2

Boundary-Value Problems for Ordinary Differential Equations: Discrete Variable Methods

INTRODUCTION

In this chapter we discuss discrete variable methods for solving BVPs for ordinary differential equations. These methods produce solutions that are defined on a set of discrete points. Methods of this type are initial-value techniques, i.e., shooting and superposition, and finite difference schemes. We will discuss initial-value and finite difference methods for linear and nonlinear BVPs, and then conclude with a review of the available mathematical software (based upon the methods of this chapter).

BACKGROUND

One of the most important subdivisions of BVPs is between linear and nonlinear problems. In this chapter linear problems are assumed to have the form

$$y' = F(\mathrm{x})\mathrm{y} + \mathrm{z}(\mathrm{x}), \qquad a < x < b \tag{2.1a}$$

with

$$A\,\mathrm{y}(a) + B\,\mathrm{y}(b) = \gamma \tag{2.1b}$$

where γ is a constant vector, and nonlinear problems to have the form

$$\mathrm{y}' = \mathrm{f}(x,\mathrm{y}), \qquad a < x < b \tag{2.2a}$$

with

$$\mathbf{g}(\mathbf{y}(a), \mathbf{y}(b)) = \mathbf{0} \tag{2.2b}$$

If the number of differential equations in systems (2.1a) or (2.2a) is n, then the number of independent conditions in (2.1b) and (2.2b) is n.

In practice, few problems occur naturally as first-order systems. Most are posed as higher-order equations that can be converted to a first-order system. All of the software discussed in this chapter require the problem to be posed in this form.

Equations (2.1b) and (2.2b) are called boundary conditions (BCs) since information is provided at the ends of the interval, i.e., at $x = a$ and $x = b$. The conditions (2.1b) and (2.2b) are called nonseparated BCs since they can involve a combination of information at $x = a$ and $x = b$. A simpler situation that frequently occurs in practice is that the BCs are separated; that is, (2.1b) and (2.2b) can be replaced by

$$A\ \mathbf{y}(a) = \boldsymbol{\gamma}_1, \qquad B\ \mathbf{y}(b) = \boldsymbol{\gamma}_2 \tag{2.3}$$

where $\boldsymbol{\gamma}_1$ and $\boldsymbol{\gamma}_2$ are constant vectors, and

$$\mathbf{g}_1(\mathbf{y}\ (a)) = \mathbf{0}, \qquad \mathbf{g}_2(\mathbf{y}\ (b)) = \mathbf{0} \tag{2.4}$$

respectively, where the total number of independent conditions remains equal to n.

INITIAL-VALUE METHODS

Shooting Methods

We first consider the single linear second-order equation

$$Ly \equiv -y'' + p(x)y' + q(x)y = r(x), \qquad a < x < b \tag{2.5a}$$

with the general linear two-point boundary conditions

$$\begin{aligned} a_0 y(a) - a_1 y'(a) &= \alpha \\ b_0 y(b) + b_1 y'(b) &= \beta \end{aligned} \tag{2.5b}$$

where a_0, a_1, α, b_0, b_1, and β are constants, such that

$$\begin{aligned} &a_0 a_1 \geqslant 0, \qquad |a_0| + |a_1| \neq 0 \\ &b_0 b_1 \geqslant 0, \qquad |b_0| + |b_1| \neq 0 \\ &|a_0| + |b_0| \neq 0 \end{aligned} \tag{2.5c}$$

We assume that the functions $p(x)$, $q(x)$, and $r(x)$ are continuous on $[a, b]$ and that $q(x) > 0$. With this assumption [and (2.5c)] the solution of (2.5) is unique

[1]. To solve (2.5) we first define two functions, $y^{(1)}(x)$ and $y^{(2)}(x)$, on $[a, b]$ as solutions of the respective initial-value problems

$$Ly^{(1)} = r(x), \qquad y^{(1)}(a) = -\alpha C_1, \qquad y^{(1)\prime}(a) = -\alpha C_0 \tag{2.6a}$$

$$Ly^{(2)} = 0, \qquad y^{(2)}(a) = a_1, \qquad y^{(2)\prime}(a) = a_0 \tag{2.6b}$$

where C_0 and C_1 are any constants such that

$$a_1 C_0 - a_0 C_1 = 1 \tag{2.7}$$

The function $y(x)$ defined by

$$y(x) \equiv y(x; s) = y^{(1)}(x) + s y^{(2)}(x), \qquad a \leqslant x \leqslant b \tag{2.8}$$

satisfies $a_0 y(a) - a_1 y'(a) = \alpha(a_1 C_0 - a_0 C_1) = \alpha$, and will be a solution of (2.5) if s is chosen such that

$$\phi(s) = b_0 y(b; s) + b_1 y'(b; s) - \beta = 0 \tag{2.9}$$

This equation is linear in s and has the single root

$$s = \frac{\beta - [b_0 y^{(1)}(b) + b_1 y^{(1)\prime}(b)]}{[b_0 y^{(2)}(b) + b_1 y^{(2)\prime}(b)]} \tag{2.10}$$

Therefore, the method involves:

1. Converting the BVP into an IVP by specifying extra initial conditions	1. (2.5) to (2.6)
2. Guessing the initial conditions and solving the IVP over the entire interval	2. Guess C_0, evaluate C_1 from (2.7), and solve (2.6)
3. Solving for s and constructing y.	3. Evaluate (2.10) for s; use s in (2.8)

The shooting method consists in simply carrying out the above procedure numerically; that is, compute approximations to $y^{(1)}(x)$, $y^{(1)\prime}(x)$, $y^{(2)}(x)$, $y^{(2)\prime}(x)$ and use them in (2.8) and (2.10). To solve the initial-value problems (2.6), first write them as equivalent first-order systems:

$$\begin{bmatrix} w^{(1)} \\ v^{(1)} \end{bmatrix}' = \begin{bmatrix} v^{(1)} \\ pv^{(1)} + qw^{(1)} - r \end{bmatrix} \tag{2.11}$$
$$w^{(1)}(a) = -\alpha C_1, \qquad v^{(1)}(a) = -\alpha C_0$$

and

$$\begin{bmatrix} w^{(2)} \\ v^{(2)} \end{bmatrix}' = \begin{bmatrix} v^{(2)} \\ pv^{(2)} + qw^{(2)} \end{bmatrix} \tag{2.12}$$
$$w^{(2)}(a) = a_1, \qquad v^{(2)}(a) = a_0$$

respectively. Now any of the methods discussed in Chapter 1 can be employed to solve (2.11) and (2.12).

Let the numerical solutions of (2.11) and (2.12) be

$$W_i^{(1)},\ V_i^{(1)},\ W_i^{(2)},\ V_i^{(2)}, \qquad i = 0, 1, \ldots, N \tag{2.13}$$

respectively, for

$$x_i = a + ih, \qquad i = 0, 1, \ldots, N$$

$$h = \frac{b - a}{N}$$

At the point $x_0 = a$, the exact data can be used so that

$$\begin{aligned} W_0^{(1)} &= -\alpha C_1, & W_0^{(2)} &= -\alpha C_0 \\ V_0^{(1)} &= a_1, & V_0^{(2)} &= a_0 \end{aligned} \tag{2.14}$$

To approximate the solution $y(x)$, set

$$Y_i = W_i^{(1)} + SW_i^{(2)} \tag{2.15}$$

where

$$Y_i \simeq y(x_i)$$

$$S = \frac{\beta - [b_0 W_N^{(1)} + b_1 V_N^{(1)}]}{[b_0 W_N^{(2)} + b_1 V_N^{(2)}]} \tag{2.16}$$

This procedure can work well but is susceptible to round-off errors. If $W_i^{(1)}$ and $W_i^{(2)}$ in (2.15) are nearly equal and of opposite sign for some range of i values, cancellation of the leading digits in Y_i can occur.

