or with sections of cubic polynomials (cubic spline interpolation) using **cspline**, **pspline**, or **lspline**. These spline functions allow you to try different ways to deal with interpolation at the end points of the data. The **lspline** function generates a spline curve that is a straight line at the end

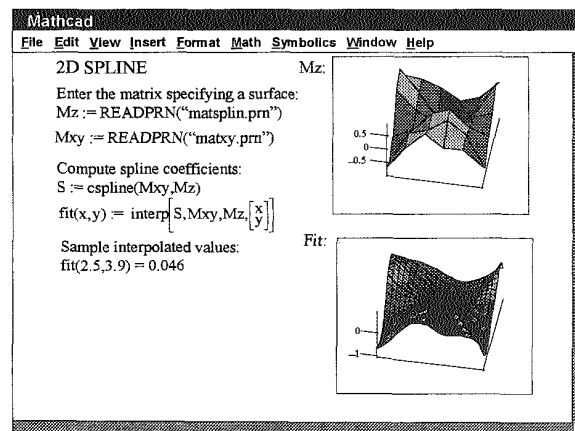
points. The **pspline** function generates a spline curve that is a parabola at the end points. The **cspline** function generates a spline curve that is cubic at the end points. The **interp** function uses the curve fitting results and returns an interpolated y value given an x value. In addition, you can perform two-dimensional cubic spline interpolation by passing a surface through a grid of points.

Mathcad contains a number of functions for performing regression. The **slope** and **intercept** functions return the slope and intercept of the least-squares regression fit line. The **regress** function is used for  $n$ th-order polynomial regression of a complete data set. The **loess** function performs localized  $n$ th-order polynomial regression over spans of the data that you can specify. The **interp** function can also be used to return intermediate values of  $y$  from a regression fit for a given  $x$  point. The **regress** and **loess** functions can also perform multivariate polynomial regression. Mathcad also provides the **linfit** function that is used to model data with a linear combination of arbitrary functions. Finally, the **genfit** function is available for cases where model coefficients appear in arbitrary form. In this case, the more difficult nonlinear equations must be solved by iteration.

Let's do an example that shows how Mathcad is used to perform two-dimensional spline interpolation (Fig. 19.22). The data we will fit is

	x						
x	0	1	2	3	4	5	
0	0.17500	0.14100	-0.13900	-0.51400	-0.29000	0.32700	
1	0.93500	0.16700	-0.76400	-0.98600	-0.30800	0.82600	
2	0.64900	-0.00302	-0.33400	-0.65900	-0.00678	0.23900	
3	-0.55300	-0.22500	0.46700	0.73600	0.10600	-0.09200	
4	-0.97900	0.17500	0.36800	0.81400	0.39000	-0.78200	
5	-0.70700	0.12600	0.76100	0.30200	0.30300	-0.16400	

**FIGURE 19.22**  
2D spline with Mathcad.

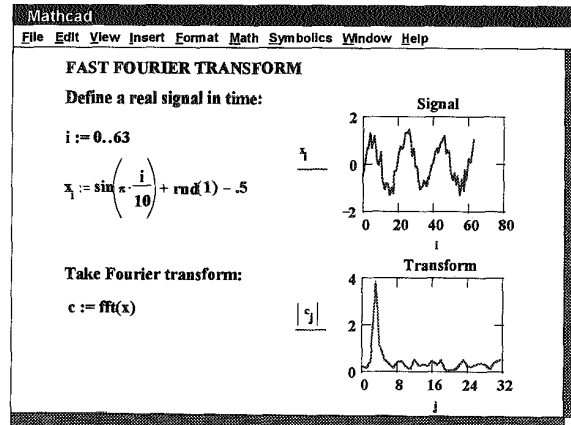


Note that the numbers along the top and left side are the x and y coordinates of the z values in the interior of the matrix.

The first step is to supply the data to Mathcad. To do this, we can create two data files called  **matsplin.prn**  and  **matxy.prn** . The first two active lines in Fig. 19.22 use the  **READ-PRN**  command to read data from these files. The  **matsplin.prn**  file is a simple text file that contains the values of the function (z) to be interpolated at various x and y locations on a rectangular grid. The dimensions of the grid are defined by the data in the  **matxy.prn**  text file. The elements of this file are pairs of x and y values that characterize the diagonal elements of the region. The definition symbol is used to assign the data from the data files to the variables Mz and Mxy. Next, the definition symbol and the  **csplin**  function are used to define the S matrix. This is a matrix that contains values of the second derivative and other numerical results at the various grid locations. This matrix, along with Mz and Mxy, are used by the  **interp**  function to return values of z as the variable fit(x,y) based on the cubic spline interpolation at input values of x and y. Mathcad designed this sequence of operations in this manner so that the interpolating polynomials would not have to be recalculated every time interpolation is required at different values of x and y. With these operations in place, you can interpolate at any location using fit(x,y), as shown with x = 2.5 and y = 3.9. You can also construct a plot of the interpolated surface as shown in Fig. 19.22.

As another example of demonstrating some of Mathcad's curve fitting capabilities, let's use the  **fft**  function for Fourier analysis as in Fig. 19.23. The first line uses the definition symbol to create i as a range variable. Next  $x_i$  is formulated using the  **rnd**  Mathcad function to impart a random component to a sinusoidal signal. The graph of the signal can be placed on the worksheet by clicking to the desired location. This places a red cross hair at that location. Then use the Insert/Graph/X-Y Plot pull down menu to place an empty plot on the worksheet with placeholders for the expressions to be graphed and for the ranges of the x and y axes. Simply type  $x_i$  in the placeholder on the y axis and 0 and 80 for

**FIGURE 19.23**  
FFT with Mathcad.



the x-axis range. Mathcad does all the rest to produce the graph shown in Fig. 19.23. Once the graph has been created you can use the Format/Graph/X-Y Plot pull down menu to vary the type of graph; change the color, type, and weight of the trace of the function; and add titles, labels, and other features. Next,  $c$  is defined as  $\text{fft}(x)$ . This function returns the Fourier transform of  $x$ . The result is a vector,  $c$ , of complex coefficients that represent values in the frequency domain. A plot of the magnitude of  $c_j$  is constructed as above.

### 19.8.3 MATLAB

As summarized in Table 19.2, MATLAB has a variety of built-in functions that span the total capabilities described in this part of the book. The following example illustrates how a few of them can be used.

**TABLE 19.2** Some of the MATLAB functions to implement interpolation, regression, splines, and the FFT.

Function	Description
polyfit	Fit polynomial to data
interp1	1-D interpolation (1-D table lookup)
interp2	2-D interpolation (2-D table lookup)
spline	Cubic spline data interpolation
fft	Discrete Fourier transform

#### EXAMPLE 19.5 Using MATLAB for Curve Fitting

**Problem Statement.** Explore how MATLAB can be employed to fit curves to data. To do this, use the sine function to generate equally spaced  $f(x)$  values from 0 to 10. Employ a step size of 1 so that the resulting characterization of the sine wave is sparse (Fig. 19.24). Then, fit it with (a) linear interpolation, (b) a fifth-order polynomial, and (c) a cubic spline.

**Solution.**

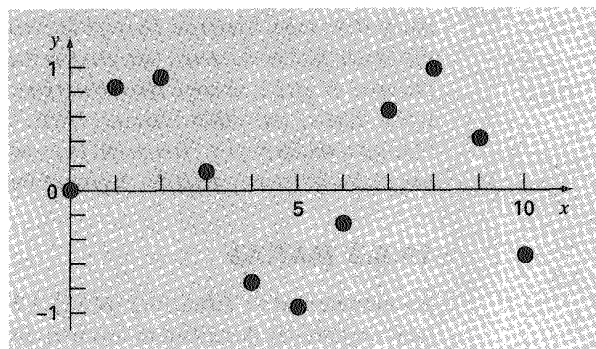
(a) The values for the independent and the dependent variables can be entered into vectors by

```
>> x=0:10;
>> y=sin(x);
```

A new, more finely spaced vector of independent variable values can be generated and stored in the vector  $\mathbf{x}_i$ ,

```
>> xi=0:.25:10;
```

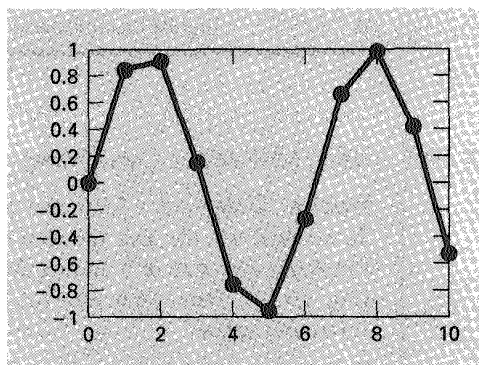
The MATLAB function **interp1** can then be used to generate dependent variable values  $y_i$  for all the  $x_i$  values using linear interpolation. Both the original data  $(x, y)$  along with

**FIGURE 19.24**

Eleven points sampled from a sinusoid.

the linearly interpolated values can be plotted together, as shown in the graph below:

```
>> yi=interp1(x,y,xi);
>> plot(x,y,'o',xi,yi)
```



