Chapter 6

Is There More Than Finite Differences?

6.1 Introduction to Projection Methods

In the previous chapters we have studied in some detail the application of finite difference methods to the approximate solution of differential equations. In this chapter we will consider another approach which has several variants known by such names as the finite element method, Galerkin's method, and the Rayleigh-Ritz method. The underlying theme of all these methods is that one attempts to approximate the solution of the differential equation by a finite linear combination of known functions. These known functions, usually called the *basis functions*, have the common property that they are relatively simple: polynomials, trigonometric functions, and, most importantly, spline functions, which will be studied in the following section. Conceptually, we regard the solution as lying in some appropriate (infinite-dimensional) function space, and we attempt to obtain an approximate solution that lies in the finite-dimensional subspace that is determined by the basis functions. The "projection" of the solution onto the finite-dimensional subspace is the approximate solution.

We will illustrate these general ideas with the linear two-point boundary-value problem

$$v''(x) + q(x)v = f(x), \qquad 0 \leq x \leq 1, \tag{6.1.1}$$

with

$$v(0) = 0, \qquad v(1) = 0, \tag{6.1.2}$$

where, for simplicity, we have taken the interval to be $[0, 1]$ and the boundary conditions to be zero (see Exercises 3.1.1 and 6.1.6).

Suppose that we look for an approximate solution of (6.1.1), (6.1.2) of the

form

$$u(x) = \sum_{j=1}^{n} c_j \phi_j(x), \tag{6.1.3}$$

where the basis functions ϕ_j satisfy the boundary conditions:

$$\phi_j(0) = \phi_j(1) = 0, \qquad j = 1, \ldots, n. \tag{6.1.4}$$

If (6.1.4) holds, then the approximate solution u given by (6.1.3) satisfies the boundary conditions. A classical example of a set of basis functions that satisfy (6.1.4) is

$$\phi_j(x) = \sin j\pi x, \qquad j = 1, \ldots, n. \tag{6.1.5}$$

Another example is the set of polynomials

$$\phi_j(x) = x^j(1-x), \qquad j = 1, \ldots, n. \tag{6.1.6}$$

In the latter case the approximate solution (6.1.3) is of the form

$$u(x) = x(1-x)(c_1 + c_2 x + \cdots + c_n x^{n-1}),$$

which is a polynomial of degree $n+1$ with the property that it vanishes at 0 and 1. Our main example of a set of basis functions, however, is spline functions, which, as mentioned previously, will be studied in the following section.

Given a set of basis functions, we need to specify in what sense (6.1.3) is to be an approximate solution; that is, what is the criterion for determining the coefficients c_k in the linear combination? There are several possible approaches, and we will discuss here only two, both of which are generally applicable and widely used.

Collocation

The first criterion is that of *collocation*. Let $x_1, \ldots, x_n$ be n (not necessarily equally spaced) grid points in the interval $[0, 1]$. We then require that the approximate solution satisfy the differential equation at these n points. Thus for the equation (6.1.1) and the approximation (6.1.3) we require that

$$\frac{d^2}{dx^2}\left(\sum_{j=1}^{n} c_j \phi_j(x)\right)\Big|_{x_i} + q(x_i)\sum_{j=1}^{n} c_j \phi_j(x_i) = f(x_i), \quad i = 1, \ldots, n, \tag{6.1.7}$$

and we assume, of course, that the basis functions are twice differentiable. If we carry out the differentiation in (6.1.7) and collect coefficients of the c_j, we have

$$\sum_{j=1}^{n} c_j[\phi_j''(x_i) + q(x_i)\phi_j(x_i)] = f(x_i), \quad i = 1, \ldots, n. \tag{6.1.8}$$

This is a system of n linear equations in the n unknowns $c_1, \ldots, c_n$. The computational problem is first to evaluate the coefficients

$$a_{ij} \equiv \phi_j''(x_i) + q(x_i)\phi_j(x_i), \tag{6.1.9}$$

and then solve the system of linear equations

$$A\mathbf{c} = \mathbf{f}, \tag{6.1.10}$$

where A is the $n \times n$ matrix (a_{ij}), $\mathbf{c} = (c_1, \ldots, c_n)^T$, and $\mathbf{f} = (f(x_1), \ldots, f(x_n))^T$.

We give a simple example. Consider the problem

$$v''(x) + x^2 v(x) = x^3, \qquad 0 \le x \le 1, \tag{6.1.11}$$

with the boundary conditions (6.1.2). Here $f(x) = x^3$ and $q(x) = x^2$. With the basis functions (6.1.5), we have

$$\phi_j'(x) = j\pi \cos j\pi x, \qquad \phi_j''(x) = -(j\pi)^2 \sin j\pi x,$$

so that the coefficients (6.1.9) are

$$a_{ij} = -(j\pi)^2 \sin j\pi x_i + x_i^2 \sin j\pi x_i.$$

The coefficients of the right-hand side of the system (6.1.10) are $f(x_i) = x_i^3$. If we use the basis functions (6.1.6), then

$$\phi_j'(x) = x^{j-1}[j - (j+1)x], \qquad \phi_j''(x) = jx^{j-2}[j - 1 - (j+1)x],$$

so that the coefficients (6.1.9) are now

$$a_{ij} = jx_i^{j-2}[j - 1 - (j+1)x_i] + x_i^{j+2}(1 - x_i).$$

Again, the components of the right-hand side are x_i^3. In both cases the system (6.1.10) is easily constructed once the grid points $x_1, \ldots, x_n$ are specified.

Galerkin's Method

We will return to a discussion of the collocation method after we consider another approach to determining the coefficients $c_1, \ldots, c_n$. This is known as *Galerkin's method* and is based on the concept of orthogonality of functions. Recall that two vectors $\mathbf{f}$ and $\mathbf{g}$ are orthogonal if the inner product satisfies

$$(\mathbf{f}, \mathbf{g}) \equiv \mathbf{f}^T\mathbf{g} = \sum_{j=1}^{n} f_j g_j = 0. \tag{6.1.12}$$

Now suppose that the components of the vectors $\mathbf{f}$ and $\mathbf{g}$ are the values of two functions f and g at n equally spaced grid points in the interval $[0, 1]$; that is,

$$\mathbf{f} = (f(h), f(2h), \ldots, f(nh)),$$

where $h = (n+1)^{-1}$ is the grid-point spacing, and similarly for $\mathbf{g}$. Then the orthogonality relation (6.1.12) is

$$\sum_{j=1}^{n} f(jh)g(jh) = 0,$$

and this relation is unchanged if we multiply by h:

$$h\sum_{j=1}^{n} f(jh)g(jh) = 0. \tag{6.1.13}$$

Now let $n \to \infty$ (or, equivalently, let $h \to 0$). Then, assuming that the functions f and g are integrable, the sum in (6.1.13) will tend to the integral

$$\int_0^1 f(x)g(x)dx = 0. \tag{6.1.14}$$

With this motivation, we define two functions f and g to be *orthogonal* on the interval $[0,1]$ if the relation (6.1.14) holds.

The rationale for the Galerkin approach is as follows. Let the *residual function* for $u(x)$ be defined by

$$r(x) = u''(x) + q(x)u(x) - f(x), \qquad 0 \le x \le 1. \tag{6.1.15}$$

If $u(x)$ were the exact solution of (6.1.1), then the residual function would be identically zero. Obviously, the residual would then be orthogonal to every function, and, in particular, it would be orthogonal to the set of basis functions. However, we cannot expect $u(x)$ to be the exact solution because we restrict $u(x)$ to be a linear combination of the basis functions. The Galerkin criterion is to choose $u(x)$ so that its residual is orthogonal to all of the basis functions $\phi_1, \ldots, \phi_n$:

$$\int_0^1 [u''(x) + q(x)u(x) - f(x)]\phi_i(x)dx = 0, \quad i = 1, \ldots, n. \tag{6.1.16}$$

If we put (6.1.3) into (6.1.16) and interchange the summation and integration, we obtain

$$\sum_{j=1}^{n} c_j \int_0^1 [\phi_j''(x) + q(x)\phi_j(x)]\phi_i(x)dx = \int_0^1 f(x)\phi_i(x)dx, \quad i = 1, \ldots, n.$$

Again, this is a system of linear equations of the form (6.1.10) with

$$f_i = \int_0^1 f(x)\phi_i(x)dx, \qquad i = 1, \ldots, n, \tag{6.1.17}$$

and

$$a_{ij} = \int_0^1 [\phi_j''(x) + q(x)\phi_j(x)]\phi_i(x)dx.$$

If we integrate the first term in this integral by parts,

$$\int_0^1 \phi_j''(x)\phi_i(x)dx = \phi_j'(x)\phi_i(x)\,\Big|_0^1 - \int_0^1 \phi_j'(x)\phi_i'(x)dx,$$

and note that the first term vanishes because ϕ_i is zero at the end points, we can rewrite a_{ij} as

$$a_{ij} = -\int_0^1 \phi_i'(x)\phi_j'(x)dx + \int_0^1 q(x)\phi_i(x)\phi_j(x)dx. \tag{6.1.18}$$

Thus the system of equations to solve for the coefficients $c_1, \ldots, c_n$ in Galerkin's method is $A\mathbf{c} = \mathbf{f}$, with the elements of A given by (6.1.18) and those of $\mathbf{f}$ by (6.1.17). An example of the evaluation of the a_{ij} of (6.1.18) is left to Exercise 6.1.5.

