Chapter 4

Approximation Theory

4.1 Introduction to approximation of functions, regression vs. interpolation, some basic concept of the functions, continuous and discrete functions
4.2 Approximation of functions: Taylor's series, Least squares
4.3 Geometric interpretation of functions, norms of functions, orthogonal functions – Legendre and Tchebycheff polynomials
4.4 Approximation of data: Interpolating polynomials in Newton, Lagrange, and Gram forms.
4.5 Spline interpolation: Quadratic and cubic splines. Hermite interpolation.
4.6 Regression: Linear, nonlinear, and multiple regression
4.7 Periodic functions and Discrete Fourier Series
4.8 Summary

4.1 Introduction

In science and engineering, we frequently encounter problems where, based on a few measurements of a *dependent* variable (**function**) at corresponding values of the *independent* variable (generally distance or time), we need to estimate the value of the function corresponding to a different value of the independent variable. For example, in our *Batman* problem, suppose that the building has 200 floors and Joker notes down the time as the wheel passes the window of every 5^{th} floor till it hits the ground. From this data of *time* (dependent variable) versus *distance of the floor from the roof* (independent variable), we may be required to

- estimate the time at which Batman (at the 13^{th} floor) sees the wheel
- estimate the initial downward velocity, u, and the gravitational acceleration, g
- estimate the velocity and acceleration at different locations/times

On the other hand, if we were to measure the velocity at different times (by a radar gun, say), we could be asked to find the distance travelled till that time. There are various methods which could be used to perform these and other similar tasks. In this chapter, we discuss

- *interpolation*[1] (estimating the function value using a curve which passes through all data points), and
- *regression* (fitting a curve which represents the general trend of the function)

and, in the next chapter, we describe

- *numerical differentiation* (estimating the function derivatives, e.g., to obtain the velocity or acceleration from distance measurements), and

[1] We will not discuss *extrapolation*, i.e., estimating function value outside the range covered by the given data, since it may lead to large errors.

- *numerical integration* (estimating the integral of the function, e.g., to obtain distance from velocity measurements)

These problems may be thought of as *approximation* problems in which we approximate the actual function, *f(x)*, by an approximate function, $\tilde{f}(x)$ over a range of interest (a,b). The need for approximation arises because of various factors. We may not know the exact nature of dependence of the function on the independent variable, *or* the functional relationship may be so complex as to preclude operations like differentiation and integration, *or* the measured data may have errors. The methodology we adopt will be largely dependent on these factors. For example, if the data is exact, we would like our approximate function to pass through each data point (interpolation). On the other hand, if the data is affected by measurement errors, the approximating function *should not* pass through all the points and it would be sufficient for it to represent the general trend (regression) [1]. Since various concepts involved in the analysis are simpler for a function rather than tabular data, we first discuss the *continuous* case in which we need to approximate a function ***f(x)*** of a *single* independent variable ***x***. Extensions to the *discrete* case (in which the form of the function is not specified, only the function values are given corresponding to a few values of x) and *multiple* independent variables is conceptually similar and is described subsequently. The continuous case, of course, is a superset of the discrete case since the function values at selected points can be readily generated if the function is known.

4.2 Approximation of Functions

The form of the approximating function may be deduced form the knowledge of the function behaviour or from looking at the data. For example, if the function shows periodicity, it may be approximated by a combination of various sine and cosine functions[2]. Similarly, if we know from the physics of the problem that the function shows an exponential decay, we could use an exponential function as the approximate function. Most of the times, however, it would not be apparent from the problem/data as to what is the exact nature of the function. Because of their simplicity, easy differentiability and integrability, and due to the fact that most functions could be expanded in terms of polynomials using the Taylor's series, polynomials have been widely used as approximating functions. Moreover, a polynomial remains a polynomial of the same degree[3] even under a linear transformation of the dependent variable (e.g., x*=ax+b). Also, there is a theorem which implies that an approximating polynomial can be brought arbitrarily close to the actual function by increasing the degree of the polynomial (Weierstrass theorem, see Theorem 4.1). Therefore, we will focus our attention mostly on polynomial approximations.

Theorem 4.1: **Weierstrass Theorem**

[1] Sometimes, even when the data does not contain error, we may want to do regression to obtain an approximate function which is smoother than the actual function.
[2] See section 4.7, for further discussion of periodic functions.
[3] The terms *degree* and *order* of a polynomial are used interchangeably by us. Some books use the degree as the highest power of x in the polynomial and the order as one larger than the degree (in effect, indicating the number of terms in a general polynomial).

The Weierstrass approximation theorem states that "If *f* is a continuous real-valued function on a closed, bounded interval [*a, b*] and if $\varepsilon > 0$, there exists a polynomial *p* such that $|f(x) - p(x)| < \varepsilon$ for all $x \in [a,b]$, i.e., a continuous function on a bounded interval can be uniformly approximated by polynomial functions. Many different proofs of this theorem are available and we mention here the one based on Bernstein polynomials.

Since any closed interval [a,b] may be transformed to [0,1] by a linear substitution, we focus our attention on functions continuous over [0,1]. Since *x* varies over [0,1], it may be thought of as a probability of occurrence of some event, say, *E*. A binomial probability, π(i,n), representing the probability of *E* occurring exactly *i* times in *n* independent trials, is then written as

$$\pi(i,n) = \binom{n}{i} x^i (1-x)^{n-i}$$

in which the binomial coefficient $\binom{n}{i}$ is equal to $\frac{n!}{i!(n-i)!}$. Note that it is an nth degree polynomial in *x*, is nonnegative over [0,1], and, from elementary probability theory, $\sum_{i=0}^{n} \pi(i,n) = 1$

The Bernstein polynomials of order *n* of a function f(x) is then written as

$$B_n(x) = \sum_{i=0}^{n} \pi(i,n) f\left(\frac{i}{n}\right)$$

It is easy to see that if *f(x)* has an upper bound, *M*, in the interval [a,b], then $B_n(x)$ will also have the same upper bound.

To prove the Weierstrass theorem, we write

$$
\begin{aligned}
\left| f(x) - B_{n(x)} \right| &= \left| f(x) - \sum_{i=0}^{n} f\left(\frac{i}{n}\right)\pi(i,n) \right| \\
&= \left| f(x)\sum_{i=0}^{n}\pi(i,n) - \sum_{i=0}^{n} f\left(\frac{i}{n}\right)\pi(i,n) \right| \\
&= \left| \sum_{i=0}^{n}\left[f(x) - f\left(\frac{i}{n}\right)\right]\pi(i,n) \right| \\
&= \sum_{\left|\frac{i}{n}-x\right|<\delta} \left| f(x) - f\left(\frac{i}{n}\right)\right|\pi(i,n) + \sum_{\left|\frac{i}{n}-x\right|\geq\delta} \left| f(x) - f\left(\frac{i}{n}\right)\right|\pi(i,n) \\
&\leq \varepsilon + 2M \sum_{\left|\frac{i}{n}-x\right|\geq\delta} \pi(i,n)
\end{aligned}
$$

In the last two lines, we have split the summation into two parts, one where the points x and i/n are within an arbitrarily close distance, δ, and the other beyond it. Since the function is continuous, $\left| f(x) - f\left(\frac{i}{n}\right)\right| < \varepsilon$ for $\left|\frac{i}{n} - x\right| < \delta$, and the first term would be less than or equal to ε. Also, since the function has an upper bound of M, the upper bound of $\left| f(x) - f\left(\frac{i}{n}\right)\right|$ will be $2M$. To evaluate $\sum_{\left|\frac{i}{n}-x\right|\geq\delta} \pi(i,n)$, we consider the summation

$$
S = \sum_{i=0}^{n}\left(\frac{i}{n} - x\right)^2 \pi(i,n) = \frac{1}{n^2}\sum_{i=1}^{n} i^2\pi(i,n) - \frac{2x}{n}\sum_{i=1}^{n} i\pi(i,n) + x^2\sum_{i=0}^{n}\pi(i,n)
$$

Note that in the first two terms on the r.h.s., the lower limit of the summation index is changed to 1 since the term corresponding to i=0 will vanish. Also the summation in the third term would be unity. For the middle term, we write

$$
\begin{aligned}
\sum_{i=1}^{n} i\pi(i,n) &= \sum_{i=1}^{n} i\frac{n!}{i!(n-i)!}x^i(1-x)^{n-i} \\
&= \sum_{i=1}^{n} nx\frac{(n-1)!}{(i-1)!(n-i)!}x^{i-1}(1-x)^{n-i} \\
&= nx\sum_{j=0}^{m}\frac{m!}{(j)!(m-j)!}x^j(1-x)^{m-j} \qquad (\text{with } j \equiv \text{i} - 1, m \equiv n-1) \\
&= nx\sum_{j=0}^{m}\pi(j,m) = nx
\end{aligned}
$$

Similarly, it can be shown that

$$\sum_{i=1}^{n} i^2 \pi(i,n) \quad = nx(nx - x + 1)$$

The summation, S, is then obtained as

$$S = \frac{1}{n^2} nx(nx - x + 1) - \frac{2x}{n} nx + x^2 = \frac{x - x^2}{n}$$

We, therefore, have (noting that the maximum value of x–x^2 in the interval [0,1] is ¼)

$$\sum_{\left|\frac{i}{n} - x\right| \geq \delta} \pi(i,n) \leq \sum_{\left|\frac{i}{n} - x\right| \geq \delta} \frac{\left(\frac{i}{n} - x\right)^2}{\delta^2} \pi(i,n) = \frac{S}{\delta^2} = \frac{x(1-x)}{n\delta^2} \leq \frac{1}{4n\delta^2}$$

So, finally, we get

$$\left| f(x) - B_{n(x)} \right| \quad \leq \varepsilon + \frac{M}{2n\delta^2}$$

which may be made arbitrarily small by taking a sufficiently large *n*, thus proving the Weierstrass theorem. Other proofs are also available using, for example, the Fourier series.

Once we choose the approximating function to be a polynomial, we have to decide what should be the degree of this polynomial. The answer clearly depends on how close to the actual function we want the approximation to be and how much computational effort are we willing to spend. In general a higher order polynomial would be closer to the function but will need more computational effort for evaluating the coefficients (for discrete case, however, a higher order polynomial may result in a worse fit, see Fig. 4.?, Runge phenomenon). We must also decide on how to quantify the nearness of the actual function, f(x), and the approximating polynomial of degree m, $f_m(x)$. For example, we may use the maximum difference between f(x) and $f_m(x)$ over the interval (a, b) as a measure of the error of the approximation. Or we may use the integral of the absolute value or the square of $[f(x)-f_m(x)]$. Finally, we should device an algorithm which would give us the m^{th}-degree polynomial *nearest* to f(x). We discuss these issues first from an analytical perspective using the *Taylor's series* and the *method of least squares* and then from a geometrical perspective in the next section.

Taylor's Series

Probably the simplest way of approximating a function by a polynomial is by using the Taylor's series expansion about a point:

$$f(a+h) = f(a) + hf'(a) + \frac{h^2}{2!} f''(a) + \ldots + \frac{h^i}{i!} f^{(i)}(a) + \ldots \qquad (4.1)$$

and truncating it after the desired number of terms. If the function and its first (m+1) derivatives are continuous over the interval (a, a+h), Taylor's theorem (Theorem 4.2) states that

$$f(a+h) = f(a) + hf'(a) + \frac{h^2}{2!} f''(a) + \ldots + \frac{h^m}{m!} f^{(m)}(a) + R_m \qquad (4.2a)$$

in which the remainder, R_m, is given by

$$R_m = \int_a^{a+h} \frac{(a+h-x)^m}{m!} f^{(m+1)}(x)\,dx = \frac{h^{m+1}}{(m+1)!} f^{(m+1)}(\xi) \qquad (4.2b)$$

where $\xi \in (a, a+h)$. Thus, to approximate a function, f(x), over the interval (a, b), by a m^{th} degree polynomial, $f_m(x)$, we write

$$f_m(x) = f(x_0) + (x-x_0) f'(x_0) + \frac{(x-x_0)^2}{2!} f''(x_0) + \ldots + \frac{(x-x_0)^m}{m!} f^{(m)}(x_0) \qquad (4.3)$$

where x_0 is a *judiciously chosen* point (in absence of any other information, the most logical choice would be the midpoint of the interval. However, suppose we know that most of the times the approximation will be used to predict the function values closer to *a* rather than *b*, we may choose x_0 nearer to *a*.) Not only does Eq. (4.3) gives us an approximation, it also provides us with an estimate of the error [Eq. (4.2b)]. However, since the point ξ is not known, the error estimate is generally not very useful. In most cases, though, the upper bound of the $(m+1)^{th}$ derivative of the function over (a,b) may be obtained and will provide an upper bound for the error (the actual error is generally much smaller!). In some cases, e.g., when f(x) is a $(m+1)^{th}$ degree polynomial, $f^{(m+1)}(x)$ may be constant or nearly constant over (a,b) and the remainder may be readily computed. For example, to approximate the function $f(x)=a_0+a_1x+a_2x^2$ by a first-degree polynomial $f_1(x)=c_0+c_1x$ over the range (a,b) (*not a very realistic example, since we would hardly ever want to approximate a polynomial by another! The reason for choosing this example would be clear when we discuss* geometrical interpretation of a function *latter in this chapter*.), we may write

$$f_1(x) = f\left(\frac{a+b}{2}\right) + \left(x - \frac{a+b}{2}\right) f'\left(\frac{a+b}{2}\right) \qquad (4.4)$$

resulting in $c_0 = a_0 - a_2 \frac{(a+b)^2}{4}$ and $c_1 = a_1 + a_2(a+b)$. The error at any point, $f(x)-f_1(x)$, is given by the remainder term, $R_1 = a_2\left(x - \frac{a+b}{2}\right)^2$.

Example 4.1: The function $f(x) = 1 + x + x^2$ has to be approximated by a linear function over the interval (0,1). Find the approximating function by using Taylor's series expansion about the points 0, 0.5, and 1, respectively.

Solution: Using Eq. (4.3), the approximating linear function is written as

$$f_1(x) = f(x_0) + (x-x_0) f'(x_0) = \begin{cases} 1+x & x_0 = 0 \\ 1.75 + 2(x-0.5) & x_0 = 0.5 \\ 3 + 3(x-1) & x_0 = 1 \end{cases}$$

Figure 4.1 shows these three approximations along with another line (D).

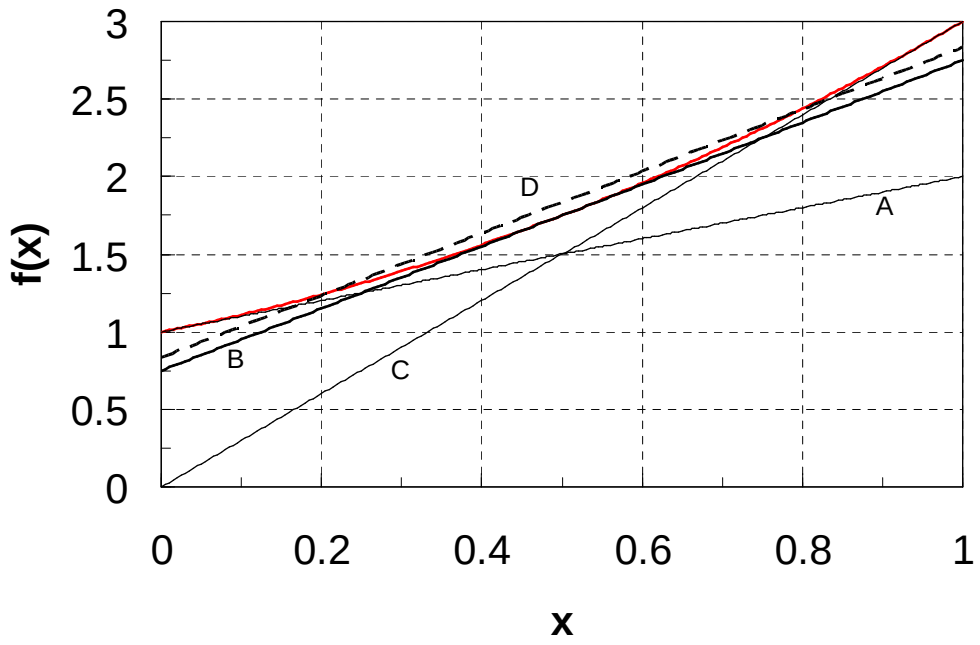


Figure 4.1 The function $f(x) = 1 + x + x^2$ and its straight line approximation over the domain (0,1). The function is shown in red and the Taylor's series approximations about points 0 (line A), 0.5 (line B), and 1.0 (line C) are also drawn. Line D is parallel to line B.

From this figure, the following observations are made:

- Taylor's series about the mid-point is much better than that about the end-points
- The error grows as we move away from the mid-point (see line B and also the expression for the remainder)
- The fit does not appear to be the *best* fit since a parallel line (D) is much *closer* to the function. This line, however, does not represent a Taylor's series expansion of f(x).

Thus, in spite of its simplicity, Taylor's series is hardly ever used for approximating a function. However, it is very useful in performing an error analysis of other approximating schemes, as we will see on numerous occasions in this and subsequent chapters.

Theorem 4.2: **Taylor's Theorem**

Writing

$$f(a+h) = f(a) + \int_a^{a+h} f'(x)dx$$

and integrating by parts, we obtain

$$
\begin{aligned}
f(a+h) &= f(a)+\left[xf'(x)\right]_a^{a+h}-\int_a^{a+h} xf''(x)dx \\
&= f(a)+(a+h)f'(a+h)-af'(a)-\int_a^{a+h} xf''(x)dx \\
&= f(a)+(a+h)\left[\int_a^{a+h} f''(x)dx+f'(a)\right]-af'(a)-\int_a^{a+h} xf''(x)dx \\
&= f(a)+hf'(a)+\int_a^{a+h}(a+h-x)f''(x)dx
\end{aligned}
$$

A similar procedure leads to

$$f(a+h)=f(a)+hf'(a)+\frac{h^2}{2}f''(a)+\frac{1}{2}\int_a^{a+h}(a+h-x)^2 f'''(x)dx$$

By repeating the process (or, formally, by induction) we obtain the Taylor's theorem

$$f(a+h)=f(a)+hf'(a)+\frac{h^2}{2!}f''(a)+\ldots+\frac{h^m}{m!}f^{(m)}(a)+R_m$$

in which the remainder, R_m, is given by $R_m=\int_a^{a+h}\frac{(a+h-x)^m}{m!}f^{(m+1)}(x)\,dx$. Using the mean value theorem for integration, which states that

If *f(x)* and *g(x)* are continuous and integrable on *[a, b]*and *g(x)* has the same sign everywhere in (a,b), then there exists a number $c\in[a,b]$ such that $\int_a^b f(x)g(x)dx=f(c)\int_a^b g(x)dx$

the remainder may be written as $R_m=\frac{h^{m+1}}{(m+1)!}f^{(m+1)}(\xi)$ in which, $\xi\in(a,a+h)$.

Method of Least Squares

In this method, we choose the approximating polynomial in such a way as to give us the least value of the square of the deviation [f(x)-f_m(x)] integrated over the relevant domain (a,b). We may write the approximating polynomial in the commonly used form, $f_m(x)=\sum_{j=0}^{m}c_j x^j$.

However, there are alternative ways of writing a polynomial which may be more suitable for

some problems. For example, the form $f_m(x)=\sum_{j=0}^{m} c_j\left(x-\frac{a+b}{2}\right)^j$ will have lower round-off errors. Other possibilities include $f_m(x)=\sum_{j=0}^{m} c_j p_j$, where p_j denotes a j^{th} degree polynomial and $f_m(x)=\sum_{j=0}^{m} c_j p_{m,j}$, where $p_{m,j}$ are m^{th} degree polynomials. We use a general notation

$$f_m(x)=\sum_{j=0}^{m} c_j\phi_j(x) \tag{4.5}$$

to represent the approximating polynomial in which the ϕ's are the (known) polynomials and the c's are the (unknown) coefficients. The problem then reduces to finding the c's which minimise $\int_a^b\left(f(x)-\sum_{j=0}^{m} c_i\phi_i(x)\right)^2 dx$. This can be done by invoking the stationary points theorem[1] to obtain a set of m+1 linear simultaneous equations of the form

$$[A]\{c\}=\{b\} \tag{4.6}$$

in which $a_{ij}=\int_a^b \phi_i(x)\phi_j(x)dx$ and $b_i=\int_a^b \phi_i(x)f(x)dx$. Eqs (4.6) are known as the *normal equations* and may be solved using any of the methods discussed in Chapter 2 to obtain the coefficients $c_0,c_1\ldots c_m$. For example, to approximate the function $f(x)=a_0+a_1x+a_2x^2$ by a first-degree polynomial $f_1(x)=c_0+c_1x$ over the range (a,b), we obtain

$$\int_a^b 1.1dx\,c_0+\int_a^b 1.xdx\,c_1=\int_a^b 1.\left(a_0+a_1x+a_2x^2\right)dx$$

$$\int_a^b x.1dx\,c_0+\int_a^b x.xdx\,c_1=\int_a^b x.\left(a_0+a_1x+a_2x^2\right)dx$$

that is,

$$\begin{aligned}(b-a)c_0+\frac{b^2-a^2}{2}c_1&=(b-a)a_0+\frac{b^2-a^2}{2}a_1+\frac{b^3-a^3}{3}a_2\\ \frac{b^2-a^2}{2}c_0+\frac{b^3-a^3}{3}c_1&=\frac{b^2-a^2}{2}a_0+\frac{b^3-a^3}{3}a_1+\frac{b^4-a^4}{4}a_2\end{aligned} \tag{4.7}$$

from which $c_0=a_0-\frac{b^2+a^2+4ab}{6}a_2$ and $c_1=a_1+(b+a)a_2$.

Example 4.2: The function $f(x)=1+x+x^2$ has to be approximated by a linear function over the interval (0,1). Find the approximating function by using the Least Squares method.

Solution: Using the basis functions as $\phi_0(x)=1$ and $\phi_1(x)=x$, and with a=0 and b=1, Eq. (4.7) is written as

[1] A continuous function of *n* variables attains an extremum only at points at which either all the *n* partial derivatives are zero (stationary points) or one or more of these derivatives do not exist.

$$c_0 + 0.5c_1 = 1 + \frac{1}{2} + \frac{1}{3} = \frac{11}{6}$$
$$\frac{1}{2}c_0 + \frac{1}{3}c_1 = \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = \frac{13}{12}$$

from which, c_0=5/6 and c_1=2. The *best* approximating linear function is thus written as $f_1(x)$=5/6+2x (Line D in Figure 4.1 shows this approximation).

This method, however, becomes quite cumbersome and the normal equations become ill-conditioned as the degree of polynomial increases. In the next section, we give a geometric interpretation of the problem of function approximation which leads to development of simpler techniques.

Exercise 4.2

1. Approximate the function exp(x) over (−1,1) by a straight line using (a) Taylor's series about x=0 and (b) the method of least squares. Plot the residual for both the methods. Using these approximations, estimate the value of exp(x) at x=0, 0.62, and 1. Compare with the exact values and comment.
2. Compare the residuals in the Taylor's series approximation at all three points in problem 1 with the analytical expression of the remainder [Eq. (4.2b)] and verify that ξ does lie between a and $a+h$.
3. Compute the L_2 norm of the residual, i.e., $\int_{-1}^{1}\left[f(x) - f_1(x)\right]^2 dx$ for both the approximations in problem 1. In order to reduce this norm, it is desired to approximate the function by a second degree polynomial. Obtain this polynomial and the associated L_2 norm. Is it possible that the L_2 norm of the residual is larger for a higher degree polynomial approximation?
4. Approximate the function exp(2x−1) over (0,1) by a straight line using the method of least squares. Compare the approximating function with that obtained in problem 1 for the domain (−1,1) and verify that these are identical (with a linear transformation of variable).
5. The function x^x has to be integrated over the interval (0,1). Since analytical integration is not possible, it is decided to approximate the function by a 3rd degree polynomial and then perform an analytical integration of the approximating polynomial. What problem is encountered in the method of least squares? Would the Taylor's series work? If yes, estimate the value of the integral $\int_0^1 x^x dx$. [Note: We will look at better methods of estimating this integral in the next chapter.]