Keller [1] posed the following example to show how cancellation of digits can occur. Suppose that the solution of the IVP (2.6) grows in magnitude as $x \rightarrow b$ and that the boundary condition at $x = b$ has $b_1 = 0$ [$y(b) = \beta$ is specified]. Then if $|\beta| << |b_0 W_N^{(1)}|$

$$S \simeq -\frac{W_N^{(1)}}{W_N^{(2)}} \tag{2.17}$$

and

$$Y_i \simeq W_i^{(1)} - \left[\frac{W_N^{(1)}}{W_N^{(2)}}\right] W_i^{(2)} \tag{2.18}$$

Clearly the cancellation problem occurs here for x_i near b. Note that the solution $W_i^{(1)}$ need not grow very fast, and in fact for $\beta = 0$ the difficulty is always potentially present. If the loss of significant digits cannot be overcome by the use of double precision arithmetic, then multiple-shooting techniques (discussed later) can be employed.

We now consider a second-order nonlinear equation of the form

$$y'' = f(x, y, y'), \qquad a < x < b \tag{2.19a}$$

subject to the general two-point boundary conditions

$$\begin{aligned} a_0 y(a) - a_1 y'(a) &= \alpha, \qquad a_i \geqslant 0 \\ b_0 y(b) + b_1 y'(b) &= \beta, \qquad b_i \geqslant 0 \\ a_0 + b_0 &> 0 \end{aligned} \tag{2.19b}$$

The related IVP is

$$u'' = f(x, u, u'), \qquad a < x < b \tag{2.20a}$$

$$\begin{aligned} u(a) &= a_1 s - c_1 \alpha \\ u'(a) &= a_0 s - c_0 \alpha \end{aligned} \tag{2.20b}$$

where

$$a_1 c_0 - a_0 c_1 = 1$$

The solution of (2.20), $u = u(x; s)$, will be a solution of (2.19) if s is a root of

$$\phi(s) = b_0 u(b; s) + b_1 u'(b; s) - \beta = 0 \tag{2.21}$$

To carry out this procedure numerically, convert (2.20) into a first-order system:

$$\begin{bmatrix} w \\ v \end{bmatrix}' = \begin{bmatrix} v \\ f(x, w, v) \end{bmatrix} \tag{2.22a}$$

with

$$\begin{aligned} w(a) &= a_1 s - c_1 \alpha \\ v(a) &= a_0 s - c_0 \alpha \end{aligned} \tag{2.22b}$$

In order to find s, one can apply Newton's method to (2.21), giving

$$\begin{aligned} s^{[k+1]} &= s^{[k]} - \frac{\phi(s^{[k]})}{\phi'(s^{[k]})}, \qquad k = 0, 1, \ldots \\ s^{[0]} &= \text{arbitrary} \end{aligned} \tag{2.23}$$

To find $\phi'(s)$, first define

$$\xi(x) = \frac{\partial w(x; s)}{\partial s} \quad \text{and} \quad \eta(x) = \frac{\partial v(x; s)}{\partial s} \tag{2.24}$$

Differentiation of (2.22) with respect to s gives

$$\begin{aligned} \xi' &= \eta, & \xi(a) &= a_1 \\ \eta' &= \frac{\partial f}{\partial v}\eta + \frac{\partial f}{\partial w}\xi, & \eta(a) &= a_0 \end{aligned} \tag{2.25}$$

Solution of (2.25) allows for calculation of ϕ' as

$$\phi' = b_0\,\xi(b;s) + b_1\,\eta(b;s) \tag{2.26}$$

Therefore, the numerical solution of (2.25) would be computed along with the numerical solution of (2.22). Thus, one iteration in Newton's method (2.23) requires the solution of two initial-value problems.

EXAMPLE 1

An important problem in chemical engineering is to predict the diffusion and reaction in a porous catalyst pellet. The goal is to predict the overall reaction rate of the catalyst pellet. The conservation of mass in a spherical domain gives

$$D\left[\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{dc}{dr}\right)\right] = k\mathscr{R}(c), \qquad 0 < r < r_p$$

where

r = radial coordinate (r_p = pellet radius)
D = diffusivity
c = concentration of a given chemical
k = rate constant
$\mathscr{R}(c)$ = reaction rate function

with

$$\frac{dc}{dr} = 0 \quad \text{at} \quad r = 0 \qquad \text{(symmetry about the origin)}$$

$$c = c_0 \quad \text{at} \quad r = r_p \qquad \text{(concentration fixed at surface)}$$

If the pellet is isothermal, an energy balance is not necessary. We define the effectiveness factor E as the average reaction rate in the pellet divided by the average reaction rate if the rate of reaction is evaluated at the surface. Thus

$$E = \frac{\int_0^{r_p} \mathscr{R}(c(r))r^2\,dr}{\int_0^{r_p} \mathscr{R}(c_0)r^2\,dr}$$

We can integrate the mass conservation equation to obtain

$$D\int_0^{r_p}\left[\frac{1}{r^2}\frac{d}{dr}\left(r^2\frac{dc}{dr}\right)\right]r^2\,dr = k\int_0^{r_p}\mathscr{R}(c)r^2\,dr = Dr_p^2\left.\frac{dc}{dr}\right|_{r_p}$$

Hence the effectiveness factor can be rewritten as

$$E = \frac{3r_p^2\,D\left.\dfrac{dc}{dr}\right|_{r_p}}{k\mathscr{R}(c_0)}$$

If $E = 1$, then the overall reaction rate in the pellet is equal to the surface value and mass transfer has no limiting effects on the reaction rate. When $E < 1$, then mass transfer effects have limited the overall rate in the pellet; i.e., the average reaction rate in the pellet is lower than the surface value of the reaction rate because of the effects of diffusion.

Now consider a sphere (5 mm in diameter) of γ-alumina upon which Pt is dispersed in order to catalyze the dehydrogenation of cyclohexane. At 700 K, the rate constant k is 4 s^{-1}, and the diffusivity D is 5×10^{-2} cm^2/s. Set up the equations necessary to calculate the concentration profile of cyclohexane within the pellet and also the effectiveness factor for a general $\mathcal{R}(c)$. Next, solve these equations for $\mathcal{R}(c) = c$, and compare the results with the analytical solution.

SOLUTION

Define

$$C = \frac{\text{concentration of cyclohexane}}{\text{concentration of cyclohexane at the surface of the sphere}}$$

$$R = \text{dimensionless radial coordinate based on the radius of the sphere } (r_p = 2.5\text{ mm})$$

Assume that the spherical pellet is isothermal. The conservation of mass equation for cyclohexane is

$$\frac{d^2C}{dR^2} + \frac{2}{R}\frac{dC}{dR} = \Phi^2 \frac{\mathcal{R}(c)}{c_0}, \qquad 0 < R < 1,$$

with

$$\frac{dC}{dR} = 0 \quad \text{at} \quad R = 0 \qquad \text{(due to symmetry)}$$

$$C = 1 \quad \text{at} \quad R = 1 \qquad \text{(by definition)}$$

where

$$\Phi = r_p\sqrt{\frac{k}{D}} \qquad \text{(Thiele modulus)}$$

Since $\mathcal{R}(c)$ is a general function of c, it may be nonlinear in c. Therefore, assume that $\mathcal{R}(c)$ is nonlinear and rewrite the conservation equation in the form of (2.19):

$$\frac{d^2C}{dR^2} = \Phi^2 \frac{\mathcal{R}(c)}{c_0} - \frac{2}{R}\frac{dC}{dR} = f(R, C, C')$$

The related IVP systems become

$$\begin{bmatrix} w \\ v \end{bmatrix}' = \begin{bmatrix} v \\ \Phi^2 \dfrac{\mathscr{R}(\mathrm{c})}{\mathrm{c}_0} - \dfrac{2}{R} v \end{bmatrix}, \quad \begin{bmatrix} \xi \\ \eta \end{bmatrix}' = \begin{bmatrix} \eta \\ \Phi^2 \dfrac{d}{dw}\left(\dfrac{\mathscr{R}(\mathrm{c})}{\mathrm{c}_0}\right) - \dfrac{2}{R} \eta \end{bmatrix}$$

with

$$\begin{aligned} w(0) &= s, \qquad & \xi(0) &= 1 \\ v(0) &= 0, \qquad & \eta(0) &= 0 \end{aligned}$$

and

$$\begin{aligned} \phi(s) &= w(1; s) - 1 \\ \phi'(s) &= \xi(1; s) \end{aligned}$$

Choose $s^{[0]}$, and solve the above system of equations to obtain a solution. Compute a new s by

$$s^{[k+1]} = s^{[k]} - \frac{w(1; s^{[k]}) - 1}{\xi(1; s^{[k]})}, \qquad k = 0, 1, \ldots$$

and repeat until convergence is achieved.