Comparison of Methods

We now make several comments regarding the finite difference, collocation, and Galerkin methods as applied to (6.1.1). In each case the central computational problem is to solve a system of linear equations. In the finite difference and collocation methods these linear systems are determined by n grid points in the interval, although the nature of the linear systems is quite different: the finite difference method gives an approximation to the solution of the differential equation at the grid points, whereas the collocation method gives the coefficients of the representation (6.1.3) of the approximate solution. With the collocation (or Galerkin) method, the value of the approximate solution at any point $\bar{x}$ in the interval is obtained by the additional evaluation

$$u(\bar{x}) = \sum_{j=1}^{n} c_j\phi_j(\bar{x}).$$

Although the finite difference method requires no additional work to obtain the approximate solution at the grid points, it is defined *only* at the grid points, and obtaining an approximation at other points in the interval necessitates an interpolation process. The collocation and Galerkin methods, on the other hand, give an approximate solution on the whole interval.

As we saw in Chapter 3, the linear system of equations of the finite difference method is easily obtained and has the important property (for the second-order difference approximations used there) that the coefficient matrix is tridiagonal; thus, the solution of the linear system requires relatively little computation, the number of arithmetic operations required being proportional

to n (Section 3.2). For the collocation method, the elements of the coefficient matrix are also evaluated relatively easily by (6.1.9), provided that the basis functions ϕ_j are suitably simple. However, the coefficient matrix will now generally be full, which means not only that all n^2 elements need to be evaluated but also that the solution time will be proportional to n^3. One of the very important properties of the spline basis functions – to be discussed in the next section – is that ϕ_i will be identically zero except in a subinterval about x_i. In the cases considered in Section 6.4, this subinterval will extend only from x_{i-2} to x_{i+2}, and the coefficient matrix will be tridiagonal.

These same comments apply to Galerkin's method: the coefficient matrix will in general be full, but the use of an appropriate spline function basis will allow us to recover a tridiagonal matrix. However, there is now another complication. The evaluation of the matrix coefficients (6.1.18) and elements of the right-hand side (6.1.17) requires integration over the whole interval. Only if the functions q and f are very simple will one be able to evaluate these integrals explicitly in closed form. Usually they must be approximated, and this leads us to the topic of numerical integration, which we consider in Section 6.3. Symbolic computation systems may also be used under certain circumstances. Finally, one advantage of the Galerkin method is that it always yields a symmetric matrix, as can be seen by (6.1.18), whereas collocation does not. No method has a clear advantage over the others. For each method there are problems for which it is best. Given a particular problem, analyses of the three methods applied to the problem may be necessary to evaluate their relative effectivenesses.

Nonlinear Problems

We end this section by indicating briefly how the collocation and Galerkin methods can be applied to nonlinear problems. For this purpose we will consider the equation

$$v'' = g(v), \qquad v(0) = v(1) = 0, \tag{6.1.19}$$

where g is a given nonlinear function of a single variable. For the collocation method applied to (6.1.19) we substitute the approximate solution (6.1.3) and evaluate at the grid points $x_1, \ldots, x_n$ as before. This then leads to the nonlinear system of equations

$$\sum_{j=1}^{n} c_j \phi_j''(x_i) = g\left(\sum_{j=1}^{n} c_j \phi_j(x_i)\right), \qquad i = 1, \ldots, n, \tag{6.1.20}$$

for the coefficients $c_1, \ldots c_n$.

Similarly, for the Galerkin method the residual function (6.1.15) now becomes

$$r(x) = \sum_{i=1}^{n} c_j \phi_j''(x) - g\left(\sum_{i=1}^{n} c_j \phi_j(x)\right),$$

so that the system of equations corresponding to (6.1.16) is

$$\int_0^1 \left[\sum_{j=1}^n c_j \phi_j''(x) - g\left(\sum_{j=1}^n c_j \phi_j(x)\right)\right] \phi_i(x)dx = 0, \quad i = 1, \ldots, n. \quad (6.1.21)$$

As before, we can integrate the first term by parts to put (6.1.21) in the form

$$-\sum_{j=1}^n c_j \int_0^1 \phi_i'(x)\phi_j'(x)dx = \int_0^1 g\left(\sum_{j=1}^n c_j \phi_j(x)\right) \phi_i(x)dx, \quad (6.1.22)$$

which is again a nonlinear system for $c_1, \ldots, c_n$. The methods of the previous chapter can be applied, in principle, to approximate solutions of both (6.1.20) and (6.1.22).

Supplementary Discussion and References: 6.1

A related approach to projection methods is by means of a *variational principle.* Consider the problem

$$\underset{v \in V}{\text{Minimize}} \int_0^1 \{[v'(x)]^2 - q(x)[v(x)]^2 - 2f(x)v(x)\}dx, \quad (6.1.23)$$

where V is a set of suitably differentiable functions that vanish at the end points $x = 0$ and $x = 1$. By results in the calculus of variations, the solution of (6.1.23) is also the solution of the differential equation (6.1.1), which is known as the *Euler equation* for (6.1.23). Thus we can solve (6.1.1) by solving (6.1.23), and we can attempt to approximate a solution to (6.1.23) in a manner analogous to the Galerkin method. This is known as the *Rayleigh-Ritz method.*

Let $\phi_1, \ldots, \phi_n$ be a set of basis functions such that $\phi_i(0) = \phi_i(1) = 0$, $i = 1, \ldots, n$. Then we wish to minimize

$$\int_0^1 \left\{\left[\sum_{i=1}^n c_i \phi_i'(x)\right]^2 - q(x)\left[\sum_{i=1}^n c_i \phi_i(x)\right]^2 - 2f(x)\sum_{i=1}^n c_i \phi_i(x)\right\} dx \quad (6.1.24)$$

over the coefficients $c_1, \ldots, c_n$. If $c_1^*, \ldots, c_n^*$ is the solution of the minimization problem, then

$$u(x) = \sum_{i=1}^n c_i^* \phi_i(x)$$

is taken as an approximate solution for (6.1.23). If we use the same basis functions for the Galerkin method applied to (6.1.1), we will obtain the same approximate solution. A good reference for the Rayleigh-Ritz and Galerkin methods is Strang and Fix [1973]. A good reference for collocation methods is Ascher et al. [1988].

An important question is when does the system of linear equations obtained by the discretization methods of this section have a unique solution. This is generally easier to ascertain in the case of the Rayleigh-Ritz method since the question reduces to when the functional (6.1.24) has a minimum. For an introduction to these existence and uniqueness theorems as well as the important question of discretization error for the Galerkin and collocation methods, see, for example, Prenter [1975] and Hall and Porsching [1990].

EXERCISES 6.1

6.1.1. **a.** For the two-point boundary-value problem $y''(x) = y(x) + x^2$, $0 \le x \le 1$, $y(0) = y(1) = 0$, write out explicitly the system of equations (6.1.8) for $n = 3$, $\phi_j(x) = \sin j\pi x$, and $x_i = i/3$, $i, j = 1, 2, 3$.

b. Repeat part **a** with $\phi_j(x) = x^j(1 - x)$, $j = 1, 2, 3$.

c. Write out the system for general n in matrix form using both (6.1.5) and (6.1.6) as the basis functions.

6.1.2. Show that the functions $\sin k\pi x$ are mutually orthogonal on the interval $[0, 1]$, that is, $\int_0^1 \sin k\pi x \sin j\pi x \, dx = 0$, $j, k = 0, 1, \ldots, j \ne k$.

6.1.3. Repeat Exercise 6.1.1 for the Galerkin equations $A\mathbf{c} = \mathbf{f}$, where $\mathbf{f}$ is given by (6.1.17) and A by (6.1.18).

6.1.4. Let $g(v) = e^v$.

a. For $n = 3$ and $\phi_j(x) = \sin j\pi x$, $j = 1, 2, 3$, write out explicitly the equations (6.1.20) and (6.1.22) for the two-point boundary-value problem (6.1.19).

b. Repeat part **a** for the basis functions $\phi_j(x) = x^j(1 - x)$, $j = 1, 2, 3$.

6.1.5. If $q(x) = x^2$, evaluate the coefficients a_{ij} of (6.1.18) for the basis functions (6.1.5) and (6.1.6).

6.1.6. Show that the boundary value problem

$$v''(x) + p(x)v'(x) + q(x)v(x) = f(x), \quad v(0) = \alpha, \; v(1) = \beta$$

can be converted to a problem with zero boundary conditions as follows. Let $u(x) = v(x) - (\beta - \alpha)x - \alpha$. Show that $u(0) = u(1) = 0$ and that u satisfies the differential equation

$$u''(x) + p(x)u'(x) + q(x)u(x) = f(x) - \alpha q(x) - [p(x) + q(x)](\beta - \alpha)x.$$

6.2 Spline Approximation

In Section 2.3 we considered the problem of approximating a function by polynomials or piecewise polynomials. In the present section we will extend this to piecewise polynomials that have additional properties.