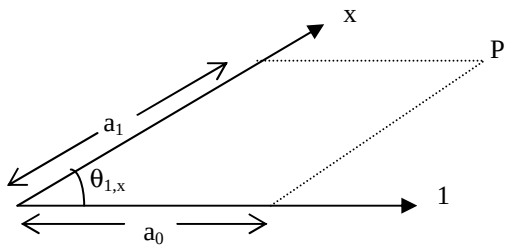
4.3 Geometric interpretation of Functions and orthogonal polynomials

We are quite familiar with the three-dimensional Euclidean vector space in which a point is represented by its distance from an origin along three orthogonal directions (e.g., East-West, North-South, Up-Down). Sometimes a fourth dimension of time is also added in the description. Although it is difficult to visualize, higher dimensional spaces may, and probably

do, exist[1]. Along similar lines, we may think of an *n*-dimensional space (Box 4.1) whose axes are not directions but functions and then define products of functions, angle between functions and similar other properties.

Box 4.1: **Geometric interpretation of Functions**

We may think of a space whose axes are not directions but functions For example, the function a_0+a_1x could be represented by a point P in a 2-dimensional plane having axes *1* and *x*, as shown in the figure below:



The axes need not be orthogonal but, as we will see a little later, the computations are much simpler if they are. Similar to a vector space, we may have an (n+1)-dimensional space to represent an n^{th} degree polynomial with its axes as 1, x, x^2,…,x^n. In fact we may choose the axes as 1, 1+x, 1+x+x^2,…,1+x+…+x^n or any other form as long as all n^{th} degree polynomials can be represented by a linear combination of these axes. (Other functions would belong to different function spaces, e.g., periodic functions are generally represented by a space with its axes as $1, \sin ax, \cos ax, \sin 2ax, \cos 2ax....$). Once this space is defined, any function can be represented by a *point* in the corresponding space. If f(x) is a polynomial of degree n it would be a point in the (n+1)-dimensional function space. If f(x) is not a polynomial, it may be represented in an infinite-dimensional polynomial space using the Taylor's series.

Similar to the *dot product* (scalar product) of two vectors, an *inner product* of two functions, f(x) and g(x) in the function space is defined as (sometimes a weight, *w(x)*, is also used in this definition, but we will introduce it later in this chapter)

$$\langle f, g \rangle = \int_a^b f(x)g(x)dx \qquad \text{(B4.1.1)}$$

where (a,b) is the domain over which these functions are defined. Analogous to the vector space, the *magnitude* (norm) of a function is given as

[1] String theory in physics, which considers the basic building blocks to be strings rather than particles, predicts the dimensionality of the Universe to be 10 (superstring theory), 11 (M-theory) or 26 (bosonic string theory)!

$$\|f\| = \sqrt{\langle f, f \rangle}$$

and the *angle* between any two functions is given by

$$\theta_{f,g} = \arccos \frac{\langle f, g \rangle}{\|f\| \|g\|}$$

Using these definitions, the angle between the 1-axis and x-axis in the figure above, $\theta_{1,x}$, is given by

$$\cos\theta_{1,x} = \frac{\left(b^2 - a^2\right)/2}{\sqrt{b-a}\sqrt{\left(b^3 - a^3\right)/3}} = \frac{\sqrt{3}}{2} \frac{b+a}{\sqrt{b^2 + a^2 + ba}}$$

Thus if the function is defined over the domain (0,1), the angle between the two axes would be 30°, and for the domain (−1,1), the axes would be orthogonal (and the inner product will be zero).

As described in Box 4.1, f(x) is represented as a point in space whose location will depend on the nature of the function. And if we decide to approximate it by a polynomial of a lower degree, $f_m(x)$, the approximating polynomial would be another point in the (m+1)-dimensional space (if f(x) is a polynomial of degree n and m is equal to or greater than n, the two points would be same). This gives us a convenient way of defining the nearness of the two functions in terms of the *distance* between these two points. Let us illustrate this with an example.

Suppose we want to approximate the function $f(x)=a_0+a_1x$ by a constant value $f_0(x)=c_0$ over the interval (a,b). Figure 4.2 shows the point f(x) in the two-dimensional function space with 1 and x as the coordinate axes (As shown in Box 4.1, the angle between the two axes depends on *a* and *b*). The *best* approximating constant function may be taken as the point on the 1-axis which is at a minimum distance from f(x). For this simple problem, it is readily obtained by drawing a perpendicular from f(x) on the 1-axis.

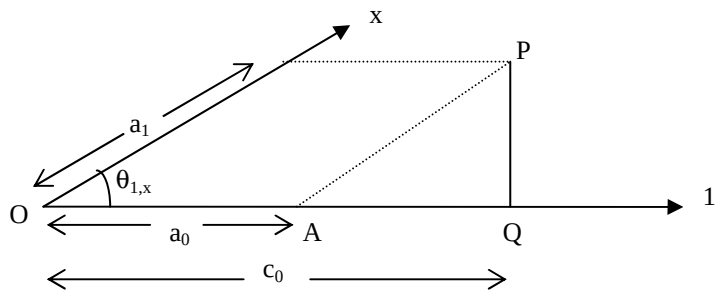


Figure 4.2 Approximating a linear function by a constant

Similarly, Fig. 4.3 shows how to approximate the function $f(x)=a_0+a_1x+a_2x^2$ over the same range by a linear function $f_1(x)=c_0+c_1x$. In this case, f(x) is shown as a point (P) in the three-dimensional $(1,x,x^2)$ space and the best approximation is obtained as the base of the perpendicular (Q) from f(x) on the (1,x) plane.

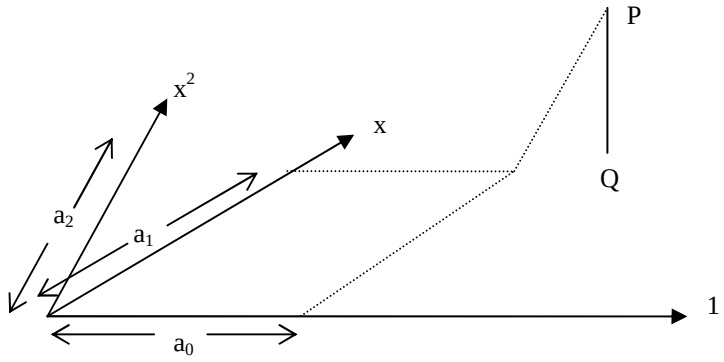


Figure 4.3 Approximating a quadratic function by a linear function

Extending this argument, we could say that for a function f(x), the best approximating polynomial of degree m would be the projection of the function f(x) on the (m+1)-dimensional space. The only thing which is left to do now is to devise a method to obtain the location of this point, i.e., to choose the coefficients $c_0, c_1,\ldots c_m$. We will do it by extending the concepts of vector space to the space representing the function.

As shown in Fig. 4.2, to obtain the best 0^{th} order polynomial to approximate the function $f(x)=a_0+a_1x$ over the range (a,b), we need to find c_0 (i.e., OQ) such that PQ is perpendicular to the 1-axis. There are several ways in which it could be done. For example, from triangle APQ, we have

$$\|AQ\|\left(=\sqrt{\langle AQ,AQ\rangle}=AQ\sqrt{b-a}\right)=\|AP\|\cos\theta_{1,x}=\sqrt{\langle a_1x,a_1x\rangle}\cos\theta_{1,x}=a_1\frac{\left(b^2-a^2\right)\Big/2}{\sqrt{b-a}}$$

resulting in $c_0=a_0+a_1(b+a)/2$. Another method would be to obtain the projection of OP on the 1-axis as

$$\|OQ\|\left(=c_0\sqrt{b-a}\right)=\frac{\langle OP,1\rangle}{\|1\|}=\frac{\int_a^b\left(a_0+a_1x\right)dx}{\sqrt{\int_a^b 1dx}}=\frac{a_0(b-a)+a_1\dfrac{b^2-a^2}{2}}{\sqrt{b-a}}$$

which, as it should, gives the same value of c_0. While these methods are good for illustration, their extension to higher dimensions is cumbersome. Fig. 4.3 depicts the problem of approximating a 3-dimensional function, $f(x)=a_0+a_1x+a_2x^2$, by a 2-dimesnional function

$f_1(x)=c_0+c_1x$. Finding OQ using geometry or inner products with the coordinate axes is possible but becomes quite involved. A much simpler technique results from the observation that the *error* term, $[f_m(x)-f(x)]$, i.e., PQ, should be orthogonal to the (m+1)-dimensional plane containing $f_m(x)$ in order for it to be minimum. This implies that PQ should be orthogonal to *all* the coordinate axes of $f_m(x)$. Thus, the problem of finding the best straight line approximation for a quadratic polynomial reduces to the following equations:

$$\langle f_1(x) - f(x), 1 \rangle = 0$$
$$\langle f_1(x) - f(x), x \rangle = 0$$

giving rise to the same two linear simultaneous equations in c_0 and c_1 as obtained using the method of least squares (Eq. 4.7).

The ideas described in the preceding paragraphs could now be formalised for the general case as follows. Let f(x) be a function which needs to be approximated over the interval x=a to x=b by an m^{th} degree polynomial $f_m(x)$. Let the coordinate axes, or the *basis*, of the (m+1)-dimensional space representing the approximating polynomial be represented by ϕ_j, j=0,m. The approximating polynomial may then be written as

$$f_m(x) = \sum_{j=0}^{m} c_j \phi_j(x) \tag{4.8}$$

where c_j are coefficients which need to be determined. Using the orthogonality conditions

$$\left\langle \sum_{j=0}^{m} c_j \phi_j(x) - f(x), \phi_i(x) \right\rangle = 0 \text{ for } i = 0,1,...,m \tag{4.9}$$

we get the same set of m+1 linear simultaneous equations as obtained earlier (Eq. 4.7)

$$[A]\{c\} = \{b\} \tag{4.10}$$

in which $a_{ij} = \langle \phi_i, \phi_j \rangle$ and $b_i = \langle \phi_i, f \rangle$ (for convenience we have dropped the x showing dependence of f and ϕ on x). Eqs (4.10) may be solved using any of the methods discussed in Chapter 2 to obtain the coefficients $c_0, c_1 \ldots c_m$. However, the computations become extremely simple if we choose an *orthogonal basis*[1]. In that case, all the non-diagonal elements of the matrix A become zero and the c's are obtained directly as $c_i = \dfrac{\langle \phi_i, f \rangle}{\langle \phi_i, \phi_i \rangle}$. And if we choose an *orthonormal basis*, we have $c_i = \langle \phi_i, f \rangle$. This provides us with a motivation to look closely at orthogonal functions.

From Box 4.1, we see that the functions 1 and x would be orthogonal if the domain is (–a,a). The most common choice is (–1,1) and if a function f(x*) has a finite[2] domain (a,b), we may

---

[1] An orthogonal set of basis functions is one for which $\langle \phi_i, \phi_j \rangle = 0$ if $i \neq j$. If, in addition, all basis functions have unit norms, i.e., $\langle \phi_i, \phi_i \rangle = 1 \; \forall \; i$, we get an orthonormal set of basis functions.

[2] For infinite domain of the form (a,∞) with a>0, a *nonlinear* transformation of the form $x = 1 - 2a/x^*$ may be used. Another possibility would be to use $x = 1 - 2\exp(x^* - a)$. However, we will not discuss these cases.

use the transformation $x = \frac{2x^* - b - a}{b - a}$ to obtain a function f(x) over (−1,1). Therefore most of the subsequent discussion is based on the assumption that the function domain is (−1,1). The basis functions $\phi_0 = 1$ and $\phi_1 = x$ are thus orthogonal, i.e., $\int_{-1}^{1} 1 \cdot x dx = 0$. If we choose $\phi_2 = x^2$, we find that ϕ_1 and ϕ_2 are orthogonal, since $\int_{-1}^{1} x \cdot x^2 dx = 0$, but ϕ_0 and ϕ_2 are not, since $\int_{-1}^{1} 1 \cdot x^2 dx \neq 0$. One way of choosing ϕ_2 is to assume it to be of the form $\alpha_0 + \alpha_1 x + \alpha_2 x^2$, and using the orthogonality with ϕ_0 and ϕ_1 to show that $\alpha_2 = -3\alpha_0$ and $\alpha_1 = 0$ such that $\phi_2 = \alpha_0(1 - 3x^2)$ would form an orthogonal basis. However, the Gram-Schmidt process (see Theorem 2.9 and Box 4.2) provides us a convenient algorithm for generating the orthogonal polynomials. For the 2nd degree polynomial, $\phi_0 = 1, \phi_1 = x - \frac{\langle x,1 \rangle}{\langle 1,1 \rangle} 1 = x$ and $\phi_2 = x^2 - \frac{\langle x^2,1 \rangle}{\langle 1,1 \rangle} 1 - \frac{\langle x^2,x \rangle}{\langle x,x \rangle} x = x^2 - \frac{1}{3}$. Note that this ϕ_2 is the same as obtained earlier with $\alpha_0 = -1/3$. Since any arbitrary constant could be used, commonly it is chosen in such a way as to make the ϕ value at x=1 equal to 1. This set of orthogonal polynomials is known as the *Legendre polynomials*, and is described in the next subsection.

Box 4.2: **Gram-Schmidt process for generating orthogonal polynomials**

Given a set of linearly independent vectors $v_0, v_1, v_2, \ldots, v_m$, a set of orthogonal vectors, O, generating the same subspace may be obtained by

$$O_0 = v_0; O_1 = v_1 - \frac{\langle v_1, O_0 \rangle}{\langle O_0, O_0 \rangle} O_0; O_2 = v_2 - \frac{\langle v_2, O_0 \rangle}{\langle O_0, O_0 \rangle} O_0 - \frac{\langle v_2, O_1 \rangle}{\langle O_1, O_1 \rangle} O_1$$

and

$$O_m = v_m - \frac{\langle v_m, O_0 \rangle}{\langle O_0, O_0 \rangle} O_0 - \frac{\langle v_m, O_1 \rangle}{\langle O_1, O_1 \rangle} O_1 - \ldots - \frac{\langle v_m, O_{m-1} \rangle}{\langle O_1, O_{m-1} \rangle} O_{m-1}$$

Legendre Polynomials

During the solution of the Laplace's equation in spherical coordinates for boundary value problems, the following differential equation is obtained

$$\frac{d}{dx}\left[\left(1-x^2\right)\frac{dp}{dx}\right]+n\left(n+1\right)p=0$$

in which *x* represents the cosine of the polar angle, and therefore has the domain (–1,1), and *p* represents the part of the potential which depends on the polar angle. This equation is known as the Legendre's equation and arises in many other applications also. For a physically realistic solution, it can be shown that *n* should be a nonnegative integer and a solution of this equation is given by an n^{th} degree polynomial. These polynomials, generally multiplied by a constant to make them equal to 1 at x=1, are called the Legendre polynomials and are denoted by $P_n(x)$. They may be obtained by using the Rodrigues' formula

$$P_n(x)=\frac{1}{2^n n!}\frac{d^n}{dx^n}\left[\left(x^2-1\right)^n\right]$$

or using Bonnet's recursive relation[1]

$$P_n(x)=\frac{2n-1}{n}xP_{n-1}(x)-\frac{n-1}{n}P_{n-2}(x) \qquad n=2,3,.... \qquad \text{with } P_0(x)=1, P_1(x)=x \quad (4.11)$$

and have the orthogonality property

$$\left\langle P_i(x),P_j(x)\right\rangle=\int_{-1}^{1}P_i(x)P_j(x)dx=\begin{cases}0 & i\neq j\\ \dfrac{2}{2i+1} & i=j\end{cases} \quad (4.12)$$

We may, of course, make these orthonormal by multiplying each polynomial by $\sqrt{n+\frac{1}{2}}$, but it does not make much of a difference as far as the computations are concerned. It is easy to see that $P_n(x)$ will be orthogonal to *all* polynomials (not only Legendre polynomials) of degree equal to or less than n–1. The first four polynomials are listed below (from the Rodrigues' formula it is seen that the coefficient of the leading term, x^n, in $P_n(x)$ will be equal to[2] $\frac{(2n)!}{2^n\left(n!\right)^2}$):

$$P_0(x)=1 \qquad P_1(x)=x \qquad P_2(x)=\frac{1}{2}\left(3x^2-1\right) \qquad P_3(x)=\frac{1}{2}\left(5x^3-3x\right)$$

Now, let us revisit the problem of approximating the function, $f(x^*)=a_0+a_1x^*+a_2x^{*2}$, by a straight line $f_1(x^*)=c_0+c_1x^*$ over the domain (a,b). We first change the domain[3] to (–1,1) by using the transformation $x=\frac{2x^*-b-a}{b-a}$ which results in

---

[1] Similarly, we have $\left(1-x^2\right)P_n'(x)=-nxP_n(x)+nP_{n-1}(x)$

[2] In the chapter on numerical integration, we will see that it is convenient to express the orthogonal polynomials as monomials. So we could re-define the Legendre polynomials by dividing them with the coefficient of the leading term. However, we would use the generally accepted definitions.

[3] It may be easily seen that if the function domain remains (a,b), the polynomial orthogonal basis functions could be written as $1,\ x-\frac{a+b}{2},\frac{3}{2}x^2-\frac{3}{2}(a+b)x+\frac{a^2+4ab+b^2}{4},...$ which reduces to the Legendre polynomials for (–1,1). However, it is more convenient to use the standard domain (–1,1).

$$f(x) = a_0 + a_1\frac{b+a}{2} + a_2\left(\frac{b+a}{2}\right)^2 + \left(a_1\frac{b-a}{2} + a_2\frac{b^2-a^2}{2}\right)x + a_2\left(\frac{b-a}{2}\right)^2 x^2$$

Then we write $f_1(x)=\alpha_0 P_0(x)+\alpha_1 P_1(x)$ and obtain the coefficients as

$$\alpha_0 = \frac{\langle P_0(x),\, f(x)\rangle}{\langle P_0(x), P_0(x)\rangle} = a_0 + a_1\frac{b+a}{2} + a_2\frac{(b+a)^2}{6}$$

$$\alpha_1 = \frac{\langle P_1(x),\, f(x)\rangle}{\langle P_1(x), P_1(x)\rangle} = a_1\frac{b-a}{2} + a_2\frac{b^2-a^2}{2}$$

and, finally, on transforming back to the variable x*

$$c_0 = a_0 - \frac{b^2+a^2+4ab}{6}a_2 \text{ and } c_1 = a_1 + (b+a)a_2$$

which are same as those obtained earlier.

Example 4.3: The function $f(x) = 1 + x + x^2$ has to be approximated by a linear function over the interval (0,1). Find the approximating function by using the Legendre polynomials.

Solution: Since the interval is (0,1), we would need to transform the variable such that the domain becomes (−1,1). We define $y = 2x - 1$ so that $f(y) = 1 + \left(\frac{y+1}{2}\right) + \left(\frac{y+1}{2}\right)^2 = 1.75 + y + 0.25y^2$. Using the basis functions as the Legendre polynomials of order 0 and 1 as $P_0(y) = 1$ and $P_1(y) = y$, and writing the approximating polynomial as $f_1(y)=c_0 P_0(y)+c_1 P_1(y)$, we get

$$c_0 = \frac{\langle P_0(y),\, f(y)\rangle}{\langle P_0(y), P_0(y)\rangle} = \frac{\int_{-1}^{1} 1\cdot\left(1.75 + y + 0.25y^2\right)dy}{2/(2\times 0+1)} = \frac{11}{6}$$

$$c_1 = \frac{\langle P_1(y),\, f(y)\rangle}{\langle P_1(y), P_1(y)\rangle} = \frac{\int_{-1}^{1} y\cdot\left(1.75 + y + 0.25y^2\right)dy}{2/(2\times 1+1)} = 1$$

Finally, transforming back to the variable *x*, we get $f_1(x) = \frac{11}{6} + 1\times(2x-1) = \frac{5}{6} + 2x$, the same as that obtained in Example 4.2.

For this simple case, the advantage of using orthogonal polynomials is not apparent. However, for fitting higher order polynomials, it results in considerable saving of computations. Also, the normal equations for nonorthogonal basis become increasingly ill-conditioned as the degree of the approximating polynomial increases and may lead to large errors in the solution. Another advantage of the orthogonal polynomials is that the coefficients are independent of the degree of polynomial and addition of a higher degree term does not require re-computation of *all* coefficients. For example, after computing the coefficients of an m^{th} degree polynomial, if we find that the approximating polynomial is not sufficiently close to the function f(x), we can add the $(m+1)^{th}$ degree basis function and find its coefficient as

$c_{m+1} = \frac{\langle \phi_{m+1}, f \rangle}{\langle \phi_{m+1}, \phi_{m+1} \rangle}$. All previously computed coefficients $c_0, c_1, \ldots, c_m$ will remain same. However, if we use the normal equations with non-orthogonal polynomial basis, we need to solve the entire system of m+2 equations to compute the values of the coefficients $c_0, c_1 \ldots c_{m+1}$.

The fact that the *integral* of the *square* of the *error* has been minimised leads us to conclude that the approximating polynomial is the *best* in an *overall* sense. It may be argued that squaring of the error term leads to more weight being given to larger errors and using the absolute value (modulus) of the error may be more logical. Also, it may be seen (Line D in Fig. 4.1) that the error is larger near the ends of the interval compared to that in the middle. So if we have to compute the function value near the end points using the approximating polynomial, it may involve large errors. If no prior information is given about the point in (a,b) at which we want to evaluate the function, it may be a better idea to use the approximation which would minimise the maximum difference between the function and its approximation over the domain (a,b). In mathematical terms, we aim for minimising the L_2 norm (least squares), L_1 norm (least absolute deviation) or the L_∞ norm (minimax or minmax). The least squares method has already been discussed and the least absolute deviation method is not used very frequently since it is not easily amenable to analytical treatment. In the next subsection we describe the minimax method and introduce the Tchebycheff polynomials closely associated with minimax (or uniform) approximation.

Minimax Approximation

The least squares method provides us with an approximation of a function which is best in an overall sense since the integral of the squared error over the relevant domain is minimum. However, if we now use this approximation to estimate the function value at a particular point, we may get a larger error than that obtained from some other approximation. In fact, if we know that the function value is to be estimated close to a specified point, the Taylor's series expansion about that point is likely to be better than the least squares method. However, since the point at which the function has to be estimated is not known *a priori* but may be anywhere in the given domain, it is logical to think of an approximation which would minimize the maximum difference between the function and its approximation over the entire domain. This will ensure that no matter where the point is within the domain, the approximated value of the function will be within a certain *distance* from the actual value and this distance would be minimum for the minimax approximation. A general technique for obtaining the minimax approximation is beyond the scope of this text. However, a closely related technique based on the Tchebycheff (or Chebyshev) polynomials is described next.

Tchebycheff polynomials

Consider the problem of obtaining the minimax approximation of a n^{th} degree polynomial, $\sum_{j=0}^{n} a_j x^j$, by a $(n-1)^{th}$ degree polynomial, $\sum_{j=0}^{n-1} \alpha_j x^j$, over the domain (−1,1) (for convenience, we assume that n>1 since some of the expressions derived in this section do not directly extend to n=1). This is equivalent to the problem:

$$\text{minimize } \sup_{x\in[-1,1]}\left|\sum_{j=0}^{n} c_j x^j\right| \text{ with } c_j = a_j - \alpha_j \text{ for } j = 0,1,...n-1 \text{ and } c_n = a_n \qquad (4.13)$$

i.e., to find the n^{th} degree polynomial, with leading coefficient c_n, having the smallest maximum norm in [-1,1]. These polynomials, with the coefficient c_n assigned a value such that the maximum norm of the polynomial becomes unity, are called Tchebycheff polynomials and are denoted by $T_n(x)$. Using the Tchebycheff alternation theorem (Theorem 4.3, and note that we are fitting a n-1 degree polynomial) and the fact that $T_n'(x)$ is a polynomial of degree n−1, which can have *at most* n−1 zeroes in [−1,1], we deduce that there are *exactly* n+1 distinct points $-1 = x_1 < x_2 < x_3 ... < x_n < x_{n+1} = 1$ such that $\left|T_n(x_i)\right| = T_{\max}$ for i =1 to n+1 (T_{max} being the maximum norm of $T_n(x)$ over [−1,1]) and, applying the stationary points theorem, $T_n'(x_i) = 0$ for i =2 to n (1 and n+1 are excluded since these are boundary points and the derivative need not vanish even if the function has an extremum).