Using the data provided, we get $\Phi = 2.236$. If $\mathscr{R}(c) = c$, then the problem is linear and no Newton iteration is required. The IMSL routine DVERK (see Chapter 1) was used to integrate the first-order system of equations. The results, along with the analytical solution calculated from [2],

$$C = \frac{\sinh(\Phi R)}{R \sinh(\Phi)}$$

are shown in Table 2.1. Notice that the computed results are the same as the analytical solution (to four significant figures). In Table 2.1 we also compare

TABLE 2.1 Results from Example 1
TOL = 10^{-6} for DVERK

R	C, Analytical Solution	C, Computed Solution (s = 0.4835)
0.0	0.4835	0.4835
0.2	0.4998	0.4998
0.4	0.5506	0.5506
0.6	0.6422	0.6422
0.8	0.7859	0.7859
1.0	1.0000	1.0000
E	0.7726	0.7727

the computed value of E, which is defined as

$$E = \frac{3 \left.\dfrac{dC}{dR}\right|_1}{\Phi^2}$$

with the analytical value from [2],

$$E = \frac{3}{\Phi}\left[\frac{1}{\tanh(\Phi)} - \frac{1}{\Phi}\right]$$

Again, the results are quite good.

Physically, one would expect the concentration of cyclohexane to decrease as R decreases since it is being consumed by reaction. Also, notice that the concentration remains finite at $R = 0$. Therefore, the reaction has not gone to completion in the center of the catalytic pellet. Since $E < 1$, the average reaction rate in the pellet is less than the surface value, thus showing the effects of mass transfer.

EXAMPLE 2

If the system described in Example 1 remains the same except for the fact that the reaction rate function now is second-order, i.e., $\mathscr{R}(c) = c^2$, compute the concentration profile of cyclohexane and calculate the value of the effectiveness factor. Let $c_0 = 1$.

SOLUTION

The material balance equation is now

$$\frac{d^2C}{dR^2} + \frac{2}{R}\frac{dC}{dR} = \Phi^2 C^2, \qquad 0 < R < 1$$

$$\frac{dC}{dR} = 0 \quad \text{at} \quad R = 0$$

$$C = 1 \quad \text{at} \quad R = 1$$

$$\Phi = 2.236$$

The related IVP systems are

$$\begin{bmatrix} w \\ v \end{bmatrix}' = \begin{bmatrix} v \\ \Phi^2 w^2 - \dfrac{2}{R} v \end{bmatrix}, \qquad \begin{bmatrix} \zeta \\ \eta \end{bmatrix}' = \begin{bmatrix} \eta \\ 2\Phi^2 w\zeta - \dfrac{2}{R} \eta \end{bmatrix}$$

with

$$w(0) = s, \qquad \xi(0) = 1$$

$$v(0) = 0, \qquad \eta(0) = 0$$

and

$$\phi(s) = w(1; s) - 1$$

$$\phi'(s) = \xi(1; s)$$

The results are shown in Table 2.2. Notice the effect of the tolerances set for DVERK (TOLD) and on the Newton iteration (TOLN). At TOLN = 10^{-3}, the convergence criterion was not sufficiently small enough to match the boundary condition at $R = 1.0$. At TOLN = 10^{-6} the boundary condition at $R = 1$ was achieved. Decreasing either TOLN or TOLD below 10^{-6} produced the same results as shown for TOLN = TOLD = 10^{-6}.

In the previous two examples, the IVPs were not stiff. If a stiff IVP arises in a shooting algorithm, then a stiff IVP solver, for example, LSODE (MF = 21), would have to be used to perform the integration.

Systems of BVPs can be solved by initial-value techniques by first converting them into an equivalent system of first-order equations. Consider the system

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \qquad a < x < b \tag{2.27a}$$

with

$$A\,\mathbf{y}(a) + B\,\mathbf{y}(b) = \boldsymbol{\alpha} \tag{2.27b}$$

or more generally

$$\mathbf{g}(\mathbf{y}(a), \mathbf{y}(b)) = \mathbf{0} \tag{2.27c}$$

The associated IVP is

$$\mathbf{u}' = \mathbf{f}(x, \mathbf{u}) \tag{2.28a}$$

$$\mathbf{u}(a) = \mathbf{s} \tag{2.28b}$$

where

$$\mathbf{s} = \text{vector of unknowns}$$

TABLE 2.2 Results from Example 2

R	C,
	TOLD = 10^{-3}†
	TOLN = 10^{-3}‡
		C,
		TOLD = 10^{-6}
		TOLN = 10^{-3}
			C,
			TOLD = 10^{-6}
			TOLN = 10^{-6}

0.0	0.5924	0.5924	0.5921
0.2	0.6042	0.6042	0.6039
0.4	0.6415	0.6415	0.6411
0.6	0.7101	0.7101	0.7096
0.8	0.8220	0.8220	0.8214
1.0	1.0008	1.0008	1.0000
E	0.6752	0.6752	0.6742
s	0.5924	0.5924	0.5921

† Tolerance for DVERK.

‡ Tolerance on Newton iteration.

We now seek $\mathbf{s}$ such that $\mathbf{u}(x; \mathbf{s})$ is a solution of (2.27). This occurs if $\mathbf{s}$ is a root of the system

$$\boldsymbol{\phi}(\mathbf{s}) = A\,\mathbf{s} + B\,\mathbf{u}(b; \mathbf{s}) - \boldsymbol{\alpha} = \mathbf{0} \tag{2.29}$$

or more generally

$$\boldsymbol{\phi}(\mathbf{s}) = \mathbf{g}(\mathbf{s}, \mathbf{u}(b; \mathbf{s})) = \mathbf{0} \tag{2.30}$$

Thus far we have only discussed shooting methods that "shoot" from $x = a$. Shooting can be applied in either direction. If the solutions of the IVP grow from $x = a$ to $x = b$, then it is likely that the shooting method will be most effective in reverse, that is, using $x = b$ as the initial point. This procedure is called reverse shooting.

Multiple Shooting

Previously we have discussed some difficulties that can arise when using a shooting method. Perhaps the best known difficulty is the loss in accuracy caused by the growth of solutions of the initial-value problem. Multiple shooting attempts to prevent this problem. Here, we outline multiple-shooting methods that are used in software libraries.