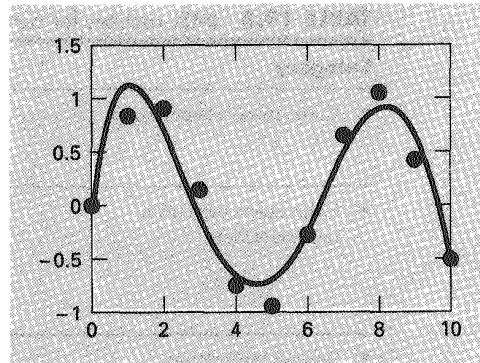
(b) Next, the MATLAB **polyfit** function can be used to generate the coefficients of a fifth-order polynomial fit of the original sparse data,

```
>> p=polyfit(x,y,5)
p=
  0.0008 -0.0290 0.3542 -1.6854 2.5860 -0.0915
```

where the vector **p** holds the polynomial's coefficients. These can, in turn, be used to generate a new set of  $y_i$  values, which can again be plotted along with the original sparse sample,

```
>> yi = polyval(p,xi);
>> plot(x,y,'o',xi,yi)
```

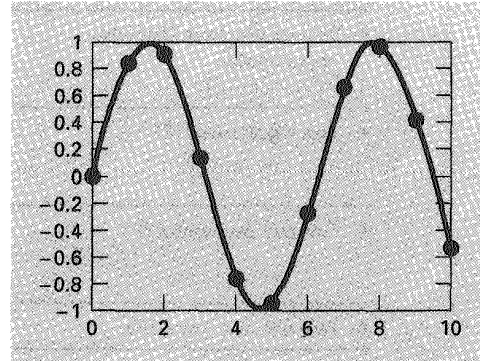




Thus, the polynomial captures the general pattern of the data, but misses most of the points.

- (c) Finally, the MATLAB **spline** function can be used to fit a cubic spline to the original sparse data in the form of a new set of  $y_i$  values, which can again be plotted along with the original sparse sample,

```
>> yi=spline(x,y,xi);
>> plot(x,y,'o',xi,yi)
```



It should be noted that MATLAB also has excellent capabilities to perform Fourier analysis. We devote Sec. 20.3 to an example of how this can be done.

#### 19.8.4 IMSL

IMSL has numerous routines for curve fitting that span all the capabilities covered in this book, and then some. A sample is presented in Table 19.3. In the present discussion, we will focus on the RCURV routine. This routine fits a least-squares polynomial to data.

RCURV is implemented by the following CALL statement:

```
CALL RCURV (NOBS, XDATA, YDATA, NDEG, B, SSPOLY, STAT)
```

**TABLE 19.3** IMSL routines for curve fitting.

Category	Routines	Description
• Cubic spline interpolation	CSIEZ CSINT CSDEC	Easy to use cubic spline routine Not-a-knot Derivative end conditions
• Cubic spline evaluation and integration	CSVAL CSDER CS1GD CSITG	Evaluation Evaluation of the derivative Evaluation on a grid Integration
• B-spline interpolation		
• Piecewise polynomial		
• Quadratic polynomial interpolation routines for gridded data		
• Scattered data interpolation		
• Least-squares approximation	RLINE RCURV FNLSQ	Linear polynomial General polynomial General functions
• Cubic spline smoothing		
• Rational weighted Chebyshev approximation		Rational weighted Chebyshev approximation
• Real trigonometric FFT	FFTRF FFTRB FFTRI	Forward transform Backward or inverse transform Initialization routine for FFTR
• Complex exponential FFT	FFTCF FFTCB FFTCI	Forward transform Backward or inverse transform Initialization routine for FFTC
• Real sine and cosine FFTs		
• Real quarter sine and quarter cosine FFTs		
• Two- and three-dimensional complex FFTs		
• Convolutions and correlations		
• Laplace transform		

where NOBS = Number of observations. (Input)

XDATA = Vector of length NOBS containing the x values. (Input)

YDATA = Vector of length NOBS containing the y values. (Input)

NDEG = Degree of polynomial. (Input)

B = Vector of length NDEG + 1 containing the coefficients.

SSPOLY = Vector of length NDEG + 1 containing the sequential sums of squares.  
 (Output) SSPOLY(1) contains the sum of squares due to the mean. For  
 $i = 1, 2, \dots, \text{NDEG}$ , SSPOLY( $i + 1$ ) contains the sum of squares due  
 to  $x^i$  adjusted for the mean,  $x, x^2, \dots$ , and  $x^{i-1}$ .

STAT = Vector of length 10 containing statistics described in Table 19.4. (Output  
 where 1 = Mean of  $x$

2 = Mean of  $y$

3 = Sample variance of  $x$

4 = Sample variance of  $y$

5 = R-squared (in percent)

6 = Degrees of freedom for regression

7 = Regression sum of squares

8 = Degrees of freedom for error

9 = Error sum of squares

10 = Number of data points ( $x, y$ ) containing NaN (not a number) as  
 an  $x$  or  $y$  value.

#### EXAMPLE 19.6 Using IMSL for Polynomial Regression

**Problem Statement.** Use RCURV to determine the cubic polynomial that provides a least-squares fit of the following data:

x	0.05	0.12	0.15	0.30	0.45	0.70	0.84	1.05
y	0.957	0.851	0.832	0.720	0.583	0.378	0.295	0.156

**Solution.** An example of a main Fortran 90 program and function using RCURV to solve this problem can be written as

```
PROGRAM Fitpoly
  use msimsl
  IMPLICIT NONE
  INTEGER::ndeg,nobs,i,j
  PARAMETER (ndeg=3, nobs=8)
  REAL::b(ndeg+1),sspoly(ndeg+1),stat(10),x(nobs),y(nobs),
        ycalc(nobs)
  DATA x/0.05,0.12,0.15,0.30,0.45,0.70,0.84,1.05/
  DATA y/0.957,0.851,0.832,0.720,0.583,0.378,0.295,0.156/
  CALL RCURV(nobs,x,y,ndeg,B,sspoly,stat)
  PRINT *, 'Fitted polynomial is'
  DO i = 1,ndeg+1
    PRINT '(1X, ''X^'',I1,'' TERM: '',F8.4)', i-1, b(i)
  END DO
  PRINT *
  PRINT '(1X, ''R^2: '',F5.2, ''%'')',stat(5)
  PRINT *
  PRINT *, 'NO.   X   Y   YCALC'
  DO i = 1,nobs
```

```

      ycalc=0.
      DO j = 1,ndeg+1
        ycalc(i)=ycalc(i)+b(j)*x(i)**(j-1)
      END DO
      PRINT '(1X,I8,3(5X,F8.4))', i, x(i), y(i), ycalc(i)
    END DO
  END

```

An example run is

```

Fitted polynomial is
X^0 TERM:   .9909
X^1 TERM:  -1.0312
X^2 TERM:   .2785
X^3 TERM:  -0.0513
R^2: 99.81%

```

NO.	X	Y	YCALC
1	.0500	.9570	.9401
2	.1200	.8510	.8711
3	.1500	.8320	.8423
4	.3000	.7200	.7053
5	.4500	.5830	.5786
6	.7000	.3780	.3880
7	.8400	.2950	.2908
8	1.0500	.1560	.1558

## PROBLEMS

**19.1** The pH in a reactor varies sinusoidally over the course of a day. Use least-squares regression to fit Eq. (19.11) to the following data. Use your fit to determine the mean, amplitude, and time of maximum pH.

Time, hr	0	2	4	5	7	8.5	12	15	20	22	24
pH	7.3	7	7.1	6.4	7.4	7.2	8.9	8.8	8.9	7.9	7

**19.2** The solar radiation for Georgetown, South Carolina has been tabulated as

Time, mo		J	F	M	A	M	J	J	A	S	O	N	D
Radiation, W/m <sup>2</sup>		122	—	188	230	267	270	252	—	196	160	138	120

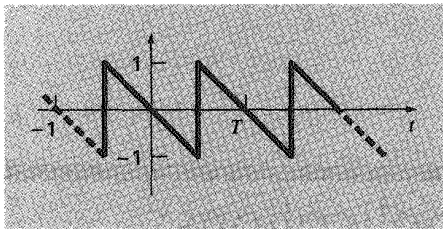
Assuming each month is 30 days long, fit a sinusoid to this data. Use the resulting equation to predict the radiation in mid-August.

**19.3** The average values of a function can be determined by

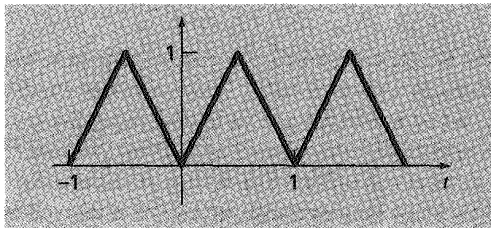
$$\overline{f(x)} = \frac{\int_0^x f(x) dx}{x}$$

Use this relationship to verify the results of Eq. (19.13).

**19.4** Use a continuous Fourier series to approximate the sawtooth wave in Fig. P19.4. Plot the first three terms along with the summation.



**FIGURE P19.4**  
A sawtooth wave.



**FIGURE P19.5**  
A triangular wave

**19.5** Use a continuous Fourier series to approximate the wave form in Fig. P19.5. Plot the first three terms along with the summation.