Piecewise Quadratic Functions

Let $a \le x_1 < x_2 < \cdots < x_n \le b$ be nodes subdividing the interval $[a, b]$, and let $y_1, \ldots, y_n$ be corresponding function values. In Section 2.3, we used piecewise polynomials that matched at certain grid points. For example, the function of (2.3.8) was a piecewise quadratic that agreed with given data at seven nodes; it was composed of three quadratics and was continuous but failed to be differentiable at the nodes where the different quadratics met. Now suppose that we wish to approximate by piecewise quadratics, but we require that the approximating function be differentiable everywhere. Then we need a different approach than that of Section 2.3. To illustrate this approach let $n = 4$, $I_i = [x_i, x_{i+1}]$, $i = 1, 2, 3$, and

$$q_i(x) = a_{i2}x^2 + a_{i1}x + a_{i0}, \qquad i = 1, 2, 3. \tag{6.2.1}$$

We will define a piecewise quadratic function q such that $q(x) = q_i(x)$ if $x \in I_i$, $i = 1, 2, 3$, as illustrated in Figure 6.1. For q to be continuous and take on the prescribed values y_i at the nodes, we require that

$$\begin{aligned} &q_1(x_1) = y_1, \qquad q_1(x_2) = y_2, \qquad q_2(x_2) = y_2, \\ &q_2(x_3) = y_3, \qquad q_3(x_3) = y_3, \qquad q_3(x_4) = y_4. \end{aligned} \tag{6.2.2}$$

If we also wish that q be differentiable at the nodes, then q_1' must equal q_2' at x_2, and q_2' must equal q_3' at x_3:

$$q_1'(x_2) = q_2'(x_2), \qquad q_2'(x_3) = q_3'(x_3). \tag{6.2.3}$$

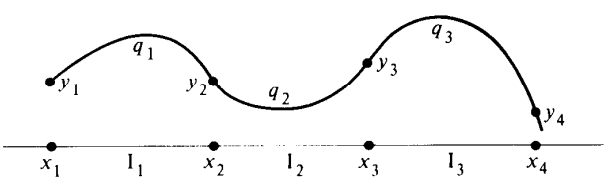


Figure 6.1: *A Piecewise Quadratic Function*

The function q is determined by the nine coefficients in (6.2.1) that define q_1, q_2, and q_3. The relations (6.2.2) and (6.2.3) give only eight conditions that

these nine coefficients must satisfy, and hence another relation must be specified to determine q uniquely. Usually a value of q' at some node is specified, for example,

$$q_1'(x_1) = d_1, \tag{6.2.4}$$

where d_1 is some given value. The nine relations (6.2.2), (6.2.3), and (6.2.4) are then a system of nine linear equations for the coefficients of the q_i.

This approach is easily extended to an arbitrary number, n, of nodes. In this case there will be $n-1$ intervals I_i and $n-1$ quadratics q_i defined on these intervals. The conditions (6.2.2) and (6.2.3) become

$$q_i(x_i) = y_i \qquad q_i(x_{i+1}) = y_{i+1}, \qquad i = 1, \ldots, n-1, \tag{6.2.5}$$

and

$$q_i'(x_{i+1}) = q_{i+1}'(x_{i+1}), \qquad i = 1, \ldots, n-2. \tag{6.2.6}$$

These relations give $3n-4$ linear equations for the $3n-3$ unknown coefficients of the polynomials $q_1, \ldots, q_{n-1}$. Again, one additional condition is needed, and we can use, for example, (6.2.4). Thus, to determine the piecewise quadratic, we need to solve $3n-3$ linear equations.

Writing out equations (6.2.5), (6.2.6), and (6.2.4) for the quadratics q_i, we have

$$\begin{aligned}
&a_{i2}x_i^2 + a_{i1}x_i + a_{i0} = y_i, && i = 1, \ldots, n-1,\\
&a_{i2}x_{i+1}^2 + a_{i1}x_{i+1} + a_{i0} = y_{i+1}, && i = 1, \ldots, n-1,\\
&2a_{i2}x_{i+1} + a_{i1} = 2a_{i+1,2}x_{i+1} + a_{i+1,1}, && i = 1, \ldots, n-2,\\
&2a_{12}x_1 + a_{11} = d_1.
\end{aligned}$$

If we order the unknowns as a_{12}, a_{11}, a_{10}, a_{22}, a_{21}, a_{20} and so on, the coefficient matrix of this system has the structure

$$\begin{bmatrix}
x_1^2 & x_1 & 1 & & & & & & & & & \\
x_2^2 & x_2 & 1 & & & & & & & & & \\
 & & & x_2^2 & x_2 & 1 & & & & & & \\
 & & & x_3^2 & x_3 & 1 & & & & & & \\
 & & & & & & \ddots & & & & & \\
 & & & & & & & & & x_{n-1}^2 & x_{n-1} & 1 \\
 & & & & & & & & & x_n^2 & x_n & 1 \\
2x_2 & 1 & 0 & -2x_2 & -1 & & & & & & & \\
 & & & 2x_3 & 1 & 0 & -2x_3 & -1 & & & & \\
 & & & & & & \ddots & & & & & \\
 & & & & & & & 2x_{n-1} & 1 & 0 & -2x_{n-1} & -1 & 0 \\
2x_1 & 1 & & & & & & & & & & &
\end{bmatrix}. \tag{6.2.7}$$

Cubic Splines

For the purpose of approximating solutions to differential equations – as well as for many other situations – it is necessary that the approximating functions be at least twice continuously differentiable. This is not possible with piecewise quadratics unless the data are such that a single quadratic will suffice. We are thus led to consider a piecewise cubic polynomial $c(x)$ with the following properties:

$$c \text{ is twice continuously differentiable} \tag{6.2.8}$$

$$\text{In each interval } I_i = [x_i, x_{i+1}], c \text{ is a cubic polynomial.} \tag{6.2.9}$$

Such a function is called a *cubic spline*, the name being derived from a flexible piece of wood used by draftsmen for drawing curves.

The function c will be represented by

$$c(x) = c_i(x) = a_{i3}x^3 + a_{i2}x^2 + a_{i1}x + a_{i0}, \quad x \in I_i, \quad i = 1, \ldots, n-1. \tag{6.2.10}$$

The condition (6.2.8) implies that both c and c' are also continuous on the whole interval I. Hence we must have

$$c_{i-1}(x_i) = c_i(x_i) \qquad c'_{i-1}(x_i) = c'_i(x_i) \qquad c''_{i-1}(x_i) = c''_i(x_i), \tag{6.2.11}$$

for $i = 2, \ldots, n-1$, which are $3n-6$ conditions. Since there are $4n-4$ unknown coefficients a_{ij} to be obtained for the function c of (6.2.10), we need $n+2$ additional conditions. Especially for the purpose of interpolation or approximation, we will require that c take on the prescribed values

$$c(x_i) = y_i, \qquad i = 1, \ldots, n, \tag{6.2.12}$$

which gives another n conditions. We still need two more conditions, and there are various possibilities for this. The *natural cubic spline* satisfies the additional conditions

$$c''(x_1) = c''(x_n) = 0. \tag{6.2.13}$$

It can be shown that if $\hat{c}$ is any other cubic spline that satisfies (6.2.8), (6.2.9), and (6.2.12), then

$$\int_a^b [c''(x)]^2 dx \leq \int_a^b [\hat{c}''(x)]^2 dx, \tag{6.2.14}$$

so that the natural cubic spline has "minimum curvature."

We could determine c by solving the system of linear equations given by (6.2.11) – (6.2.13) for the unknown coefficients a_{ij}. In this case the coefficient matrix would have a somewhat unwieldy structure similar to (6.2.7). However, for the natural cubic spline there is another approach that will lead to a simple tridiagonal system of equations in which the unknowns are the values of the second derivatives of c at the nodes. Then by integration we can determine c

itself. Obtaining this tridiagonal system requires a good deal of manipulation, which we now begin.

Computation of the Natural Cubic Spline

We first note that c_i'' is linear since c_i is a cubic. Therefore the formula for linear interpolation yields

$$c_i''(x) = c_i''(x_i) + \frac{(x - x_i)}{h_i}[c_i''(x_{i+1}) - c_i''(x_i)], \tag{6.2.15}$$

where we have set $h_i = x_{i+1} - x_i$, $i = 1, \ldots, n-1$. We now integrate (6.2.15) twice to obtain an expression for $c(x)$:

$$\begin{aligned} c_i'(x) &= c_i'(x_i) + \int_{x_i}^{x} c_i''(t)dt = c_i'(x_i) + c_i''(x_i)(x - x_i) \\ &\quad + \frac{[c_i''(x_{i+1}) - c_i''(x_i)]}{2h_i}(x - x_i)^2, \end{aligned} \tag{6.2.16}$$

$$\begin{aligned} c_i(x) &= c_i(x_i) + \int_{x_i}^{x} c_i'(t)dt = c_i(x_i) + c_i'(x_i)(x - x_i) \\ &\quad + c_i''(x_j)\frac{(x - x_i)^2}{2} + \frac{[c_i''(x_{i+1}) - c_i''(x_i)]}{6h_i}(x - x_i)^3. \end{aligned} \tag{6.2.17}$$

For convenience we will henceforth use the notation

$$y_i = c_i(x_i) = c_{i-1}(x_i), \qquad y_i' = c_i'(x_i) = c_{i-1}'(x_i), \tag{6.2.18}$$
$$y_i'' = c_i''(x_i) = c_{i-1}''(x_i),$$

where we have invoked the conditions (6.2.11). Now replace i by $i-1$ in (6.2.16), and then set $x = x_i$ to obtain

$$y_i' = y_{i-1}' + (y_i'' + y_{i-1}'')\frac{h_{i-1}}{2}. \tag{6.2.19}$$

Next, set $x = x_{i+1}$ in (6.2.17) and solve for y_i':

$$y_i' = \frac{y_{i+1} - y_i}{h_i} - y_{i+1}''\frac{h_i}{6} - y_i''\frac{h_i}{3}. \tag{6.2.20}$$

Equating the right-hand sides of (6.2.19) and (6.2.20) gives

$$y_{i-1}' + (y_i'' + y_{i-1}'')\frac{h_{i-1}}{2} = \frac{y_{i+1} - y_i}{h_i} - y_{i+1}''\frac{h_i}{6} - y_i''\frac{h_i}{3}. \tag{6.2.21}$$

We next wish to eliminate y'_{i-1} from (6.2.21). To do this, replace i by $i-1$ in (6.2.20) and substitute the resulting expression for y'_{i-1} into (6.2.21):

$$\frac{y_i - y_{i-1}}{h_{i-1}} - y''_i \frac{h_{i-1}}{6} - y''_{i-1} \frac{h_{i-1}}{3} + (y''_i + y''_{i-1}) \frac{h_{i-2}}{2} = \frac{y_{i+1} - y_i}{h_i} - y''_{i+1} \frac{h_i}{6} - y''_i \frac{h_i}{3}.$$