Multiple shooting is designed to reduce the growth of the solutions of the IVPs that must be solved. This is done by partitioning the interval into a number of subintervals, and then simultaneously adjusting the "initial" data in order to satisfy the boundary conditions and appropriate continuity conditions. Consider a system of n first-order equations of the form (2.27), and partition the interval as

$$a = x_0 < x_1 < \ldots < x_{N-1} < x_N = b \tag{2.31}$$

Define

$$\begin{aligned} h_i &= x_i - x_{i-1} \\ t &= \frac{x - x_{i-1}}{h_i} \\ \boldsymbol{\tau}_i(t) &= \mathbf{y}(x) = \mathbf{y}(x_{i-1} + th_i) \\ \mathbf{r}_i(t, \boldsymbol{\tau}_i) &= h_i\,\mathbf{f}(x_{i-1} + th_i, \boldsymbol{\tau}_i) \end{aligned} \tag{2.32}$$

for

$$i = 1, 2, \ldots, N$$

With this change of variables, (2.27) becomes

$$\frac{d\boldsymbol{\tau}_i}{dt} = \mathbf{r}_i(t, \mathbf{r}_i), \qquad 0 < t < 1 \tag{2.33}$$

for

$$i = 1, 2, \ldots, N$$

The boundary conditions are now

$$A\,\boldsymbol{\tau}_1(0) + B\,\boldsymbol{\tau}_N(1) = \boldsymbol{\alpha} \qquad \text{[for (2.27b)]} \tag{2.34a}$$

or

$$\mathbf{g}(\boldsymbol{\tau}_1(0), \boldsymbol{\tau}_N(1)) = \mathbf{0} \qquad \text{[for (2.27c)]} \tag{2.34b}$$

In order to have a continuous solution to (2.27), we require

$$\boldsymbol{\tau}_{i+1}(0) - \boldsymbol{\tau}_i(1) = \mathbf{0}, \qquad i = 1, 2, \ldots, N-1 \tag{2.35}$$

The N systems of n first-order equations can thus be written as

$$\frac{d}{dt}\boldsymbol{\psi} = \mathcal{R}(t, \boldsymbol{\psi}) \tag{2.36}$$

with

$$P\,\boldsymbol{\psi}(0) + Q\,\boldsymbol{\psi}(1) = \boldsymbol{\gamma}$$

or

$$\mathbf{G} = \mathbf{0}$$

where

$$\begin{aligned}
\boldsymbol{\psi} &= [\boldsymbol{\tau}_1(t), \boldsymbol{\tau}_2(t), \ldots, \boldsymbol{\tau}_N(t)]^T \\
\mathcal{R}(t, \boldsymbol{\psi}) &= [\mathbf{r}_1(t, \boldsymbol{\tau}_1), \mathbf{r}_2(t, \boldsymbol{\tau}_2), \ldots, \mathbf{r}_N(t, \boldsymbol{\tau}_N)]^T \\
\boldsymbol{\gamma} &= [\boldsymbol{\alpha}, \mathbf{0}, \ldots, \mathbf{0}]^T \\
\mathbf{0} &= [\mathbf{0}, \mathbf{0}, \ldots, \mathbf{0}]^T
\end{aligned}$$

$$P = \begin{bmatrix} A & & & & \\ & I & & \mathbf{0} & \\ & & \cdot & & \\ & \mathbf{0} & & \cdot & \\ & & & & \cdot \\ & & & & I \end{bmatrix}$$

$$Q = \begin{bmatrix} \mathbf{0} & & & & B \\ -I & \mathbf{0} & & & \\ & \cdot & \cdot & & \\ & & \cdot & \cdot & \\ & & & \cdot & \cdot \\ & & & -I & \mathbf{0} \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} \boldsymbol{\tau}_2(0) - \boldsymbol{\tau}_1(1) \\ \boldsymbol{\tau}_3(0) - \boldsymbol{\tau}_2(1) \\ \vdots \\ \boldsymbol{\tau}_N(0) - \boldsymbol{\tau}_{N-1}(1) \\ \mathbf{g}(\boldsymbol{\tau}_1(0), \boldsymbol{\tau}_N(1)) \end{bmatrix}$$

The related IVP problem is

$$\frac{d}{dt}\mathbf{U} = \mathcal{R}(t, \mathbf{U}), \qquad 0 < t < 1 \tag{2.37}$$

with

$$\mathbf{U}(0) = \mathbf{S}$$

where

$$\mathbf{S} = [\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_N]^T$$

$$\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_N]^T$$

The solution of (2.37) is a solution to (2.36) if **S** is a root of

$$\boldsymbol{\Phi}(\mathbf{S}) = P\,\mathbf{S} + Q\,\mathbf{U}(1; \mathbf{S}) - \boldsymbol{\gamma} = \mathbf{0}$$

or (2.38)

$$\boldsymbol{\Phi}(\mathbf{S}) = \mathbf{G} = \mathbf{0}$$

depending on whether the BCs are of form (2.27b) or (2.27c). The solution procedure consists of first guessing the "initial" data **S**, then applying ordinary shooting on (2.37) while also performing a Newton iteration on (2.38). Obviously, two major considerations are the mesh selection, i.e., choosing x_i, $i = 1, \ldots, N - 1$, and the starting guess for **S**. These difficulties will be discussed in the section on software.

An alternative shooting procedure would be to integrate in both directions up to certain matching points. Formally speaking, this method includes the previous method as a special case. It is not clear a priori which method is preferable [3].

Superposition

Another initial-value method is called superposition and is discussed in detail by Scott and Watts [4]. We will outline the method for the following linear equation

$$\mathbf{y}'(x) = F(x)\mathbf{y}(x) + \mathbf{g}(x), \qquad a < x < b \tag{2.39a}$$

with

$$A\,\mathbf{y}(a) = \boldsymbol{\alpha} \qquad \text{(2.39b)}$$

$$B\,\mathbf{y}(b) = \boldsymbol{\beta}$$

The technique consists of finding a solution $\mathbf{y}(x)$ such that

$$\mathbf{y}(x) = \mathbf{v}(x) + U(x)\mathbf{c} \tag{2.40}$$

where the matrix U satisfies

$$U'(x) = F(x)U(x) \tag{2.41a}$$

$$A\,U(a) = \mathbf{0} \tag{2.41b}$$

the vector $\mathbf{v}(x)$ satisfies

$$\mathbf{v}'(x) = F(x)\,\mathbf{v}(x) + \mathbf{g}(x) \tag{2.42a}$$

$$\mathbf{v}(a) = \boldsymbol{\alpha} \tag{2.42b}$$

and the vector of constants $\mathbf{c}$ is chosen to satisfy the boundary conditions at $x = b$:

$$B\,U(b)\mathbf{c} = -B\,\mathbf{v}(b) + \boldsymbol{\beta} \tag{2.43}$$

The matrix $U(x)$ is often referred to as the fundamental solution, and the vector $\mathbf{v}(x)$ the particular solution.

In order for the method to yield accurate results, $\mathbf{v}(x)$ and the columns of $U(x)$ must be linearly independent [5]. The initial conditions (2.41b) and (2.42b) theoretically ensure independence; however, due to the finite world length used by computers, the solutions may lose their numerical independence (see [5] for full explanation). When this happens, the resulting matrix problem (2.43) may give inaccurate answers for $\mathbf{c}$. Frequently, it is impossible to extend the precision of the computations in order to overcome this difficulty. Therefore, the basic superposition method must be modified.