**19.6** Construct amplitude and phase line spectra for Prob. 19.4.

**19.7** Construct amplitude and phase line spectra for Prob. 19.5.

**19.8** A half-wave rectifier can be characterized by

$$y(t) = C_1 \left[ \frac{1}{\pi} + \frac{1}{2} \sin t - \frac{2}{3\pi} \cos 2t - \frac{2}{15\pi} \cos 4t - \frac{2}{35\pi} \cos 6t - \dots \right]$$

where  $C_1$  is the amplitude of the wave. Plot the first four terms along with the summation.

**19.9** Construct amplitude and phase line spectra for Prob. 19.8.

**19.10** Develop a user-friendly subprogram for the DFT based on the algorithm from Fig. 19.12. Test it by duplicating Fig. 19.13.

**19.11** Use the program from Prob. 19.10 to compute a DFT for the

triangular wave from Prob. 19.8. Sample the wave from  $t = 0$  to  $4T$ . Use 32, 64, and 128 sample points. Time each run and plot execution versus  $N$  to verify Fig. 19.14.

**19.12** Develop a user-friendly subprogram for the FFT based on the algorithm from Fig. 19.18. Test it by duplicating Fig. 19.13.

**19.13** Repeat Prob. 19.11 using the software you developed in Prob. 19.12.

**19.14** Use Excel's Trendline command to fit a power equation to

x	2.5	3.5	5	6	7.5	10	12.5	15	17.5	20
y	7	5.5	3.9	3.6	3.1	2.8	2.6	2.4	2.3	2.3

Plot  $y$  versus  $x$  along with the power equation and  $r^2$ .

**19.15** Use the Excel Data Analysis Toolpack to develop a fourth-order regression polynomial to the following data for the dissolved oxygen concentration of fresh water versus temperature at sea level.

$T, ^\circ\text{C}$	0	8	16	24	32	40
$a, \text{mg/l}$	14.621	11.843	9.870	8.418	7.305	6.413

**19.16** Use the Excel Data Analysis Toolpack to fit a straight line to the following data. Determine the 90% confidence interval for the intercept. If it encompasses zero, redo the regression, but with the intercept forced to be zero (this is an option on the **Regression** dialogue box)

x	2	4	6	8	10	12	14
y	5.7	6.4	12.3	17.7	18.9	25.7	26.8

**19.17** Use Mathcad to fit a cubic spline (with a straight line at the end points) to the following data:

x	0	2	4	7	10	12
y	20	20	12	7	6	5.6

Determine the value of  $y$  at  $x = 3$ .

**19.18** Use Mathcad to generate 64 points from the function

$$f(t) = \cos(3t) + \sin(10t)$$

from  $t = 0$  to  $2\pi$ . As in Sec. 19.8.2 add a random component to the signal. Take an FFT of these values and plot the results.

**19.19** In a fashion similar to Sec. 19.8.3, use MATLAB to fit the data from Prob. 19.17 using (a) linear interpolation, (b) a fifth-order polynomial, and (c) a spline.

**19.20** Repeat Prob. 19.18, but use MATLAB to perform the analysis.

**19.21** Repeat Prob. 19.15, but use the IMSL routine, RCURV.

# CHAPTER 20

## Engineering Applications: Curve Fitting

The purpose of this chapter is to use the numerical methods for curve fitting to solve some engineering problems. The first application, which is taken from chemical engineering, demonstrates how a nonlinear model can be linearized and fit to data using linear regression. The second application employs splines to study a problem that has relevance to the environmental area of civil engineering: heat and mass-transport in a stratified lake.

The third application illustrates how a fast Fourier transform can be employed in electrical engineering to analyze a signal by determining its major harmonics. The final application demonstrates how multiple linear regression is used to analyze experimental data for a fluids problem taken from mechanical and aerospace engineering.

### 20.1 LINEAR REGRESSION AND POPULATION MODELS (CHEMICAL/PETROLEUM ENGINEERING)

**Background.** Population growth models are important in many fields of engineering. Fundamental to many of the models is the assumption that the rate of change of the population ( $dp/dt$ ) is proportional to the actual population ( $p$ ) at any time ( $t$ ), or in equation form,

$$\frac{dp}{dt} = kp \quad (20.1)$$

where  $k$  = a proportionality factor called the specific growth rate and has units of  $\text{time}^{-1}$ . If  $k$  is a constant, then the solution of Eq. (20.1) can be obtained from the theory of differential equations:

$$p(t) = p_0 e^{kt} \quad (20.2)$$

where  $p_0$  = the population when  $t = 0$ . It is observed that  $p(t)$  in Eq. (20.2) approaches infinity as  $t$  becomes large. This behavior is clearly impossible for real systems. Therefore, the model must be modified to make it more realistic.

**Solution.** First, it must be recognized that the specific growth rate  $k$  cannot be constant as the population becomes large. This is the case because, as  $p$  approaches infinity, the organism being modeled will become limited by factors such as food shortages and toxic waste production. One way to express this mathematically is to use a saturation-growth-rate

model such that

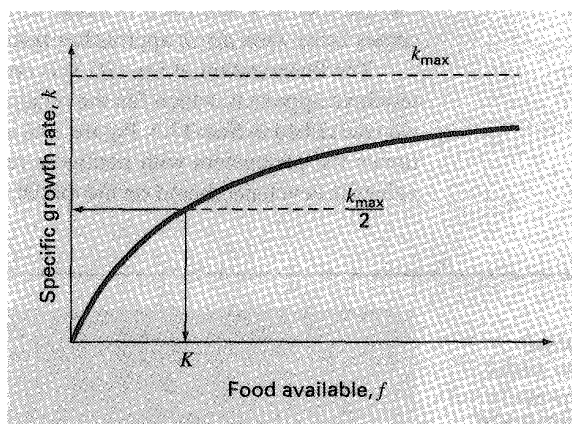
$$k = k_{\max} \frac{f}{K + f} \quad (20.3)$$

where  $k_{\max}$  = the *maximum attainable growth rate* for large values of food ( $f$ ) and  $K$  = the *half-saturation constant*. The plot of Eq. (20.3) in Fig. 20.1 shows that when  $f = K$ ,  $k = k_{\max}/2$ . Therefore,  $K$  is that amount of available food that supports a population growth rate equal to one-half the maximum rate.

The constants  $K$  and  $k_{\max}$  are empirical values based on experimental measurements of  $k$  for various values of  $f$ . As an example, suppose the population  $p$  represents a yeast employed in the commercial production of beer and  $f$  is the concentration of the carbon source to be fermented. Measurements of  $k$  versus  $f$  for the yeast are shown in Table 20.1. It is

**FIGURE 20.1**

Plot of specific growth rate versus available food for the saturation-growth-rate model used to characterize microbial kinetics. The value  $K$  is called a half-saturation constant because it conforms to the concentration where the specific growth rate is half its maximum value.



**TABLE 20.1** Data used to evaluate the constants for a saturation-growth-rate model to characterize microbial kinetics.

$f$ , mg/L	$k$ , day <sup>-1</sup>	$1/f$ , L/mg	$1/k$ , day
7	0.29	0.14286	3.448
9	0.37	0.11111	2.703
15	0.48	0.06666	2.083
25	0.65	0.04000	1.538
40	0.80	0.02500	1.250
75	0.97	0.01333	1.031
100	0.99	0.01000	1.010
150	1.07	0.00666	0.935

required to calculate  $k_{\max}$  and  $K$  from this empirical data. This is accomplished by inverting Eq. (20.3) in a manner similar to Eq. (17.17) to yield

$$\frac{1}{k} = \frac{K + f}{k_{\max} f} = \frac{K}{k_{\max}} \frac{1}{f} + \frac{1}{k_{\max}} \quad (20.4)$$

By this manipulation, we have transformed Eq. (20.3) into a linear form; that is,  $1/k$  is a linear function of  $1/f$ , with slope  $K/k_{\max}$  and intercept  $1/k_{\max}$ . These values are plotted in Fig. 20.2.

Because of this transformation, the linear least-squares procedures described in Chap. 17 can be used to determine  $k_{\max} = 1.23 \text{ day}^{-1}$  and  $K = 22.18 \text{ mg/L}$ . These results combined with Eq. (20.3) are compared to the untransformed data in Fig. 20.3, and when substituted into the model in Eq. (20.1), give

$$\frac{dp}{dt} = 1.23 \frac{f}{22.18 + f} p \quad (20.5)$$

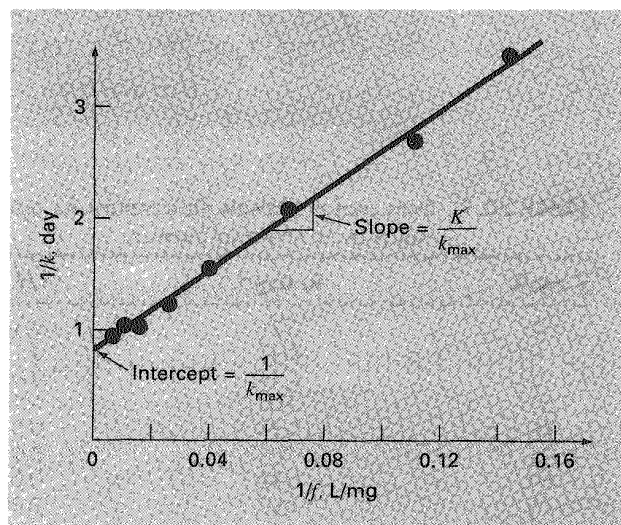
Note that the fit yields a sum of the squares of the residuals (as computed for the untransformed data) of 0.001305.