These properties, along with the facts listed below enable us to write expressions for the Tchebycheff polynomials as follows:

$T_{\max}^2 - T_n^2(x)$ has simple zeroes at x = −1 and x = 1 and double zeroes at $x_2, x_3, \ldots, x_n$,[1] therefore

$$T_{\max}^2 - T_n^2(x) = A\left(1-x^2\right)\left[\prod_{i=2}^{n}(x-x_i)\right]^2$$

where *A* is a constant. Since the leading term in $T_n^2(x)$ is $c_n^2 x^{2n}$, it follows that

$$T_n^2(x) = T_{\max}^2 - c_n^2\left(1-x^2\right)\left[\prod_{i=2}^{n}(x-x_i)\right]^2$$

Similarly, since $T_n'(x_i) = 0$ for i =2 to n and the leading term in $T_n'(x)$ is $nc_n x^{n-1}$,

$$T_n'(x) = nc_n\prod_{i=2}^{n}(x-x_i) = \pm n\sqrt{\frac{T_{\max}^2 - T_n^2(x)}{1-x^2}} \qquad (4.14)$$

Eq. (4.14) may be differentiated and the terms rearranged (using the fact that $\frac{d}{dx}\sqrt{T_{\max}^2 - T_n^2(x)} = \frac{-T_n(x)T_n'(x)}{\sqrt{T_{\max}^2 - T_n^2(x)}} = \frac{\mp nT_n(x)}{\sqrt{1-x^2}}$) to get the following differential equation

$$\frac{d}{dx}\left[\sqrt{1-x^2}\,\frac{dT_n(x)}{dx}\right] + \frac{n^2}{\sqrt{1-x^2}}T_n(x) = 0 \qquad (4.15)$$

which is known as the Tchebycheff differential equation (note the similarity with the Legendre equation, Eq. 4.11). On putting $x = \cos\theta$ and using the conditions that these are polynomials with a maximum norm of unity, we get

$$T_n(x) = \cos\left(n\cos^{-1}x\right) \qquad (4.16)$$

---

[1] Recall that the derivative $T_n'(x_i)$ is not zero at the end points and vanishes only at points $x_2, x_3, \ldots x_n$.

Theorem 4.3: **Tchebycheff alternation theorem**

The theorem states that

An n^{th} degree polynomial, $f_n(x)$, is the (unique) minimax approximation of the continuous function f(x) over [a,b], if and only if there are *at least* n+2 points $a \le x_1 < x_2 \ldots < x_{n+2} \le b$ at which the residual, $f(x)-f_n(x)$, attains its maximum magnitude with alternating signs.

Proof:

We first prove the sufficient condition, i.e., show that if there are at least n+2 such points, $f_n(x)$ would be minimax approximation of f(x).

We denote the maximum magnitude of the residual, $\left\| f(x) - f_n(x) \right\|_\infty = M$. Now, let us assume that there is another n^{th} degree polynomial, $g_n(x)$, such that $\left\| f(x) - g_n(x) \right\|_\infty < M$. Since $g_n(x) - f_n(x) = \left[f(x) - f_n(x) \right] - \left[f(x) - g_n(x) \right]$, and $f(x) - f_n(x) = \pm M$ at all the alternation points ($x_1, x_2 \ldots x_{n+2}$), it is obvious that $g_n(x) - f_n(x)$ would have the same sign as $f(x) - f_n(x)$ at all these points. Therefore, $g_n(x) - f_n(x)$ would also have alternating signs at the *n*+2 alternation points (generally called the *critical points*) and, consequently, must have at least *n*+1 zeroes. However, since $g_n(x) - f_n(x)$ is an n^{th} degree polynomial, it must be identically zero, contradicting the assumption that $g_n(x)$ is different from $f_n(x)$.

A rigorous proof of the necessary condition, i.e., to show that if there are less than n+2 such points, an approximating polynomial *better than* $f_n(x)$ could be obtained, is quite involved. Here we provide a brief outline:

Let there be *m*+1 critical points in order of increasing *x* as $x_0(\Box a), x_1, x_2, \ldots x_m(\Box b)$ such that the residual $f(x_i) - f_n(x_i) = (-1)^i M$ [Note that we have assumed the residual to be positive at the first critical point (x_0). The proof is similar for the case when it is negative]. We now divide the domain [*a*,*b*] into *m*+1 intervals $[a,\chi_1]$, $[\chi_1,\chi_2]$, … $[\chi_m,b]$ such that each χ_i is in the open interval (x_{i-1},x_i). This ensures that within each interval there is only one critical point. It is then obvious that in each of these intervals there is only one point (the critical point) where the magnitude of the residual is equal to *M* and everywhere else it is smaller. For example, considering the first interval $[a,\chi_1]$, the residual is +M at x_0 and lies in the **open** range (–M, M) at all other points. This implies that we can find a positive δ_1 smaller than *M*, such that the residual in the first interval satisfies $M \ge f(x) - f_n(x) \ge -M + \delta_1$. If we have a function, say, *g(x)*, which is nonnegative over the closed interval $[a,\chi_1]$ and has a maximum magnitude of N_1, it is easily shown that *in the first interval* $\left\| f(x) - f_n(x) - \alpha_1 g(x) \right\| < M$ where $0 \le \alpha_1 \le \frac{\delta_1}{N_1}$ (to account for the case when *a* is a critical point, we stipulate that *g(x)*

does not vanish at *a*. However, since the χ^s, by definition, are not critical points, *g(x)* may be allowed to be zero there). Similarly, for the second interval, the residual is equal to $-M$ at one point (x_1) and there exists another positive number δ_2 smaller than *M*, such that $M-\delta_2 \geq f(x)-f_n(x) \geq -M$. Again, if there is a function *g(x)* which is nonpositive over this interval and has a maximum magnitude of N_2, we have *in the second interval* $\left\| f(x)-f_n(x)-\alpha_2 g(x) \right\| < M$ where $0 \leq \alpha_2 \leq \frac{\delta_2}{N_2}$. Extending this argument, it can be concluded that if a function *g(x)* is alternately nonnegative and nonpositive in the intervals $[a,\chi_1]$, $[\chi_1,\chi_2]$, … $[\chi_m,b]$, and does not vanish at $x=a$ and $x=b$, then the function $f_n(x)+\alpha g(x)$ would be a better approximation [compared to $f_n(x)$] of *f(x)* in the maximum norm over [a,b], α being a positive number less than the minimum of $\frac{\delta_1}{N_1},\frac{\delta_2}{N_2},...,\frac{\delta_{m+1}}{N_{m+1}}$. It turns out that the function $g(x)=\prod_{i=1}^{m}(\chi_i - x)$ satisfies the required conditions. Since this is a polynomial of order *m*, $f_n(x)+\alpha g(x)$ would be a polynomial of order *n* if $m \leq n$ and it would contradict the fact that $f_n(x)$ is the best approximating n^{th} degree polynomial in the maximum norm. Therefore, *m* must be greater than or equal to *n*+1 and the number of critical points (which was assumed as *m*+1) must be greater than or equal to *n*+2.

The first few Tchebycheff polynomials are listed below:

$T_0(x)=1 \qquad T_1(x)=x \qquad T_2(x)=2x^2-1 \qquad T_3(x)=4x^3-3x \qquad T_4(x)=8x^4-8x^2+1$

and it can be easily shown that the leading coefficient for $T_n(x)$ is 2^{n-1} for n>0. Returning to our original problem of approximating the n^{th} degree polynomial, with a leading coefficient a_n, by a $(n-1)^{th}$ degree polynomial using the minimax criterion, it can be seen that the residual $\sum_{j=0}^{n} c_j x^j$ should be equal to $2^{1-n} a_n T_n(x)$, from which the coefficients α could be obtained.

Example 4.4: The function $f(x)=1+2x+3x^2$ has to be approximated by a linear function over the interval (–1,1). Find the approximating function by minimizing the L_2 and L_1 norms.

Solution: The minimization of L_2 norm is achieved by using the Legendre polynomials, and the approximating function is obtained as

$$f_1(x) = \frac{\langle P_0(x), f(x)\rangle}{\langle P_0(x), P_0(x)\rangle} P_0(x) + \frac{\langle P_1(x), f(x)\rangle}{\langle P_1(x), P_1(x)\rangle} P_1(x)$$

$$= \frac{\int_{-1}^{1} 1\cdot\left(1+2x+3x^2\right)dx}{2/(2\times 0+1)} + \frac{\int_{-1}^{1} x\cdot\left(1+2x+3x^2\right)dx}{2/(2\times 1+1)} x = 2+2x$$

The minimization of the L_1 norm is obtained from the Tchebycheff polynomial of degree 2 (note that n=2, and the leading coefficient a_n=3), by writing the second degree polynomial with a leading coefficient of 3 and having a minimum L_1 norm as $c_0 + c_1 x + c_2 x^2 = 2^{-1} 3\left(2x^2 - 1\right) = 3x^2 - \frac{3}{2}$, giving $c_0 = -3/2$, $c_1 = 0$, and $c_2 = 3$. From Eq. (4.13), $\alpha_0 = a_0 - c_0 = \frac{5}{2}$ and $\alpha_1 = a_1 - c_1 = 2$ indicating that the straight line 5/2+2x is the best fit in the minimax sense. Figure 4.4 shows both of these approximations.

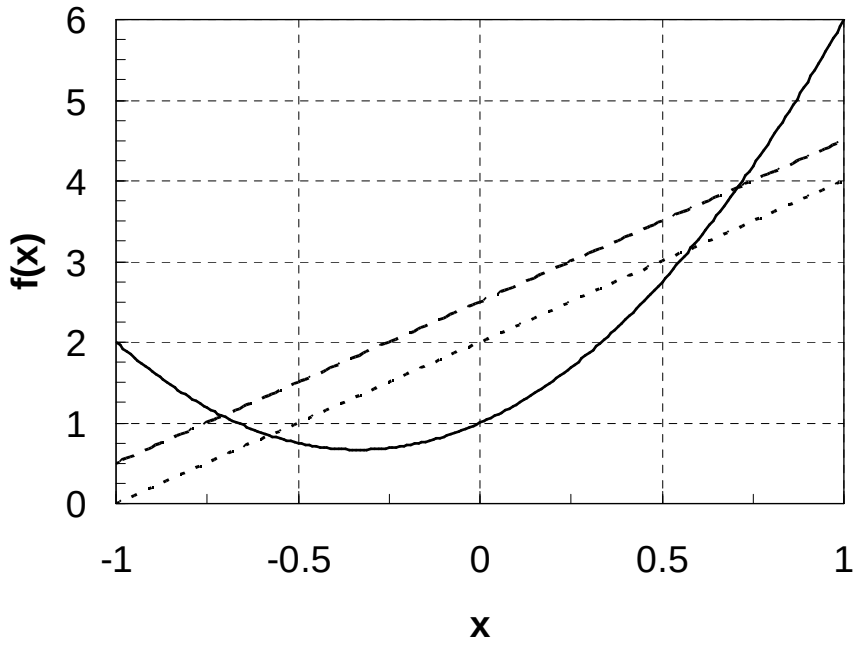


Figure 4.4 The function $f(x) = 1 + 2x + 3x^2$ and its straight line approximation over the domain (−1,1). Solid line – function, Dashed line – best L_1 approximation, Dotted line – best L_2 approximation.

The following may be noted from this figure:

- Minimax approximation of any function f(x) by a straight line (n=1) should have *at least* 3 distinct points where the residual attains its maximum magnitude with alternating signs. The residual is seen to be 1.5, −1.5, and 1.5 at x = −1, 0, and 1, respectively.
- If the function to be approximated is a polynomial of the next higher degree, there are *exactly* 3 such alternation points which include the two end points.
- The maximum magnitude of the residual using the least squares fit is 2. The least squares fit is much better in the centre but has large errors near the ends.

Some properties of Tchebycheff polynomials

(a) Similar to the Rodrigues' formula for the Legendre polynomials, we have

$$T_n(x) = \frac{(-2)^n\, n!\sqrt{1-x^2}}{2n!}\frac{d^n}{dx^n}\left[\left(1-x^2\right)^{n-\frac{1}{2}}\right]$$

(b) The recursive relation for the Tchebycheff polynomials is

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \qquad n = 2,3,.... \qquad \text{with } T_0(x) = 1, T_1(x) = x$$

(c) In the domain (−1,1), $T_n(x)$ has n zeroes, called the Tchebycheff abscissae, and n+1 extrema, where it is equal to +1 or −1. These are given by

$$\textit{zeroes}: \qquad x_i = \cos\left(\frac{2i+1}{n}\frac{\pi}{2}\right) \qquad i = n-1, n-2, ...1, 0$$

$$\textit{extrema}: \qquad x_i = \cos\left(\frac{i}{n}\pi\right) \qquad i = n, n-1, ..., 1, 0$$

(d) $T_n(x)$ has the smallest maximum norm (equal to 1) in (−1,1) out of all n^{th} degree polynomials with leading coefficient 2^{n-1}. In other words, for all monic polynomials[1] of n^{th} degree, $2^{1-n}\,T_n(x)$ has the smallest maximum norm of 2^{1-n}.

Orthogonality of Tchebycheff polynomials

Using the orthogonality of the Cosine function[2], it is easily seen that

$$\int_{-1}^{1} T_m(x)T_n(x)\frac{1}{\sqrt{1-x^2}}\,dx = \begin{cases} 0 & m \neq n \\ \pi & m = n = 0 \\ \pi/2 & m = n \neq 0 \end{cases} \qquad (4.17)$$

Thus if we modify our earlier definition of the inner product (Eq. B4.1.1) as

$$\langle f, g \rangle = \int_a^b w(x)\, f(x) g(x) dx \qquad (4.18)$$

where w(x) is a weight function[3], the Tchebycheff polynomials are orthogonal over the interval (−1,1) with respect to the weight function $1/\sqrt{1-x^2}$. Introduction of the weight allows us to manipulate the problem to account for desired objective and/or additional information[4]. For example, if we wish to approximate a function over the interval (−1,1) but know that its argument is more likely to lie in the range (−0.5,0.5) than outside it, we will

---

[1] A monic polynomial is one in which the coefficient of the highest order term is unity.

[2] $\int_0^{\pi} \cos m\theta \cos n\theta\, d\theta = 0$ for $m \neq n$

[3] The weight function is continuous and positive over [a,b] but may have integrable singularities at the end points. The simplest form, w(x)=1, has already been discussed. Some other possibilities are: (i) $w(x)=\exp(-x^2)$ over the interval (-∞,∞) which gives rise to orthogonal polynomials called the *Hermite polynomials* and (ii) w(x)=exp(-x) over (0,∞) which results in *Laguerre polynomials*.

[4] The Legendre and Tchebycheff polynomials are special cases of the Jacobi polynomials which are orthogonal with respect to the weight function $w(x)=(1-x)^{\alpha}(1+x)^{\beta}$. For Legendre polynomials $\alpha=\beta=0$ and for Tchebycheff, $\alpha = \beta = -\frac{1}{2}$.

choose the weight accordingly. Similarly, as discussed earlier, since the *unweighted* (or *ordinary*) least squares fit resulted in larger errors near the end points, having a larger weight near the ends tends to provide a fit which distributes the error more evenly over the interval. The orthogonality condition may be used to obtain the approximation of any function in terms of the Tchebycheff polynomials as described in the following example.

Example 4.5: The function $f(x)=\sqrt{1-x^2}$ has to be approximated by a 2nd degree polynomial over the interval (–1,1). Find the approximating function by using Tchebycheff polynomials.

Solution: We write the approximating polynomial as $f_2(x)=c_0T_0(x)+c_1T_1(x)+c_2T_2(x)$ and obtain the coefficients by using the orthogonality property. We then have

$$f_2(x) = \frac{\langle T_0(x), f(x)\rangle}{\langle T_0(x), T_0(x)\rangle}T_0(x)+\frac{\langle T_1(x), f(x)\rangle}{\langle T_1(x), T_1(x)\rangle}T_1(x)+\frac{\langle T_2(x), f(x)\rangle}{\langle T_2(x), T_2(x)\rangle}T_2(x)$$

$$= \frac{\int_{-1}^{1} 1\cdot\sqrt{1-x^2}\cdot\frac{1}{\sqrt{1-x^2}}dx}{\pi}+\frac{\int_{-1}^{1} x\cdot\sqrt{1-x^2}\cdot\frac{1}{\sqrt{1-x^2}}dx}{\pi/2}x+\frac{\int_{-1}^{1}\left(2x^2-1\right)\cdot\sqrt{1-x^2}\cdot\frac{1}{\sqrt{1-x^2}}dx}{\pi/2}\left(2x^2-1\right)$$

$$= \frac{10}{3\pi}-\frac{8}{3\pi}x^2$$

Fig. 4.5 shows the plot of the function and its approximation.

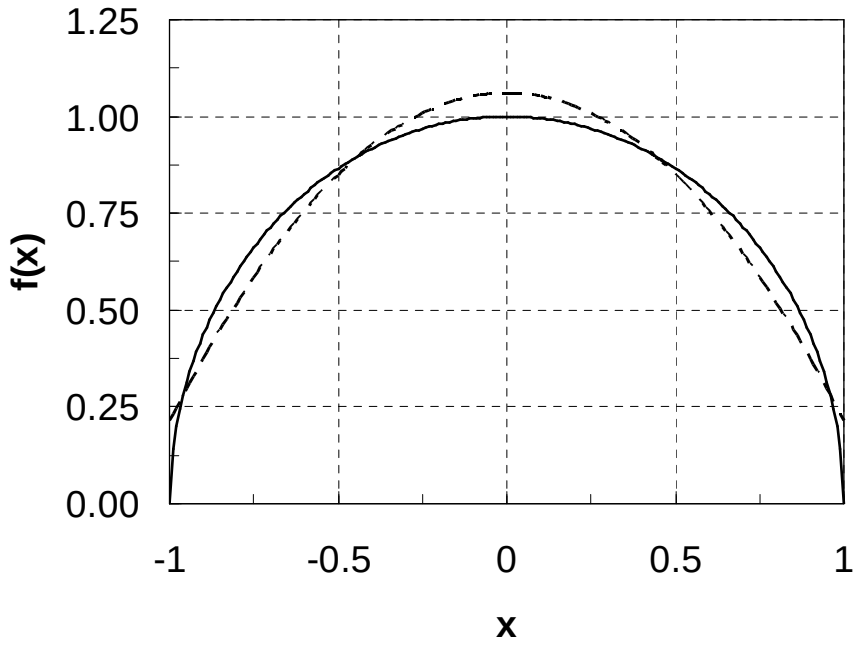


Figure 4.5 The function $f(x)=\sqrt{1-x^2}$ and its 2nd degree Tchebycheff polynomial approximation over the domain (–1,1). Solid line – function, Dashed line – Tchebycheff approximation.

Remarks on use of orthogonal polynomials:

- The function, f(x), should be of a form which allows easy evaluation of the integrals required in the computations. If the integrals cannot be obtained analytically, e.g., $f(x)=x^x$, we may perform numerical integration (described in the next chapter) or may go for discrete data fit (described in the next section).
- For some function, e.g., $f(x)=e^x$, the Legendre polynomial fit may be easily obtained but the Tchebycheff fit requires numerical integration.

Many times the functional relationship between the dependent variable and the independent variable(s) is not known. The only information we have is the observed (or computed) value of the function, f(x), at a few values of x. While numerical integration is a possibility in such cases, it is generally better to go for discrete equivalents of the methods discussed in this section. These are described in the next section.

Exercise 4.3

1. Approximate the function exp(x) over (–1,1) by a straight line using (a) Legendre polynomials and (b) Tchebycheff polynomials. [For the Tchebycheff method use the following values: $\int_{-1}^{1} \frac{e^x dx}{\sqrt{1-x^2}} = 3.97746$, $\int_{-1}^{1} \frac{xe^x dx}{\sqrt{1-x^2}} = 1.77550$]
2. For the previous problem, plot the residual in the Tchebycheff approximation and ascertain whether it is a minimax approximation using the alternation theorem. Comment on the result.
3. Although we have not described the methods for obtaining the best L_∞ and L_1 approximations for a general function, we list these approximations[1] for the exponential function over the interval (–1,1): (a) Minimax: $f_1(x) = 1.26428 + 1.17520x$ (b) Least absolute deviation: $f_1(x) = 1.12763 + 1.04219x$. Plot these two approximations along with the two obtained in problem 1 and comment on their relative positions.
4. For all four approximations, Legendre (minimizing L_2), Minimax (minimizing L_1), LAD (minimizing L_∞), and Tchebycheff (minimizing L_1 if the function is a polynomial of one degree higher than the approximating polynomial), obtain the L_1, L_2, and L_∞ norms of the residuals. (Note: while L_2 is straightforward to compute, other two norms cannot be readily obtained. For this problem, it would be possible to compute these norms by observing that there are two points of intersection and the residual is negative between these points and positive elsewhere. The points of intersection are- Legendre : –0.533232 and 0.620700, Minimax : –0.616401 and 0.779467, LAD : –0.5 and 0.5, and Tchebycheff : –0.665210 and 0.746749)

---

[1] The exact values of the coefficients are – Minimax: $c_1 = 0.5\left(e - e^{-1}\right), c_0 = 0.5\left(e - c_1 \ln c_1\right)$ and LAD: $c_0 = 0.5\left(e^{0.5} + e^{-0.5}\right), c_1 = e^{0.5} - e^{-0.5}$.

5. Consider the approximation of the function $f(\theta) = \dfrac{\pi^2}{2\pi^2 + \theta^2 - 2\pi\theta}$ in the interval [0, 2π]. Approximate the function by employing a Legendre basis $\left\{P_j(x)\right\}_{j=0}^{j=3}$ after mapping the θ-domain to the x-domain in such a way that [0,2π] maps into [-1,1]. Graphically compare the function (in the x domain) and the approximating polynomial.
6. Approximate the function $f(x) = \dfrac{1}{1+25x^2}$ over the interval (−1,1) by a 2nd order polynomial minimizing the L_2 norm. Improve the fit by using a fourth order polynomial and plot both these approximations.
7. Consider the polynomial $\{p_n(x)\}$ given by $p_n(x) = \dfrac{\sin(n+1)\phi}{\sin\phi}$ where $x = \cos\phi$. Show that this polynomial satisfies the same recursion formula as the Tchebycheff polynomial. What are the first three polynomials, $p_0(x)$, $p_1(x)$ and $p_2(x)$? Show that $\{p_n(x)\}$ forms an orthogonal system of polynomials with the weight function $w(x) = (1 - x^2)^{1/2}$, $x \in [-1, 1]$.

4.4 Approximation of data

As described in the beginning of this chapter, if we have a table of data listing the values of the dependent variable, f(x), corresponding to a few values of the independent variable, x, we may choose the approximating function, $\tilde{f}(x)$, either to pass through *all* data points (interpolation) or to represent the general trend of the data (regression). We denote the data points by the set of values[1] $\{(x_k, f(x_k)), k=0,1,\ldots,n\}$. The form of the approximating function would depend on the type of data but we again assume it to be an m^{th} -degree polynomial, $f_m(x)$, represented by Eq. (4.8), i.e.,

$$f_m(x) = \sum_{j=0}^{m} c_j \phi_j(x) \tag{4.19}$$

in which the ϕ's are the polynomial basis functions and the c's are coefficients. Obviously, for an interpolation problem, in general m=n (a lower degree polynomial *may* interpolate the given data *in some cases*, e.g., three points lying on a straight line: n=2, m=1), and for regression, m<n [when m>n, we will have infinite solutions, e.g., fitting a parabola (m=2), through two points (n=1)]. Clearly, the interpolation problems are conceptually simpler than the regression problems since we do not have to worry about the degree of approximating polynomial (m=n) and quantification of the residuals (all residuals are zero). Therefore, we discuss interpolation first and then move on to regression.

Interpolation

The problem can be stated as: given the function values at a set of n+1 *distinct* points, find the polynomial (of degree *at most* n) which matches the function value at these points. Although this polynomial is unique[2], it may be expressed in many different forms. We describe below these alternative forms and look at the situations for which they are suitable.

---

[1] For interpolation problems, we require all x's to be distinct.

[2] If the function values are given at *distinct* points, as shown in the next subsection

Conventional Form

In the conventional representation, $\phi_j(x) = x^j$, and the coefficients c_j are obtained from the following set of linear equations representing the equality of the approximating polynomial and the function value at all data points:

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{Bmatrix} c_0 \\ c_1 \\ . \\ . \\ c_n \end{Bmatrix} = \begin{Bmatrix} f(x_0) \\ f(x_1) \\ . \\ . \\ f(x_n) \end{Bmatrix} \tag{4.20}$$

This matrix is called the Vandermonde matrix and it is relatively straightforward to show that its determinant may be written as multiplication of terms of the form $\left(x_i - x_{j(\neq i)}\right)$. This implies that a unique solution will exist for distinct points[1]. However, the Vandermonde matrix is known to be ill-conditioned for large n[2]. Also, the addition of one or more data points or change in function values at an existing point necessitates the re-computation of all the coefficients. Therefore this method is not recommended.