Analogous to using multiple shooting to overcome the difficulties with shooting methods, one can modify the superposition method by subdividing the interval as in (2.31), and then defining a superposition solution on each subinterval by

$$\begin{aligned} \mathbf{y}_i(x) &= \mathbf{v}_i(x) + U_i(x)\mathbf{c}_i(x), \qquad x_{i-1} \leqslant x \leqslant x_i \\ i &= 1, 2, \ldots, N, \end{aligned} \tag{2.44}$$

where

$$\begin{aligned} U_i'(x) &= F(x)\,U_i(x) \\ U_i(x_{i-1}) &= U_{i-1}(x_{i-1}), \qquad A\,U_1(a) = \mathbf{0} \end{aligned} \tag{2.45}$$

$$\begin{aligned} \mathbf{v}_i'(x) &= F(x)\mathbf{v}_i(x) + \mathbf{g}(x) \\ \mathbf{v}_i(x_{i-1}) &= \mathbf{v}_{i-1}(x_{i-1}), \qquad \mathbf{v}_1(a) = \boldsymbol{\alpha} \end{aligned} \tag{2.46}$$

and

$$\mathbf{y}_i(x_i) = \mathbf{y}_{i+1}(x_i) \tag{2.47}$$

$$B\,U_N(b)\mathbf{c}_N = -B\,\mathbf{v}_N(b) + \boldsymbol{\beta} \tag{2.48}$$

The principle of the method is then to piece together the solutions defined on the various subintervals to obtain the desired solution. At each of the mesh points x_i the linear independence of the solutions must be checked. One way to guarantee independence of solutions over the entire interval is to keep them nearly orthogonal. Therefore, the superposition algorithm must be coupled with a routine that checks for orthogonality of the solutions, and each time the vectors start to lose their linear independence, they must be orthonormalized [4,5] to regain linear independence. Obviously, one of the major problems in implementing this method is the location of the orthonormalization points x_i.

Nonlinear problems can also be solved using superposition, but they first must be "linearized." Consider the following nonlinear BVP:

$$\begin{aligned} \mathbf{y}'(x) &= \mathbf{f}(x, \mathbf{y}), \qquad a < x < b \\ A\,\mathbf{y}(a) &= \boldsymbol{\alpha} \\ B\,\mathbf{y}(b) &= \boldsymbol{\beta} \end{aligned} \tag{2.49}$$

If Newton's method is applied directly to the nonlinear function $\mathbf{f}(x, \mathbf{y})$, then the method is called quasilinearization. Quasilinearization of (2.49) gives

$$\mathbf{y}'_{(k+1)}(x) = \mathbf{f}(x, \mathbf{y}_{(k)}(x)) + J(x, \mathbf{y}_{(k)}(x))(\mathbf{y}_{(k+1)}(x) - \mathbf{y}_{(k)}(x)), \quad k = 0, 1, \ldots \tag{2.50}$$

where

$$\begin{aligned} J(x, \mathbf{y}_{(k)}(x)) &= \text{Jacobian of } \mathbf{f}(x, \mathbf{y}_{(k)}(x)) \\ k &= \text{iteration number} \end{aligned}$$

One iteration of (2.50) can be solved by the superposition methods outlined above since it is a linear system.

FINITE DIFFERENCE METHODS

Up to this point, we have discussed initial-value methods for solving boundary-value problems. In this section we cover finite difference methods. These methods are said to be global methods since they simultaneously produce a solution over the entire interval.

The basic steps for a finite difference method are as follows: first, choose a mesh on the interval of interest, that is, for $[a,b]$

$$a = x_0 < x_1 < \ldots < x_N < x_{N+1} = b \tag{2.51}$$

such that the approximate solution will be sought at these mesh points; second, form the algebraic equations required to satisfy the differential equation and the BCs by replacing derivatives with difference quotients involving only the mesh points; and last, solve the algebraic system of equations.

Linear Second-Order Equations

We first consider the single linear second-order equation

$$Ly \equiv -y'' + p(x)y' + q(x)y = r(x), \qquad a < x < b \tag{2.52a}$$

subject to the Dirichlet boundary conditions

$$\begin{aligned} y(a) &= \alpha \\ y(b) &= \beta \end{aligned} \tag{2.52b}$$

On the interval $[a, b]$ impose a uniform mesh,

$$x_i = a + ih, \qquad i = 0, 1, \ldots, N + 1,$$

$$h = \frac{b - a}{N + 1}$$

The parameter h is called the mesh-size, and the points x_i are the mesh points. If $y(x)$ has continuous derivatives of order four, then, by Taylor's theorem,

$$y(x + h) = y(x) + hy'(x) + \frac{h^2}{2!}y''(x) + \frac{h^3}{3!}y'''(x) + \frac{h^4}{4!}y''''(\xi),$$

$$x_i \leqslant \xi \leqslant x_i + h \tag{2.53}$$

$$y(x - h) = y(x) - hy'(x) + \frac{h^2}{2!}y''(x) - \frac{h^3}{3!}y'''(x) + \frac{h^4}{4!}y''''(\bar{\xi}),$$

$$x_i - h \leqslant \bar{\xi} \leqslant x_i \tag{2.54}$$

From (2.53) and (2.54) we obtain

$$y'(x) = \left[\frac{y(x + h) - y(x)}{h}\right] - \frac{h}{2!}y''(x) - \frac{h^2}{3!}y'''(x) - \frac{h^3}{4!}y''''(\xi) \tag{2.55}$$

$$y'(x) = \left[\frac{y(x) - y(x - h)}{h}\right] + \frac{h}{2!}y''(x) - \frac{h^2}{3!}y'''(x) + \frac{h^3}{4!}y''''(\bar{\xi}) \tag{2.56}$$

respectively. The forward and backward difference equations (2.55) and (2.56) can be written as

$$y'(x_i) = \frac{y_{i+1} - y_i}{h} + 0(h) \tag{2.57}$$

$$y'(x_i) = \frac{y_i - y_{i-1}}{h} + 0(h) \tag{2.58}$$

respectively, where

$$y_i = y(x_i)$$

Thus, Eqs. (2.57) and (2.58) are first-order accurate difference approximations to the first derivative. A difference approximation for the second derivative is obtained by adding (2.54) and (2.53) to give

$$y(x + h) + y(x - h) = 2y(x) + h^2y''(x) + \frac{h^4}{4!}[y''''(\xi) + y''''(\bar{\xi})] \tag{2.59}$$

from which we obtain

$$y''(x_i) = \frac{(y_{i+1} - 2y_i + y_{i-1})}{h^2} + 0(h^2) \tag{2.60}$$

If the BVP under consideration contains both first and second derivatives, then one would like to have an approximation to the first derivative compatible with the accuracy of (2.60). If (2.54) is subtracted from (2.53), then

$$2hy'(x) = y(x + h) - y(x - h) - \frac{h^3}{3!}y'''(x) + \frac{h^4}{4!}[y''''(\bar{\xi}) - y''''(\xi)] \tag{2.61}$$

and hence

$$y'(x_i) = \left[\frac{y_{i+1} - y_{i-1}}{2h}\right] + 0(h^2) \tag{2.62}$$

which is the central difference approximation for the first derivative and is clearly second-order accurate.

To solve the given BVP, at each *interior* mesh point x_i we replace the derivatives in (2.52a) by the corresponding second-order accurate difference approximations to obtain

$$L_hu_i \equiv -\left[\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}\right] + p(x_i)\left[\frac{u_{i+1} - u_{i-1}}{2h}\right] + q(x_i)u_i = r(x_i)$$

$$i = 1, \ldots, N \tag{2.63}$$

and

$$u_0 = \alpha, \qquad u_{N+1} = \beta$$

where

$$u_i \simeq y_i$$

The result of multiplying (2.63) by $h^2/2$ is

$$\frac{h^2}{2}L_hu_i = a_iu_{i-1} + b_iu_i + c_iu_{i+1} = \frac{h^2}{2}r(x_i), \qquad i = 1, 2, \ldots, N$$

$$u_0 = \alpha, \qquad u_{N+1} = \beta \tag{2.64}$$

where

$$a_i = -\frac{1}{2}\left[1 + \frac{h}{2}p(x_i)\right]$$

$$b_i = \left[1 + \frac{h^2}{2}q(x_i)\right]$$

$$c_i = -\frac{1}{2}\left[1 - \frac{h}{2}p(x_i)\right]$$

The system (2.64) in vector notation is

$$A\,\mathbf{u} = \mathbf{r} \tag{2.65}$$

where

$$\mathbf{u} = [u_1, u_2, \ldots, u_N]^T$$

$$\mathbf{r} = \frac{h^2}{2}\left[r(x_1) - \frac{2a_1\alpha}{h^2}, r(x_2), \ldots, r(x_{N-1}) - \frac{2c_N\beta}{h^2}\right]^T$$

$$A = \begin{bmatrix} b_1 & c_1 & & & & \\ a_2 & b_2 & c_2 & & & 0 \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & & & & a_N & b_N \end{bmatrix}$$

A matrix of the form A is called a tridiagonal. This special form permits a very efficient application of the Gaussian elimination procedure (described in Appendix C).