Equation (20.5) can be solved using the theory of differential equations or using numerical methods discussed in Chap. 25 when  $f(t)$  is known. If  $f$  approaches zero as  $p$  becomes large, then  $dp/dt$  approaches zero and the population stabilizes.

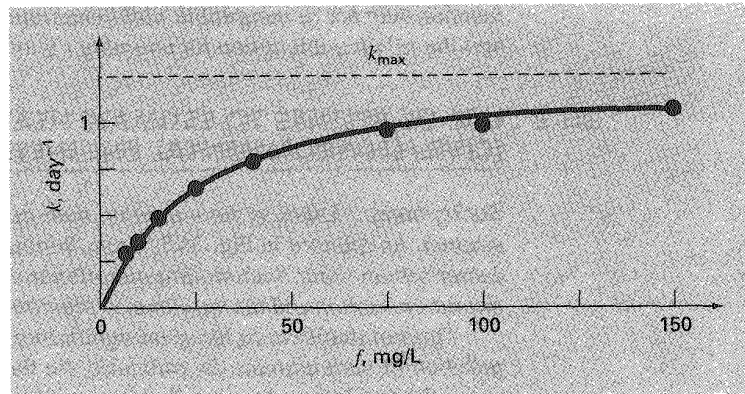
The linearization of Eq. (20.3) is one way to evaluate the constants  $k_{\max}$  and  $K$ . An alternative approach, which fits the relationship in its original form, is the nonlinear regression described in Sec. 17.5. Figure 20.4 shows how the Excel Solver tool can be used to estimate the parameters with nonlinear regression. As can be seen, a column of predicted values is developed based on the model and the parameter guesses. These are used to gen-

**FIGURE 20.2**

Linearized version of the saturation-growth-rate model. The line is a least-squares fit that is used to evaluate the model coefficients  $k_{\max} = 1.23 \text{ day}^{-1}$  and  $K = 22.18 \text{ mg/L}$  for a yeast that is used to produce beer.





**FIGURE 20.3**

Fit of the saturation-growth-rate model to a yeast employed in the commercial production of beer.

	A	B	C	D
1	kmax	1.2301		
2	K	22.1386		
3				
4	f	k	k-predicted	Res <sup>2</sup>
5	7	0.29	0.295508	0.000030
6	9	0.37	0.355536	0.000209
7	15	0.48	0.496828	0.000283
8	25	0.65	0.652384	0.000006
9	40	0.80	0.791842	0.000067
10	75	0.97	0.949751	0.000410
11	100	0.99	1.007134	0.000294
12	150	1.07	1.071897	0.000004
13				
14			SSR	0.001302

$=B\$1 * A5 / (B\$2 + A5)$   
 $=(B5 - C5)^2$   
 $=SUM(D5..D12)$

**FIGURE 20.4**

Nonlinear regression to fit the saturation-growth-rate model to a yeast employed in the commercial production of beer.

erate a column of squared residuals that are summed, and the result is placed in cell D14. The Excel Solver is then invoked to minimize cell D14 by adjusting cells B1:B2. The result, as shown in Fig. 20.4, yields estimates of  $k_{\max} = 1.23$  and  $K = 22.14$ , with an  $S_r = 0.001302$ . Thus, although, as expected, the nonlinear regression yields a slightly better fit, the results are almost identical. In other applications, this may not be true (or the

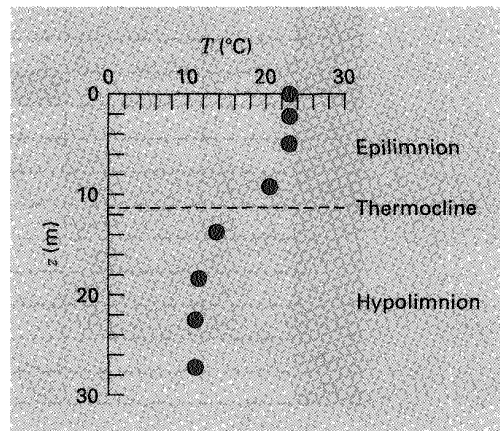
function may not be compatible with linearization) and nonlinear regression could represent the only feasible option for obtaining a least-squares fit.

## 20.2 USE OF SPLINES TO ESTIMATE HEAT TRANSFER (CIVIL/ENVIRONMENTAL ENGINEERING)

**Background.** Lakes in the temperate zone can become thermally stratified during the summer. As depicted in Fig. 20.5, warm, buoyant water near the surface overlies colder, denser bottom water. Such stratification effectively divides the lake vertically into two layers: the *epilimnion* and the *hypolimnion* separated by a plane called the *thermocline*.

Thermal stratification has great significance for environmental engineers studying the pollution of such systems. In particular, the thermocline greatly diminishes mixing between the two layers. As a result, decomposition of organic matter can lead to severe depletion of oxygen in the isolated bottom waters.

The location of the thermocline can be defined as the inflection point of the temperature-depth curve—that is, the point at which  $d^2T/dz^2 = 0$ . It is also the point at which the absolute value of the first derivative or gradient is a maximum. Use cubic splines to determine the thermocline depth for Platte Lake (Table 20.2). Also use the splines to determine the value of the gradient at the thermocline.



**FIGURE 20.5**

Temperature versus depth during summer for Platte Lake, Michigan.

**TABLE 20.2** Temperature versus depth during summer for Platte Lake, Michigan.

$T, ^\circ\text{C}$	22.8	22.8	22.8	20.6	13.9	11.7	11.1	11.1
$z, \text{m}$	0	2.3	4.9	9.1	13.7	18.3	22.9	27.2

Solution. The data is analyzed with a program that was developed based on the pseudocode from Fig. 18.18. The results as displayed in Table 20.3 list the spline predictions along with first and second derivatives at intervals of 1 m down through the water column.

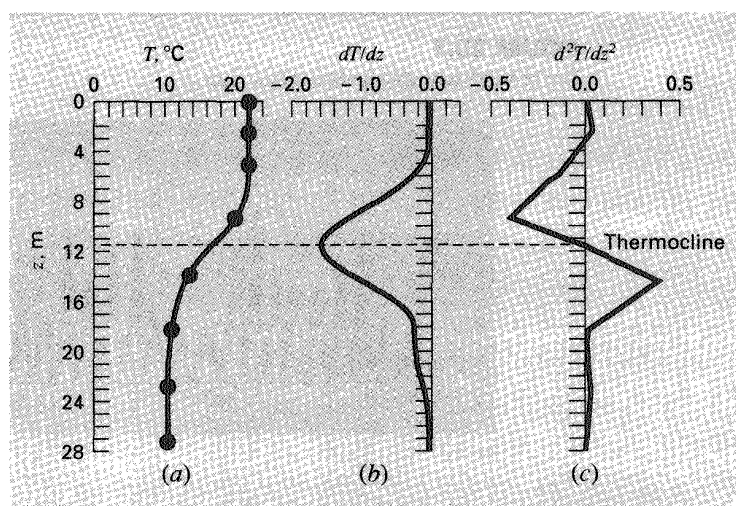
The results are plotted in Fig. 20.6. Notice how the thermocline is clearly located at the depth where the gradient is highest (i.e., the absolute value of the derivative is greatest) and the second derivative is zero. The depth is 11.35 m and the gradient at this point is  $-1.61^{\circ}\text{C}/\text{m}$ .

**TABLE 20.3** Output of spline program based on pseudocode from Fig. 18.18.

Depth (m)	T (C)	dT/dz	d <sup>2</sup> T/dz <sup>2</sup>	Depth (m)	T (C)	dT/dz	d <sup>2</sup> T/dz <sup>2</sup>
0.	22.8000	-.0115	.0000	15.	12.7652	-.6518	.3004
1.	22.7907	-.0050	.0130	16.	12.2483	-.3973	.2086
2.	22.7944	.0146	.0261	17.	11.9400	-.2346	.1167
3.	22.8203	.0305	-.0085	18.	11.7484	-.1638	.0248
4.	22.8374	-.0055	-.0635	19.	11.5876	-.1599	.0045
5.	22.7909	-.0966	-.1199	20.	11.4316	-.1502	.0148
6.	22.6229	-.2508	-.1884	21.	11.2905	-.1303	.0251
7.	22.2665	-.4735	-.2569	22.	11.1745	-.1001	.0354
8.	21.6531	-.7646	-.3254	23.	11.0938	-.0596	.0436
9.	20.7144	-1.1242	-.3939	24.	11.0543	-.0212	.0332
10.	19.4118	-1.4524	-.2402	25.	11.0480	.0069	.0229
11.	17.8691	-1.6034	-.0618	26.	11.0646	.0245	.0125
12.	16.2646	-1.5759	.1166	27.	11.0936	.0318	.0021
13.	14.7766	-1.3702	.2950	28.	11.1000	.0000	.0000
14.	13.5825	-.9981	.3923				

**FIGURE 20.6**

Plots of (a) temperature, (b) gradient, and (c) second derivative versus depth (m) generated with the cubic spline program. The thermocline is located at the inflection point of the temperature-depth curve.



### 20.3 FOURIER ANALYSIS (ELECTRICAL ENGINEERING)

**Background.** Fourier analysis is used in many areas of engineering. However, it is extensively employed in electrical engineering applications such as signal processing.