Lagrange Form

The polynomials ϕ are chosen in such a way that each of them is an n^{th} degree polynomial (denoted by L) and satisfies the following:

$$L_i(x_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

These are known as the Lagrange interpolating polynomials[3] and it is apparent that these may be written as

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \tag{4.21}$$

Another useful form in which the Lagrange polynomials could be written is based on the fact that $\frac{d}{dx}\left[\prod_{j=0}^{n}(x - x_j)\right] = \sum_{i=0}^{n}\left[\prod_{\substack{j=0 \\ j \neq i}}^{n}(x - x_j)\right]$, which is the denominator of Eq. 4.21, resulting in

[1] The uniqueness may also be proved by arguing that if there is another n^{th} degree interpolating polynomial, the difference of the two interpolating polynomials would also be an n^{th} degree polynomial, which will vanish at all n+1 grid points (since both the interpolating polynomials reproduce the function value at the grid points). This implies that the difference is identically zero.

[2] For example, if we take the domain to be (-1,1), increasing the number of points will (i) cause the points to come nearer to one another, and (ii) make the higher powers approach zero. Both these result in ill-conditioning of the matrix.

[3] Proposed by Waring in 1779 and later by Lagrange in 1795.

$$L_i(x) = \frac{\prod_{j=0}^{n}(x - x_j)}{(x - x_i)\left[\frac{d}{dx}\left\{\prod_{j=0}^{n}(x - x_j)\right\}\right]_{x=x_i}} \qquad (4.22)$$

The equality condition at the data points then implies that the coefficients c_i are simply the function values at corresponding points, i.e., $c_i=f(x_i)$. This method is preferred when the grid points (i.e., the values of the independent variable) remain fixed but the function values keep changing, e.g., measurement of water depth at a few selected locations in a channel. If we want to predict the function value at an intermediate point, the L's have to be computed only once for that point. The interpolated function value at that point is readily obtained for any particular set of observed values at the grid points. However, if we change the grid points, e.g., by adding or deleting an observation, the Lagrange polynomials need to be recomputed. Similarly, if the degree of the interpolating polynomial is not known *a priori* but is to be obtained from looking at the proximity of the fit to observed data for increasing number of grid points, Lagrange polynomials would not work very efficiently. For such cases Newton's (divided difference) form of the interpolating polynomials works better.

Newton's (Divided Difference) Form

The basis functions are chosen in such a way that ϕ_i is an i^{th}-degree polynomial defined as

$$\phi_0(x) = 1 \qquad \phi_i(x) = \prod(x - x_0)(x - x_1)...(x - x_{i-1}) \quad \text{for } i = 1, 2, ..., n+1 \qquad (4.23)$$

These are known as the Newton's divided difference polynomials and satisfy the following:

$$\phi_i(x_j) = 0 \qquad \text{for } j < i$$

The coefficients, c_i, are again obtained by the equality of the interpolating polynomial and the given function at the grid points. For example, the equality at $x=x_0$ results in $c_0=f(x_0)$ since all other basis functions (except ϕ_0) are zero at this point. At the next point ($x=x_1$), only ϕ_0 and ϕ_1 are non-zero with $\phi_0=1$ and $\phi_1=x_1-x_0$. Therefore, the coefficient c_1 is obtained from the equality condition, $f(x_1) = c_0\phi_0(x_1) + c_1\phi_1(x_1) = f(x_0) + c_1(x_1 - x_0)$, as $c_1 = \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$.

To obtain the next coefficient, c_2, we use the equality at $x=x_2$ as

$f(x_2) = c_0\phi_0(x_2) + c_1\phi_1(x_2) + c_2\phi_2(x_2) = f(x_0) + c_1(x_2 - x_0) + c_2(x_2 - x_0)(x_2 - x_1)$ and obtain

$$c_2 = \frac{\dfrac{f(x_2) - f(x_1)}{x_2 - x_1} - \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}.$$

For a more efficient recursive determination of the coefficients, we may think of the *additional* term, $c_2\phi_2(x_2)$, as the difference (at the grid point x_2) between the linear interpolating function over (x_0,x_1) and that over (x_1,x_2), see Fig. 4.6. Further, we note that the *unique* interpolating polynomial does not depend on the order of the grid points. For example, the linear interpolation over (x_0,x_1) would be same even when we use x_1 as the first point and x_0 as the second. The difference between these two interpolating functions at the point x_2 may

then be written as $f(x_1)+(x_2-x_1)\dfrac{f(x_2)-f(x_1)}{x_2-x_1}-\left[f(x_1)+(x_2-x_1)\dfrac{f(x_1)-f(x_0)}{x_1-x_0}\right]$ and, on equating it to $c_2(x_2-x_0)(x_2-x_1)$, we get the value of c_2.

Figure 4.6 Newton interpolating polynomial

The recursive relationship may be developed by assuming that all the coefficients up to c_i have been determined. The additional term, $c_{i+1}\phi_{i+1}(x_{i+1})$, is written as the difference (at the grid point x_{i+1}) of the i^{th} degree interpolating polynomial passing through the points $(x_1,x_2,x_3,...,x_i,x_{i+1})$ and that passing through the points $(x_1,x_2,...,x_{i-1},x_i,x_0)$ (note the change in order of points). Clearly, since the first *i* grid points are same, the coefficients and basis functions up to the $(i-1)^{th}$ degree term would be identical for both these interpolating polynomials and the only difference will be in the i^{th} term (even for this term, the basis function would be the same, see Eq. 4.23). This difference is written as

$$c_i'(x_{i+1}-x_1)(x_{i+1}-x_2)(x_{i+1}-x_3)...(x_{i+1}-x_i)-c_i(x_{i+1}-x_1)(x_{i+1}-x_2)(x_{i+1}-x_3)...(x_{i+1}-x_i)$$

where c_i is the coefficient in the *original* grid point arrangement and c_i' represents the coefficient in the *modified* arrangement (removing x_0 and adding x_{i+1}). Equating this difference to $c_{i+1}\phi_{i+1}(x_{i+1})$, we obtain $c_{i+1}=\dfrac{c_i'-c_i}{x_{i+1}-x_0}$.

We now use the standard definitions of the divided differences (i.e., the *difference* of a function *divided* by the *difference* of ordinates) as

First divided difference: $f\left[x_j, x_i\right] = \dfrac{f\left(x_j\right) - f\left(x_i\right)}{x_j - x_i} \quad \left(= f\left[x_i, x_j\right]\right)$

Second divided difference: $f\left[x_k, x_j, x_i\right] = \dfrac{f\left[x_k, x_j\right] - f\left[x_j, x_i\right]}{x_k - x_i} \qquad \left(= f\left[x_k, x_i, x_j\right] = \ldots = f\left[x_i, x_j, x_k\right]\right)$

Third divided difference: $f\left[x_l, x_k, x_j, x_i\right] = \dfrac{f\left[x_l, x_k, x_j\right] - f\left[x_k, x_j, x_i\right]}{x_l - x_i}$

...

n^{th} divided difference: $f\left[x_n, x_{n-1}, \ldots, x_1, x_0\right] = \dfrac{f\left[x_n, x_{n-1}, \ldots, x_1\right] - f\left[x_{n-1}, \ldots, x_1, x_0\right]}{x_n - x_0}$

(4.24)

The coefficients c_i are then given by the divided differences as

$$\begin{aligned} c_0 &= f\left[x_0\right] \equiv f\left(x_0\right) \\ c_1 &= f\left[x_1, x_0\right] \\ c_2 &= f\left[x_2, x_1, x_0\right] \\ &\cdot \\ &\cdot \\ c_i &= f\left[x_i, x_{i-1}, \ldots, x_0\right] \\ &\cdot \\ &\cdot \\ c_n &= f\left[x_n, x_{n-1}, \ldots, x_0\right] \end{aligned} \qquad (4.25)$$

and, for hand-computations, are efficiently obtained by using a divided difference table (see Example 4.6). The remainder, R_n, at any point x, is obtained, by using an argument similar to that used in the derivation of the recursive relation, as

$$R_n\left(x\right) = f\left(x\right) - f_n\left(x\right) = \phi_{n+1}\left(x\right) f\left[x, x_n, x_{n-1}, \ldots, x_0\right] \qquad (4.26)$$

Since f(x) is unknown, the divided difference in Eq. (4.26) can not be obtained. However, if an additional point $(x_{n+1}, f(x_{n+1}))$ is available, it may be used in place of f(x) to *approximate* the divided difference[1]. A significant advantage of the Newton's method over the previous two forms is, therefore, the ease of estimation of error. On the other hand, if f(x) is known in the functional form and its derivatives up to order n+1 exist, we may use the relation between the divided difference and the function derivative (Box 4.3) and write (also see Eq. 3.6)

$$R_n\left(x\right) = \phi_{n+1}\left(x\right) \frac{f^{(n+1)}\left(\xi\right)}{\left(n+1\right)!} \qquad (4.27)$$

[1] This concept is similar to the one used in chapter 1 where, in absence of the true value, an approximate error was defined as the difference between two successive iterations. Here, $R_n(x)$ is equivalent to $f_{n+1}(x)$-$f_n(x)$.

in which $\xi \in (x, x_0, x_1, ..., x_n)$. Again, since ξ is not known, we will not be able to compute the error at any point, *x*. We may, however, obtain an upper bound of the error from the behaviour of the $(n+1)^{th}$ derivative of the function.

Box 4.3: **Relation between the divided difference and the derivative**

The function $g(x) = f(x) - f_n(x)$ will have *at least* n+1 zeroes (at all the grid points) in the interval spanned by the grid points $(x_0, x_1, ..., x_n)$. Repeated application of Rolle's theorem indicates that there would be at least one point, say ξ, in this interval at which the n^{th} derivative of *g(x)* will be zero. Hence $f^n(\xi) = \left.\frac{d^n f_n(x)}{dx^n}\right|_{x=\xi} = \left.\frac{d^n \sum_{i=0}^{n} c_n \phi_n(x)}{dx^n}\right|_{x=\xi}$. The basis functions up to the order n−1 will not contribute to the n^{th} derivative, and the only term would be $\frac{d^n c_n \phi_n(x)}{dx^n} = n!\, f[x_n, x_{n-1}, ..., x_1, x_0]$. Therefore, a relation between the divided difference and the function derivative is obtained as $f[x_n, x_{n-1}, ..., x_1, x_0] = \frac{f^{(n)}(\xi)}{n!}$, where $\xi \in (x_0, x_1, ..., x_n)$.

Remarks:

- The points x_j do not have to be in any particular order
- While all the Lagrange polynomials are of the same order (n), the Newton polynomials are of increasing order (0 to n).
- After computing the coefficients, additional effort is required in computing the interpolated value at a non-grid point. There is an algorithm[1], similar to the divided difference algorithm, which provides the interpolated value at a point with almost same amount of computational effort as used in computing the coefficients. However, we will not discuss it in detail.
- We assume that interpolation is applicable to the given data. In some cases, e.g., if C and E are located at higher ground, interpolation may not be directly applicable. We may use a *weighted* interpolation assigning higher weight to the measurement at C to estimate the temperature at E. However, this will not be discussed here.

Example 4.6: Three weather stations, A, B, and C, are located along a straight road such that B is equidistant from A and C. There are two cities, D and E, on the same road, D being

[1] Neville's algorithm

equidistant from A and B, and E from B and C. The temperatures (in °C) at a given time are recorded at A, B, and C, as 10.1, 11.3, and 11.9, respectively. Estimate the temperatures at D and E.

Solution:

Although we could use any interval, we choose to have the domain AC as (–1,1). The coordinates of various points are, therefore, A: –1, B: 0, C: 1, D: –0.5, and E: 0.5. Using the measured values at x_0= –1, x_1=0, and x_2=1, we obtain the 2nd degree interpolating polynomial in the three different forms as follows:

(a) Conventional form:

The interpolating polynomial is written as $f_2(x) = c_0 + c_1 x + c_2 x^2$ and the set of equations (see Eq. 4.20) is written as

$$\begin{bmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{bmatrix} \begin{Bmatrix} c_0 \\ c_1 \\ c_2 \end{Bmatrix} = \begin{Bmatrix} f(x_0) \\ f(x_1) \\ f(x_2) \end{Bmatrix} \Rightarrow \begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{Bmatrix} c_0 \\ c_1 \\ c_2 \end{Bmatrix} = \begin{Bmatrix} 10.1 \\ 11.3 \\ 11.9 \end{Bmatrix}$$

Which provides the solution as c_0=11.3, c_1=0.9, and c_2= –0.3. The estimated values at D (x= –0.5) is obtained as 10.775 °C and at E (x=0.5) as 11.675 °C.

(b) Lagrange form:

Using Eq. (4.21) we obtain the Lagrange polynomials as

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} = \frac{x(x-1)}{2}; L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} = \frac{(x+1)(x-1)}{-1}; L_2(x) = \frac{(x+1)x}{2}$$

The interpolating polynomial is written as $f_2(x) = f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x)$ and it may be easily verified that it reduces to the same form as in (a). To estimate the temperature at the points D and E, we compute the values of the Lagrange polynomials at these points as $L_0(-0.5) = 0.375; L_1(-0.5) = 0.75; L_2(-0.5) = -0.125$ and $L_0(0.5) = -0.125; L_1(0.5) = 0.75; L_2(0.5) = 0.375$. The interpolated temperatures are, therefore,

at D: 10.1 x 0.375 + 11.3 x 0.75 + 11.9 x (–0.125) = 10.775 °C
at E: 10.1 x (–0.125) + 11.3 x 0.75 + 11.9 x 0.375 = 11.675 °C

(c) Newton form:

The interpolating polynomial is written as $f_2(x) = c_0 + c_1(x-x_0) + c_2(x-x_0)(x-x_1)$ with $c_0 = f(x_0), c_1 = f[x_0, x_1], c_2 = f[x_0, x_1, x_2]$. Since the order of the points is not important, we would use the point B (x=0) as x_0 and A (x= –1) as x_1. The divided differences are computed in a tabular form as shown below:

The divided differences are shown in the topmost number in each column giving us $f[x_0, x_1] = 1.2$; $f[x_0, x_1, x_2] = -0.3$ and $f_2(x) = 11.3 + 1.2x - 0.3x(x+1) = 11.3 + 0.9x - 0.3x^2$, the same as before.

The example above shows different ways of obtaining the *unique*[1] interpolating polynomial. The polynomial will pass through the function values at all the grid points used for its development. However, at any intermediate point, the estimated function value may be in error (unless, of course, the function itself is an n^{th} degree polynomial or we are lucky enough to choose a point where the interpolating polynomial matches with the function value). As mentioned earlier, the interpolating polynomial is unique and, therefore, the error is also same irrespective of whether we express it in the conventional, Lagrange, or Newton form. Since the true value of the function will generally be unknown at any point other than the grid points, it is generally not possible to estimate the true error. On the other hand, for the interpolation to be useful, we must have some way of estimating the error. Eqs. (4.26) and (4.27) are two alternative ways of doing it.

While the three forms discussed above are applicable to any distribution of the grid points, most of the times we will have equally spaced points $x_0, x_1, \ldots, x_n$. Also, sometimes we will have complete flexibility in choosing these points, e.g., when deciding on times at which to measure the distance travelled by an object. For the first case, a set of orthonormal polynomials, closely related to Legendre polynomials and known as Gram's polynomials, are useful. For the second case, if we aim at minimising the maximum error of interpolation, discrete form of Legendre or Tchebycheff polynomials are used. These are described next.

Gram's Polynomial

As in Legendre polynomials, we assume that the range of the (**equidistant**) data is normalised such that $x_0 = -1$ and $x_n = 1$, which implies that $x_i = -1 + 2i/n$, $i = 0$ to n. The Gram's

---

[1] It is obvious that higher order polynomials would be non-unique since if $f_n(x)$ interpolates the function value at all grid points, any polynomial of the form $f_n(x) + A(x)\prod_{j=0}^{n}(x - x_j)$ would also do so since the second term vanishes at all grid points. One may try to get a higher order interpolating polynomial by omitting a few lower order terms, e.g., for three data points, x_0, x_1 and x_2, using an interpolating polynomial of the form $c_0 + c_2x^2 + c_3x^3$. However, a *unique* solution (or even *a solution*) is not guaranteed in such cases. If, for example, $f_3(x)$ is an interpolating polynomial and the grid points are located such that $(x - x_0)(x - x_1)(x - x_2)$ is of the form $\alpha_0 + \alpha_2x^2 + \alpha_3x^3$, then $f_3(x) + \lambda(x - x_0)(x - x_1)(x - x_2)$ would also be an interpolating polynomial of the same form. See exercise 4.4.? (fitting $c_0 + c_1x + c_3x^3$ to (0,1), (1,2), (2,6); (-1,0),(0,1),(1,2) and (-1,0),(0,1),(1,3)). Also, the computation of the interpolating polynomial is not very efficient since it cannot be expressed in Lagrange or Newton forms.

polynomials then satisfy the orthonormality condition (note that the inner product is now defined by a sum as opposed to the integral used when the function was given[1])

$$\left\langle G_i(x), G_j(x) \right\rangle = \sum_{k=0}^{n} G_i(x_k) G_j(x_k) = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \qquad (4.28)$$

The general equation for generating Gram's polynomials of order *n* is

$$G_{i+1}(x) = \alpha_i x G_i(x) - \frac{\alpha_i}{\alpha_{i-1}} G_{i-1}(x) \quad \text{for } i = 0,1,2,...,n-1$$
$$\text{with } G_{-1}(x) = 0; G_0(x) = \frac{1}{\sqrt{n+1}} \text{ and } \alpha_i = \frac{n}{i+1}\sqrt{\frac{4(i+1)^2 - 1}{(n+1)^2 - (i+1)^2}} \qquad (4.29)$$

Thus, for n=1: $G_0 = \frac{1}{\sqrt{2}}; G_1 = \frac{x}{\sqrt{2}}$, and for n=2: $G_0 = \frac{1}{\sqrt{3}}; G_1 = \frac{x}{\sqrt{2}}; G_2 = \sqrt{\frac{3}{2}}x^2 - \sqrt{\frac{2}{3}}$.

The interpolation formula is then given by

$$f_n(x) = \sum_{i=0}^{n} c_i G_i(x) \qquad (4.30)$$

and the coefficients are obtained from the equality, $f_n(x_k) = f(x_k)$, and orthonormality property, Eq. 4.28, as

$$c_i = \sum_{k=0}^{n} f(x_k) G_i(x_k) \qquad (4.31)$$

Example 4.7: Re-solve the problem described in example 4.6 using the Gram's polynomials and estimate the temperature at the point E.

Solution:

The interpolating polynomial is written as $f_2(x) = c_0 G_0(x) + c_1 G_1(x) + c_2 G_2(x)$ with $G_0 = \frac{1}{\sqrt{3}}; G_1 = \frac{x}{\sqrt{2}}; G_2 = \sqrt{\frac{3}{2}}x^2 - \sqrt{\frac{2}{3}}$, and the computations of the coefficients is shown in the table below:

k	x_k	$f(x_k)$	$G_0(x_k)$	$G_1(x_k)$	$G_2(x_k)$
0	-1	10.1	0.577350	-0.707107	0.408248
1	0	11.3	0.577350	0	-0.816497
2	1	11.9	0.577350	0.707107	0.408248
		$c_i = \Sigma f(x_k) G_i(x_k) =$	19.2258	1.27279	-0.244949

The interpolating polynomial is again seen to be the same if we expand in powers of x. To estimate the temperatures at the point E (x=0.5) we compute the Gram's polynomials at this

[1] The notation used here for inner product, <…>, is sometimes used to denote *only* discrete inner product with (…) used for the continuous inner product. However, we will use the same notation for both the continuous and the discrete inner products.

point as $G_0 = 0.577350; G_1 = 0.353553; G_2 = -0.510310$ and obtain the temperature as $f(0.5) = 19.2258 \times 0.577350 + 1.27279 \times 0.353553 + 0.244949 \times 0.510310 = 11.675$ °C.

While interpolation with equidistant points works well and the accuracy typically increases with a finer grid, Runge showed that, in some cases, increasing the number of grid points leads to larger errors. For example, interpolating the function $f(x) = \frac{1}{1+25x^2}$ over the interval (–1,1) using a 10th degree polynomial we get the following:

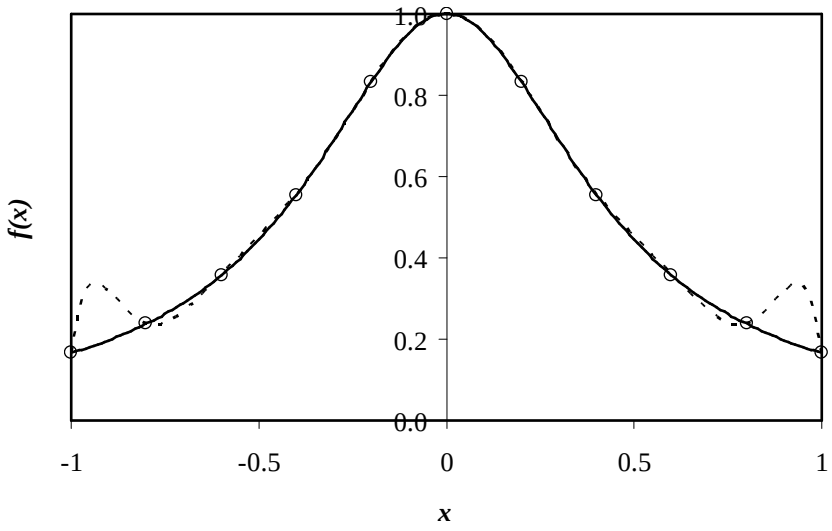


Figure 4.7 (a) Interpolation of the function $f(x) = \frac{1}{1+25x^2}$ using 11 equidistant points

If we now use a 20th degree polynomial, the interpolating polynomial is as shown below:

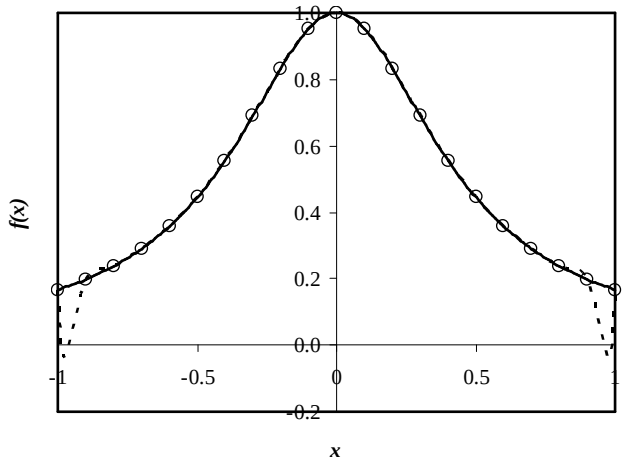


Figure 4.7 (b) Interpolation of the function $f(x) = \dfrac{1}{1+25x^2}$ using 21 equidistant points

Therefore, in this case, equidistant interpolation may result in large errors, especially near the end points of the interval. One way to avoid this problem is to use unevenly spaced grid points (provided we have complete freedom in choosing the grid points). The pertinent question would then be: is there an arrangement of points which would be the "best?" In the next subsection we try to answer this question.

Unevenly spaced grid points

From the definition of the residual, Eq. 4.27 and Eq. 4.23, it is apparent that the error of interpolation is a $(n+1)^{th}$ degree polynomial, which vanishes at all the grid points. Location of the point ξ will depend on the location of the grid points and the point at which we have to estimate the error. However, since the function behaviour is not in our control, the best one can do is to choose the grid points in such a way as to minimise some norm of ϕ_{n+1}. If we choose the L_2 norm, the grid points turn out to be the zeroes of the Legendre polynomial, $P_{n+1}(x)$, and if we choose the L_∞ norm, the grid points should be located at the zeroes of the Tchebycheff polynomial, $T_{n+1}(x)$ [1]. The interpolating polynomial may then be obtained using the conventional form, Lagrange form, or the Newton form. However, the orthogonality property for the discrete case may be utilized to obtain the interpolating polynomial as described below:

Legendre Polynomials For discrete case:

If x_k are the zeroes of $P_{n+1}(x)$, then the orthogonality condition is given by, for i, j = 0,1,2,…,n (compare with Eq. 4.12 for the case when the function is given and particularly note the presence of the weight):

$$\left\langle P_i, P_j \right\rangle = \sum_{k=0}^{n} P_i(x_k) P_j(x_k) w_k \quad \begin{cases} = 0 & i \neq j \\ = \dfrac{2}{2i+1} & i = j \end{cases} \tag{4.32}$$

where the weights are given by

$$w_k = \frac{2}{\left(1-x_k^2\right)\left[P'_{n+1}(x_k)\right]^2} \tag{4.33}$$

The interpolating polynomial is then obtained as

$$f_n(x) = \sum_{i=0}^{n} c_i P_i(x) \tag{4.34}$$

in which the coefficients are given by

[1] The L_1 norm may also be used but is not very common because of the difficulty in obtaining the grid points compared to the other two cases where the zeroes of the Legendre and Tchebycheff polynomials are easily computed.

$$c_i = \frac{\langle f, P_i \rangle}{\langle P_i, P_i \rangle} = \frac{2i+1}{2} \sum_{k=0}^{n} f(x_k) P_i(x_k) w_k \tag{4.35}$$

Clearly, the computational complexity has increased and it may be simpler and more efficient to use the Lagrange or Newton form of the interpolating polynomials with the grid points located at the zeroes of $P_{n+1}(x)$. As we see in the next subsection, the discrete version of the Tchebycheff polynomials is much simpler and is widely used.