After rearranging, this becomes

$$y''_{i-1} h_{i-1} + 2y''_i (h_i + h_{i-1}) + y''_{i+1} h_i = \gamma_i, \qquad i = 2, \cdots, n-1, \tag{6.2.22}$$

where

$$\gamma_i = 6 \left[\frac{y_{i+1} - y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}} \right], \qquad i = 2, \ldots, n-1. \tag{6.2.23}$$

Since $y''_1 = y''_n = 0$ by the condition (6.2.13), (6.2.22) is a system of $n-2$ linear equations in the $n-2$ unknowns $y''_2, \ldots, y''_{n-1}$; it is of the form $H\mathbf{y} = \boldsymbol{\gamma}$, where $\mathbf{y} = (y''_2, \ldots, y''_{n-1})$, $\boldsymbol{\gamma} = (\gamma_2, \ldots, \gamma_{n-1})$, and

$$H = \begin{bmatrix} 2(h_1 + h_2) & h_2 & & & \\ h_2 & 2(h_2 + h_3) & h_3 & & \\ & h_3 & \ddots & \ddots & \\ & & \ddots & & h_{n-2} \\ & & & h_{n-2} & 2(h_{n-2} + h_{n-1}) \end{bmatrix}. \tag{6.2.24}$$

The matrix H is tridiagonal and clearly diagonally dominant. (It is also symmetric and positive-definite.) Hence, the system $H\mathbf{y} = \boldsymbol{\gamma}$ can be easily and efficiently solved by Gaussian elimination with no interchanges.

After the y''_i are computed, we still need to obtain the polynomials $c_1, \ldots, c_{n-1}$. The first derivatives y'_i at the node points can be obtained from (6.2.20) since the y_i are also known; thus

$$\begin{aligned} y'_i &= c'_i(x_i) = c'_{i-1}(x_i) \\ &= \frac{y_{i+1} - y_i}{h_i} - y''_{i+1} \frac{h_i}{6} - y''_i \frac{h_i}{3}, \qquad i = 1, \ldots, n-1. \end{aligned} \tag{6.2.25}$$

Then, the c_i themselves can be computed from (6.2.17), which we write in terms of y_i, y'_i, and y''_i for $i = 1, \cdots, n-1$:

$$c_i(x) = y_i + y'_i (x - x_i) + y''_i \frac{(x - x_i)^2}{2} + (y''_{i+1} - y''_i) \frac{(x - x_i)^3}{6h_i}. \tag{6.2.26}$$

Note that if we wish to evaluate c for some particular value $\hat{x}$, we first need to ascertain the interval I_i in which the point $\hat{x}$ lies and thus select the correct c_k for the evaluation.

An Example

We now give a simple example of the computation of a natural cubic spline. Suppose that we have the following nodes and function values:

$$\begin{array}{lllll} x_1 = 0 & x_2 = \frac{1}{4} & x_3 = \frac{1}{2} & x_4 = \frac{3}{4} & x_5 = 1 \\ y_1 = 1 & y_2 = 2 & y_3 = 1 & y_4 = 0 & y_5 = 1. \end{array}$$

Here, $n = 5$ and the h_i are all equal to $\frac{1}{4}$. The matrix H of (6.2.24) and the vector γ whose components are given by (6.2.23) are

$$H = \frac{1}{4}\begin{bmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{bmatrix}, \qquad \gamma = \begin{bmatrix} -48 \\ 0 \\ 48 \end{bmatrix}.$$

Hence, the quantities y_2'', y_3'', and y_4'' are obtained as the solution of the linear system

$$\begin{aligned} 4y_2'' + y_3'' &= -192 \\ y_2'' + 4y_3'' + y_4'' &= 0 \\ y_3'' + 4y_4'' &= 192, \end{aligned}$$

which is easily solved by Gaussian elimination. Since $y_1'' = y_5'' = 0$ by the condition (6.2.13), we have

$$y_1'' = 0, \qquad y_2'' = -48, \qquad y_3'' = 0, \qquad y_4'' = 48, \qquad y_5'' = 0.$$

Using these values of the y_i'', we next compute the y_i' from (6.2.25):

$$y_1' = 6, \quad y_2' = 0, \quad y_3' = -6, \quad y_4' = 0,$$

and with these we obtain the cubic polynomials c_1, c_2, c_3, and c_4 from (6.2.26). This then gives the cubic spline

$$c(x) \begin{cases} = c_1(x) = 1 + 6x - 32x^3, & 0 \le x \le \frac{1}{4}, \\ = c_2(x) = 2 - 24(x - \frac{1}{4})^2 + 32(x - \frac{1}{4})^3, & \frac{1}{4} \le x \le \frac{1}{2}, \\ = c_3(x) = 1 - 6(x - \frac{1}{2}) + 32(x - \frac{1}{2})^3, & \frac{1}{2} \le x \le \frac{3}{4}, \\ = c_4(x) = 24(x - \frac{3}{4})^2 - 32(x - \frac{3}{4})^3, & \frac{3}{4} \le x \le 1. \end{cases}$$

If we wish to evaluate c at some point, say $x = 0.35$, we note that $0.35 \in [\frac{1}{4}, \frac{1}{2}]$, and thus use c_2 for the evaluation:

$$c(0.35) = c_2(0.35) = 2 - 24(0.1)^2 + 32(0.1)^3 = 1.792.$$

In Section 6.4, we will return to the collocation and Galerkin methods using cubic splines as the basis functions. We stress again, however, that spline functions are extremely useful in many types of approximation applications.

Supplementary Discussion and References: 6.2

For further reading on spline functions, see Prenter [1975] and de Boor [1978]. In particular, splines using polynomials of degree higher than cubic are sometimes very useful and are developed in these references.

EXERCISES 6.2

6.2.1. Assume that f is a given function for which the following values are known: $f(1) = 2$, $f(2) = 3$, $f(3) = 5$, $f(4) = 3$, $f(4) = 3$. For these data:

a. Find the interpolating polynomial of degree 3 and write it in the form $a_0 + a_1x + a_2x^2 + a_3x^3$.

b. Find the quadratic spline function that satisfies the condition $q'(1) = 0$. (*Hint:* Start from the left.)

c. Find the cubic spline function that satisfies $c''(1) = 6$, $c''(4) = -9$. (*Hint:* Try the polynomial of part **a.**)

6.2.2. Reorder the unknowns in the system of equations (6.2.7) so as to obtain a coefficient matrix with as small a bandwidth as you can.

6.2.3. Use (6.2.11) – (6.2.13) to write out the system of equations for the unknown coefficients a_{ij} of the cubic spline (6.2.10).

6.2.4. For the function of Exercise 6.2.1, find the cubic spline c that satisfies $c'(1) = 1$, $c'(4) = -1$, rather than the condition (6.2.13). (*Hint:* Think.)

6.2.5. Write a computer program to obtain the natural cubic spline for a given set of nodes $x_1 < \cdots < x_n$ and corresponding function values $y_1, \ldots, y_n$ by first solving the tridiagonal system with the coefficient matrix (6.2.24) and then using (6.2.25) and (6.2.26). Also write a program for evaluating this cubic spline at a given value x. Test your program on the example given in the text.

6.3 Numerical Integration

The Galerkin method described in Section 6.1 requires the evaluation of definite integrals of the form

$$I(f) = \int_a^b f(x)dx,$$

and the need to evaluate such integrals also arises in a number of other problems in scientific computing. The integrand, $f(x)$, may be given in one of three ways:

1. An explicit formula for $f(x)$ is given; for example, $f(x) = (\sin x)e^{-x^2}$.

2. The function $f(x)$ is not given explicitly but can be computed for any value of x in the interval $[a, b]$, usually by means of a computer program.

3. A table of values $\{x_i, f(x_i)\}$ is given for a fixed, finite set of points x_i in the interval.

Functions in the first category are sometimes amenable to methods of symbolic computation, either by hand or by computer systems, although many integrands will not have a "closed form" integral. The integrals of functions that fall into the second and third categories – as well as the first category if symbolic methods are not used – are usually approximated by numerical methods; such methods are called *quadrature rules* and are derived by approximating the function $f(x)$ by some other function, $\tilde{f}(x)$, whose integral is relatively easy to evaluate. Any class of simple functions may be used to approximate $f(x)$, such as polynomials, piecewise polynomials, and trigonometric, exponential, or logarithmic functions. The choice of the class of functions used may depend on some particular properties of the integrand, but the most common choice, which we will use here, is polynomials or piecewise polynomials.

The Newton-Cotes Formulas

The simplest polynomial is a constant. In the *rectangle rule*, f is approximated by its value at the end point a (or, alternatively, at b) so that

$$I(f) \doteq R(f) = (b-a)f(a). \tag{6.3.1}$$

We could also approximate f by another constant obtained by evaluating f at a point interior to the interval; the most common choice is $(a+b)/2$, the center of the interval, which gives the *midpoint rule*

$$I(f) \doteq M(f) = (b-a)f\left(\frac{a+b}{2}\right). \tag{6.3.2}$$

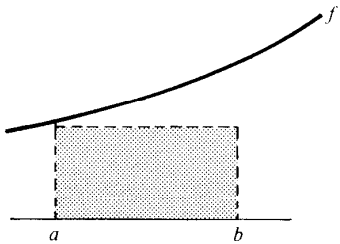
The rectangle and midpoint rules are illustrated in Figure 6.2.