In 1848, Rudolph Wolf devised a method for quantifying solar activity by counting the number of individual spots and groups of spots on the sun's surface. He computed a quantity, now called a *Wolf sunspot number*, by adding 10 times the number of groups plus the total count of individual spots. As in Fig. 20.7, the record of this number extends back to 1770. On the basis of the early historical records, Wolf determined the cycle's length to be 11.1 years.

Use a Fourier analysis to confirm this result by applying an FFT to the data from Fig. 20.3. Pinpoint the period by developing a power versus period plot.

**Solution.** The data for year and sunspot number was downloaded from the web<sup>1</sup> and stored in a tab-delimited file: sunspot.dat. The file can be loaded into MATLAB and the year and number information assigned to vectors of the same name,

```
>> load sunspot.dat
>> year=sunspot(:,1);number=sunspot(:,2);
```

Next, an FFT can be applied to the sunspot numbers

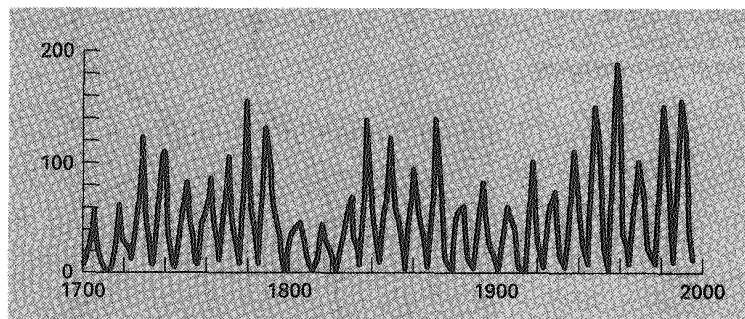
```
>> y=fft(number);
```

After getting rid of the first harmonic, the length of the FFT is determined ( $n$ ) and then the power and frequency calculated,

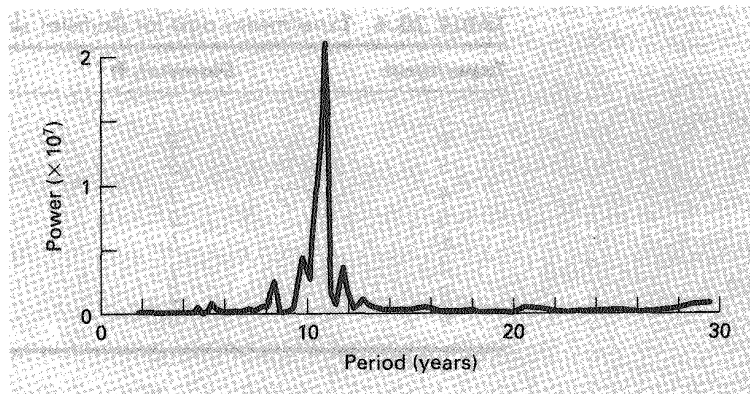
```
>> y(1)=[];
>> n=length(y);
>> power=abs(y(1:n/2)).^2;
>> nyquist=1/2;
>> freq=(1:n/2)/(n/2)*nyquist;
```

**FIGURE 20.7**

Plot of Wolf sunspot number versus year.



<sup>1</sup>At the time of this book's printing, the html was <http://www.ngdc.noaa.gov/stp/SOLAR/SSN/ssn.html>.

**FIGURE 20.8**

Power spectrum for Wolf sunspot numbers.

At this point, the power spectrum is a plot of power versus frequency. However, because period is more meaningful in the present context, we can determine the period and a power-period plot,

```
>> period=1./freq
>> plot(period,power);
```

The result, as shown in Fig. 20.8, indicates a peak at about 11 years. The exact value can be computed with

```
>> index=find(power==max(power));
>> period(index)
```

```
ans=
    10.9259
```

## 20.4 ANALYSIS OF EXPERIMENTAL DATA (MECHANICAL/AEROSPACE ENGINEERING)

**Background.** Engineering design variables are often dependent on several independent variables. Often this functional dependence is best characterized by multivariate power equations. As discussed in Sec. 17.3, a multiple linear regression of log-transformed data provides a means to evaluate such relationships.

For example, a mechanical engineering study indicates that fluid flow through a pipe is related to pipe diameter and slope (Table 20.4). Use multiple linear regression to analyze this data. Then use the resulting model to predict the flow for a pipe with a diameter of 2.5 ft and a slope of 0.025 ft/ft.

**Solution.** The power equation to be evaluated is

$$Q = a_0 D^{a_1} S^{a_2} \quad (20.6)$$

**TABLE 20.4** Experimental data for diameter, slope, and flow of concrete circular pipes.

Experiment	Diameter, ft	Slope, ft/ft	Flow, ft <sup>3</sup> /s
1	1	0.001	1.4
2	2	0.001	8.3
3	3	0.001	24.2
4	1	0.01	4.7
5	2	0.01	28.9
6	3	0.01	84.0
7	1	0.05	11.1
8	2	0.05	69.0
9	3	0.05	200.0

where  $Q$  = flow (ft<sup>3</sup>/s),  $S$  = slope (ft/ft),  $D$  = pipe diameter (ft), and  $a_0$ ,  $a_1$ , and  $a_2$  = coefficients. Taking the logarithm of this equation yields

$$\log Q = \log a_0 + a_1 \log D + a_2 \log S$$

In this form, the equation is suited for multiple linear regression because  $\log Q$  is a linear function of  $\log S$  and  $\log D$ . Using the logarithm (base 10) of the data in Table 20.4, we can generate the following normal equations expressed in matrix form [recall Eq. (17.22)]:

$$\begin{bmatrix} 9 & 2.334 & -18.903 \\ 2.334 & 0.954 & -4.903 \\ -18.903 & -4.903 & 44.079 \end{bmatrix} \begin{Bmatrix} \log a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 11.691 \\ 3.945 \\ -22.207 \end{Bmatrix}$$

This system can be solved using Gauss elimination for

$$\log a_0 = 1.7475$$

$$a_1 = 2.62$$

$$a_2 = 0.54$$

If  $\log a_0 = 1.7475$ ,  $a_0 = 10^{1.7475} = 55.9$ , and Eq. (20.6) is

$$Q = 55.9D^{2.62}S^{0.54} \quad (20.7)$$

Eq. (20.7) can be used to predict flow for the case of  $D = 2.5$  ft and  $S = 0.025$  ft/ft, as in

$$Q = 55.9(2.5)^{2.62}(0.025)^{0.54} = 84.1 \text{ ft}^3/\text{s}$$

It should be noted that Eq. (20.7) can be used for other purposes besides computing flow. For example, the slope is related to head loss  $h_L$  and pipe length  $L$  by  $S = h_L/L$ . If this relationship is substituted into Eq. (20.7) and the resulting formula solved for  $h_L$ , the following equation can be developed:

$$h_L = \frac{L}{1721} Q^{1.85} D^{4.85}$$

This relationship is called the *Hazen-Williams equation*.

**PROBLEMS**

**Chemical/Petroleum Engineering**

**20.1** Perform the same computation as in Sec. 20.1, but use linear regression and transformations to fit the data with a power equation. Ignore the first point when fitting the equation.

**20.2** You perform experiments and determine the following values of heat capacity  $c$  at various temperatures  $T$  for a gas:

$T$	-40	-20	10	70	100	120
$c$	1250	1280	1350	1480	1580	1700

Use regression to determine a model to predict  $c$  as a function of  $T$ .

**20.3** The saturation concentration of dissolved oxygen in water as a function of temperature and chloride concentration is listed in Table P20.3. Use interpolation to estimate the dissolved oxygen level for  $T = 18^\circ\text{C}$  with chloride = 10,000 mg/L.

**20.4** For the data in Table P20.3, use polynomial interpolation to derive a second-order predictive equation for dissolved oxygen concentration as a function of temperature for the case where chloride concentration is equal to 20,000 mg/L. Use the equation to estimate the dissolved oxygen concentration for  $T = 8^\circ\text{C}$ .

**20.5** Use multiple linear regression to derive a predictive equation for dissolved oxygen concentration as a function of temperature and chloride based on the data from Table P20.3. Use the equation

to estimate the concentration of dissolved oxygen for a chloride concentration of 15,000 mg/L at  $T = 12^\circ\text{C}$ .

**20.6** As compared to the models from Probs. 20.4 and 20.5, a somewhat more sophisticated model that accounts for the effect of both temperature and chloride on dissolved oxygen saturation can be hypothesized as being of the form,

$$o_s = f_3(T) + f_1(c)$$

That is, a third-order polynomial in temperature and a linear relationship in chloride is assumed to yield superior results. Use the general linear least-squares approach to fit this model to the data in Table P20.3. Use the resulting equation to estimate the dissolved oxygen concentration for a chloride concentration of 20,000 mg/L at  $T = 30^\circ\text{C}$ .

**20.7** It is known that the tensile strength of a plastic increases as a function of the time it is heat-treated. The following data is collected:

Time	10	15	20	25	40	50	55	60	75
Tensile strength	4	20	18	50	33	48	80	60	78

Fit a straight line to this data and use the equation to determine the tensile strength at a time of 30 min.

**20.8** The following data was gathered to determine the relationship between pressure and temperature of a fixed volume of 1 kg of nitrogen. The volume is  $10\text{ m}^3$ .