To estimate the error in the numerical solution of BVPs by finite difference methods, first define the local truncation errors $\tau_i[\emptyset]$ in L_h as an approximation of L, for any smooth function $\emptyset(x)$ by

$$\tau_i[\emptyset] = L_h\emptyset(x_i) - L\emptyset(x_i), \qquad i = 1, 2, \ldots, N \tag{2.66}$$

If $\emptyset(x)$ has continuous fourth derivatives in $[a, b]$, then for L defined in (2.52) and L_h defined in (2.63),

$$\tau_i[\emptyset] = -\left[\frac{\emptyset(x_i + h) - 2\emptyset(x_i) + \emptyset(x_i - h)}{h^2}\right] + \emptyset''(x_i) + p(x_i)\left[\frac{\emptyset(x_i + h) - \emptyset(x_i - h)}{2h} - \emptyset'(x_i)\right] \tag{2.67}$$

or by using (2.59) and (2.61),

$$\tau_i[\emptyset] = -\frac{h^2}{4!}[\emptyset''''(\gamma_i) - 2p(x_i)\emptyset'''(\bar{\gamma}_i)], \qquad i = 1, \ldots, N \tag{2.68}$$

$$x_{i-1} \leqslant \gamma_i \leqslant x_{i+1}, \qquad x_{i-1} \leqslant \bar{\gamma}_i \leqslant x_{i+1}$$

From (2.67) we find that L_h is consistent with L, that is, $\tau_i[\emptyset] \to 0$ as $h \to 0$, for all functions $\emptyset(x)$ having continuous second derivatives on $[a, b]$. Further, from (2.68), it is apparent that L_h has second-order accuracy (in approximating L) for functions $\emptyset(x)$ with continuous fourth derivatives on $[a, b]$. For sufficiently small h, L_h is *stable*, i.e., for all functions v_i, $i = 0, 1, \ldots, N + 1$ defined on x_i, $i = 0, 1, \ldots, N + 1$, there is a positive constant M such that

$$|v_i| \leqslant M\left\{\max(|v_0|, |v_{N+1}|) + \max_{1\leqslant i\leqslant N}|L_h v_i|\right\}$$

for $i = 0, 1, \ldots, N + 1$. If L_h is stable and consistent, it can be shown that the error is given by

$$|u_i - y(x_i)| \leq M \max_{1 \leq j \leq i} |\tau_j[y]|, \qquad i = 1, \ldots, N \tag{2.69}$$

(for proof see Chapter 3 of [1]).

Flux Boundary Conditions

Consider a one-dimensional heat conduction problem that can be described by Fourier's law and is written as

$$\frac{1}{z^s}\frac{d}{dz}\left[z^s k \frac{dT}{dz}\right] = g(z), \qquad 0 < z < 1 \tag{2.70}$$

where

k = thermal conductivity

$g(z)$ = heat generation or removal function

s = geometric factor: 0, rectangular; 1, cylindrical; 2, spherical

In practical problems, boundary conditions involving the flux of a given component occur quite frequently. To illustrate the finite difference method with flux boundary conditions, consider (2.70) with $s = 0$, $g(z) = z$, $k =$ constant, and

$$T = T_0 \quad \text{at} \quad z = 0 \tag{2.71}$$

$$\frac{dT}{dz} + \lambda_1 T = \lambda_2 \quad \text{at} \quad z = 1 \tag{2.72}$$

where λ_1 and λ_2 are given constants. Since the governing differential equation is (2.70), the difference formula is

$$u_{i+1} - 2u_i + u_{i-1} = \frac{h^2}{k} z_i, \qquad i = 1, 2, \ldots, N \tag{2.73a}$$

with

$$u_0 = T_0 \tag{2.73b}$$

$$\frac{dT}{dz} + \lambda_1 T = \lambda_2 \quad \text{at} \quad z = 1 \tag{2.73c}$$

Since u_{N+1} is now an unknown, a difference equation for (2.73c) must be determined in order to solve for u_{N+1}.

To determine u_{N+1}, first introduce a "fictitious" point x_{N+2} and a corresponding value u_{N+2}. A second-order correct approximation for the first derivative at $z = 1$ is

$$\frac{dT}{dz} \simeq \frac{T_{N+2} - T_N}{2h} \tag{2.74}$$

Therefore, approximate (2.73c) by

$$\frac{u_{N+2} - u_N}{2h} + \lambda_1 u_{N+1} = \lambda_2 \tag{2.75}$$

and solve for u_{N+2}

$$u_{N+2} = 2h(\lambda_2 - \lambda_1 u_{N+1}) + u_N \tag{2.76}$$

The substitution of (2.76) into (2.73a) with $i = N + 1$ gives

$$(\lambda_2 - \lambda_1 u_{N+1})h - u_{N+1} + u_N = \frac{h^2}{2k} \tag{2.77}$$

Notice that (2.77) contains two unknowns, u_N and u_{N+1}, and together with the other $i = 1, 2, \ldots, N$ equations of type (2.73a), maintains the tridiagonal structure of the matrix A. This method of dealing with the flux condition is called the method of fictitious boundaries for obvious reasons.

EXAMPLE 3

A simple but practical application of heat conduction is the calculation of the efficiency of a cooling fin. Such fins are used to increase the area available for heat transfer between metal walls and poorly conducting fluids such as gases. A rectangular fin is shown in Figure 2.1. To calculate the fin efficiency one must first calculate the temperature profile in the fin. If $L >> B$, no heat is lost from the end or from the edges, and the heat flux at the surface is given by

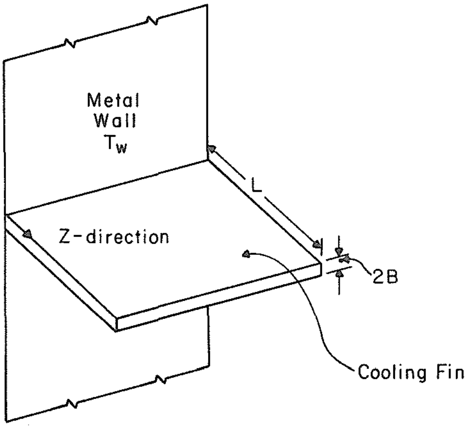

FIGURE 2.1 Cooling fin.

$q = \eta(T - Ta)$ in which the convective heat transfer coefficient η is constant as is the surrounding fluid temperature Ta, then the governing differential equation is

$$\frac{d^2T}{dz^2} = \frac{\eta}{kB}(T - Ta)$$

where

$$k = \text{thermal conductivity of the fin}$$

and

$$T(0) = T_w$$

$$\frac{dT}{dz}(L) = 0$$

Calculate the temperature profile in the fin, and demonstrate the order of accuracy of the finite difference method.