The next simplest polynomial is a linear function. If it is chosen so that it agrees with f at the end points a and b, then a trapezoid is formed, as illustrated in Figure 6.3. The area of this trapezoid – the integral of the linear function – is the approximation to the integral of f and is given by

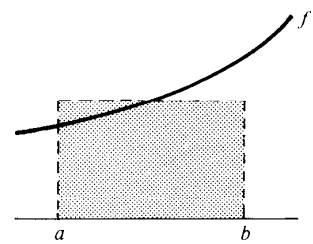
$$I(f) \doteq T(f) = \frac{(b-a)}{2}[f(a) + f(b)]. \tag{6.3.3}$$

This is known as the *trapezoid rule.*

To obtain one further formula, we next approximate f by an interpolating quadratic polynomial that agrees with f at the end points a and b and the

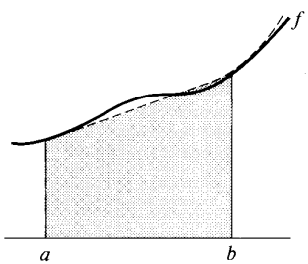


(a) Rectangle rule

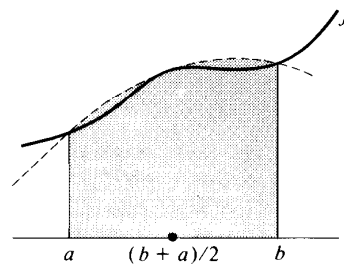


(b) Midpoint rule

Figure 6.2: *Integration Approximations*



(a) Trapezoid rule



(b) Simpson's rule

Figure 6.3: *More Approximations*

midpoint $(a+b)/2$. The integral of this quadratic is given by (see Exercise 6.3.1)

$$I(f) \doteq S(f) = \frac{(b-a)}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right], \tag{6.3.4}$$

which is *Simpson's rule* and is illustrated in Figure 6.3. We note that Simpson's rule may also be viewed as a linear combination of the trapezoid rule and the midpoint rule since

$$\frac{1}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right] = \frac{1}{3}\left[\frac{f(a)+f(b)}{2}\right] + \frac{2}{3}f\left(\frac{a+b}{2}\right).$$

We can continue the preceding method of generating quadrature formulas by using polynomials of still higher degree. The interval $[a,b]$ is divided by m equally spaced points, an interpolating polynomial of degree $m+1$ is

constructed to agree with f at these m points plus the two end points, and this polynomial is then integrated from a to b to give an approximation to the integral. Such quadrature formulas are called the *Newton-Cotes formulas* (See the Supplementary Discussion.)

Error Formulas

We consider next the error made in using the quadrature rules that have been described. In all cases f is approximated by an interpolating polynomial p of degree n over the interval $[a, b]$, and the integral of p is the approximation to the integral. Hence the error in this approximation is

$$E = \int_a^b [f(x) - p(x)]dx. \tag{6.3.5}$$

By the Interpolation Error Theorem 2.3.2, this can be written as

$$E = \frac{1}{(n+1)!} \int_a^b (x - x_0) \cdots (x - x_n) f^{(n+1)}(z(x))dx, \tag{6.3.6}$$

where $x_0, x_1, \ldots, x_n$ are the interpolation points, and $z(x)$ is a point in the interval $[a, b]$ that depends on x. We now apply (6.3.6) to some specific cases.

For the rectangle rule (6.3.1), $n = 0$ and $x_0 = a$; hence (6.3.6) becomes

$$|E_R| = \left| \int_a^b (x - a) f'(z(x))dx \right| \leq M_1 \int_a^b (x - a)dx = \frac{M_1}{2}(b - a)^2, \tag{6.3.7}$$

where M_1 is a bound for $|f'(x)|$ over the interval $[a, b]$. Note that the bound (6.3.7) will not be small unless M_1 is small, which means that f is close to constant, or the length of the interval is small; we shall return to this point later when we discuss the practical use of these quadrature formulas. For the trapezoid rule (6.3.3), $n = 1$, $x_0 = a$, and $x_1 = b$. Hence, again applying (6.3.6), we have

$$|E_T| = \frac{1}{2} \left| \int_a^b (x - a)(x - b) f''(z(x))dx \right| \leq \frac{M_2}{12}(b - a)^3, \tag{6.3.8}$$

where M_2 is a bound on $|f''(x)|$ over $[a, b]$.

Consider next the midpoint rule (6.3.2), in which $n = 0$ and $x_0 = (a+b)/2$. If we apply (6.3.6) and proceed as in (6.3.7), we obtain

$$|E_M| = \left| \int_a^b \left[x - \frac{(a+b)}{2} \right] f'(z(x))dx \right| \leq \frac{M_1}{4}(b - a)^2. \tag{6.3.9}$$

This, however, is not the best bound we can obtain. We shall instead expand the integrand of (6.3.5) in a Taylor series about $m = (a+b)/2$. Since the interpolating polynomial is simply the constant $p(x) = f(m)$, this gives

$$f(x) - p(x) = f'(m)(x-m) + \tfrac{1}{2}f''(z(x))(x-m)^2,$$

where z is a point in the interval and depends on x. Thus the error in the midpoint rule is

$$\begin{aligned} |E_M| &= \left| \int_a^b [f'(m)(x-m)dx + \tfrac{1}{2}f''(z(x))(x-m)^2]dx \right| \qquad (6.3.10) \\ &\leq \left| f'(m) \int_a^b (x-m)dx \right| + \frac{1}{2} \left| \int_a^b f''(z(x))(x-m)^2 dx \right| \\ &\leq \frac{M_2}{24}(b-a)^3, \end{aligned}$$

since

$$\int_a^b (x-m)dx = 0, \qquad \int_a^b (x-m)^2 dx = \frac{(b-a)^3}{12}.$$

In a similar way we can derive the following bound for the error in Simpson's rule (6.3.4), which we state without proof (M_4 is a bound for the fourth derivative):

$$|E_S| \leq \frac{M_4}{2880}(b-a)^5. \qquad (6.3.11)$$

Composite Formulas

The above error bounds all involve powers of the length $b-a$ of the interval, and unless this length is small the bounds will not, in general, be small. However, in practice, we will only apply these quadrature formulas to sufficiently small intervals which we obtain by subdividing the given interval $[a,b]$. Thus we partition the interval $[a,b]$ into n subintervals $[x_{i-1}, x_i]$, $i = 1, \ldots, n$, where $x_0 = a$ and $x_n = b$. Then

$$I(f) = \int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)dx.$$

If we apply the rectangle rule to each subinterval $[x_{i-1}, x_i]$, we obtain the *composite rectangle rule*

$$I(f) \doteq I_{CR}(f) = \sum_{i=1}^{n} h_i f(x_{i-1}), \qquad (6.3.12)$$

where $h_i = x_i - x_{i-1}$. The *composite midpoint, trapezoid,* and *Simpson's rules* are obtained in the same way by applying the basic rule to each subinterval; they are given by

$$I_{CM}(f) = \sum_{i=1}^{n} h_i f\left(\frac{x_i + x_{i-1}}{2}\right), \tag{6.3.13}$$

$$I_{CT}(f) = \sum_{i=1}^{n} \frac{h_i}{2}[f(x_{i-1}) + f(x_i)], \tag{6.3.14}$$

$$I_{CS}(f) = \frac{1}{6}\sum_{i=1}^{n} h_i \left[f(x_{i-1}) + 4f\left(\frac{x_{i-1} + x_i}{2}\right) + f(x_i)\right]. \tag{6.3.15}$$

We note that the composite rules may all be viewed as approximating the integrand f on the interval $[a, b]$ by a piecewise polynomial function (see Section 2.3) and then integrating the piecewise polynomial to obtain an approximation to the integral. For the midpoint and rectangle rules, the approximating function is piecewise constant; for the trapezoid rule it is piecewise linear, and for Simpson's rule it is piecewise quadratic.

We can now apply the previous error bounds on each subinterval. For example, for the rectangle rule, we use (6.3.7) to obtain the following bound on the error in the composite rule:

$$E_{CR} \leq \frac{M_1}{2}\sum_{i=1}^{n} h_i^2. \tag{6.3.16}$$

Note that we have used the maximum M_1 of $|f'(x)|$ on the whole interval $[a, b]$, although a better bound in (6.3.16) could be obtained if we used the maximum of $|f'(x)|$ separately on each subinterval.

In the special case that the subintervals are all of the same length, $h_i = h = (b - a)/n$, (6.3.16) becomes

$$E_{CR} \leq \frac{M_1}{2}(b - a)h \qquad \text{(Composite rectangle rule error)}, \tag{6.3.17}$$

which shows that the composite rectangle rule is a first-order method; that is, the error reduces only linearly in h. In a similar fashion, we can obtain bounds for the errors in the other composite rules by using (6.3.8), (6.3.10), and (6.3.11). The following bounds are given in the case that the intervals are all of the same length h:

$$E_{CM} \leq \frac{M_2}{24}(b - a)h^2 \qquad \text{(Composite midpoint rule error)}, \tag{6.3.18}$$

$$E_{CT} \leq \frac{M_2}{12}(b - a)h^2 \qquad \text{(Composite trapezoid rule error)}, \tag{6.3.19}$$

$$E_{CS} \leq \frac{M_4}{2880}(b-a)h^4 \qquad \text{(Composite Simpson's rule error).} \qquad (6.3.20)$$

Thus the composite midpoint and trapezoid rules are both second order, whereas the composite Simpson's rule is fourth order. Because of its relatively high accuracy and simplicity, the composite Simpson's rule is an often-used method.