Tchebycheff Polynomials For discrete case:

If x_k are the zeroes of $T_{n+1}(x)$, the orthogonality condition is given by, for i, j = 0,1,2,…,n (compare with Eq. 4.17 for the continuous case and note that the weight is unity):

$$\begin{aligned} \langle T_i, T_j \rangle = \sum_{k=0}^{n} T_i(x_k) T_j(x_k) \qquad &= 0 \qquad && i \neq j \\ &= n+1 \qquad && i = j = 0 \\ &= \frac{n+1}{2} \qquad && i = j \neq 0 \end{aligned} \tag{4.36}$$

The interpolating polynomial is again given as

$$f_n(x) = \sum_{i=0}^{n} c_i T_i(x) \tag{4.37}$$

in which the coefficients are given by

$$c_i = \frac{\langle f, T_i \rangle}{\langle T_i, T_i \rangle} = \frac{2}{n+1} \sum_{k=0}^{n} f(x_k) T_i(x_k) \quad \text{for } i \geq 1 \tag{4.38}$$

and $c_0 = \frac{1}{n+1} \sum_{k=0}^{n} f(x_k) T_0(x_k)$.

Example 4.8: Obtain the 4th degree interpolating polynomial to exp(2x) over (–1,1) by generating 5 data points at zeroes of the Legendre and Tchebycheff polynomials, respectively. Estimate the value of the function at the point x=0.85.

Solution:

For a 4th degree polynomial, we need the Legendre polynomials up to the order 5, which are listed below:

$$P_0(x) = 1, P_1(x) = x, P_2(x) = \frac{1}{2}\left(3x^2 - 1\right), P_3(x) = \frac{1}{2}\left(5x^3 - 3x\right)$$

$$P_4(x) = \frac{1}{8}\left(35x^4 - 30x^2 + 3\right), P_5(x) = \frac{1}{8}\left(63x^5 - 70x^3 + 15x\right)$$

The grid points for the data generation are located at the zeroes of $P_5(x)$ which are obtained as 0,±0.538469, ±0.906180. The weights (Eq. 4.33) are obtained as

$$w_k = \frac{2}{\left(1 - x_k^2\right) \frac{1}{64} \left[315x_k^4 - 210x_k^2 + 15\right]^2}$$

The interpolating polynomial is written as $f_4(x) = \sum_{i=0}^{4} c_i P_i(x)$ and the coefficients are obtained in the table below [Note: The values of Legendre polynomials at a grid point may be directly computed by using the expressions listed above. However, it is more efficient to use the recursive relation, Eq. (4.11), to compute P_2, P_3, and P_4] :

k	x_k	$f(x_k)$	w_k	$P_0(x_k)$	$P_1(x_k)$	$P_2(x_k)$	$P_3(x_k)$	$P_4(x_k)$
0	-0.906180	0.163268	0.236927	1.00000	-0.906180	0.731743	-0.501031	0.245735
1	-0.538469	0.340637	0.478629	1.00000	-0.538469	-0.065076	0.417382	-0.344501
2	0.000000	1.00000	0.568889	1.00000	0.000000	-0.500000	0.000000	0.375000
3	0.538469	2.93568	0.478629	1.00000	0.538469	-0.065076	-0.417382	-0.344501
4	0.906180	6.12488	0.236927	1.00000	0.906180	0.731743	0.501031	0.245735
			$\Sigma f(x_k)P_i(x_k)w_k=$	3.62686	1.94876	0.703681	0.189276	0.039213
			(From Eq. 4.35) $c_i=$	1.81343	2.92314	1.75920	0.662465	0.176458

The value of the function at x=0.85 is obtained by computing the Legendre polynomials at this point as $P_0 = 1, P_1 = 0.85, P_2 = 0.583750, P_3 = 0.260313, P_4 = -0.050598$. We then have $f_4(0.85)$=1.81343×1+2.92314×0.85+1.75920×0.583750+0.662465×0.260313−0.176458×0.050598 =5.48855. The exact value of the function is exp(1.7)=5.47395 indicating an error of only about 0.27%.

For the Tchebycheff polynomial, the required polynomials are listed below:
$T_0(x) = 1, T_1(x) = x, T_2(x) = 2x^2 - 1, T_3(x) = 4x^3 - 3x, T_4(x) = 8x^4 - 8x^2 + 1, T_5(x) = 16x^5 - 20x^3 + 5x$
The zeroes of $T_5(x)$ are obtained as 0,±0.587785, ±0.951057. The coefficients are computed as shown in the table below:

k	x_k	$f(x_k)$	$T_0(x_k)$	$T_1(x_k)$	$T_2(x_k)$	$T_3(x_k)$	$T_4(x_k)$
0	-0.951057	0.149253	1.00000	-0.951057	0.809017	-0.587785	0.309017
1	-0.587785	0.308643	1.00000	-0.587785	-0.309017	0.951057	-0.809017
2	0.000000	1.000000	1.00000	0.000000	-1.00000	0.000000	1.00000
3	0.587785	3.239991	1.00000	0.587785	-0.309017	-0.951057	-0.809017
4	0.951057	6.700037	1.00000	0.951057	0.809017	0.587785	0.309017
		$\Sigma f(x_k)T_i(x_k)=$	11.3979	7.95317	3.44460	1.06258	0.245642
		(From Eq. 4.38) $c_i=$	2.27958	3.18127	1.37784	0.425031	0.0982568

The value of the function at x=0.85 is obtained by computing the Tchebycheff polynomials at this point as $T_0 = 1, T_1 = 0.85, T_2 = 0.445, T_3 = -0.0935, T_4 = -0.60395$. We then have
$f_4(0.85) = 2.27958 \times 1 + 3.18127 \times 0.85 + 1.37784 \times 0.445 - 0.425031 \times 0.0935 - 0.0982568 \times 0.60395$ =5.49772, indicating an error of only about 0.43%. Note that the error is a little larger than that obtained using the Legendre polynomials. One of the reasons may be that the point x=0.85 is closer to the grid point of Legendre polynomial (0.906180) than that of the Tchebycheff polynomial (0.951057).

As discussed earlier, if one is not at a freedom to choose the grid points, use of higher order interpolating polynomials may result in large errors near the end points. In this case, it may be

better to use piecewise interpolation[1], i.e., using a smaller degree polynomial to pass through a *subset* of data. For example, in Runge's problem (see Fig. 4.7a), if we connect consecutive data points by straight lines, we get a much superior interpolating polynomial than the 10^{th} degree polynomial passing through all data points. These types of interpolating polynomials are known as *splines* and are described next.

Exercise 4.4

1. Estimate the value of the function at $x = 4$ from the table of data given below, using (a) Lagrange interpolating polynomial of 2^{nd} order using x=2,3,5 (b) Newton's interpolating polynomial of 4^{th} order, and (c) Gram's polynomial of 2^{nd} order using x=1,3,5.

x	***f(x)***
1	1
2	12
3	54
5	375
6	756

2. The function $f(x) = 0.5\left(x^3 + x^4\right)$ is to be approximated over the range (1,6) using a 2^{nd} order polynomial. Use the discrete Legendre and Tchebycheff polynomials to obtain the approximation and estimate the value of the function at x=4.
3. The function x^x has to be integrated over the interval (0,1). Since analytical integration is not possible, it is decided to approximate the function by a 3^{rd} degree polynomial and then perform an analytical integration of the approximating polynomial. Estimate the value of the integral $\int_0^1 x^x dx$ using both Gram's polynomials and discrete Tchebycheff polynomials by generating 4 data points, obtaining the approximating polynomial, and integrating it.
4. The function $f(x) = \cos x$ is to be approximated over the interval (0, π/4) by a 2^{nd} order interpolating polynomial by using (i) Newton's divided difference with three points (0, π/8, π/4) and (ii) Taylor's series expansion about π/8. Obtain the interpolating polynomials for both methods, write the error of interpolation in terms of the 3^{rd} derivative of the function and estimate the maximum possible error over the interval (0, π/4). Find the true error at x=0.4 and 0.75 and give reasons for the comparative performance of the two methods at these points.

4.5 Spline Interpolation

Spline interpolation uses piecewise polynomial fitting to the set of data points. For easier presentation, we now assume that the grid points $x_0, x_1, \ldots x_n$ are arranged in increasing order of x values. The grid points are also called *nodes* or *knots* (x_0 and x_n would be the *corner* nodes and the others are called *interior* nodes) and the portion between two consecutive nodes is

[1] Another use of piecewise interpolation is in the solution of nonlinear equations, where a few function values in the neighbourhood of the *likely root* are utilized to perform an *inverse interpolation*, i.e., to express *x* as a polynomial of *f(x)*. From this inverse interpolating polynomial, the root is directly estimated by putting *f(x)*=0.

called a *segment* or *knot span*. There would thus be *n+1* knots and *n* spans. Our objective is then to obtain the splines S_i, which are polynomials of some pre-selected order, for the i^{th} segment (i=0 to n-1[1]) such that it passes through the nodes on either end of the segment (x_i and x_{i+1}). The degree of these polynomials is used to classify the spline as *linear*, *quadratic*, or *cubic* spline[2]. Linear splines are the simplest and are uniquely defined by the data given whereas, as we will see in a short while, the quadratic and cubic splines require specification of additional constraints for a unique definition.

Linear Splines

For the i^{th} segment, function value at the nodes at either end, $f(x_i)$ and $f(x_{i+1})$, uniquely define the linear spline as

$$S_i(x) = f(x_i) + (x - x_i) f[x_{i+1}, x_i] \qquad i = 0,1,2,...,n-1 \tag{4.39}$$

The computations are therefore quite straightforward. Prediction of the function value at any point within the segment requires the determination of the relevant segment and the use of the corresponding spline. The drawback, as can be seen in Fig. 4.8, is that the curve is not smooth[3].

Figure 4.8 Spline fitting using a linear spline

A number of times we are interested in finding the first or second derivatives from the observed data (e.g., computing velocity or acceleration from distance measurements). While the first derivative of the linear spline fit would show a discontinuity at the knots, the second derivative is zero within the segment and is not defined at the knots. The linear splines are, therefore, not of much practical interest.

Quadratic Splines

The discussion in the previous paragraph implies that even a quadratic polynomial would not be very practical since it would not have a continuous second derivative at the knots. However, we briefly discuss these to introduce the concept of *degree of freedom* and *constraint*. A linear spline for the i^{th} segment, say $c_{0,i} + c_{1,i}x$, had two undetermined constants which were uniquely defined by the function value at the nodes at either end. A quadratic

[1] i=1 to n may also be used. In that case, the nodes at the end of the i^{th} segment would be x_{i-1} and x_i.
[2] Generally polynomials of higher order than cubic are not used due to their computational complexity
[3] We assume that the function which has generated the data is "smooth." However, even if the function has a finite jump discontinuity at the knots, the spline fitting, because of its piecewise polynomial nature, could be made to work. In all subsequent discussion, though, we assume the function to be sufficiently smooth.

spline has three coefficients while the equality of the spline interpolant and function values at the end nodes provides only two conditions. Thus we have a *degree of freedom* of one in each segment, for a total of *n*, and need to provide *n* additional *constraints* in order to get a unique definition of the spline. One obvious constraint is the continuity of the first derivative (C^1 continuity) at the *interior* nodes, i.e., $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1})$ for i=0,1,2,...,n – 2, which provides n–1 constraints, leaving one degree of freedom. The additional constraint required to uniquely define the quadratic spline may be chosen on the basis of any available information about the function behaviour. For a general case, some commonly used options are listed below:

- An arbitrary free parameter, say t_0, may be chosen and the quadratic spline may be expressed in terms of this parameter. By changing this parameter and looking at the resulting interpolating polynomial, one may decide the *proper* value of t_0. Considering two consecutive segments and using the C^0 continuity at the three knots and C^1 continuity at the middle knot, the quadratic spline may be written as:

$$S_i(x) = f(x_i) + (x - x_i)t_i + (x - x_i)^2 \frac{t_{i+1} - t_i}{2h_i} \qquad i = 0,1,2,...,n-1 \qquad (4.40)$$

  in which $h_i\,(= x_{i+1} - x_i)$ is the length of the i^{th} segment (note that t_i represents the first derivative of the quadratic spline at $x=x_i$). The coefficient of the quadratic term is obtained by the continuity of the first derivative $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1})$, and the $t^{'s}$ satisfy the recurrence relationship obtained from the condition, $S_i(x_{i+1}) = f(x_{i+1})$, as

$$t_{i+1} = 2f[x_{i+1}, x_i] - t_i \qquad i = 0,1,2,...,n-1 \qquad (4.41)$$

  If, it is known that the first derivative is zero at $x=x_0$ (for example, when the function represents distance at various times and it is known that the initial velocity is zero), t_0 is equal to zero and all other $t^{'s}$, and therefore S_i, are readily computed.
- A zero second derivative at the first[1] corner node: $S_0''(x_0) = 0$. This implies that the 0^{th} segment is a straight line joining $f(x_0)$ to $f(x_1)$, which is readily computed. The first segment, then, has three constraints, viz., the two function values at x_1 and x_2 and the first derivative at x_1 (which should be equal to $f[x_1,x_0]$ due to the C^1 continuity), that can be used to obtain the three coefficients in $S_1(x) = c_{0,1} + c_{1,1}x + c_{2,1}x^2$. Each successive segment is similarly computed. The same computation may be done by using the recurrence relation listed above (Eq. 4.41), with $t_0=f[x_1,x_0]$.
- If the function is a 2^{nd} degree polynomial, one may expect a quadratic spline to fit it exactly. However, this does not happen if the first segment is specified as linear. The not-a-knot condition is an option which does not suffer from this drawback. It requires C^2 continuity at the first interior node: $S_0''(x_1) = S_1''(x_1)$, which implies that $S_0(x) \equiv S_1(x)$, hence the name not-a-knot. The three function values; $f(x_0)$, $f(x_1)$, and $f(x_2)$; are used to obtain the three coefficients $c_{0,0}\,(= c_{0,1}), c_{1,0}\,(= c_{1,1})$, and $c_{2,0}\,(= c_{2,1})$. Successive segments then have three constraints as in the previous case (function values at x_i and x_{i+1} and the first derivative at x_i) and are sequentially computed. Alternatively, one could use the recurrence relationship with the continuity of the second derivative at the first interior node resulting in

[1] Any other node could also be chosen but the computations become more complicated.

$$\frac{t_1 - t_0}{h_0} = \frac{t_2 - t_1}{h_1}$$

which becomes, after using the recursion (Eq. 4.41),

$$t_0 = f\left[x_1, x_0\right] - h_0 f\left[x_2, x_1, x_0\right]$$

Example 4.9: Obtain the quadratic spline which fits $f(x)=x^2$ sampled at points x=0,1,2, and 3.

Solution:

We first use t_0 as an arbitrary parameter and obtain the corresponding quadratic splines. The following table shows the computations:

i	x_i	$f(x_i)$	$f[x_{i+1},x_i]$	Computed t_i from Eq. (4.41) for		
				t_0=0	t_0=1	t_0=2
0	0	0	1	0	1	2
1	1	1	3	2	1	0
2	2	4	5	4	5	6
3	3	9		6	5	4

Fig. 4.9 shows the plots of the three quadratic splines using Eq. (4.40) with the *t* values computed above.

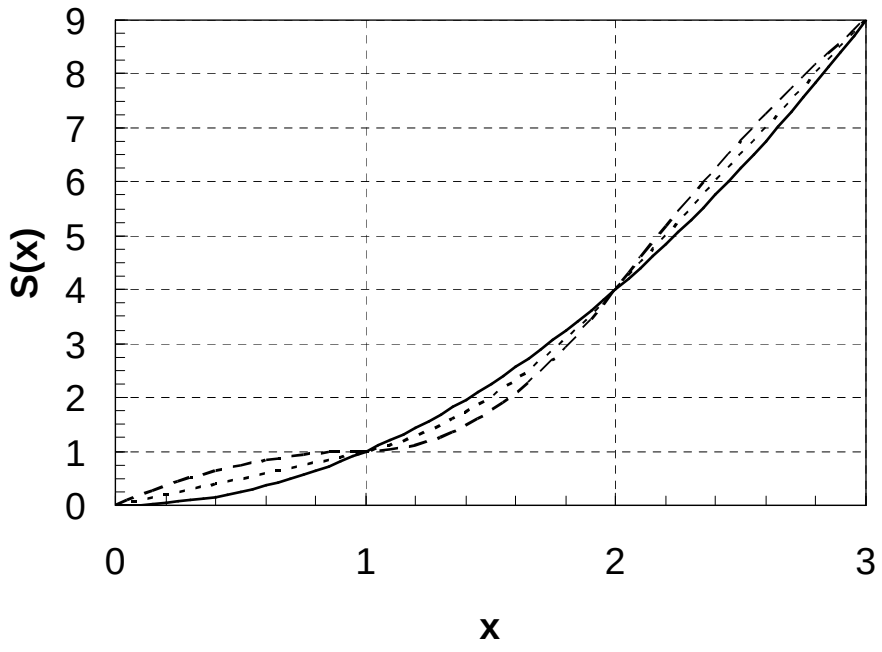


Figure 4.9 Quadratic spline interpolation using different values of the slope at the first point, t_0. Solid line – t_0=0, Dotted line – t_0=1, Dashed line – t_0=2.

From the figure, it appears that the solid line (t_0=0) is a better interpolating spline as it is more smooth compared to the other curves. However, if we do not have any information about the function, there is no reason to prefer one of these curves over the others.

We now assume that the second derivative is zero at the first corner point (x_0). This results in the first segment being a straight line and $t_0 = f[x_1, x_0] = 1$. Using the recursive relations, we get the same values as shown in the table above for t_0=1.

Finally, if we apply the not-a-knot condition, we get $t_0 = f[x_1, x_0] - h_0 f[x_2, x_1, x_0]$=1-1 x (3-1)/(2-0)=0. Hence we get the same interpolating spline as shown by the solid line in Fig. 4.9, which, incidentally, matches the exact function.

Now we are ready to discuss the most widely used spline, the cubic spline.

Cubic Splines

For a cubic spline, the degree of freedom is 2 for each segment since there are 4 coefficients and only two constraints of the function value at either end. Again, a logical constraint would be to impose the C^2 continuity at all the *internal* nodes which provides us with 2n–2 constraints, one for the first derivative and the other for the second derivative. The system will then have 2 degrees of freedom and, therefore, two additional constraints have to be provided to define the unique cubic spline. We saw earlier that a unique interpolating polynomial could be expressed in different forms. Similarly, once the two additional constraints are imposed, the unique cubic spline could also be expressed in different forms. We first discuss some ways of specifying the additional constraints and then look at a few options for expressing (and computing) the resulting cubic polynomial.

Constraints

- Clamped: When the function is clamped on each corner node forcing both ends to have some fixed slope, say, s_0 and s_n. This implies $S_0'(x_0) = s_0$ and $S_{n-1}'(x_n) = s_n$.
- Natural: Curvature at the corner nodes is zero, i.e., $S_0''(x_0) = S_{n-1}''(x_n) = 0$
- Cyclic or periodic: When the function is cyclic, with x_0 and x_n corresponding to the beginning and end of a cycle, respectively [necessarily, then, $f(x_0)=f(x_n)$]. The additional constraints are: $S_0'(x_0) = S_{n-1}'(x_n)$ and $S_0''(x_0) = S_{n-1}''(x_n)$
- Not-a-knot: When the first and last interior nodes have C^3 continuity, i.e., these do not act as a knot. Thus $S_0(x) \equiv S_1(x)$ and $S_{n-2}(x) \equiv S_{n-1}(x)$

Different forms

Similar to the quadratic spline, we may write

$$S_i(x) = f(x_i) + (x - x_i)u_i + (x - x_i)^2 v_i + (x - x_i)^3 \frac{v_{i+1} - v_i}{3h_i} \qquad i = 0,1,2,...,n-1 \qquad (4.42)$$

in which the coefficient of the cubic term is obtained by using the C^2 continuity at x_{i+1}. The u's and v's (note that u_i represents the first derivative of the spline at $x=x_i$ and v_i is one-half of the second derivative) are given by the recursive relations (obtained by the C^1 and C^0 continuity, respectively):

$$v_{i+1} = \frac{u_{i+1} - u_i}{h_i} - v_i \qquad \text{for i=0,1,2,…,n−1} \qquad (4.43)$$

$$u_{i+1} = 3f\left[x_{i+1}, x_i\right] - 2u_i - h_i v_i$$

Unlike the quadratic spline, however, sequential computations of the coefficients *u* and *v* will generally not be possible. In some cases both u_0 and v_0 may be specified (for example, when the function represents distance, both the initial velocity and acceleration being zero will lead to $u_0=v_0=0$) and then the spline is readily obtained using the recursive relations to compute (in the order) $u_1,v_1,u_2,v_2\ldots,u_n,v_n$ [note that v_n is needed in $S_{n-1}(x)$]. However, as seen in the previous paragraph, typically the constraints are specified as one at each corner. For example, for a clamped spline, $u_0 = s_0$ and $u_n = s_n$; for a natural spline, $v_0 = v_n = 0$; for a cyclic spline, $u_0 = u_n$ and $v_0 = v_n$; and for the not-a-knot condition, $\frac{v_1 - v_0}{h_0} = \frac{v_2 - v_1}{h_1}$ and $\frac{v_{n-1} - v_{n-2}}{h_{n-2}} = \frac{v_n - v_{n-1}}{h_{n-1}}$. In all these cases, a set of linear simultaneous equation is obtained which can be solved using any of the techniques described in Chapter 2. The recursive form listed in Eq. (4.43) gives rise to a system of 2n equations in 2n+2 unknowns[1], but may be easily modified to obtain a *tridiagonal* system of *n–1* equations involving *n+1* unknowns. Two commonly used methods for achieving this are discussed below.

(a) *Using the second derivatives as primary variables*:

From the recursive relations, we may express u in terms of v as

$$u_i = f\left[x_{i+1}, x_i\right] - \frac{h_i\left(v_{i+1} + 2v_i\right)}{3} \qquad i = 0,1,2,...,n-1 \qquad (4.44)$$

and then write a tridiagonal system of equations as (Box 4.4)

$$h_{i-1}v_{i-1} + 2\left(h_{i-1} + h_i\right)v_i + h_i v_{i+1} = 3f\left[x_{i+1}, x_i\right] - 3f\left[x_i, x_{i-1}\right] \qquad i = 1,2,...,n-1 \qquad (4.45)$$

which can be solved using the Thomas algorithm utilising the two constraints.

(b) *Using the first derivatives as primary variables*:

As is clear from the above formulation, the second derivatives at the knots are obtained directly as part of the solution process. However, if one needs to find the first derivatives, differentiation of the cubic spline has to be performed. So if, for example, from the distance measurement at different times, it is desired to estimate the acceleration at these times, this formulation would work well. However, it is more likely that the first derivative at the nodes will be looked-for more often than the second derivative. We describe, therefore, an alternative form of the cubic spline

[1] The two boundary conditions enable us to get a *unique* solution.

which will be more suitable for such cases. From the recursive relations, we may express v in terms of u as

$$v_i = \frac{3f\left[x_{i+1}, x_i\right] - 2u_i - u_{i+1}}{h_i} \qquad i = 0,1,2,...,n-1 \qquad (4.46)$$

and then write a tridiagonal system of equations (for i=1 to n–1) as (Box 4.4)

$$h_i u_{i-1} + 2\left(h_{i-1} + h_i\right)u_i + h_{i-1}u_{i+1} = 3h_{i-1}f\left[x_{i+1}, x_i\right] + 3h_i f\left[x_i, x_{i-1}\right] \qquad (4.47)$$

which can again be solved using the Thomas algorithm utilising the two constraints.