SOLUTION

Define

$$\theta = \frac{T - Ta}{T_w - Ta}$$

$$x = \frac{z}{L}$$

$$H = \sqrt{\frac{\eta L^2}{kB}}$$

The problem can be reformulated as

$$\frac{d^2\theta}{dx^2} = H^2\theta, \qquad \theta(0) = 1, \qquad \frac{d\theta}{dx}(1) = 0$$

The analytical solution to the governing differential equation is

$$\theta = \frac{\cosh H(1 - x)}{\cosh H}$$

For this problem the finite difference method (2.63) becomes

$$a_i u_{i-1} + b_i u_i + c_i u_{i+1} = 0, \qquad i = 1, 2, \ldots, N$$

where

$$a_i = 1$$
$$c_i = 1$$
$$b_i = -(2 + h^2H^2)$$

with

$$u_0 = 1$$

and

$$2u_N - (2 + h^2H^2)u_{N+1} = 0$$

Numerical results are shown in Table 2.3. Physically, one would expect θ to decrease as x increases since heat is being removed from the fin by convection. From these results we demonstrate the order of accuracy of the finite difference method.

If the error in approximation is $0(h^P)$ [see (2.68)], then an estimate of P can be determined as follows. If $u_j(h)$ is the approximate solution calculated using a mesh-size h and

$$e_j(h) = y(x_j) - u_j(h), \qquad j = 1, \ldots, N + 1$$

with

$$\|e(h)\| = \max_j |y(x_j) - u_j(h)|$$

then let

$$\|e(h)\| \simeq \psi h^P$$

where ψ is a constant. Use a sequence of h values, that is, $h_1 > h_2 > \ldots$, and write

$$\|e(h_t)\| = \psi h_t^P$$

TABLE 2.3 Results of $(d^2\theta)/(dx^2) = 4\theta$, $\theta(0) = 1$, $\theta'(1) = 0$

x	Analytical solution	θ, $h = 0.2$	Error†, $h = 0.2$	Error, $h = 0.1$	Error, $h = 0.05$	Error, $h = 0.02$
0.0	1.00000	1.00000	—	—	—	—
0.2	0.68509	0.68713	2.0 (−3)	5.1 (−4)	1.2 (−4)	2.0 (−5)
0.4	0.48127	0.48421	2.9 (−3)	7.4 (−4)	1.8 (−4)	2.9 (−5)
0.6	0.35549	0.35876	3.2 (−3)	8.2 (−4)	2.0 (−4)	3.3 (−5)
0.8	0.28735	0.29071	3.3 (−3)	8.4 (−4)	2.1 (−4)	3.4 (−5)
1.0	0.26580	0.26917	3.3 (−3)	8.5 (−4)	2.1 (−4)	3.4 (−5)

† Error = θ − analytical solution.

The value of P can be determined as

$$P = \frac{\ln\left[\dfrac{\|e(h_{t-1})\|}{\|e(h_t)\|}\right]}{\ln\left[\dfrac{h_{t-1}}{h_t}\right]}$$

Using the data in Table 2.3 gives:

t	h_t	$\|e(h_t)\|$	$\ln\left[\dfrac{\|e_{t-1}\|}{\|e_t\|}\right]$	$\ln\left[\dfrac{h_{t-1}}{h_t}\right]$	P
1	0.20	3.3 (−3)	—	—	—
2	0.10	8.5 (−4)	1.356	0.693	1.96
3	0.05	2.1 (−4)	1.398	0.693	2.01
4	0.02	3.4 (−5)	1.820	0.916	1.99

One can see the second-order accuracy from these results.

Integration Method

Another technique can be used for deriving the difference equations. This technique uses integration, and a complete description of it is given in Chapter 6 of [6].

Consider the following differential equation

$$-\frac{d}{dx}\left[\omega(x)\frac{dy}{dx}\right] + p(x)\frac{dy}{dx} + q(x)y = r(x)$$
$$a < x < b \tag{2.78}$$
$$\alpha_1 y(a) - \beta_1 y'(a) = \gamma_1$$
$$\alpha_2 y(b) + \beta_2 y'(b) = \gamma_2$$

where $\omega(x), p(x), q(x)$, and $r(x)$ are only piecewise continuous and hence possess a number of jump discontinuities. Physically, such problems arise from steady-state diffusion problems for heterogeneous materials, and the points of discontinuity represent interfaces between successive homogeneous compositions. For such problems y and $\omega(x)y'$ must be continuous at an interface $x = \eta$, that is,

$$y(\eta^-) = y(\eta^+)$$
$$\omega(\eta^+)\left.\frac{dy}{dx}\right|_{\eta^+} = \omega(\eta^-)\left.\frac{dy}{dx}\right|_{\eta^-} \tag{2.79}$$
$$a < \eta < b$$

Choose any set of points $a = x_0 < x_1 < \ldots < x_{N+1} = b$ such that the discontinuities of ω, p, q, and r are a subset of these points, that is, $\eta = x_i$ for some i. Note that the mesh spacings $h_i = x_{i+1} - x_i$ need not be uniform. Integrate (2.78) over the interval $x_i \leqslant x \leqslant x_i + h_i/2 \equiv x_{i+1/2}$, $1 \leqslant i \leqslant N$, to give:

$$-\omega_{i+1/2}\frac{dy_{i+1/2}}{dx} + \omega(x_i^+)\frac{dy(x_i^+)}{dx} + \int_{x_i}^{x_{i+1/2}} p(x)\frac{dy}{dx}\,dx + \int_{x_i}^{x_{i+1/2}} y(x)q(x)\,dx = \int_{x_i}^{x_{i+1/2}} r(x)\,dx \tag{2.80}$$

We can also integrate (2.78) over the interval $x_{i-1/2} \leqslant x \leqslant x_i$ to obtain:

$$-\omega(x_i^-)\frac{dy(x_i^-)}{dx} + \omega_{i-1/2}\frac{dy_{i-1/2}}{dx} + \int_{x_{i-1/2}}^{x_i} p(x)\frac{dy}{dx}\,dx + \int_{x_{i-1/2}}^{x_i} y(x)q(x)\,dx = \int_{x_{x-1/2}}^{x_i} r(x)\,dx \tag{2.81}$$

Adding (2.81) and (2.80) and employing (2.79) gives

$$-\omega_{i+1/2}\frac{dy_{i+1/2}}{dx} + \omega_{i-1/2}\frac{dy_{i-1/2}}{dx} + \int_{x_{i-1/2}}^{x_{i+1/2}} p(x)\frac{dy}{dx}\,dx + \int_{x_{i-1/2}}^{x_{i+1/2}} y(x)q(x)\,dx = \int_{x_{i-1/2}}^{x_{i+1/2}} r(x)\,dx \tag{2.82}$$

The derivatives in (2.82) can be approximated by central differences, and the integrals can be approximated by

$$\int_{x_{i-1/2}}^{x_{i+1/2}} g(x)\,dx = \int_{x_{i-1/2}}^{x_i} g(x)\,dx + \int_{x_i}^{x_{i+1/2}} g(x)\,dx \simeq g_i^-\left(\frac{h_{i-1}}{2}\right) + g_i^+\left(\frac{h_i}{2}\right) \tag{2.83}$$

where

$$g_i^- = g(x_i^-)$$
$$g_i^+ = g(x_i^+)$$

Using these approximations in (2.82) results in

$$-\omega_{i+1/2}\left[\frac{u_{i+1} - u_i}{h_i}\right] + \omega_{i-1/2}\left[\frac{u_i - u_{i-1}}{h_{i-1}}\right] + p_i^+\left[\frac{u_{i+1} - u_i}{2}\right] + p_i^-\left[\frac{u_i - u_{i-1}}{2}\right] + u_i\left[\frac{q_i^- h_{i-1} + q_i^+ h_i}{2}\right] = \left[\frac{r_i^- h_{i-1} + r_i^+ h_i}{2}\right],$$

$$1 \leqslant i \leqslant N \tag{2.84}$$

At the left boundary condition, if $\beta_1 = 0$, then $u_0 = \gamma_1/\alpha_1$. If $\beta_1 > 0$, then u_0 is unknown. In this case, direct substitution of the boundary condition into (2.80) for $i = 0$ gives

$$-\omega_{1/2}\left[\frac{u_1 - u_0}{h_0}\right] + \omega_0\left[\frac{\alpha_1 u_0 - \gamma_1}{\beta_1}\right] + p_0\left[\frac{u_1 - u_0}{2}\right] + q_0 u_0 \frac{h_0}{2} = \frac{r_0 h_0}{2} \tag{2.85}$$