Supplementary Discussion and References: 6.3

A difficulty with quadrature rules, as well as with other numerical methods that we have discussed earlier, is that some choice of the step sizes, h_j, must be made. If the numerical integration schemes were to be used as described previously, the user would be required to specify h_j a priori. In practice, high-quality quadrature software will employ some automatic adaptive scheme that will vary the step size depending on estimates of the error obtained during the computation. The user will be required to specify an acceptable tolerance for the error, and the program will automatically specify the step size as it is computing.

The solution at $x = b$ of the initial-value problem

$$y'(x) = f(x), \qquad y(a) = 0, \qquad a \leq x \leq b, \qquad (6.3.21)$$

is $y(b) = \int_a^b f(x)dx$. Hence integration may be viewed as the "trivial" subcase of solving an initial-value problem in which the right-hand side is independent of y. Any of the methods discussed in Chapter 2 may be applied to (6.3.21), in principle. In fact, most of those methods reduce to some quadrature rule that we have discussed. For example, Euler's method is the composite rectangle rule, the second-order Runge-Kutta method is the composite trapezoid rule, and the fourth-order Runge-Kutta method is the composite Simpson's rule (see Exercise 6.3.7).

The Newton-Cotes formulas, mentioned in the text as being derived by integrating an interpolating polynomial of degree n, can be written in the form

$$I(f) \doteq \sum_{i=0}^{n} \alpha_i f(x_i), \qquad (6.3.22)$$

where the x_i are equally spaced points in the interval $[a, b]$, with $x_0 = a$, $x_n = b$; Simpson's rule is the case $n = 2$. For $n \leq 7$, the coefficients α_i are all positive, but beginning with $n = 8$ certain coefficients will be negative; this has a deleterious effect on rounding error since cancellations will occur. The Newton-Cotes formulas also have the unsatisfactory theoretical property that as $n \to \infty$, convergence to the integral will not necessarily occur, even for infinitely differentiable functions.

The representation (6.3.22) provides another approach to the derivation of quadrature formulas — the *method of undetermined coefficients*. Assume first that the x_i are given. If we seek to determine the α_i so that the formula is

exact for polynomials of as high a degree as possible, then in particular it must be exact for $1, x, x^2, \ldots, x^m$, where m is to be as large as possible. This means that we must have

$$\sum_{i=0}^{n} \alpha_i x_i^j = \frac{b^{j+1} - a^{j+1}}{j+1}, \qquad j = 0, 1, \ldots, m, \tag{6.3.23}$$

where the right-hand sides of these relations are the exact integrals of the powers of x. The relations (6.3.23) constitute a system of linear equations for the unknown coefficients α_i. If $m = n$, then the coefficient matrix is the Vandermonde matrix discussed in Section 2.3. It is nonsingular if the x_i are all distinct, and hence the α_i are uniquely determined for $m = n$. If the x_i are equally spaced, then this approach again gives the Newton-Cotes formulas.

Now assume that we do not specify the points x_i in advance but consider them to be unknowns in the relations (6.3.23). Then if $m = 2n + 1$, (6.3.23) is a system of $2n + 2$ equations in the $2n + 2$ unknowns $\alpha_0, \alpha_1, \ldots, \alpha_n$ and $x_0, x_1, \ldots, x_n$. The solution of these equations for the α_i and x_i give the *Gaussian quadrature formulas*. For example, in the case $n = 1$ on the interval $[a, b] = [-1, 1]$, the formula is

$$\int_{-1}^{1} f(x)dx \doteq f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right).$$

In general, the abscissas x_i of these quadrature formulas are roots of certain orthogonal polynomials. Gaussian quadrature rules are popular because of their high-order accuracy, and the weights are always non-negative.

We remarked in the text that Simpson's rule can be viewed as a linear combination of the trapezoid and midpoint rules. By taking suitable linear combinations of the trapezoid rule for different spacings h, we can also derive higher-order quadrature formulas. This is known as *Romberg integration* and is a special case of Richardson extrapolation discussed earlier. The basis for the derivation of Romberg integration is that the trapezoid approximation can be shown to satisfy

$$T(h) = I(f) + C_2 h^2 + C_4 h^4 + \cdots + C_{2m} h^{2m} + 0(h^{2m+2}) \tag{6.3.24}$$

where the C_i depend on f and the interval but are independent of h. The expansion (6.3.24) holds provided that f has $2m + 2$ derivatives. Now define a new approximation to the integral by

$$T_1(h) = \tfrac{1}{3}\left[4T\left(\frac{h}{2}\right) - T(h)\right]. \tag{6.3.25}$$

The coefficients of this linear combination are chosen so that when the error in (6.3.25) is computed using (6.3.24), the coefficient of the h^2 term is zero. Thus

$$T_1(h) = I(f) + C_4^{(1)} h^4 + \cdots + 0(h^{2m+2}),$$

so that T_1 is a fourth-order approximation to the integral. One can continue the process by combining $T_1(h)$ and $T_1(h/2)$ in a similar fashion to eliminate the h^4 term in the error for T_1. More generally, we can construct the triangular array

$$\begin{array}{lll} T(h) & & \\ T(h/2) & T_1(h) & \\ T(h/4) & T_1(h/2) & T_2(h) \\ \vdots & \vdots & \ddots \end{array}$$

where

$$T_k\left(\frac{h}{2^{j-1}}\right) = \frac{4^j T_{k-1}\left(\frac{h}{2^j}\right) - T_{k-1}\left(\frac{h}{2^{j-1}}\right)}{4^j - 1}.$$

The elements in the ith column of this array converge to the integral at a rate depending on h^{2i}. Provided that f is infinitely differentiable, however, the elements on the diagonal of the array converge at a rate that is superlinear, that is, faster than any power of h.

We have not touched at all upon several other important topics in numerical integration: techniques for handling integrands with a singularity, integrals over an infinite interval, multiple integrals, and adaptive procedures that attempt to fit the grid spacing automatically to the integrand. For a discussion of these matters, as well as further reading on the topics covered in this section, see Davis and Rabinowitz [1984] and Stroud [1971].

EXERCISES 6.3

6.3.1. Write down explicitly the interpolating quadratic polynomial that agrees with f at the three points a, b, and $(a+b)/2$. Integrate this quadratic from a to b to obtain Simpson's rule (6.3.4).

6.3.2. Show that the trapezoid rule integrates any linear function exactly and that Simpson's rule integrates any cubic polynomial exactly. (*Hint:* Expand the cubic about the midpoint.)

6.3.3. Apply the rectangle, midpoint, trapezoid, and Simpson's rules to the function $f(x) = x^4$ on the interval $[0, 1]$. Compare the actual error in the approximations to the bounds given by (6.3.7), (6.3.8), (6.3.10), and (6.3.11).

6.3.4. Based on the bound (6.3.19), how small would h need to be to guarantee an error no larger than 10^{-6} in the composite trapezoid rule approximation for $f(x) = x^4$ on $[0, 1]$. How small for the composite Simpson's rule?

6.3.5. Write a computer program to carry out the composite trapezoid and Simpson's rules for an "arbitrary" function on the interval $[a, b]$ and with an arbitrary subdivision of $[a, b]$. Test your program on $f(x) = x^4$ on $[0, 1]$ and

find the actual h needed, in the case of an equal subdivision, to achieve an error of less than 10^{-6} with the composite trapezoid rule. Do the same for $f(x) = e^{-x^2}$.

6.3.6. Derive the four-point quadrature formula based on interpolation of the integrand by a cubic polynomial at equally spaced points. (Hint: Think. The calculation can be simplified somewhat.)

6.3.7. Show that Euler's method applied to the initial value problem (6.3.21) is the composite rectangle rule, that the second-order Runge-Kutta method is the composite trapezoid rule, and that the fourth-order Runge-Kutta method is the composite Simpson rule.

6.4 The Discrete Problem Using Splines

We now return to the original problem (6.1.1), (6.1.2) of this chapter: for the two-point boundary-value problem

$$v''(x) + q(x)v(x) = f(x), \qquad 0 \le x \le 1, \tag{6.4.1}$$

and

$$v(0) = v(1) = 0, \tag{6.4.2}$$

we wish to find an approximate solution of the form

$$u(x) = \sum_{j=1}^{n} c_j \phi_j(x), \tag{6.4.3}$$

where $\phi_1, \ldots, \phi_n$ are given functions.

Collocation

Recall from Section 6.1 that the collocation method for (6.4.1) requires solving the linear system of equations

$$A\mathbf{c} = \mathbf{f}, \tag{6.4.4}$$

where the elements of the matrix A are

$$a_{ij} = \phi_j''(x_i) + q(x_i)\phi_j(x_i), \qquad i, j = 1, \ldots, n, \tag{6.4.5}$$

$\mathbf{c}$ is the vector of unknown coefficients $c_1, \ldots, c_n$, $\mathbf{f}$ is the vector of values $f(x_1), \ldots, f(x_n)$, and $x_1, \ldots, x_n$ are given points in the interval $[0, 1]$. In Section 6.1, we considered the choice of the basis functions ϕ_j as either polynomials or trigonometric functions and saw that, in general, the coefficient matrix A was dense – that is, it had few zero elements – in contrast to the tridiagonal coefficient matrix that was obtained in Chapter 3 using the finite difference

method. In the present section we shall use spline functions, and we will see that in the simplest case this again leads to a tridiagonal coefficient matrix.