The two end conditions have to be expressed in terms of the primary variables (u or v). Thus a natural spline or the not-a-knot condition does not pose any problem for the form (a) while a clamped spline is easily accounted for in the form (b). While using the natural spline with the form (b), the condition $v_0 = 0$ could be written using Eq. (4.46) as

$$2u_0 + u_1 = 3f\left[x_1, x_0\right]$$

while $v_n = 0$ translates into [using Eqs. (4.43)]

$$u_{n-1} + 2u_n = 3f\left[x_n, x_{n-1}\right]$$

Similarly, for a clamped spline with the form (a), the following could be written

$$2v_0 + v_1 = \frac{3f\left[x_1 - x_0\right] - 3s_0}{h_0}$$

$$v_{n-1} + 2v_n = \frac{3s_n - 3f\left[x_n, x_{n-1}\right]}{h_{n-1}}$$

Once the two additional constraints are specified, the cubic spline is uniquely defined. Which form of the cubic spline to use in a particular problem would depend on several factors like the ease of applying boundary conditions, requirement of first or second derivatives, and the data storage and computational efficiency requirements. For example, the recursive relations Eqs. (4.43) require larger storage (both u and v) but will be more computationally efficient. The other forms require less storage (either S' or S'') but are likely to require more computation time. These issues are not addressed here.

Box 4.4: **Alternative methods of obtaining Cubic Spline equations**

Equations (4.45) could also be obtained by starting from the second derivatives as primary variables, expressing them as linear function of x (which is the linear interpolation between S_i'' and S_{i+1}''), integrating twice, applying the C^0 continuity conditions at the two nodes to evaluate the constants of integration, and using the C^1 continuity to obtain the tridiagonal system of equations. The resulting equations are generally written as

$$\left(x_i - x_{i-1}\right)S_{i-1}'' + 2\left(x_{i+1} - x_{i-1}\right)S_i'' + \left(x_{i+1} - x_i\right)S_{i+1}'' = 6\frac{f\left(x_{i+1}\right) - f(x_i)}{x_{i+1} - x_i} - 6\frac{f\left(x_i\right) - f(x_{i-1})}{x_i - x_{i-1}}$$

and

$$S_i(x)=\frac{S''_{i+1}\left(x-x_i\right)^3+S''_i\left(x_{i+1}-x\right)^3}{6\left(x_{i+1}-x_i\right)}+\left[\frac{f\left(x_{i+1}\right)}{x_{i+1}-x_i}-\frac{\left(x_{i+1}-x_i\right)S''_{i+1}}{6}\right]\left(x-x_i\right)$$
$$+\left[\frac{f\left(x_i\right)}{x_{i+1}-x_i}-\frac{\left(x_{i+1}-x_i\right)S''_i}{6}\right]\left(x_{i+1}-x\right)$$

Similarly, Equations (4.47) could be obtained by starting from the first derivatives as primary variables, expressing them as a quadratic function of x (with an undetermined coefficient, since only two conditions are available: the quadratic function should be equal to S'_i at $x=x_i$ and S'_{i+1} at $x=x_{i+1}$), integrating once, applying the C^0 continuity conditions at the two nodes to evaluate the constants of integration, and using the C^1 continuity to obtain the tridiagonal system of equations. The resulting equations are usually written as

$$\left(x_{i+1}-x_i\right)S'_{i-1}+2\left(x_{i+1}-x_{i-1}\right)S'_i+\left(x_i-x_{i-1}\right)S'_{i+1}=3\left(x_i-x_{i-1}\right)\frac{f\left(x_{i+1}\right)-f(x_i)}{x_{i+1}-x_i}$$
$$+3\left(x_{i+1}-x_i\right)\frac{f\left(x_i\right)-f\left(x_{i-1}\right)}{x_i-x_{i-1}}$$

and

$$S_i(x)=\frac{x_{i+1}-x}{x_{i+1}-x_i}f\left(x_i\right)+\frac{x-x_i}{x_{i+1}-x_i}f\left(x_{i+1}\right)$$
$$+\frac{\left(x-x_i\right)\left(x_{i+1}-x\right)}{\left(x_{i+1}-x_i\right)^2}\left\{\left(x_{i+1}-x\right)S'_i-\left(x-x_i\right)S'_{i+1}-\left(x_{i+1}-2x+x_i\right)\left[f\left(x_{i+1}\right)-f\left(x_i\right)\right]\right\}$$

Example 4.10: Obtain the natural cubic spline which interpolates $f(x)=1/(1+x^2)$ sampled at points x=-2,-1,0,1, and 2 by using the second derivatives as the primary variables. Estimate the function value at x=1.6.

Solution:

The following table shows the function values and the first divided differences:

i	x_i	$f(x_i)$	$f[x_{i+1},x_i]$
0	-2	0.2	0.3
1	-1	0.5	0.5
2	0	1.0	-0.5
3	1	0.5	-0.3
4	2	0.2	

Eq. (4.45) is then written as (note that $v_0=v_4=0$ for a natural spline)

$$\begin{bmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{bmatrix} \begin{Bmatrix} v_1 \\ v_2 \\ v_3 \end{Bmatrix} = \begin{Bmatrix} 0.6 \\ -3 \\ 0.6 \end{Bmatrix}$$

giving the solution as $v_1=v_3=0.385714$, $v_2= -0.942857$. Eq. (4.44) is used to obtain the coefficients u as $u_0=0.171429, u_1=0.557143, u_2=0, u_3= -0.557143$. Eq. (4.42) is then used to write the cubic spline in the last segment (which contains the point 1.6) as

$$S_3(x) = f(x_3) + (x-x_3)u_3 + (x-x_3)^2 v_3 + (x-x_3)^3 \frac{v_4 - v_3}{3h_3}$$

$$= 0.5 - 0.557143(x-1) + 0.385714(x-1)^2 - 0.128571(x-1)^3$$

and the function value at x=1.6 is estimated as 0.276800. The true value at this point is 0.280899 indicating an error of about 1.5%.

Hermite interpolation

Till now we have assumed that the *function* values are given at a set of grid points. Sometimes, however, additional information about the function may be available at these points in terms of its derivatives. For example, if we are tracking the path of an object, in addition to the distance measured at different times, we may have some velocity measurements also. This extra information will usually lead to a more accurate prediction of the position of the object at a time not coinciding with the grid points. It will enable us to use a higher degree polynomial than that possible when only the function values are given. Thus, given the function value at a single data point we use a zero-order interpolation (a constant). But if we are given its derivative also, we will be able to use a linear interpolation with a line passing through the given function value and having a slope equal to the given derivative. Similarly, if the function and its first derivatives are given at two points, a cubic parabola[1] may be used as the interpolating polynomial. This method of interpolation, using the function and its derivative(s)[2], is known as the Hermite interpolation or osculating interpolation.

Based on the fact that *m+1* data points can generally be interpolated with an m^{th} degree polynomial, we may surmise that given the values of the function and its first derivative at

[1] If we use linear interpolation, the slope of this line at the grid points may not match the values given. Similarly, if we use a parabola passing through the two given points, there are infinite possibilities but only one of them will have the required slope at the, say, first point and it may not match the slope at the second point. If we use a 4^{th} degree polynomial, there would be infinite possibilities which will satisfy all four given data.

[2] Higher order derivatives may also be used, but we will only consider the first derivative.

n+1 points, there exists a unique interpolating polynomial of degree (at most) *2n+1*. For convenience, we would express it in a form similar to the Lagrange polynomials and, without loss of generality, as we did for orthogonal polynomials, we will assume that the range of data points is (–1,1)[1].

Given the set of values $\left\{\left[x_k,\, f\left(x_k\right),\, f'\left(x_k\right)\right],\ k=0,1,...,n\right\}$ we write the general form of the Hermite interpolating formula as

$$f_{(d+1)(n+1)-1}(x)=\sum_{i=0}^{n}\sum_{j=0}^{d}H_{i,j}\,f^{\,j}\left(x_i\right) \tag{4.48}$$

in which *d* represents the order of derivatives specified at each point (if the first derivatives are specified, *d=1*, if the second derivatives are *also* specified, *d=2*), $f^{\,j}\left(x_i\right)$ denotes the j^{th} derivative at x_i (*j=0* corresponds to the function value itself), and $H_{i,j}$ are the Hermite polynomials (of degree at most *(d+1)(n+1)-1*). These are obtained from the conditions that the function value and its derivative(s) obtained for the approximating polynomial should be equal to those specified in the problem. In other words, the polynomial $H_{i,0}$ would be equal to 1 at $x=x_i$ and 0 at all other grid points and its derivative $H'_{i,0}$ would be zero at all grid points. Similarly, the polynomial $H_{i,1}$ would be zero at all grid points and its derivative will be 1 at $x=x_i$ and will vanish at all other grid points. Since we consider only the cases where the function and its first derivatives are given, we write

$$f_{2n+1}(x)=\sum_{i=0}^{n}\left[H_{i,0}\,f\left(x_i\right)+H_{i,1}\,f'\left(x_i\right)\right] \tag{4.49}$$

For example, for two points (*n=1*, $x_0=-1$, $x_1=1$) we have

$$f_3(x)=H_{0,0}\,f\left(x_0\right)+H_{1,0}\,f\left(x_1\right)+H_{0,1}\,f'\left(x_0\right)+H_{1,1}\,f'\left(x_1\right)$$

where all $H^{'s}$ are cubic polynomials. As discussed in the previous paragraph, $H_{0,0}$ will satisfy the following:

$$H_{0,0}(-1)=1 \quad H'_{0,0}(-1)=0 \quad H_{0,0}(1)=0 \quad H'_{0,0}(1)=0$$

One could write a cubic polynomial for $H_{0,0}$ in the conventional form, obtain its four coefficients using the conditions above and get

$$H_{0,0}=\frac{1}{2}-\frac{3x}{4}+\frac{x^3}{4}$$

However, it will be inefficient for large number of grid points. An alternative technique based on the Lagrange form of interpolating polynomials is described below.

Since the Lagrange polynomials, L_i (see Eq. 4.21) are n^{th} degree polynomials which are 1 at the corresponding node (x_i) and 0 at all other nodes, if we square them we get a polynomial of degree *2n* which will not only preserve the nodal values of 1 and 0 but the first derivative will also vanish at all nodes other than x_i. The value of the first derivative at $x=x_i$ would be equal to $2L_i\left(x_i\right)L'_i\left(x_i\right)$. Since the Hermite polynomials are of degree *2n+1*, and essentially have similar properties, we should be able to write (*j=0,1*)

$$H_{i,j}=\left(\alpha_{i,j}+\beta_{i,j}x\right)\left[L_i(x)\right]^2$$

[1] Which implies that we are not considering the trivial case of a single grid point, i.e., *n* has to be greater than 0.

with the conditions

$$\alpha_{i,0}+\beta_{i,0}x_i=1 \qquad 2\left(\alpha_{i,0}+\beta_{i,0}x_i\right)L_i'(x_i)+\beta_{i,0}=0$$
$$\alpha_{i,1}+\beta_{i,1}x_i=0 \qquad 2\left(\alpha_{i,1}+\beta_{i,1}x_i\right)L_i'(x_i)+\beta_{i,1}=1$$

Therefore, the expressions for Hermite polynomials are obtained as

$$\begin{aligned} H_{i,0} &= \left[1-2\left(x-x_i\right)L_i'(x_i)\right]\left[L_i(x)\right]^2 \\ H_{i,1} &= \left(x-x_i\right)\left[L_i(x)\right]^2 \end{aligned} \tag{4.50}$$

For example, with two points $\left(x_0=-1, x_1=1\right)$ we have $L_0(x)=\frac{1-x}{2}, L_1(x)=\frac{1+x}{2}$ and the derivatives as $L_0'(x)=-\frac{1}{2}, L_1'(x)=\frac{1}{2}$. From Eq. (4.50), therefore,

$$H_{0,0}=\left[1-2(x+1)\left(-\frac{1}{2}\right)\right]\left(\frac{1-x}{2}\right)^2=\frac{1}{2}-\frac{3x}{4}+\frac{x^3}{4}$$
$$H_{1,0}=\left[1-2(x-1)\left(\frac{1}{2}\right)\right]\left(\frac{1+x}{2}\right)^2=\frac{1}{2}+\frac{3x}{4}-\frac{x^3}{4}$$
$$H_{0,1}=(x+1)\left(\frac{1-x}{2}\right)^2=\frac{1}{4}\left(1-x-x^2+x^3\right)$$
$$H_{1,1}=(x-1)\left(\frac{1+x}{2}\right)^2=\frac{1}{4}\left(-1-x+x^2+x^3\right)$$

The interpolated value at the midpoint (x=0) is therefore

$$\tilde{f}\left(\text{midpoint}\right)=\frac{f(x_0)}{2}+\frac{f(x_1)}{2}+\frac{f'(x_0)}{4}-\frac{f'(x_1)}{4}$$

Note that the first two terms on the r.h.s. represent the linear interpolation which would be used in absence of any data on the function derivatives. The last two terms represent a correction from the linear trend, increasing the value for a curve which tends to get flatter with increase in *x* and reducing it if the curve becomes steeper.

The error of interpolation could be obtained by following a methodology similar to that used for polynomial interpolation (Eq. 4.27). However, we may visualize the Hermite interpolation as the limiting case of a polynomial interpolation using only the function values at the grid points and points located at an infinitesimal distance from the grid points[1], i.e., the data set

$$\left\{\left(x_i, f(x_i)\right), \left(x_i+\varepsilon, f(x_i)+\varepsilon f'(x_i)\right)\right\}$$

i=0,1,…,n. The remainder term is then obtained as

$$\begin{aligned} R &= \lim_{\varepsilon\to 0}\prod_{i=0}^{n}\left(x-x_i\right)\left(x-x_i-\varepsilon\right) f\left[x, x_n+\varepsilon, x_n, x_{n-1}+\varepsilon, x_{n-1}, \ldots, x_0+\varepsilon, x_0\right] \\ &= \frac{f^{2n+2}(\xi)}{(2n+2)!}\prod_{i=0}^{n}\left(x-x_i\right)^2 \end{aligned} \tag{4.51}$$

[1] We could use this methodology to obtain the expressions for the Hermite polynomials also but leave it to the reader to do so, if desired.

with □ having its usual meaning of being in the relevant interval.

Similar to the spline interpolation, we could use a piecewise-cubic Hermite interpolation if the function and its derivatives are given at the nodes. It will have continuous first derivatives at the nodes but the second derivative may not be continuous since each cubic polynomial may be obtained independent of neighbouring segments. We could achieve continuity of the second derivative at the knots by using higher order interpolating polynomial, but it is not very common to do so.

One of the reasons for choosing a piecewise polynomial fitting is the presence of large oscillations in a single polynomial fit over the whole range. However, even the cubic polynomial suffers from large oscillations, though less frequently. If the data is subjected to measurement errors in the function value (or the independent variable) an interpolating polynomial would not work very well since it will pass through the "erroneous" points also. We would then like to go for some approximating function (again, generally polynomial) which represents the general trend of the data but does not necessarily pass through *all* the data points. This is achieved by *regression* as described next.

Exercise 4.5

1. Estimate the value of the function at $x = 4$ from the table of data given below, using (a) Quadratic spline (assume that the derivative of the function is zero at x=0) and (b) Cubic spline (using the not-a-knot condition at either ends).

x	***f(x)***
1	1
2	12
3	54
5	375
6	756

2. The location of an object and its velocity were measured at different times as shown in the table below:

t (s)	***Distance (m)***	***Velocity (m/s)***
1	1	3.5
2	12	22

Use Hermite interpolation to approximate the distance as a cubic polynomial in time.

4.6 Regression

The problem can be stated as: given the function values at a set of n+1 points (*not necessarily distinct*[1]), find the polynomial[2] (of degree m<n) which is *nearest*[1] to the function value at

[1] Since we allow for the possibility of measurement errors, it is now permissible to have two different function values at the same node.
[2] Some other types of approximating functions may also be used and are described latter.

these points. In general, this polynomial will depend on the selected measure of nearness. However, if the data is such that an m^{th} degree polynomial is able to interpolate it, e.g., three points on a straight line (n=2, m=1), the regressing and interpolating polynomials will be identical (*and unique*). In the next subsection we describe the least-squares regression, which aims at minimising the L_2 norm of the residuals.

Least-squares regression

Let $(x_0,f(x_0))$, $(x_1,f(x_1))$,….$(x_n,f(x_n))$ be the set of function values at corresponding n+1 grid points over the interval x=a to x=b. We need to find the m^{th} degree polynomial $f_m(x)$ such that the sum of squared residuals at the grid points is a minimum. As before, if the desired polynomial is

$$f_m(x) = \sum_{j=0}^{m} c_j \phi_j(x) \tag{4.52}$$

the c_j are to be chosen such that $\sum_{i=0}^{n}\left(f(x_i) - \sum_{j=0}^{m} c_j \phi_j(x_i) \right)^2$ is a minimum[2]. Using the stationary point theorem[3], we get

$$\sum_{i=0}^{n} \phi_k(x_i)\left(f(x_i) - \sum_{j=0}^{m} c_j \phi_j(x_i) \right) = 0 \text{ for k} = 0,1,...,m \tag{4.53}$$

a set of m+1 *linear* simultaneous equations[4] of the form

$$[A]\{c\} = \{b\} \tag{4.54}$$

in which $a_{ij} = \sum_{k=0}^{n} \phi_i(x_k)\phi_j(x_k)$ and $b_i = \sum_{k=0}^{n} \phi_i(x_k) f(x_k)$. Eqs 4.54 are the *normal equations* (see section 4.2) and may be solved to obtain the coefficients $c_0,c_1…c_m$. For example, if the bases, ϕ, are chosen in the conventional form $\phi_i(x) = x^i$, we get the normal equation

---

[1] As we did in the continuous case, some measure of *nearness* has to be decided before attempting to obtain the *best* polynomial fit. We will use the L_2 norm (sum of squares) of the residual to quantify the error since it leads to a unique and mathematically convenient solution. While the fit based on L_1 norm (sum of absolute value) may be non-unique and difficult to obtain, that based on the L_∞ norm (maximum absolute value) is also rather tedious. However, it should be mentioned that squaring the residual tends to give more weight to the large residuals. Some other criterion may very well be used. For example, if a straight road is to be constructed near a few villages, sum of the perpendicular distance (or its square) of the villages from the road may be a good *norm* to minimize.

[2] We assume that the independent variable, *x*, is not subject to measurement errors. Regression techniques for such cases, where both the dependent and the independent variables may have errors, are available but are not discussed here.

[3] An inherent assumption here is that the bases, ϕ, do not involve the coefficients, c. Sometimes it may not be possible to separate the bases and the coefficients which leads to *nonlinear regression* which is discussed latter in this chapter.

[4] In terms of function space, Eq. 4.53 indicates that the residual is orthogonal to the basis functions.

$$\begin{bmatrix} n+1 & \sum x_i & \sum x_i^2 & \dots & \sum x_i^m \\ \sum x_i & \sum x_i^2 & \sum x_i^3 & \dots & \sum x_i^{m+1} \\ . & . & . & . & . \\ . & . & . & . & . \\ \sum x_i^m & \sum x_i^{m+1} & \sum x_i^{m+2} & \dots & \sum x_i^{2m} \end{bmatrix} \begin{Bmatrix} c_0 \\ c_1 \\ . \\ . \\ c_m \end{Bmatrix} = \begin{Bmatrix} \sum f(x_i) \\ \sum x_i f(x_i) \\ . \\ . \\ \sum x_i^m f(x_i) \end{Bmatrix} \tag{4.55}$$

in which the summation is over i=0 to n. As before, this system of equations is ill-conditioned and it is preferable to choose an orthogonal basis, for which the coefficient matrix [A] becomes diagonal and the c's are obtained directly. For equidistant grid points, Gram's polynomial (of order m) could be used as the orthogonal basis, while if we are free to locate the points, discrete versions of Legendre polynomial of order m [with grid points at the zeroes of $P_n(x)$] or Tchebycheff polynomial of order m [grid points at zeroes of $T_{n+1}(x)$] could be used. For arbitrarily spaced points, the Givens or Householder transforms or Singular Value Decomposition (see chapter 2) could be used to improve the condition number.

Example 4.11: The mass of a radioactive substance is measured at 2-day intervals till 8 days. Unfortunately, the reading could not be taken at 6 days due to equipment malfunction. The following table shows the other readings:

Time (d)	0	2	4	8
Mass (g)	1.000	0.7937	0.6300	0.3968

Estimate the mass at 6 days using a second order polynomial regression.

Solution:

A quadratic regression is obtained by writing $m = c_0 + c_1 t + c_2 t^2$ and writing the normal equations as

$$\begin{bmatrix} n+1 & \sum t_i & \sum t_i^2 \\ \sum t_i & \sum t_i^2 & \sum t_i^3 \\ \sum t_i^2 & \sum t_i^3 & \sum t_i^4 \end{bmatrix} \begin{Bmatrix} c_0 \\ c_1 \\ c_2 \end{Bmatrix} = \begin{Bmatrix} \sum m_i \\ \sum t_i m_i \\ \sum t_i^2 m_i \end{Bmatrix}$$

The following table shows the computations:

i	t_i	m_i	t^2	t^3	t^4	t m	t^2 m
0	0	1.0000	0	0	0	0.000	0.000
1	2	0.7937	4	8	16	1.587	3.175
2	4	0.6300	16	64	256	2.520	10.080
3	8	0.3968	64	512	4096	3.174	25.395
	Σ=	2.8205	84	584	4368	7.2818	38.65

Solution of the normal equations results in c_0=0.9991, c_1=-11.02, and c_2=0.004370. The estimated mass at t=6 days is, therefore, 0.4951 g.

Since the radioactive decay follows an exponential equation, $m(t) = m(0)e^{-\lambda t}$, a linear regression between ln(m) and t would provide the value of the decay constant which could be used to estimate the mass at 6 days (see Exercise 4.6.1).

Once the least-square polynomial fit is obtained, one would like to ascertain the nearness of the fitted polynomial to the given data points. As shown in Fig. 4.10, we may qualitatively say that the fit (a) is nearer to the data than (b). To quantify it, we describe (very briefly, since we assume reader familiarity with statistical analysis) the most commonly used indicator, the coefficient of determination.

Figure 4.10 Goodness of fit for an approximating function

Coefficient of determination

We denote the spread of data about its mean by S_t, which is the sum of the squares of the deviations from mean:

$$S_t = \sum_{i=0}^{n} \left[f(x_i) - \bar{f} \right]^2 \tag{4.56}$$

where $\bar{f}$ is the mean defined by $\bar{f} = \frac{\sum_{i=0}^{n} f(x_i)}{n+1}$. Further, we use S_r to represent the sum of the squares of the deviations from the best-fit polynomial:

$$S_r = \sum_{i=0}^{n} \left[f(x_i) - f_m(x_i) \right]^2 \tag{4.57}$$

Obviously, since the best-fit polynomial minimizes the sum of squares of the residual, S_r will be less than S_t (or, at worst, they will be equal when the polynomial regression has only the constant term and all higher order coefficients are zero). The difference, $S_t - S_r$, may be thought of as the amount of variability in the data *explained* by the regression and the ratio of the explained variability to the inherent data variability, S_t, is called the coefficient of determination, r^2:

$$r^2 = \frac{S_t - S_r}{S_t} \tag{4.58}$$

and its square root, r, is the correlation coefficient[1]. It varies from zero (when $f_m(x) = \bar{f}$) to 1 (when the polynomial passes through all data points). Typically, a value of r^2 less than 0.3

[1] If a linear relationship is used (i.e., m=1), this correlation coefficient is identical to the Pearson correlation coefficient which measures the strength of a linear relationship between two variables and is given by

indicates a poor fit while more than 0.8 indicates a good fit of the data by the regressing polynomial. Since a polynomial of a particular degree is a superset of all polynomials of a lower dwgree, it is obvious that the fit will improve (or, at worst, remain same) with increase in *m*. Thus a possible (but not very efficient) strategy for polynomial regression would be to start with m=1 and then keep on increasing m till a desirable value of r^2 is achieved. However, if there are large measurement errors or if the data does not follow a polynomial relation, a higher order polynomial regression may produce large oscillations. Thus a large value of the coefficient of determination does not necessarily mean that the regression has *improved*. See exercise ? (linear relation with small errors in data). A plot of the data and the regression curve, therefore, must be looked at before accepting any regression.