$T, ^\circ\text{C}$	-20	0	20	40	50	70	100	120
$p, \text{N/m}^2$	7500	8104	8700	9300	9620	10,200	11,200	11,700

Employ the ideal gas law  $pV = nRT$  to determine  $R$  on the basis of this data. Note that for the law  $T$  must be expressed in kelvins.

**TABLE P20.3** Dependency of dissolved oxygen concentration on temperature and chloride concentration.

Temperature, $^\circ\text{C}$	Dissolved Oxygen (mg/L) for Stated Concentration of Chloride and Temperature		
	Chloride = 0 mg/L	Chloride = 10,000 mg/L	Chloride = 20,000 mg/L
5	12.8	11.6	10.5
10	11.3	10.3	9.2
15	10.0	9.1	8.2
20	9.0	8.2	7.4
25	8.2	7.4	6.7
30	7.4	6.8	6.1

**20.9** The specific volume of a superheated steam is listed in steam tables for various temperatures. For example, at a pressure of 2950 lb/in<sup>2</sup>, absolute:

$T, ^\circ\text{F}$	700	720	740	760	780
$v$	0.1058	0.1280	0.1462	0.1603	0.1703

Determine  $v$  at  $T = 750^\circ\text{F}$ .

**20.10** A reactor is thermally stratified as in the following table:

Depth, m	0	0.5	1.0	1.5	2.0	2.5	3.0
Temperature, $^\circ\text{C}$	70	68	55	22	13	11	10

As depicted in Fig. P20.10, the tank can be idealized as two zones separated by a strong temperature gradient or *thermocline*. The depth of this gradient can be defined as the inflection point of the temperature-depth curve—that is, the point at which  $d^2T/dz^2 = 0$ . At this depth, the heat flux from the surface to the bottom layer can be computed with Fourier's law,

$$J = -k \frac{dT}{dz}$$

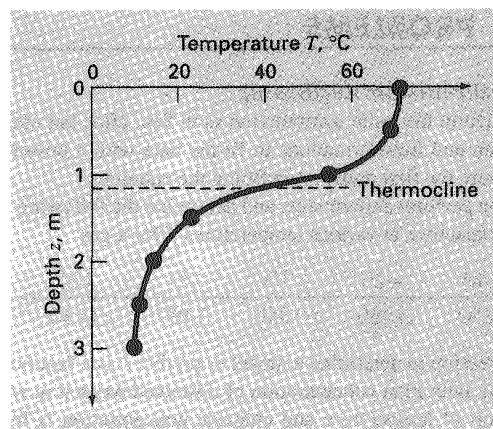
#### Civil/Environment Engineering

**20.11** The shear stress, in kips per square foot (ksf), of nine specimens taken at various depths in a clay stratum are

Depth, m	1.9	3.1	4.2	5.1	5.8	6.9	8.1	9.3	10.0
Stress, ksf	0.3	0.6	0.4	0.9	0.7	1.1	1.5	1.3	1.6

Estimate the shear stress at a depth of 4.5 m.

**20.12** A transportation engineering study was conducted to determine the proper design of bike lanes. Data was gathered on bike-lane widths and average distance between bikes and passing cars. The data from 11 streets is



**FIGURE P20.10**

Use a cubic spline fit of this data to determine the thermocline depth. If  $k = 0.01 \text{ cal}/(\text{s} \cdot \text{cm} \cdot ^\circ\text{C})$  compute the flux across this interface.

Distance $x$ , ft	3	8	5	8	6	6	10	10	4	5	7
lane width $y$ , ft	5	10	7	7.5	7	6	8	9	5	5.5	8

- Plot the data.
- Fit a straight line to the data with linear regression. Add this line to the plot.
- If the minimum safe average distance between bikes and passing cars is considered to be 7 ft, determine the corresponding minimum lane width.

**20.13** In water-resources engineering the sizing of reservoirs depends on accurate estimates of water flow in the river that is being impounded. For some rivers, long-term historical records of such flow data are difficult to obtain. In contrast, meteorological data on precipitation is often available for many years past. Therefore it is often useful to determine a relationship between flow and precipitation. This relationship can then be used to estimate flows for years when only precipitation measurements were made. The following data is available for a river that is to be dammed:

Precipitation, cm	88.9	101.6	104.1	139.7	132.1	94.0	116.8	121.9	99.1
Flow, m <sup>3</sup> /s	114.7	172.0	152.9	269.0	206.4	161.4	175.8	239.0	130.0

- Plot the data.
- Fit a straight line to the data with linear regression. Superimpose this line on your plot.
- Use the best-fit line to predict the annual water flow if the precipitation is 120 cm.



20.14 The concentration of total phosphorus ( $p$  in  $\text{mg}/\text{m}^3$ ) and chlorophyll  $a$  ( $c$  in  $\text{mg}/\text{m}^3$ ) for each of the Great Lakes is

	$p$	$c$
Lake Superior	4.5	0.8
Lake Michigan	8.0	2.0
Lake Huron	5.5	1.2
Lake Erie:		
West basin	39.0	11.0
Central basin	19.5	4.4
East basin	17.5	3.8
Lake Ontario	21.0	5.5

Chlorophyll  $a$  is a parameter that indicates how much plant life is suspended in the water. As such, it indicates how unclear and unsightly the water appears. Use the above data to determine a relationship to predict  $c$  as a function of  $p$ . Use this equation to predict the level of chlorophyll that can be expected if waste treatment is used to lower the phosphorus concentration of western Lake Erie to  $15 \text{ mg}/\text{m}^3$ .

20.15 The vertical stress  $\sigma_z$  under the corner of a rectangular area subjected to a uniform load of intensity  $q$  is given by the solution of Boussinesq's equation:

$$\sigma_z = \frac{q}{4\pi} \left[ \frac{2mn\sqrt{m^2+n^2+1}}{m^2+n^2+1+m^2n^2} \frac{m^2+n^2+2}{m^2+n^2+1} + \sin^{-1} \left( \frac{2mn\sqrt{m^2+n^2+1}}{m^2+n^2+1+m^2n^2} \right) \right]$$

Because this equation is inconvenient to solve manually, it has been reformulated as

$$\sigma_z = qf_z(m, n)$$

where  $f_z(m, n)$  is called the influence value and  $m$  and  $n$  are dimensionless ratios, with  $m = a/z$  and  $n = b/z$  and  $a$  and  $b$  as defined in Fig. P20.36. The influence value is then listed in a table, a portion of which is given here:

$m$	$n = 1.2$	$n = 1.4$	$n = 1.6$
0.1	0.02926	0.03007	0.03058
0.2	0.05733	0.05894	0.05994
0.3	0.08323	0.08561	0.08709
0.4	0.10631	0.10941	0.11135
0.5	0.12626	0.13003	0.13241
0.6	0.14309	0.14749	0.15027
0.7	0.15703	0.16199	0.16515
0.8	0.16843	0.17389	0.17739

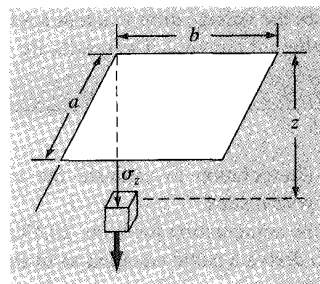


FIGURE P20.15

(a) If  $a = 4.8$  and  $b = 16$ , use a third-order interpolating polynomial to compute  $\sigma_z$  at a depth 10 m below the corner of a rectangular footing that is subject to a total load of 200 t (metric tons). Express your answer in tonnes per square meter. Note that  $q$  is equal to the load per area.

(b) Solve Part (a) but use Mathcad's cspline function as described in Sec. 19.8.2.

20.16 Three disease-carrying organisms decay exponentially in lake water according to the following model.

$$p(t) = Ae^{-1.5t} + Be^{-0.3t} + Ce^{-0.05t}$$

Estimate the initial population of each organism ( $A$ ,  $B$ , and  $C$ ) given the following measurements:

$t, \text{hr}$	0.5	1	2	3	4	5	6	7	8	9
$p(t)$	7	5.2	3.8	3.2	2.5	2.1	1.8	1.5	1.2	1.1

20.17 The mast of a sailboat has a cross-sectional area of  $10.65 \text{ cm}^2$  and is constructed of an experimental aluminum alloy. Tests were performed to define the relationship between stress and strain. The test results are

Strain, $\text{cm}/\text{cm}$	0.002	0.0045	0.0060	0.0013	0.0085	0.0005
Stress, $\text{N}/\text{cm}^2$	4965	5172	5517	3586	6896	1241

The stress caused by wind can be computed as  $F/A_c$ ; where  $F$  = force in the mast and  $A_c$  = mast's cross-sectional area. This value can then be substituted into Hooke's law to determine the mast's deflection:  $\Delta L = \text{strain} \times L$ ; where  $L$  = the mast's length. If the wind force is 25,069 N, use the data to estimate the deflection of a 9.14 m mast.

**Electrical Engineering**

20.18 Perform the same computations as in Sec. 20.3, but analyze data generated with  $f(t) = 5 \cos(7t) - 2 \sin(4t) + 6$ .

**20.19** You measure the voltage drop  $V$  across a resistor for a number of different values of current  $i$ . The results are

$i$	0.25	0.75	1.25	1.5	2.0
$V$	-0.45	-0.6	0.70	1.88	6.0

Use polynomial interpolation to estimate the voltage drop for  $i = 1.1$ . Interpret your results.