Since the coefficients a_{ij} use $\phi''(x_i)$, it is necessary for the basis functions to have a second derivative at the nodes $x_1, \ldots, x_n$. Thus linear and quadratic splines will not suffice; cubic splines, however, are twice differentiable, and we will consider them first as our basis functions. We will need to choose the basis functions so that they are linearly independent in a sense to be made clear shortly. We would also like to choose them so that the coefficient matrix A has as small a bandwidth as possible. To illustrate this last point, let us attempt to make the coefficient matrix tridiagonal. Assuming that the function q has no special properties, we see from (6.4.5) that this will only be achieved if we can choose the ϕ_j so that

$$\phi_j''(x_i) = \phi_j(x_i) = 0, \qquad |i - j| > 1. \tag{6.4.6}$$

This, in turn, will be true if we can choose ϕ_i such that it vanishes identically outside the interval $[x_{i-2}, x_{i+2}]$, and if

$$\phi_i''(x_{i-2}) = \phi_i(x_{i-2}) = \phi_i''(x_{i+2}) = \phi_i(x_{i+2}) = 0. \tag{6.4.7}$$

B-Splines

Now recall that a cubic spline was defined by the conditions (6.2.8) and (6.2.9). These conditions, together with a specification of the function values at the node points $x_1, \ldots, x_n$, give $4n - 6$ relations to determine the $4n - 4$ unknown coefficients that define the cubic spline. In Section 6.2 we used the additional two conditions (6.2.13), which determine a natural cubic spline; unfortunately, this natural cubic spline cannot satisfy the condition (6.4.6) unless it is identically zero. However, if we do not impose the additional conditions (6.2.13), we can obtain a cubic spline that does indeed satisfy the conditions (6.4.6). We denote this spine by $B_i(x)$ and define it explicitly by

$$\begin{aligned}
&\frac{1}{4h^3}(x - x_{i-2})^3, && x_{i-2} \le x \le x_{i-1}, \\
&\frac{1}{4} + \frac{3}{4h}(x - x_{i-1}) + \frac{3}{4h^2}(x - x_{i-1})^2 - \frac{3}{4h^3}(x - x_{i-1})^3, && x_{i-1} \le x \le x_i, \\
&\frac{1}{4} + \frac{3}{4h}(x_{i+1} - x) + \frac{3}{4h^2}(x_{i+1} - x)^2 - \frac{3}{4h^3}(x_{i+1} - x)^3, && x_i \le x \le x_{i+1}, \\
&\frac{1}{4h^3}(x_{i+2} - x)^3, && x_{i+1} \le x \le x_{i+2}, \quad \text{otherwise},
\end{aligned} \tag{6.4.8}$$

where we have now assumed that the node points $x_1, \ldots, x_n$ are equally spaced

with spacing h. It is straightforward (Exercise 6.4.1) to verify that this function is a cubic spline with the function values

$$B_i(x_i) = 1, \qquad B_i(x_{i\pm1}) = \tfrac{1}{4},$$

and zero at the other nodes. Moreover, if $\phi_i = B_i$ the conditions (6.4.6) and (6.4.7) are satisfied (Exercise 6.4.1). Such a spline function, which is illustrated in Figure 6.4, is called a *cubic basis spline*, or *cubic B-spline* for short, since any cubic spline on the interval $[a, b]$ may be written as a linear combination of B-splines. More precisely, we state the following theorem without proof:

Theorem 6.4.1 *Let $c(x)$ be a cubic spline for the equally spaced node points $x_1 < \cdots < x_n$. Then there are constants $\alpha_0, \alpha_1, \ldots, \alpha_{n+1}$ such that*

$$c(x) = \sum_{i=0}^{n+1} \alpha_i B_i(x). \tag{6.4.9}$$

Note that the functions B_0, B_1, B_n, and B_{n+1} used in (6.4.9) require the introduction of the auxiliary grid points x_{-2}, x_{-1}, x_0 and $x_{n+1}, x_{n+2}, x_{n+3}$.

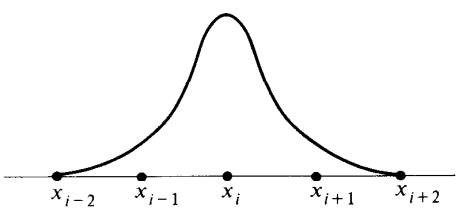


Figure 6.4: *A Cubic B-Spline*

Application to the Boundary-Value Problem

We now return to the boundary-value problem (6.4.1). We assume again that the node points are equally spaced with spacing h, and that $x_1 = 0$, $x_n = 1$. We wish to take the basis functions $\phi_1, \ldots, \phi_n$ to be the B-splines $B_1, \ldots, B_n$. However, although $B_3, \ldots, B_{n-2}$ satisfy the zero boundary conditions, B_1, B_2, B_{n-1}, and B_n do not. Therefore we define the ϕ_i to be

$$\begin{aligned} \phi_i(x) &= B_i(x), \qquad i = 3, \ldots, n-2 \\ \phi_1(x) &= B_1(x) - 4B_0(x), \quad \phi_2(x) = 4B_2(x) - B_1(x), \\ \phi_{n-1}(x) &= 4B_{n-1}(x) - B_n(x), \quad \phi_n(x) = B_n(x) - 4B_{n+1}(x). \end{aligned} \tag{6.4.10}$$

It is easy to verify from the definition (6.4.8) that $\phi_1(0) = \phi_2(0) = \phi_{n-1}(0) = \phi_n(0) = 0$. Moreover, it is clear that any linear combination (6.4.3) is a cubic spline and satisfies $u(0) = u(1) = 0$.

We next need to evaluate the coefficients (6.4.5); for this, we will need B_i, B_i', and B_i'' evaluated at the nodal points. This is easily done (Exercise 6.4.2), and we summarize the results in Table 6.1. Note that by the definition of the B_i, all values at node points not indicated in Table 6.1 are zero. This implies, in particular, that the coefficients a_{ij} of (6.4.5) are all zero unless $|i-j| \leq 1$ or, possibly, if $i = 1$ or n.

Table 6.1: Values of B_i, B_i', B_i''

	x_{i-1}	x_i	x_{i-1}
B_i	$1/4$	1	$1/4$
B_i'	$3/(4h)$	0	$-3/(4h)$
B_i''	$3/(2h^2)$	$-3/h^2$	$3/(2h^2)$

For the evaluation of the nonzero coefficients a_{ij} of (6.4.5), we set $q_i = q(x_i)$. Then using Table 6.1 we have

$$\begin{aligned} a_{ii} &= B_i''(x_i) + q_i B_i(x_i) = \frac{-3}{h^2} + q_i, \quad i = 3, \ldots, n-2, \\ a_{i,i+1} &= B_{i+1}''(x_i) + q_i B_{i+1}(x_i) = \frac{3}{2h^2} + \frac{q_i}{4}, \quad i = 2, \ldots, n-3, \\ a_{i,i-1} &= B_{i-1}''(x_i) + q_i B_{i-1}(x_i) = \frac{3}{2h^2} + \frac{q_i}{4}, \quad i = 4, \ldots, n-1. \end{aligned} \tag{6.4.11}$$

For the remaining coefficients we use the functions ϕ_1, ϕ_2, ϕ_{n-1} and ϕ_n of (6.4.10) and obtain

$$\begin{aligned} a_{11} &= -\frac{9}{h^2}, & a_{n-2,n-1} &= \frac{6}{h^2} + q_{n-2}, \\ a_{12} &= \frac{9}{h^2}, & a_{n-1,n-1} &= \frac{27}{2h^2} + \frac{15}{4} q_{n-1} \\ a_{21} &= \frac{3}{2h^2} - \frac{1}{4} q_2, & a_{n,n-1} &= \frac{3}{2h} + \frac{1}{4} q_{n-1}, \\ a_{22} &= -\frac{27}{2h^2} + \frac{15}{4} q_2, & a_{n,n-1} &= \frac{9}{h^2}, \\ a_{32} &= \frac{6}{h^2} + q_3, & a_{nn} &= -\frac{9}{h^2}. \end{aligned} \tag{6.4.12}$$

The components of the right-hand side $\mathbf{f}$ of the system (6.4.4) are $f(x_1), \ldots, f(x_n)$. Then the solution of the system (6.4.4) with the coefficients of the tridiagonal

matrix A defined by (6.4.11) and (6.4.12) yields the coefficients $c_1, \ldots, c_n$ of the linear combination (6.4.3).

We note that in the special case $q(x) \equiv 0$, so that (6.4.1) is $v''(x) = f(x)$, the coefficient matrix A reduces to

$$A = -\frac{3}{2h^2}\begin{bmatrix} 6 & -6 & & & & & & & & \\ -1 & 9 & -1 & & & & & & & \\ & -4 & 2 & -1 & & & & & & \\ & & -1 & 2 & \ddots & & & & & \\ & & & \ddots & \ddots & & & & & \\ & & & & & & -1 & & & \\ & & & & & -1 & 2 & -4 & & \\ & & & & & & -1 & 9 & -1 & \\ & & & & & & & -6 & 6 & \end{bmatrix},$$

which, aside from the factor $-3/2(h^2)$ and the elements of the first and last three rows and columns, is the $(2, -1)$ coefficient matrix obtained by the finite difference method of Section 3.1.

It can be shown that if the solution v of (6.4.1) is sufficiently differentiable, the preceding method has a discretization error of order h^2, the same as the finite difference method of Section 3.1 using centered differences. Higher-order accuracy can be obtained by using splines of higher degree. For example, fourth-order accuracy can be obtained with quintic splines, functions that are four times continuously differentiable and that reduce to fifth-degree polynomials on each subinterval $[x_i, x_{i+1}]$.