The methodology used for a single independent variable (*x*) may be easily extended to the situation where the dependent variable is a function of multiple variables to obtain a *multiple regression*. For example, with two independent variables, say *x* and *y*, we may write a regression polynomial of order m_1 in *x* and m_2 in *y* as

$$f_{m_1,m_2}(x,y)=\sum_{k=0}^{m_2}\sum_{j=0}^{m_1}c_{j,k}x^j y^k \tag{4.59}$$

the $c_{j,k}$ are to be chosen such that $\sum_{i=0}^{n}\left(f(x_i,y_i)-\sum_{k=0}^{m_2}\sum_{j=0}^{m_1}c_{j,k}x_i^j y_i^k\right)^2$ is a minimum where the total number of data points available is *n+1*. Using the stationary point theorem, we get

$$\sum_{i=0}^{n}x_i^{j_1}y_i^{k_1}\left(f(x_i,y_i)-\sum_{k=0}^{m_2}\sum_{j=0}^{m_1}c_{j,k}x^j y^k\right)=0 \text{ for } j_1=0,1,...,m_1 \text{ and } k_1=0,1,...,m_2 \tag{4.60}$$

a set of $(m_1+1)(m_2+1)$ *linear* simultaneous equations which may be solved to obtain the coefficients $c_{j,k}$ for j=0 to m_1 and k=0 to m_2. For example, if we use terms up to the first order, i.e.,

$$f_{1,1}(x,y)=c_{0,0}+c_{1,0}x+c_{0,1}y+c_{1,1}xy$$

the set of equations is written as

$$\begin{bmatrix} n+1 & \sum x_i & \sum y_i & \sum x_i y_i \\ \sum x_i & \sum x_i^2 & \sum x_i y_i & \sum x_i^2 y_i \\ \sum y_i & \sum x_i y_i & \sum y_i^2 & \sum x_i y_i^2 \\ \sum x_i y_i & \sum x_i^2 y_i & \sum x_i y_i^2 & \sum x_i^2 y_i^2 \end{bmatrix}\begin{Bmatrix} c_{0,0} \\ c_{1,0} \\ c_{0,1} \\ c_{1,1} \end{Bmatrix}=\begin{Bmatrix} \sum f(x_i,y_i) \\ \sum x_i f(x_i,y_i) \\ \sum y_i f(x_i,y_i) \\ \sum x_i y_i f(x_i,y_i) \end{Bmatrix} \tag{4.61}$$

in which the summation is over i=0 to n. Use of this method is illustrated by the following example.

$r=\dfrac{(n+1)\sum x_i y_i-\left(\sum x_i\right)\left(\sum y_i\right)}{\sqrt{(n+1)\sum x_i^2-\left(\sum x_i\right)^2}\sqrt{(n+1)\sum y_i^2-\left(\sum y_i\right)^2}}$, which varies from –1 (for a perfectly linearly decreasing function) to +1 (for a perfectly linearly increasing function). Note, however, that Eq. ? gives only positive *r*, indicating the strength of correlation but not direction. Other statistical parameters, e.g., confidence interval for the regression parameters, may also be used but are not discussed here.

Example 4.12: The cost of fuel consumed by a truck was assumed to be linearly related to the travel distance and the load carried. Over a certain period, the following data was recorded by the driver. Obtain the underlying relationship (add the constraint that there is no cost when both the distance and the load are zero).

Distance (km)	88	210	320	88	210	320	245	65
Load Factor	0.33	0.42	0.50	0.17	0.28	0.67	0.32	1.00
Cost (Rs.)	140	270	400	110	250	450	280	225

Solution:

We assume a relationship between the cost, C, the distance, x, and the load factor, y, as $C(x,y) = f_{1,1}(x,y) = c_{0,0} + c_{1,0}x + c_{0,1}y + c_{1,1}xy$. Since the cost is zero when both the distance and the load are zero, it follows that $c_{0,0}$=0. Also, since the relationship is linear with both variables, we may take $c_{1,1}$=0 (see Exercise 4.6.3). The set of equations is then written as

$$\begin{bmatrix} \sum x_i^2 & \sum x_i y_i \\ \sum x_i y_i & \sum y_i^2 \end{bmatrix} \begin{Bmatrix} c_{1,0} \\ c_{0,1} \end{Bmatrix} = \begin{Bmatrix} \sum x_i f(x_i, y_i) \\ \sum y_i f(x_i, y_i) \end{Bmatrix} \tag{4.61}$$

in which the summation is over i=0 to 7. The following table shows the computations:

i	x_i	y_i	f_i	x^2	xy	y^2	x f	y f
0	88	0.33	140	7744	29.04	0.1089	12320	46.20
1	210	0.42	270	44100	88.20	0.1764	56700	113.4
2	320	0.50	400	102400	160.0	0.2500	128000	200.0
3	88	0.17	110	7744	14.96	0.0289	9680	18.70
4	210	0.28	250	44100	58.80	0.0784	52500	70.00
5	320	0.67	450	102400	214.4	0.4489	144000	301.5
6	245	0.32	280	60025	78.40	0.1024	68600	89.60
7	65	1.00	225	4225	65.00	1.000	14625	225.0
			Σ=	**372738**	**708.8**	**2.1939**	**486425**	**1064.4**

The solution is obtained as $c_{1,0}$=0.9917, and $c_{0,1}$=164.8. Thus we have the Cost (in Rs.) = 0.9917 x Distance (in km) + 164.8 x Load Factor.

While linear regression is the preferred method, sometimes the relationship between two (or more) variables may be such that it is not possible to write it in the form of Eq. (4.52) or (4.59). For example, the exponential relationship $f(x) = c_0 e^{c_1 x}$ and the logistic function $f(x) = \dfrac{c_0}{1 + c_1 e^{c_2 x}}$ are not of this form. However, the exponential relationship may be readily manipulated to obtain a form suitable for linear regression as $\ln[f(x)] = \ln c_0 + c_1 x$, indicating that x should be used as the dependent variable and ln[f(x)] as the dependent

variable. Similarly, linear regression for a relationship of the type $f(x)=\frac{c_0 x}{1+c_1 x}$ may be used by regression of 1/f(x) on 1/x; and a power-law relationship $f(x)=c_0 x^{c_1}$ we may perform regression of ln[f(x)] on ln(x). However, for the logistic function no such manipulation is possible and we have to perform a *nonlinear regression*.

Nonlinear regression

Not all relationships between two (or more) variables could be expressed in the form of Eqs. (4.52) or (4.59). For example, the logistic model of population growth

$$f(x)=\frac{c_0}{1+c_1 e^{c_2 x}} \tag{4.62}$$

is not in a form suitable for linearization. In such cases, we may write the regression function as

$$f_m\left(x^l\right)\equiv f_m\left(x^0, x^1, ...x^l; c_0, c_1, ..., c_m\right) \tag{4.63}$$

to approximate a function of *l+1* independent variables using *m+1* coefficients[1]. To simplify the description, in the rest of this subsection we assume that there is only one independent variable and denote it by *x*.

As before, we define a residual at each of the *n*+1 data points and compute the sum of the squares of the deviations as $\sum_{i=0}^{n}\left[f(x_i)-f_m\left(x_i;c_0,c_1,...,c_m\right)\right]^2$. Again, the parameters are chosen in such a way as to minimize this sum. However, due to the nonlinear nature of the function, f_m, iterative techniques have to be used. A variety of nonlinear optimization techniques are available for this purpose but are beyond the scope of this book. Here we describe a technique based on the Newton method of linearization (see section 3.7).

We start with an initial guess for the parameters $c_0,c_1,\ldots,c_m$ and assume that the nonlinear function f_m may be linearized by using a truncated Taylor series as

$$f_m\left(x_i;c_0^{(k+1)},c_1^{(k+1)},...,c_m^{(k+1)}\right)=f_m\left(x_i;c_0^{(k)},c_1^{(k)},...,c_m^{(k)}\right)+\sum_{j=0}^{m}\left.\frac{\partial f_m}{\partial c_j}\right|_{\left(x_i;c_0^{(k)},c_1^{(k)},...,c_m^{(k)}\right)}\left[c_j^{(k+1)}-c_j^{(k)}\right]$$

in which the superscript (k), as usual, denotes the iteration number. Starting from the known (or assumed) values at iteration level (k), the objective function to be minimized becomes linear in the unknown parameters, which are $c_0,c_1,\ldots c_m$, at the iteration level (k+1). Application of the stationary point theorem then leads to the following equations (compare with Eq. 4.53):

[1] The total number of data points representing the value of the dependent function corresponding to a set of values of the independent variables is assumed to be more than *m*+1. If it is equal to *m*+1, we are likely to get a unique answer by solving the set of nonlinear equations. If it is less than *m*+1, a solution may not be possible.

$$\sum_{i=0}^{n} \left.\frac{\partial f_m}{\partial c_l}\right|_{\left(x_i, c^{(k)}\right)} \left(r^{(k)}\left(x_i\right) - \sum_{j=0}^{m} \left.\frac{\partial f_m}{\partial c_j}\right|_{\left(x_i, c^{(k)}\right)} \Delta c_j^{(k)} \right) = 0 \text{ for } l = 0,1,...,m$$

in which $r^{(k)}\left(x_i\right)$ is the residual, i.e., $f - f_m$, at the i[th] data point and k[th] iteration and $\Delta c_j^{(k)}$ is the change in the parameter value, $c_j^{(k+1)} - c_j^{(k)}$, which is expected to minimize the objective function[1]. The normal equations are then written in a matrix form as

$$[A]^{(k)}\{\Delta c\}^{(k)} = \{b\}^{(k)} \tag{4.64}$$

in which $a_{ij} = \sum_{l=0}^{n} \left.\frac{\partial f_m}{\partial c_i}\frac{\partial f_m}{\partial c_j}\right|_{\left(x_l, c^{(k)}\right)}$ and $b_i = \sum_{l=0}^{n} \left.\frac{\partial f_m}{\partial c_i}\right|_{\left(x_l, c^{(k)}\right)} r^{(k)}(x_l)$. If we define a Jacobian matrix, J, as (compare with Eq. 3.52 and note the difference that there we had derivatives of different functions in each row but here we have derivatives of the same function evaluated at different points)

$$J = \begin{bmatrix} \left.\frac{\partial f_m}{\partial c_0}\right|_{\left(x_0, c^{(k)}\right)} & \left.\frac{\partial f_m}{\partial c_1}\right|_{\left(x_0, c^{(k)}\right)} & \cdot & \left.\frac{\partial f_m}{\partial c_m}\right|_{\left(x_0, c^{(k)}\right)} \\ \left.\frac{\partial f_m}{\partial c_0}\right|_{\left(x_1, c^{(k)}\right)} & \left.\frac{\partial f_m}{\partial c_1}\right|_{\left(x_1, c^{(k)}\right)} & \cdot & \left.\frac{\partial f_m}{\partial c_m}\right|_{\left(x_1, c^{(k)}\right)} \\ \cdot & \cdot & \cdot & \cdot \\ \left.\frac{\partial f_m}{\partial c_0}\right|_{\left(x_n, c^{(k)}\right)} & \left.\frac{\partial f_m}{\partial c_1}\right|_{\left(x_n, c^{(k)}\right)} & \cdot & \left.\frac{\partial f_m}{\partial c_m}\right|_{\left(x_n, c^{(k)}\right)} \end{bmatrix}_{(n+1)\times(m+1)} \tag{4.65}$$

the normal equations may be written as

$$[J]^T[J]\{\Delta c\} = [J]^T\{r\} \tag{4.66}$$

The following example illustrates the use of the iterative procedure.

Example 4.13: The following table shows the population of a city:

Year	1951	1961	1971	1981	1991	2001
Population (in millions)	0.63	0.88	1.16	1.48	1.88	2.50

It is assumed that the growth follows a logistic model (Eq. 4.62). Estimate the parameters c_0, c_1, and c_2 by nonlinear regression.

Solution:

[1] The approximation introduced due to the truncation of Taylor series implies that this choice of the change will generally not be optimal.

The elements of the Jacobian matrix require the derivative of the approximate function with respect to the model parameters. We have, $f_2(x) \equiv f_2(x; c_0, c_1, c_2) = \frac{c_0}{c_1 + e^{c_2 x}}$, from which $\partial f_2 / \partial c_0 = \frac{1}{c_1 + e^{c_2 x}}; \partial f_2 / \partial c_1 = -\frac{c_0}{\left(c_1 + e^{c_2 x}\right)^2}; \partial f_2 / \partial c_2 = -\frac{c_0 x e^{c_2 x}}{\left(c_1 + e^{c_2 x}\right)^2}$. We take 1951 as the base year (t=0)[1] and assume the starting values of the coefficients[2] as c_0=0.7 million, c_1=0.0, and c_2= –0.025 per year. The iterations are shown below:

1st iteration: $c_0^{(k-1)} = 0.7, c_1^{(k-1)} = 0.0, c_2^{(k-1)} = -0.025$

	-0.07000		1.000	0.700	0.000	
	-0.01882		1.284	1.154	8.988	-0.028721
r=	0.00590	J=	1.649	1.903	23.082	Δc= 0.014084
	-0.00190		2.117	3.137	44.457	-0.002160
	-0.02280		2.718	5.172	76.112	
	0.05676		3.490	8.528	122.162	

2nd iteration: $c_0^{(k-1)} = 0.6713, c_1^{(k-1)} = 0.01408, c_2^{(k-1)} = -0.02716$

	-0.03196		0.986	0.653	0.000	
	0.01522		1.288	1.114	8.491	-0.001424
r=	0.03174	J=	1.681	1.896	22.031	Δc= -0.002638
	0.01051		2.189	3.217	42.726	0.000055
	-0.02969		2.845	5.433	73.327	
	0.02531		3.687	9.123	117.310	

3rd iteration: $c_0^{(k-1)} = 0.6699, c_1^{(k-1)} = 0.01144, c_2^{(k-1)} = -0.0271$

	-0.03228		0.989	0.655	0.000	
	0.01458		1.292	1.118	8.526	-0.000076
r=	0.03035	J=	1.686	1.905	22.157	Δc= -0.000230
	0.00749		2.198	3.237	43.064	0.000015
	-0.03595		2.860	5.480	74.129	
	0.01289		3.713	9.234	119.071	

4th iteration: $c_0^{(k-1)} = 0.6698, c_1^{(k-1)} = 0.01121, c_2^{(k-1)} = -0.02709$

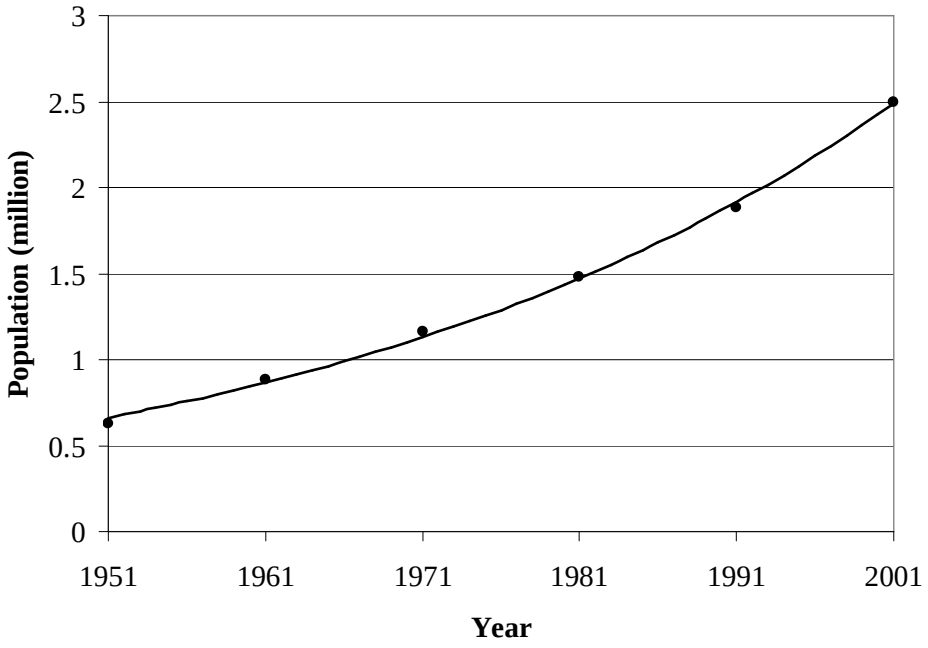
	-0.03235		0.989	0.655	0.000	
	0.01455		1.292	1.118	8.529	-0.000003
r=	0.03038	J=	1.687	1.905	22.165	Δc= -0.000006
	0.00758		2.198	3.237	43.084	0.000000
	-0.03585		2.860	5.480	74.176	

[1] If we take a different base year (e.g., 1900), the coefficients c_0 and c_1 would be different.

[2] It is obvious from the form of the logistic equation that the initial population is $c_0/(c_1+1)$ and the saturation population is c_0/c_1. Assuming c_1 to be small, a good guess for c_0 would be *a little more* than the *initial* population. Also, neglecting c_1, the last data point gives an estimate of c_2 as $\ln(c_0 / f_n) / x_n$. For this example, we use the starting guess as c_1=0, c_0=0.7 million, and c_2= –0.025.

0.01288 3.713 9.235 119.177

We stop the iterations here and estimate the values as $c_0 = 0.6698, c_1 = 0.01121, c_2 = -0.02709$. The figure below shows the observed data and fitted logistic curve. Although the fit appears to be reasonable, it should be kept in mind that the saturation population of about 60 million (c_0/c_1) appears to be unrealistic. In fact a sight change in data (e.g., changing the population in 1951 from 0.63 million to 0.65 million) causes c_1 to be negative, which is unrealistic. Hence, we may say that the logistic model is not a very good model for the population growth of the city under consideration. Another point to be noted is the sensitivity of the results to the starting guess. If we change $c_2^{(0)}$ to -0.01, the iterations do not convervge! It should also be noted that the derivatives of a nonlinear function are required in computation of the *J* matrix. If the function is such that derivatives are not easily obtainable analytically, we may have to use finite differences to estimate their values (as described in details in the next chapter).



In the analysis so far, we have assumed all data points to be equally important. However, sometimes we have more confidence in a particular measurement as compared to other data points. For example, the newer census data (year 2001) may be thought of as more reliable than the older one (1951) due to improvements in methodologies. We may then assign a higher *weight* to some observations to reflect a higher confidence. Another scenario is when we want to minimise the sum of squares of the relative errors rather than the absolute errors. In this case, the weights may be thought of as the inverse of the squares of the function values. This leads to a *weighted regression* which is not discussed in this book as it is conceptually similar to the ordinary regression.

Most of the times we have used polynomial basis functions to perform the interpolation or regression. Sometimes, the function to be interpolated (or regressed) may be periodic in nature (e.g., the temporal variation of temperature is typically periodic with a time period of 24 hours). One option in such cases could be to consider a single period and fit a polynomial to the function within that particular period only. With a shift of origin, this polynomial could be used to describe the function over subsequent periods also. However, use of periodic basis functions leads to a fitting function which is applicable over the entire range. In the next section we discuss some aspects of approximations of periodic functions.

Exercise 4.6

1. For the problem solved in Example 4.11, use the linear regression between ln *m* and *t* to estimate the half-life of the substance and estimate the mass at 6 days. Also find the coefficient of determination and comment on the goodness of fit.

2. The enzyme reaction is assumed to follow a depletion model $\frac{ds}{dt} = -\frac{\mu_{\max} s}{s_{half} + s}$ in which *s* is the substrate concentration at any time, *t*; $\mu_{\max}$ is the maximum substrate consumption rate; and s_{half} is the substrate concentration corresponding to half the maximum consumption rate. This model can be used to predict the substrate concentration at any time, starting from an initial concentration of s_0, from the nonlinear equation $s_0 - s + s_{half} \ln\frac{s_0}{s} = \mu_{\max} t$. The following data was obtained starting from an initial concentration, s_0, of 100 ppm:

t (min)	10	20	30	40	50
s (mM/L)	91	82	73	64	56

Estimate the parameters s_{half} and $\mu_{\max}$ by performing a multiple linear regression of *t* on the parameters $s_0 - s$ and $\ln\frac{s_0}{s}$, i.e., $t = c_{1,0}(s_0 - s) + c_{0,1}\ln\frac{s_0}{s}$. (Note that based on physics of the problem, we have assumed $c_{0,0}$ to be zero. If we include this parameter, the regression may result in a non-zero value.)

3. Re-solve example 4.12 including the coefficient $c_{1,1}$ in the approximate relation. Compare with the solution of example 4.12 and comment on the relative performance of the two models. Is it possible that including $c_{1,1}$ would lead to deterioration in the model performance?

4. In Artificial Neural Network (ANN) modelling, a sigmoid function is used to effect the transformation of input to a neuron into its output. The relationship between the input, *I*, and the scaled output, *O*, is written as $O = \frac{1}{1+e^{-c_0 I}}$. The following table shows a few measurements of the input and output:

I	–10	–5	0	5	10	15
O	0.10	0.25	0.49	0.75	0.89	0.96

Estimate the value of c_0 by using nonlinear regression of *O* on *I*. Then perform a linear regression of $\ln\left(\frac{1}{O}-1\right)$ on *I* and estimate c_0. Compare these two values and explain the result.

4.7 Periodic functions

Let *f(x)* be a periodic function with a period of 2π (if a function of *t* has a period of *T*, we simply define $x=2\pi t/T$. Although the use of variable *t* and time period *T* is more practical, a period of 2π considerably simplifies the mathematical analysis). As discussed before, either the function is given (continuous case) or its values at some grid points are known (discrete case) and we need to find an approximation which uses basis functions which are periodic over the distance (or time) 2π. Clearly, $\sin x$ and $\cos x$ could be taken as basis functions and correspond to the *fundamental frequency* of a Fourier series representation of the function (see Box 4.5). In fact it is easily seen that $\sin jx$ and $\cos jx$ are periodic over an interval of 2π for all integer *j* (representing the *harmonics*) and could act as the basis functions for an approximating expression. We first discuss the continuous case and then the discrete case, in which both interpolation and regression are possible.

Box 4.5: Fourier Series

Although a Fourier series may be written in terms of any set of orthogonal basis functions, we use here the usual meaning of the Fourier series with sine and cosine as the basis functions. Given a periodic function, *f(t)*, which has a period *T*, and is piecewise continuous and square integrable (i.e., $\int_0^T \left[f(t) \right]^2 dt$ is finite), it can be expanded in a Fourier series as

$$f(t) = \frac{a_0}{2} + \sum_{j=1}^{\infty} \left(a_j \cos j\omega t + b_j \sin j\omega t \right) \qquad \text{(B4.5.1)}$$

in which $\omega \left(= \frac{2\pi}{T} \right)$ is the fundamental frequency and $j\omega$ is the j^{th} harmonic. The coefficients are obtained by multiplying the above expression by $1, \cos k\omega t$, and $\sin k\omega t \left(k = 1,2,...,\infty \right)$, integrating over the interval (0,T) and using the following identities $\left(\forall j,k \geq 1 \right)$

$$\int_0^T \sin j\omega t \sin k\omega t\, dt = 0 \qquad j \neq k$$

$$= T/2 \quad j = k$$

$$\int_0^T \cos j\omega t \cos k\omega t\, dt = 0 \qquad j \neq k$$

$$= T/2 \quad j = k$$

$$\int_0^T \sin j\omega t \cos k\omega t\, dt = \int_0^T \sin j\omega t\, dt = \int_0^T \cos j\omega t\, dt = 0$$

as

$$a_j = \frac{2}{T}\int_0^T f(t)\cos j\omega t\, dt \qquad \forall j \geq 0$$

$$b_j = \frac{2}{T}\int_0^T f(t)\sin j\omega t\, dt \qquad \forall j \geq 1 \qquad \text{(B4.5.2)}$$

It is mathematically more convenient to express the Fourier series in its canonical form by using the transformation $x=2\pi t/T=\omega t$, such that the period becomes 2π. The Fourier series then becomes

$$f(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty}\left(a_j \cos jx + b_j \sin jx\right)$$

with the coefficients $\left(\text{for } j \geq 0, \text{ note that } b_0 \text{ is equal to 0, though not needed in the series}\right)$

$$a_j = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos jx\, dx \qquad b_j = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin jx\, dx$$

[we could use the integration over the interval $(0,2\pi)$ but prefer the more commonly used interval of $(-\pi,\pi)$]. The Fourier series converges to *f(x)* at all points when the function and its derivative are continuous. If the function has a finite number of jump discontinuities in each period, the Fourier series converges to the mean of the left and right limits. It should also be noted that for even functions all b^s will be zero and for odd functions all a^s would be zero, and we get the Fourier cosine and sine series, respectively.