**20.20** Duplicate the computation for Prob. 20.19, but use polynomial regression to derive a cubic equation to fit the data. Plot and evaluate your results.

**20.21** The current in a wire is measured with great precision as a function of time:

$t$	0	0.1250	0.2500	0.3750	0.5000
$i$	0	6.2402	7.7880	4.8599	0.0000

Determine  $i$  at  $t = 0.22$ .

**20.22** The following data was taken from an experiment that measured the current in a wire for various imposed voltages:

$V, V$	2	3	4	5	7	10
$i, A$	5.2	7.8	10.7	13	19.3	27.5

On the basis of a linear regression of this data, determine current for a voltage of 6 V. Plot the line and the data and evaluate the fit. Determine whether it is a good assumption that the intercept is zero. If so, redo the regression and force the intercept to be zero.

**20.23** It is known that the voltage drop across an inductor follows Faraday's law:

$$V_L = L \frac{di}{dt}$$

where  $V_L$  is the voltage drop (in volts),  $L$  is inductance (in henrys;  $1 \text{ H} = 1 \text{ V}\cdot\text{s}/\text{A}$ ), and  $i$  is current (in amperes). Employ the following data to estimate  $L$ :

$di/dt, \text{ A/s}$	1	2	4	6	8	10
$V_L, V$	5	12	18	31	39	50

What is the meaning, if any, of the intercept of the regression equation derived from this data?

**20.24** Ohm's law states that the voltage drop  $V$  across an ideal resistor is linearly proportional to the current  $i$  flowing through the resistor as in  $V = iR$ , where  $R$  is the resistance. However, real resistors may not always obey Ohm's law. Suppose that you performed some very precise experiments to measure the voltage drop and corresponding current for a resistor. The results, as listed in Table P20.24, suggest a curvilinear relationship rather than the straight line represented by Ohm's law. In order to quantify this relationship, a curve must be fit to the data. Because of measurement

**TABLE P20.24** Experimental data for voltage drop across a resistor subjected to various levels of current.

$i$	-1.00	-0.50	-0.25	0.25	0.50	1.00
$V$	-193	-41	-13.5625	13.5625	41	193

error, regression would typically be the preferred method of curve fitting for analyzing such experimental data. However, the smoothness of the relationship, as well as the precision of the experimental methods, suggests that interpolation might be appropriate. Use a fifth-order interpolating polynomial to fit the data and compute  $V$  for  $i = 0.10$ .

**20.25** Repeat Prob. 20.24 but determine the coefficients of the fifth-order equation (Sec. 18.4) that fit the data in Table P20.24.

**20.26** An experiment is performed to determine the % elongation of electrical conducting material as a function of temperature. The resulting data is

Temperature, °C	200	250	300	375	425	475	525	600
% elongation	11	13	13	15	17	19	20	23

Predict the % elongation for a temperature of 400°F.

**20.27** Bessel functions often arise in advanced engineering analyses such as the study of electric fields. These functions are usually not amenable to straightforward evaluation and, therefore, are often compiled in standard mathematical tables. For example,

$x$	1.8	2.0	2.2	2.4	2.6
$J_0(x)$	0.3400	0.2239	0.1104	0.0025	0.0968

Estimate  $J_0(2.1)$ , (a) using an interpolating polynomial and (b) using cubic splines. Note that the true value is 0.1666.

**20.28** The population ( $p$ ) of a small community on the outskirts of a city grows rapidly over a 20-year period:

$t$	0	5	10	15	20
$p$	100	212	448	949	2009

As an engineer working for a utility company, you must forecast the population 5 years into the future in order to anticipate the demand for power. Employ an exponential model and linear regression to make this prediction.

### Mechanical/Aerospace Engineering

**20.29** Based on Table 20.4, use linear and quadratic interpolation to compute  $Q$  for  $D = 1.23 \text{ ft}$  and  $S = 0.01 \text{ ft/ft}$ . Compare your results with the same value computed with the formula derived in Sec. 20.4.

**20.30** Reproduce Sec. 20.4, but develop an equation to predict diameter as a function of slope and flow. Compare your results with the formula from Sec. 20.4 and discuss your results.

**20.31** Kinematic viscosity of water,  $\nu$ , is related to temperature in the following manner:

$T, ^\circ\text{C}$	0	4	8	12	16	20	24
$\nu, 10^{-2} \text{ cm}^2/\text{s}$	1.7923	1.5676	1.3874	1.2396	1.1168	1.0105	0.9186

Plot this data.

- (a) Use interpolation to predict  $\nu$  at  $T = 7.5 ^\circ\text{C}$ .
- (b) Use polynomial regression to fit a parabola to the data in order to make the same prediction.

**20.32** Hooke's law, which holds when a spring is not stretched too far, signifies that the extension of the spring and the applied force are linearly related. The proportionality is parameterized by the spring constant  $k$ . A value for this parameter can be established experimentally by placing known weights onto the spring and measuring the resulting compression. Such data is contained in Table P20.32 and plotted in Fig. P20.32. Notice that above a weight of  $40 \times 10^4 \text{ N}$ , the linear relationship between the force and displacement breaks down. This sort of behavior is typical of what is termed a "hardening spring." Employ linear regression to determine a value of  $k$  for the linear portion of this system. In addition, fit a nonlinear relationship to the nonlinear portion.

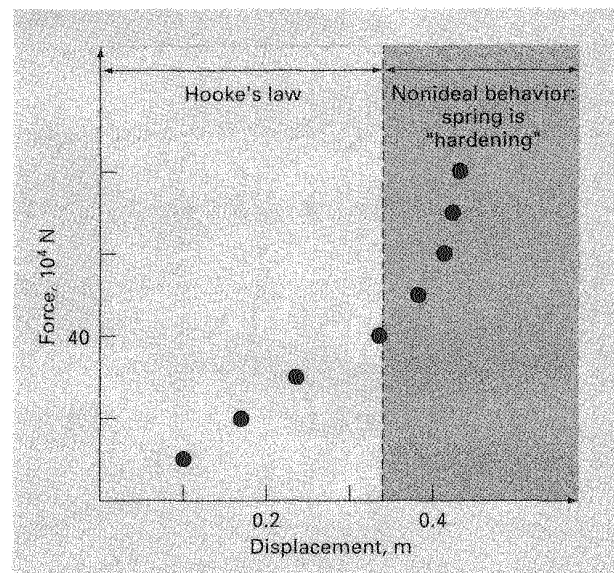
**20.33** Repeat Prob. 20.32 but fit a power curve to all the data in Table P20.32. Comment on your results.

**20.34** The distance required to stop an automobile is a function of its speed. The following experimental data was collected to quantify this relationship:

Speed, mi/h	15	20	25	30	40	50	60
Stopping distance, ft	16	20	34	40	60	90	120

Estimate the stopping distance for a car traveling at 45 mi/h.

**FIGURE P20.32** Plot of force (in  $10^4$  newtons) versus displacement (in meters) for the spring from the automobile suspension system.



**TABLE P20.32** Experimental values for elongation  $x$  and force  $F$  for the spring on an automobile suspension system.

Displacement, m	0.10	0.17	0.27	0.35	0.39	0.42	0.43	0.44
Force, $10^4 \text{ N}$	10	20	30	40	50	60	70	80

**20.35** An experiment is performed to define the relationship between applied stress and the time to fracture for a stainless steel. Eight different values of stress are applied, and the resulting data is

Applied stress, $x$ , kg/mm <sup>2</sup>	5	10	15	20	25	30	35	40
Fracture time, $y$ , h	40	30	25	40	18	20	22	15

- (a) Plot the data.  
 (b) Fit a straight line to the data with linear regression. Superimpose this line on your plot.

(c) Use the best-fit equation to predict the fracture time for an applied stress of 17 kg/mm<sup>2</sup>.

**20.36** The acceleration due to gravity at an altitude  $y$  above the surface of the earth is given by

$y$ , m	0	20,000	40,000	60,000	80,000
$g$ , m/s <sup>2</sup>	9.8100	9.7487	9.6879	9.6278	9.5682

Compute  $g$  at  $y = 55,000$  m.

# EPILOGUE: PART FIVE

## PT5.4 TRADE-OFFS

Table PT5.4 provides a summary of the trade-offs involved in curve fitting. The techniques are divided into two broad categories, depending on the uncertainty of the data. For imprecise measurements, regression is used to develop a “best” curve that fits the overall trend of the data without necessarily passing through any of the individual points. For precise measurements, interpolation is used to develop a curve that passes directly through each of the points.

All the regression methods are designed to fit functions that minimize the sum of the squares of the residuals between the data and the function. Such methods are termed least-squares regression. Linear least-squares regression is used for cases where a dependent and an independent variable are related to each other in a linear fashion. For situations where a dependent and an independent variable exhibit a curvilinear relationship, several options are available. In some cases, transformations can be used to linearize the relationship. In these instances, linear regression can be applied to the transformed variables to determine the best straight line. Alternatively, polynomial regression can be employed to fit a curve directly to the data.