The Galerkin Method

We now turn to the Galerkin method. We will assume again that we wish to approximate a solution of (6.4.1),(6.4.2) by a linear combination of the form (6.4.3), where now the coefficients $c_1, \ldots, c_n$ are determined by the Galerkin criterion discussed in Section 6.1. This leads us to the solution of the linear system (6.4.4) where from (6.1.18)

$$a_{ij} = -\int_0^1 \phi_i'(x)\phi_j'(x)dx + \int_0^1 q(x)\phi_i(x)\phi_j(x)dx, \quad i, j = 1, \ldots, n, \quad (6.4.13)$$

and from (6.1.17)

$$f_i = \int_0^1 f(x)\phi_i(x)dx, \qquad i = 1, \ldots, n. \quad (6.4.14)$$

For the basis functions ϕ_i we will again use piecewise polynomials. As the simplest possibility, let us first consider piecewise linear functions. In particular, assuming that the grid points $0 = x_0, x_1, \ldots, x_n, x_{n+1} = 1$, are

equally spaced with spacing h, we will take the basis functions ϕ_i, $i = 1, \ldots, n$, to be

$$\begin{aligned} \phi_i(x) &= \frac{1}{h}(x - x_{i-1}), && x_{i-1} \le x \le x_i, \\ &= -\frac{1}{h}(x - x_{i+1}), && x_i \le x \le x_{i+1}, \\ &= 0 && x < x_{i-1},\ x > x_{i+1}. \end{aligned} \tag{6.4.15}$$

These particular piecewise linear functions are called *hat functions*, or *linear B-splines*, and are illustrated in Figure 6.5. It is intuitively clear, and easily shown (Exercise 6.4.4), that any piecewise linear function that is defined on the nodes $x_0, x_1, \ldots, x_{n+1}$ and that vanishes at x_0 and x_{n+1} can be expressed as a linear combination of these $\phi_1, \ldots, \phi_n$.

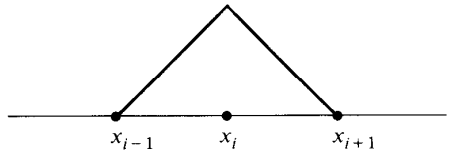


Figure 6.5: *A Hat Function*

We now wish to use the basis functions (6.4.15) in Galerkin's method. At first glance, it would seem that there is a difficulty in using the ϕ_i in the computation of the a_{ij} since this requires ϕ_i', which does not exist at the points x_{i-1}, x_i, and x_{i+1}. Note, however, that ϕ_i' is simply the piecewise constant function

$$\begin{aligned} \phi_i'(x) &= \frac{1}{h}, && x_{i-1} < x < x_i, \\ &= -\frac{1}{h}, && x_i < x < x_{i+1}, \\ &= 0, && x < x_{i-1}, x > x_{i+1}. \end{aligned} \tag{6.4.16}$$

There are discontinuities in this function at the points x_{i-1}, x_i, and x_{i+1}, but these do not affect the integration, and the integrations in (6.4.13) can be carried out on each subinterval to give

$$a_{ij} = \sum_{k=0}^{n} \int_{x_k}^{x_{k+1}} [-\phi_i'(x)\phi_i'(x) + q(x)\phi_i(x)\phi_j(x)]dx. \tag{6.4.17}$$

By the definition of the ϕ_i, the products $\phi_i\phi_j$ and $\phi_i'\phi_j'$ vanish identically unless $i - 1 \le j \le i + 1$. Thus

$$a_{ij} = 0, \qquad \text{if } |i - j| > 1. \tag{6.4.18}$$

To evaluate the other a_{ij} we first introduce the quantities

$$R_i = \int_{x_i}^{x_{i+1}} q(x)(x - x_{i+1})^2 dx, \qquad Q_i = \int_{x_{i-1}}^{x_i} q(x)(x - x_{i-1})^2 dx, \tag{6.4.19}$$

$$S_i = \int_{x_{i-1}}^{x_i} q(x)(x - x_{i-1})(x - x_i) dx,$$

and note that

$$\int_{x_i}^{x_{i+1}} \phi_i'(x)\phi_i'(x)dx = \frac{1}{h}, \qquad \int_{x_{i-1}}^{x_i} \phi_i'(x)\phi_i'(x)dx = \frac{1}{h},$$

$$\int_{x_i}^{x_{i+1}} \phi_i'(x)\phi_{i+1}'(x)dx = -\frac{1}{h}, \qquad \int_{x_i}^{x_{i+1}} q(x)\phi_i(x)\phi_i(x)dx = \frac{1}{h^2}R_i,$$

$$\int_{x_{i-1}}^{x_i} q(x)\phi_i(x)\phi_i(x)dx = \frac{1}{h^2}Q_i, \qquad \int_{x_{i-1}}^{x_i} q(x)\phi_{i-1}(x)\phi_i(x)dx = -\frac{1}{h^2}S_i.$$

Therefore, from (6.4.17),

$$\begin{aligned} a_{ii} &= \frac{1}{h^2}(-2h + Q_i + R_i), && i = 1, \ldots, n, \\ a_{i,i+1} &= \frac{1}{h^2}(h - S_{i+1}), && i = 1, \ldots, n-1, \\ a_{i,i-1} &= \frac{1}{h^2}(h - S_i), && i = 2, \ldots, n, \end{aligned} \tag{6.4.20}$$

and the right-hand side components of (6.4.4) are given for $i = 1, \cdots, n$ by

$$f_i = \frac{1}{h}\int_{x_{i-1}}^{x_i} f(x)(x - x_{i-1})dx + \frac{1}{h}\int_{x_i}^{x_{i+1}} f(x)(x_{i+1} - x)dx. \tag{6.4.21}$$

Thus the linear system (6.4.4) to be solved for the coefficients $c_1, \ldots, c_n$ of (6.4.3) consists of the tridiagonal matrix A whose components are given by (6.4.20), and the right-hand side $\mathbf{f}$ with components given by (6.4.21). We note that unless $q(x)$ and $f(x)$ are such that the integrals in (6.4.19) and (6.4.21) can be evaluated exactly, we would use the numerical integration techniques of the previous section to approximate these integrals. In the special case $q(x) \equiv 0$, all Q_i, R_i, and S_i are zero; hence

$$a_{ii} = \frac{-2}{h} \qquad a_{i,i+1} = \frac{1}{h} \qquad a_{i-1,i} = \frac{1}{h}.$$

If we then multiply the equations (6.4.4) by $-h^{-1}$, the new coefficient matrix will be exactly the $(2, -1)$ tridiagonal matrix that arose in Chapter 3 from the finite difference approximation to $v'' = f$. The right-hand side of the Galerkin equations will be different, however, involving the integrals of f given in (6.4.21).

Provided that the solution of (6.4.1) is sufficiently differentiable, it can be shown that the Galerkin procedure using the piecewise linear functions (6.4.15) is second-order accurate; that is, the discretization error is $0(h^2)$. By using cubic splines it is possible to increase the order of accuracy by two, so as to make the discretization error $0(h^4)$.

Comparison of Methods

We now compare the three methods that we have discussed for two-point boundary-value problems: finite differences, collocation, and Galerkin. The finite difference method is conceptually simple, easy to implement, and yields second-order accuracy with the centered differences that we used in Chapter 3. The collocation method with cubic splines is slightly more difficult to implement but still relatively easy. For the Galerkin method, however, we must evaluate the integrals of (6.4.19) and (6.4.21), and generally this will require the use of numerical integration or symbolic computation systems. In all three cases the system of linear equations to be solved has a tridiagonal coefficient matrix. All three methods have higher-order versions, which are, naturally, more complicated. It is probably fair to say that for ordinary differential equations the simplicity of the finite difference and collocation methods allows them to be preferred in most cases. The power of the Galerkin method becomes more apparent for partial differential equations.

Supplementary Discussion and References: 6.4

The books by Ascher et.al. [1988], deBoor [1978] and Prenter [1975] are good sources for further reading on the material of this section and for a proof of Theorem 6.4.1. See also Strang and Fix [1973] and Hall and Porsching [1990] for further discussion of the Galerkin method.

EXERCISES 6.4

6.4.1. Show that the function defined by (6.4.8) is a cubic spline on the interval $[0, 1]$ and satisfies the conditions (6.4.6) and (6.4.7).

6.4.2. Show that the values of B_i, B_i', and B_i'' at x_{i-1}, x_i, x_{i+1} are as given in Table 6.1.

6.4.3. Consider the two-point boundary-value problem

$$-v'' + (1 + x^2)v = x^2, \quad v(0) = 0, \quad v(1) = 0.$$

a. Let $h = \frac{1}{4}$ and write out the coefficients a_{ij} of (6.4.11) and (6.4.12) for the collocation method and then the complete system of linear equations (6.4.4). Ascertain whether the coefficient matrix is symmetric positive definite and diagonally dominant. Solve the system and express the approximate solution in the form (6.4.3), where the basis functions are given by (6.4.10).

b. Repeat part **a** for the Galerkin method using the basis functions (6.4.15).

6.4.4. Let $f(x)$ be a piecewise linear function with nodes $x_i = ih$, $i = 0, 1, \ldots, n+1$, $h = (n+1)^{-1}$, and that vanishes at x_0 and x_{n+1}. Show that there are constants $\alpha_1, \ldots, \alpha_n$ such that $f(x) = \sum_{i=1}^{n} \alpha_i \phi_i(x)$, where the ϕ_i are defined by (6.4.15).

6.4.5. Consider the boundary-value problem

$$v''(x) - 3v(x) = x^2, \ 0 \leq x \leq 1, \ v(0) = v(1) = 0.$$

a. Derive the system of tridiagonal equations to be solved to carry out the collocation method using the basis functions of (6.4.10) with the points $x_i = (i-1)h, i = 1, \cdots, n, h = 1/(n-1)$.

b. Derive the system of tridiagonal equations to be solved to carry out Galerkin's method using the basis functions (6.4.15).

c. Add the nonlinear term $10[v(x)]^3$ to the right hand side of the differential equation and repeat parts **a.** and **b.** Then compute the Jacobian matrices for these systems and discuss how to carry out Newton's method.