The Fourier series can easily be expressed in a compact form in terms of complex exponentials by writing

$$f(x) = \sum_{j=-\infty}^{\infty} C_j e^{ijx} \qquad \text{(B4.5.3)}$$

in which i is the imaginary unit[1] $\left(=\sqrt{-1}\right)$, and using the identity

$$\int_{-\pi}^{\pi} e^{\mathrm{i}jx} e^{\mathrm{i}kx} dx \quad = 2\pi \quad \text{for } j+k=0$$
$$= 0 \quad \text{otherwise}$$

integration of Eq. (B4.5.3) provides the values of the coefficients as

$$C_j = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x) e^{-\mathrm{i}jx} dx \qquad \text{for } j = -\infty \text{ to } \infty$$

It is easy to show that these coefficients are related to those in the usual Fourier series (Eq. B4.5.1) as (for $j \geq 0$)

$$C_j = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x) e^{-\mathrm{i}jx} dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)\cos jx dx - \mathrm{i}\frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)\sin jx dx = \frac{a_j - \mathrm{i}b_j}{2}$$
$$C_{-j} = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x) e^{\mathrm{i}jx} dx = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)\cos jx dx + \mathrm{i}\frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)\sin jx dx = \frac{a_j + \mathrm{i}b_j}{2}$$

Although the complex form of the Fourier series is more compact and mathematically convenient, from a numerical perspective we would like to avoid complex computations. Therefore, we do not discuss the complex form of the Fourier series. However, the orthogonality condition is useful in writing its counterpart for a periodic function sampled at discrete points as shown below.

When the function is not known in an explicit form but is only sampled at a few discrete points *equispaced*[2] over a period, we assume that a single period $(-\pi,\pi)$ is divided into $n+1$ equal segments such that the sampling locations are $x_l = -\pi + \frac{2\pi l}{n+1} \qquad l = 0,1,2,...n$. Note that the endpoint π is not included since the periodic nature of the function makes it redundant to sample both $-\pi$ and π. The orthogonality condition is obtained by noting that $e^{\mathrm{i}mx_l} \qquad l = 0,1,2,...n$ is a geometric sequence with the first term equal to $e^{-\mathrm{i}m\pi}$, i.e., $(-1)^m$, and the common ratio equal to $e^{\mathrm{i}\frac{2m\pi}{n+1}}$. The sum of the series would be $e^{-\mathrm{i}m\pi}\frac{1-e^{\mathrm{i}2\pi m}}{1-e^{\mathrm{i}\frac{2\pi m}{n+1}}}$, which is, clearly, zero for all integer m, unless $m/(n+1)$ is also an integer[3], in which case the common

[1] Sometimes j is used to represent the imaginary unit. Although it may be a little confusing since we use i and j as subscripts also, we decide to use this notation in keeping with the general usage.
[2] For unequal spacing of data points, the basis functions become non-orthogonal and the analysis is quite complicated. We do not discuss these cases.
[3] Which may be positive, negative, or zero.

ratio is 1 and the sum is readily obtained as $(-1)^m(n+1)$. Therefore, we have the following orthogonality condition[1]:

$$\sum_{l=0}^{n} e^{\mathrm{i}jx_l} e^{\mathrm{i}kx_l} = (-1)^{j+k}(n+1) \quad \text{for integer } \frac{j+k}{n+1}$$
$$= 0 \quad \text{otherwise}$$

To avoid complex numbers, we may write this orthogonality conditions in terms of sines and cosines as

$$\sum_{l=0}^{n} \sin jx_l = \sum_{l=0}^{n} \sin jx_l \cos kx_l = 0; \qquad \sum_{l=0}^{n} \cos jx_l = (-1)^j (n+1) \text{ if } \frac{j}{n+1} \text{ is integer}$$
$$= 0 \quad \text{otherwise}$$

$$\sum_{l=0}^{n} \sin jx_l \sin kx_l = (-1)^{j+k+1}\frac{n+1}{2} \quad \text{for integer } \frac{j+k}{n+1} \text{ and noninteger } \frac{j-k}{n+1}$$
$$= (-1)^{j+k}\frac{n+1}{2} \quad \text{for integer } \frac{j-k}{n+1} \text{ and noninteger } \frac{j+k}{n+1}$$
$$= 0 \quad \text{otherwise}$$

$$\sum_{l=0}^{n} \cos jx_l \cos kx_l = (-1)^{j+k}\frac{n+1}{2} \quad \text{when either } \frac{j-k}{n+1} \text{ or } \frac{j+k}{n+1} \text{ is integer}$$
$$= (-1)^{j+k}(n+1) \quad \text{when both } \frac{j-k}{n+1} \text{ and } \frac{j+k}{n+1} \text{ are integers}$$
$$= 0 \quad \text{otherwise}$$

(B4.5.4)

Continuous case

Although any interval of 2π could be used to define the function, we consider the interval to be $(-\pi,\pi)$ instead of $(0,2\pi)$ [if a function of t is defined over the interval $(0,T)$, we transform the variable as $x=2\pi t/T-\pi$]. We assume that the approximating function involves harmonics up to and including m and write it as

$$f_m(x) = \frac{c_0}{2} + \sum_{j=1}^{m}\left(c_j \cos jx + s_j \sin jx\right) \tag{4.67}$$

Note that the fundamental frequency is the first harmonic, and the coefficient of the zeroth harmonic (for which the sine term, obviously, vanishes) is written as $c_0/2$ to be consistent with the definition of coefficients of the cosine term in other harmonics (see Box 4.5). Also note

[1] Sometimes the interval $(0,2\pi)$ is used in which case the term $(-1)^m$ is removed since the first sampling location is x=0. It would make the orthogonality conditions much simpler to write. However, to maintain uniformity with the continuous case, we use the interval $(-\pi,\pi)$.

that while a polynomial of degree m had m+1 coefficients, the approximation using harmonics up to m has 2m+1 coefficients.

To minimize the L_2 norm of the residual[1], i.e., $\int_{-\pi}^{\pi}\left[f(x)-f_m(x)\right]^2 dx$, we use the linear regression as before (see Eq. 4.54) and obtain the following equation

$$[A]\{c\}=\{b\} \tag{4.68}$$

in which the unknown vector {c} is $\{c_0, c_1, s_1, c_2, s_2, ..., c_m, s_m\}^T$, and the elements of the matrix [A] and vector {b} are given by $a_{ij}=\langle\phi_i,\phi_j\rangle$ and $b_i=\langle\phi_i, f\rangle$, with the sine and cosine terms as the basis functions[2] which form an orthogonal set, see Box 4.5. It follows that the coefficients of the approximating function are nothing but the Fourier series coefficients and we get the result that the *Fourier series truncated at the m^{th} harmonic is the best approximation in L_2 norm using (2m+1) terms*.

Error in Fourier approximation

The error of approximation at any *x* is, naturally, equal to the sum of the remaining terms of the Fourier series. We may, therefore, analyze the error in the x-domain (since periodic functions are generally associated with temporal periodicity, we can call this the *time-domain* analysis). However, there is an alternative technique of analyzing periodic functions which would be quite useful in a latter chapter for analyzing the error of numerical solution of differential equations. Here we provide a brief introduction and a detailed analysis is left for Chapter 6.

Looking at the form of the Fourier series representation of a periodic function (Eq. B4.5.1) it is readily seen that the two waveforms (sine and cosine) corresponding to a particular frequency (jω) could be combined into a single term of the form $A\cos(j\omega t+\theta)$, in which *A* is called the amplitude $\left(A=\sqrt{a_j^2+b_j^2}\right)$ and θ is the phase angle $\left[\theta=\tan^{-1}\left(-b_j/a_j\right)\right]$ (we could also write it as a sine function with the same amplitude but a phase angle of π/2+θ). The convention is to use the range $(-\pi,\pi)$ for the phase which is defined as the angular distance of the nearest positive peak from the origin, with positive values used if this peak occurs before zero (an advanced peak) and negative if the peak occurs after zero (a delayed peak), see Fig. 4.11.

[1] We could minimize some other norm, e.g., L_1 or L_∞, but it is not mathematically convenient.

[2] The (2m+1) basis functions are (see Eq. 4.67) $1/2, \cos x, \sin x, \cos 2x, \sin 2x, ..., \cos mx, \sin mx$.

Figure 4.11 A waveform, its amplitude and phase

This leads to an alternative representation of the periodic function in terms of its different harmonics (i.e., in *frequency domain*) with their associated amplitudes and phases. It is known as spectral analysis and the plots of amplitude versus frequency and phase versus frequency are called the amplitude line[1] spectra and phase line spectra, respectively.

Example 4.14 For a periodic triangular function defined by $f(x) = \pi - |x|$ over a period of $(-\pi, \pi)$, obtain the Fourier series. Plot the amplitude line spectra and the phase line spectra. What would be the spectra for the approximation of this function using the terms up to the third harmonic.

Solution:

The Fourier series is written as $f(x) = \frac{a_0}{2} + \sum_{j=1}^{\infty} \left(a_j \cos jx + b_j \sin jx \right)$. Since the function is symmetric, all the b's would be zero. The coefficients *a* are obtained as

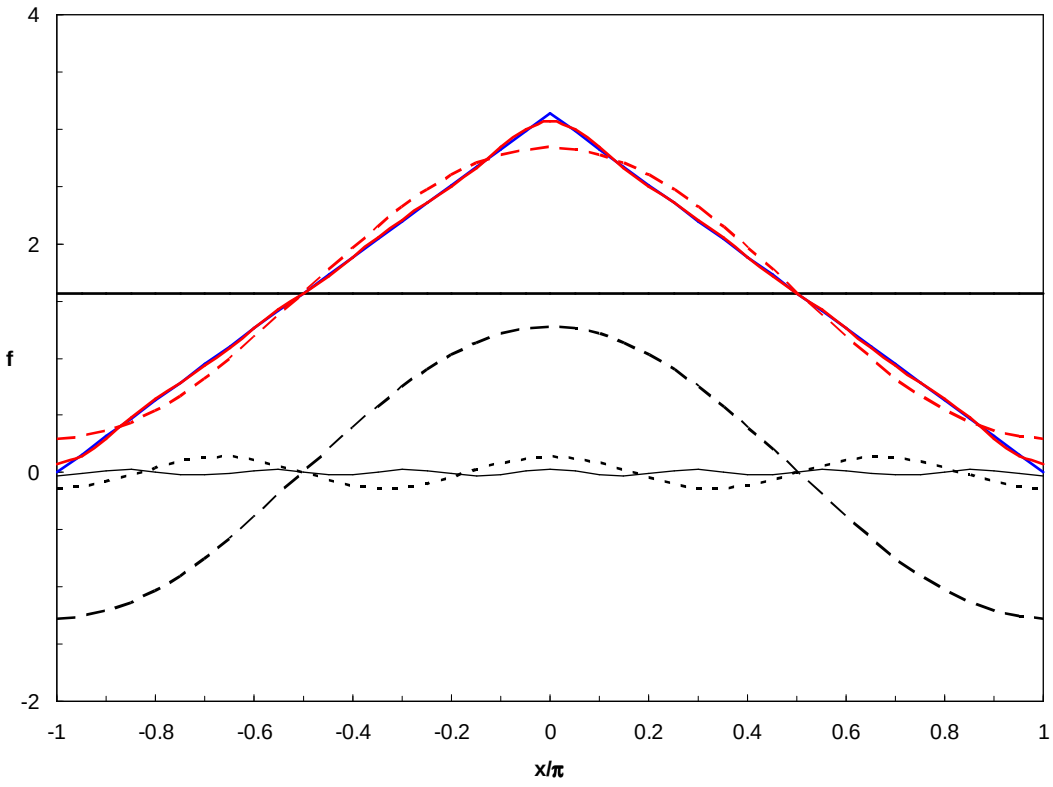
$$a_j = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos jx \, dx = \frac{1}{\pi} \left[\int_{-\pi}^{0} (\pi + x) \cos jx \, dx + \int_{0}^{\pi} (\pi - x) \cos jx \, dx \right]$$

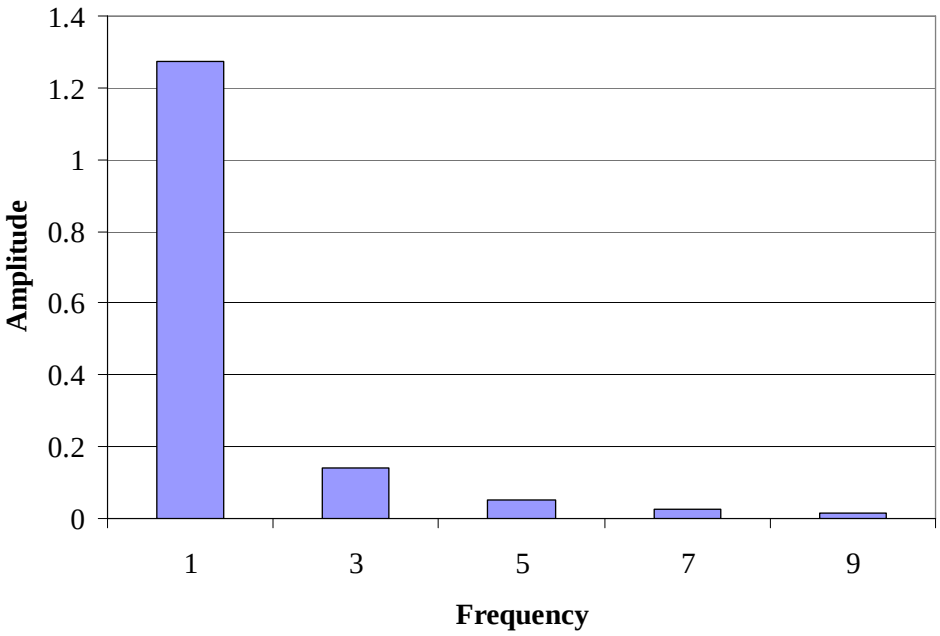
$$= \int_{-\pi}^{\pi} \cos jx \, dx - \frac{2}{\pi} \int_{0}^{\pi} x \cos jx \, dx$$

$$= \begin{cases} \pi & \text{for } j=0 \\ \frac{2}{\pi} \frac{1-(-1)^j}{j^2} & \text{otherwise} \end{cases}$$

resulting in the Fourier series $f(x) = \frac{\pi}{2} + \frac{4}{\pi} \sum_{j=1,3,5\ldots}^{\infty} \frac{\cos jx}{j^2}$. Clearly, the phase angle is zero for all frequencies and the amplitude of the j[th] harmonic is $\frac{4}{\pi j^2}$. A plot of different harmonics in the time domain is shown below and the amplitude line spectra is also shown.

[1] The term line refers to the fact that there are discrete frequencies. For a nonperiodic function, we get a continuous distribution of frequencies and get the amplitude and phase spectra.





If we include terms up to the third harmonic only, the amplitude spectra would have zero amplitude for the 5^{th} and higher harmonics.

Thus we have an alternative way of defining the error of approximation in terms of the *amplitude error* and the *phase error*. If both the given periodic function and the approximation are expressed in terms of the fundamental frequency and the harmonics, each frequency may be thought of as having an amplitude error, equal to the difference between the true amplitude and the approximate amplitude, and a phase errors, equal to the difference of true and approximate phases. For example, a truncated Fourier series approximation using up to two harmonics will have zero amplitude and phase errors corresponding to the fundamental frequency and the first two harmonics, but will have amplitude errors equal to the amplitude of the true spectra for higher harmonics. Obviously, both amplitude and phase errors are important to consider and more discussion about these appears in Chapter 6.

Discrete case

For a periodic function, sampled at the n+1 points $x_l = -\pi + \frac{2\pi l}{n+1} \qquad l = 0,1,2,...n$, we would have a choice of approximating by an interpolating truncated Fourier series (which passes through all sampled points) or a regressing series (which would minimize L_2 norm of the residual at the sampling points). The discrete form of the orthogonality condition of the Fourier series (see Box 4.5) is used to derive these approximations as follows.

Interpolation

We assume that *n is even*, such that we have an odd number of grid points. In this case, we express an interpolating function which matches the sampled function values at all grid points as

$$f_m(x) = \frac{c_0}{2} + \sum_{j=1}^{m} \left(c_j \cos jx + s_j \sin jx \right) \tag{4.69}$$

where m=n/2. The orthogonality condition results in the values of these coefficients as

$$c_j = \frac{2}{n+1} \sum_{l=0}^{n} f(x_l) \cos jx_l \quad \text{and} \quad s_j = \frac{2}{n+1} \sum_{l=0}^{n} f(x_l) \sin jx_l \tag{4.70}$$

If n is odd, m will be equal to (n−1)/2 and we add an additional term, $\frac{c_{m+1}}{2} \cos(m+1)x$, in which the coefficient is given by the same formula listed above[1]. Note that the coefficient is divided by 2 since $j=k=m+1$ implies that both $\frac{j-k}{n+1}$ and $\frac{j+k}{n+1}$ are integers and the relevant orthogonality condition (see Box 4.5) has to be used.

Regression

If we use fewer number of harmonics than that used for interpolation [i.e., n/2 for even n and (n+1)/2 for odd n], we would obtain a regressing series which would not pass through all the

[1] It is easily seen that $\sin(m+1)x$ will vanish at all grid points and $\cos(m+1)x_l$ will alternate between ± 1.

data points. A least square regression would again lead to the conclusion that a truncated Fourier series is the best approximation. Thus the form of the series and its coefficients would be same as that listed above for the interpolating series.

The discussion regarding Fourier series for periodic function may be extended to nonperiodic functions by considering them to be periodic with an infinite period. In this case, the Fourier series becomes a Fourier integral (or Fourier Transform). We do not discuss it here. Another topic which we avoid discussing here is the efficient evaluation of the Fourier approximation for discrete data using the *Fast Fourier Transform* (FFT). We believe that this topic may be suitable for only a small section of the targeted audience of this book. We have provided the basic theory of the method and interested reader should refer to specialized books on this topic for further discussion.

Example 4.15 For the periodic triangular function of the previous example, let the function be given in terms of 7 equally spaced points $(-\pi, -5\pi/7, -3\pi/7, -\pi/7, \pi/7, 3\pi/7, 5\pi/7)$. Obtain the interpolating Fourier series. Also obtain the least squares approximation of this function using the terms up to the fundamental frequency (i.e., the constant term and the first harmonic).

Solution:

The following table shows the sampled values of the function:

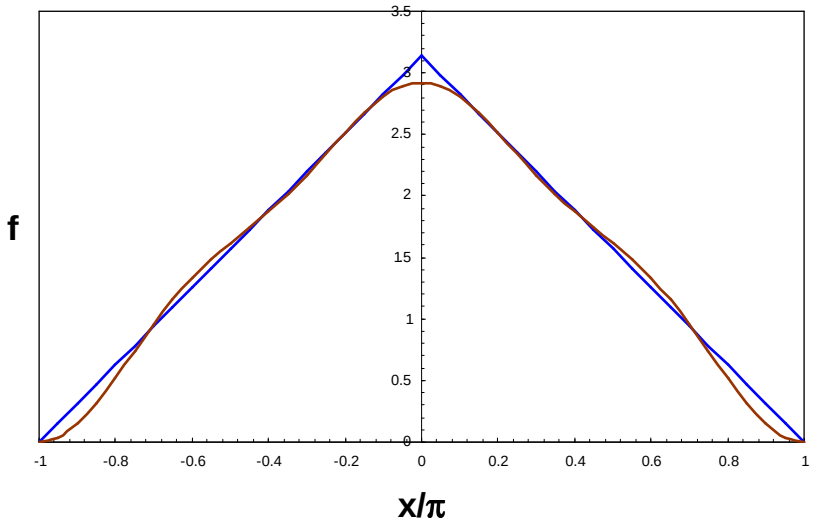
i	0	1	2	3	4	5	6
x_i	-3.141593	-2.243995	-1.346397	-0.448799	0.448799	1.346397	2.243995
$f(x_i)$	0.000000	0.897598	1.795196	2.692794	2.692794	1.795196	0.897598

Since we have an odd number of grid points (n=6), Eq. (4.69) is used to interpolate the function with m=3. The following table shows the computations of the coefficients using Eq. (4.70) (Note that the symmetry would lead to all the coefficients *s* being zero. However, we show these computations also):

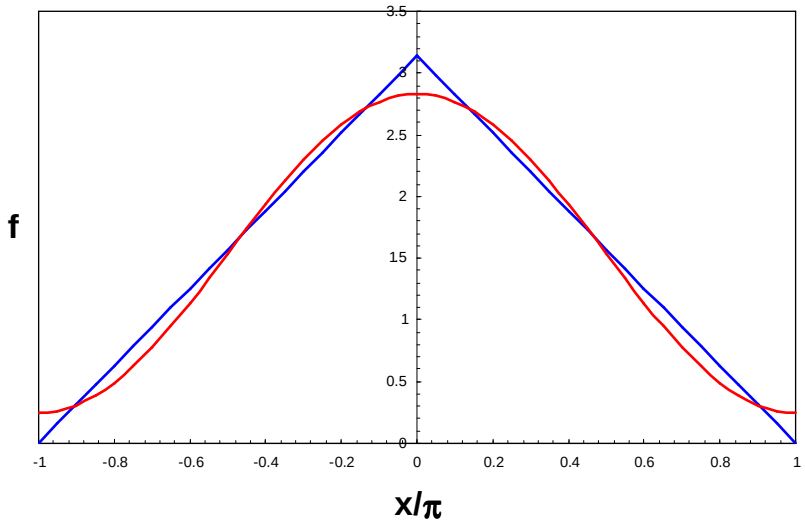
l	x_l	$f(x_l)$	$f(x_l)\cos jx_l$				$f(x_l)\sin jx_l$		
			j=0	j=1	j=2	j=3	j=1	j=2	j=3
0	-3.141593	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1	-2.243995	0.897598	0.897598	-0.559643	-0.199734	0.808708	-0.701770	0.875093	-0.389453
2	-1.346397	1.795196	1.795196	0.399469	-1.617416	-1.119286	-1.750186	-0.778906	1.403541
3	-0.448799	2.692794	2.692794	2.426123	1.678929	0.599203	-1.168359	-2.105311	-2.625280
4	0.448799	2.692794	2.692794	2.426123	1.678929	0.599203	1.168359	2.105311	2.625280
5	1.346397	1.795196	1.795196	0.399469	-1.617416	-1.119286	1.750186	0.778906	-1.403541
6	2.243995	0.897598	0.897598	-0.559643	-0.199734	0.808708	0.701770	-0.875093	0.389453

Sum	10.771175	4.531898	-0.276441	0.577249	0.000000	0.000000	0.000000
Coefficients, cj and sj	3.077479	1.294828	-0.078983	0.164928	0.000000	0.000000	0.000000

The interpolating function is therefore $f_3(x) = 1.53874 + 1.29483\cos x - 0.0789831\cos 2x + 0.164928\cos 3x$. As a comparison, we note that the corresponding coefficients of the Fourier series of this function (Example 4.14) are 1.57080, 1.27324, 0, and 0.141471, respectively. While the phase angle is again zero for all frequencies, the amplitude error is readily obtainable. Note particularly the presence of a small amplitude corresponding to the second harmonic although the *true amplitude* is zero. The following figure shows a plot of this interpolating function (which, naturally, must pass through all 7 data points):



As discussed, the least squares fit using the constant and the fundamental frequency is given by $f_1(x) = 1.53874 + 1.29483\cos x$ which is plotted below:



Exercise 4.7

1. Find the Fourier series of the saw-tooth wave of period T, defined as $f(t) = t/T$ for t = 0 to T (you may want to define $x = \pi(2t/T - 1)$ to change the period into the standard range of $-\pi$ to π. Also note that there is a jump discontinuity at the ends of the period). Why do all cosine terms (except the constant) vanish?

2. A periodic function is defined over a period $(-\pi,\pi)$ as $f(x) = x(\pi - x)$. Find its Fourier series. Now generate a set of 8 function values starting at $x=-\pi$ at intervals of $\pi/8$ and use these to obtain the interpolating Fourier series.

3. For the function described in the question above, obtain the least square fit using terms up to the second harmonic for both the continuous function and the discrete data. Compare these approximations and obtain the phase and amplitude errors.

4.8 Summary

In this chapter, we discussed the motivation for approximating a given function (either as a function or as discretely sampled values) by another (generally polynomial or trigonometric). Various methods including the Taylor's series, least squares, and orthogonal polynomials, were described for approximating a function given in a continuous form and their relative merits and drawbacks were discussed. The concept of a best-fit was discussed in terms of various error norms. For discretely sampled functions, interpolation and regression were discussed. For interpolation various alternative forms of the unique interpolating polynomials were described and their utility in particular conditions was emphasized. Piecewise interpolation was introduced as a method to reduce the possibility of large errors associated with equidistant interpolation. Linear and nonlinear regression was then discussed and the

correlation coefficient was described. Finally, for periodic functions, both in continuous and in discrete forms, the Fourier series was used to approximate these using sine and cosine functions.

Clearly, the methods described here may be used to approximate the derivatives and the integrals of a given function (or values) by first finding the approximating function and then performing an analytical differentiation or integration. However, since the problems of estimating the derivative or integral occur quite frequently, it is desirable to obtain these approximations directly rather than finding the approximation of the function first. Although the basic philosophy is similar to what we have discussed in this chapter, there are enough differences to warrant a separate chapter. Therefore, the next chapter describes various techniques of numerical differentiation and integration.