**TABLE PT5.4** Comparison of the characteristics of alternative methods for curve fitting.

Method	Error Associated with Data	Match of Individual Data Points	Number of Points Matched Exactly	Programming Effort	Comments
<i>Regression</i>					
Linear regression	Large	Approximate	0	Easy	Round-off error becomes pronounced for higher-order versions
Polynomial regression	Large	Approximate	0	Moderate	
Multiple linear regression	Large	Approximate	0	Moderate	
Nonlinear regression	Large	Approximate	0	Difficult	
<i>Interpolation</i>					
Newton's divided-difference polynomials	Small	Exact	$n + 1$	Easy	Usually preferred for exploratory analyses
Lagrange polynomials	Small	Exact	$n + 1$	Easy	Usually preferred when order is known
Cubic splines	Small	Exact	Piecewise fit of data points	Moderate	First and second derivatives equal at knots

Multiple linear regression is utilized when a dependent variable is a linear function of two or more independent variables. Logarithmic transformations can also be applied to this type of regression for some cases where the multiple dependency is curvilinear.

Polynomial and multiple linear regression (note that simple linear regression is a member of both) belong to a more general class of linear least-squares models. They are classified in this way because they are linear with respect to their coefficients. These models are typically implemented using linear algebraic systems that are sometimes ill-conditioned. However, in many engineering applications (that is, for lower-order fits), this does not come into play. For cases where it is a problem, alternative approaches are available. For example, a technique called orthogonal polynomials is available to perform polynomial regression (see Sec. PT5.6).

Equations that are not linear with respect to their coefficients are called nonlinear. Special regression techniques are available to fit such equations. These are approximate methods that start with initial parameter estimates and then iteratively home in on values that minimize the sum of the squares.

Polynomial interpolation is designed to fit a unique  $n$ th-order polynomial that passes exactly through  $n + 1$  precise data points. This polynomial is presented in two alternative formats. Newton's divided-difference interpolating polynomial is ideally suited for those cases where the proper order of the polynomial is unknown. Newton's polynomial is appropriate for such situations because it is easily programmed in a format to compare results with different orders. In addition, an error estimate can be simply incorporated into the technique. Thus, you can compare and choose from results using several different-order polynomials.

The Lagrange interpolating polynomial is an alternative formulation that is appropriate when the order is known a priori. For these situations, the Lagrange version is somewhat simpler to program and does not require the computation and storage of finite divided differences.

Another approach to curve fitting is spline interpolation. This technique fits a low-order polynomial to each interval between data points. The fit is made smooth by setting the derivatives of adjacent polynomials to the same value at their connecting points. The cubic spline is the most common version. Splines are of great utility when fitting data that is generally smooth but exhibits local areas of abrupt change. Such data tends to induce wild oscillations in higher-order interpolating polynomials. Cubic splines are less prone to these oscillations because they are limited to third-order variations.

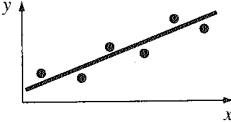
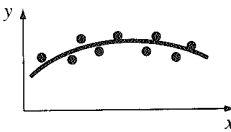
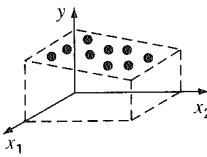
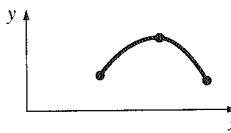
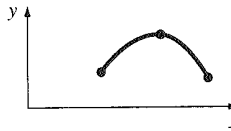
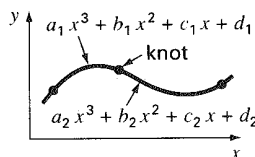
The final method covered in this part of the book is Fourier approximation. This area deals with using trigonometric functions to approximate waveforms. In contrast to the other techniques, the major emphasis of this approach is not to fit a curve to data points. Rather, the curve fit is employed to analyze the frequency characteristics of a signal. In particular, a fast Fourier transform is available to very efficiently transform a function from the time to the frequency domain to elucidate its underlying harmonic structure.

## **PT5.5 IMPORTANT RELATIONSHIPS AND FORMULAS**

---

Table PT5.5 summarizes important information that was presented in Part Five. This table can be consulted to quickly access important relationships and formulas.

**TABLE PT5.5** Summary of important information presented in Part Four.

Method	Formulation	Graphical Interpretation	Errors
Linear regression	$y = a_0 + a_1x$ where $a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$ $a_0 = \bar{y} - a_1 \bar{x}$		$s_{y/x} = \sqrt{\frac{S_r}{n-2}}$ $r^2 = \frac{S_t - S_r}{S_t}$
Polynomial regression	$y = a_0 + a_1x + \dots + a_mx^m$ (Evaluation of $a$ 's equivalent to solution of $m+1$ linear algebraic equations)		$s_{y/x} = \sqrt{\frac{S_r}{n-(m+1)}}$ $r^2 = \frac{S_t - S_r}{S_t}$
Multiple linear regression	$y = a_0 + a_1x_1 + \dots + a_mx_m$ (Evaluation of $a$ 's equivalent to solution of $m+1$ linear algebraic equations)		$s_{y/x} = \sqrt{\frac{S_r}{n-(m+1)}}$ $r^2 = \frac{S_t - S_r}{S_t}$
Newton's divided-difference interpolating polynomial*	$f_2(x) = b_0 + b_1(x - x_0) + b_2(x - x_0)(x - x_1)$ where $b_0 = f(x_0)$ $b_1 = f[x_1, x_0]$ $b_2 = f[x_2, x_1, x_0]$		$R_2 = (x - x_0)(x - x_1)(x - x_2) \frac{f^{(3)}(\xi)}{6}$ or $R_2 = (x - x_0)(x - x_1)(x - x_2)f[x_3, x_2, x_1, x_0]$
Lagrange interpolating polynomial*	$f_2(x) = f(x_0) \left( \frac{x - x_1}{x_0 - x_1} \right) \left( \frac{x - x_2}{x_0 - x_2} \right)$ $+ f(x_1) \left( \frac{x - x_0}{x_1 - x_0} \right) \left( \frac{x - x_2}{x_1 - x_2} \right)$ $+ f(x_2) \left( \frac{x - x_0}{x_2 - x_0} \right) \left( \frac{x - x_1}{x_2 - x_1} \right)$		$R_2 = (x - x_0)(x - x_1)(x - x_2) \frac{f^{(3)}(\xi)}{6}$ or $R_2 = (x - x_0)(x - x_1)(x - x_2)f[x_3, x_2, x_1, x_0]$
Cubic splines	A cubic: $a_1x^3 + b_1x^2 + c_1x + d_1$ is fit to each interval between knots. First and second derivatives are equal at each knot		

\*Note: For simplicity, second-order versions are shown.

## PT5.6 ADVANCED METHODS AND ADDITIONAL REFERENCES

---

Although polynomial regression with normal equations is adequate for many engineering applications, there are problem contexts where its sensitivity to round-off error can represent a serious limitation. An alternative approach based on *orthogonal polynomials* can mitigate these effects. It should be noted that this approach does not yield a best-fit equation, but rather, yields individual predictions for given values of the independent variable. Information on orthogonal polynomials can be found in Shampine and Allen (1973) and Guest (1961).

Whereas the orthogonal polynomial technique is helpful for developing a polynomial regression, it does not represent a solution to the instability problem for the general linear regression model [Eq. (17.23)]. An alternative approach based on *single-value decomposition*, called the SVD method, is available for this purpose. Forsythe et al. (1977), Lawson and Hanson (1974), and Press et al. (1992) contain information on this approach.

In addition to the Gauss-Newton algorithm, there are a number of optimization methods that can be used to directly develop a least-squares fit for a nonlinear equation. These nonlinear regression techniques include Marquardt's and the steepest-descent methods (recall Part Four). General information on regression can be found in Draper and Smith (1981).

All the methods in Part Five have been couched in terms of fitting a curve to data points. In addition, you may also desire to fit a curve to another curve. The primary motivation for such *functional approximation* is to represent a complicated function by a simpler version that is easier to manipulate. One way to do this is to use the complicated function to generate a table of values. Then the techniques discussed in this part of the book can be used to fit polynomials to these discrete values.

An alternative approach is based on the *minimax principle* (recall Fig. 17.2c). This principle specifies that the coefficients of the approximating polynomial be chosen so that the maximum discrepancy is as small as possible. Thus, although the approximation may not be as good as that given by the Taylor series at the base point, it is generally better across the entire range of the fit. *Chebyshev economization* is an example of an approach for functional approximation based on such a strategy (Ralston and Rabinowitz, 1978; Gerald and Wheatley, 1984; and Carnahan, Luther, and Wilkes, 1969).

An important area in curve fitting is the combining of splines with least-squares regression. Thus, a cubic spline is generated that does not intercept every point, but rather, minimizes the sum of the squares of the residuals between the data points and the spline curves. The approach involves using the so-called *B splines* as basis functions. These are so named because of their use as *basis* function, but also because of their characteristic bell shape. Such curves are consistent with a spline approach in that their value and their first and second derivatives would have continuity at their extremes. Thus, continuity of  $f(x)$  and its lower derivatives at the knots is ensured. Wold (1974), Prenter (1974), and Cheney and Kincaid (1994) present discussions of this approach.



---

In summary, the foregoing is intended to provide you with avenues for deeper exploration of the subject. Additionally, all the above references provide descriptions of the basic techniques covered in Part Five. We urge you to consult these alternative sources to broaden your understanding of numerical methods for curve fitting.