The treatment of the right-hand boundary is straightforward. Thus, these expressions can be written in the form

$$-L_i u_{i-1} + D_i u_i - U_i u_{i+1} = R_i, \qquad i = 1, 2, \ldots, N$$

where

$$\begin{aligned} L_i &= \frac{\omega_{i-1/2}}{h_{i-1}} + \frac{p_i^-}{2} \\ U_i &= \frac{\omega_{i+1/2}}{h_i} - \frac{p_i^+}{2} \\ D_i &= L_i + U_i + \frac{q_i^- h_{i-1} + q_i^+ h_i}{2} \\ R_i &= \frac{r_i^- h_{i-1} + r_i^+ h_i}{2} \end{aligned} \tag{2.86}$$

Again, if $\beta_1 > 0$, then

$$\begin{aligned} L_0 &= 0 \\ U_0 &= \frac{\omega_{1/2}}{h_0} - \frac{p_0}{2} \\ D_0 &= L_0 + U_0 + \frac{\omega_0 \alpha_1}{\beta_1} + \frac{q_0 h_0}{2} \\ R_0 &= \frac{r_0 h_0}{2} + \frac{\omega_0 \gamma_1}{\beta} \end{aligned} \tag{2.87}$$

Summarizing, we have a system of equations

$$A\,\mathbf{u} = \mathbf{R} \tag{2.88}$$

where A is an $m \times m$ tridiagonal matrix where $m = N$, $N + 1$, or $N + 2$ depending upon the boundary conditions; for example $m = N + 1$ for the combination of one Dirichlet condition and one flux condition.

EXAMPLE 4

A nuclear fuel element consists of a cylindrical core of fissionable material surrounded by a metal cladding. Within the fissionable material heat is produced as a by-product of the fission reaction. A single fuel element is pictured in Figure 2.2. Set up the difference equations in order to calculate the radial temperature profile in the element.

Data: Let

$$T_c = T(r_c)$$
$$k_f = \text{thermal conductivity of core, } k_f \neq k_f(r)$$
$$k_c = \text{thermal conductivity of the cladding, } k_c \neq k_c(r)$$
$$S(r) = \text{source function of thermal energy, } S = 0 \text{ for } r > r_f$$

SOLUTION

Finite Difference Formulation

The governing differential equation is:

$$\frac{d}{dr}\left(rk\frac{dT}{dr}\right) = rS$$

$$\frac{dT}{dr} = 0 \quad \text{at} \quad r = 0$$

$$T = T_c \quad \text{at} \quad r = r_c$$

with

$$S = \begin{cases} S(r), & 0 \leqslant r \leqslant r_f \\ 0, & r > r_f \end{cases}$$

and

$$k_f\frac{dT}{dr}\bigg|_{r_f^-} = k_c\frac{dT}{dr}\bigg|_{r_f^+}$$

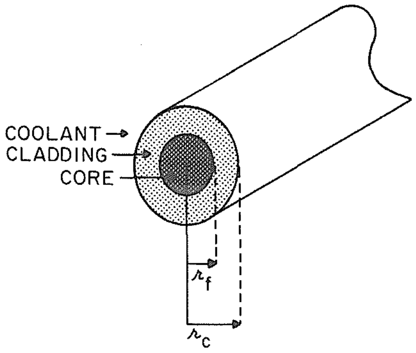

FIGURE 2.2 Nuclear fuel element.

By using (2.84), the difference formula becomes

$$-r_{i+\frac{1}{2}}k\left[\frac{u_{i+1}-u_i}{h_i}\right] + r_{i-\frac{1}{2}}k\left[\frac{u_i-u_{i-1}}{h_{i-1}}\right] = \left[\frac{h_{i-1}+h_i}{2}\right] r_i S_i$$

If $i = 0$ is the center point and $i = j$ is the point r_f, then the system of difference equations becomes

$$u_1 - u_0 = 0$$

$$-\left(\frac{r_{i+\frac{1}{2}}k_f}{h_i}\right)u_{i+1} + k_f\left[\frac{r_{i+\frac{1}{2}}}{h_i}+\frac{r_{i-\frac{1}{2}}}{h_{i-1}}\right]u_i - \left(\frac{r_{i-\frac{1}{2}}k_f}{h_{i-1}}\right)u_{i-1} = \frac{S_i}{2}[h_{i-1}\, r_i + h_i\, r_i],$$

$$i=1,\ldots,j-1$$

$$-\left(\frac{r_{j+\frac{1}{2}}k_c}{h_j}\right)u_{j+1} + \left[\frac{r_{j+\frac{1}{2}}k_c}{h_j}+\frac{r_{j-\frac{1}{2}}k_f}{h_{j-1}}\right]u_j - \left(\frac{r_{j-\frac{1}{2}}k_f}{h_{j-1}}\right)u_{j-1} = \frac{1}{2}[h_{j-1}\, r_j S_j], i=j$$

$$-\left(\frac{r_{i+\frac{1}{2}}k_c}{h_i}\right)u_{i+1} + k_c\left[\frac{r_{i+\frac{1}{2}}}{h_i}+\frac{r_{i-\frac{1}{2}}}{h_{i-1}}\right]u_i - \left(\frac{r_{i-\frac{1}{2}}k_c}{h_{i-1}}\right)u_{i-1} = 0, i=j+1,\ldots,N$$

with $u_{N+1} = T_c$

Nonlinear Second-Order Equations

We now consider finite difference methods for the solution of nonlinear boundary-value problems of the form

$$Ly(x) \equiv -y'' + f(x, y, y') = 0, \qquad a < x < b \tag{2.89a}$$

$$y(a) = \alpha, \qquad y(b) = \beta \tag{2.89b}$$

If a uniform mesh is used, then a second-order difference approximation to (2.89) is:

$$L_h u_i \equiv -\left[\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}\right] + f\left(x_i, u_i, \frac{u_{i+1} - u_{i-1}}{2h}\right) = 0,$$

$$i = 1, 2, \ldots, N \tag{2.90}$$

$$u_0 = \alpha, \qquad u_{N+1} = \beta$$

The resulting difference equations (2.90) are in general nonlinear, and we shall use Newton's method to solve them (see Appendix B). We first write (2.90) in the form

$$\mathbf{\Phi}(\mathbf{u}) = \mathbf{0} \tag{2.91}$$

where

$$\mathbf{u} = [u_1, u_2, \ldots, u_N]^T$$

$$\mathbf{\Phi} = [\Phi_1(\mathbf{u}), \Phi_2(\mathbf{u}), \ldots, \Phi_N(\mathbf{u})]^T$$

and

$$\Phi_i(\mathbf{u}) = \frac{h^2}{2} L_h u_i$$

The Jacobian of $\mathbf{\Phi}(\mathbf{u})$ is the tridiagonal matrix

$$J(\mathbf{u}) = \frac{\partial \mathbf{\Phi}(\mathbf{u})}{\partial \mathbf{u}} = \begin{bmatrix} B_1(\mathbf{u}) & C_1(\mathbf{u}) & & & 0 \\ A_2(\mathbf{u}) & B_2(\mathbf{u}) & C_2(\mathbf{u}) & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & B_{N-1}(\mathbf{u}) & C_{N-1}(\mathbf{u}) \\ 0 & & & A_N(\mathbf{u}) & B_N(\mathbf{u}) \end{bmatrix} \tag{2.92}$$

where

$$A_i(\mathbf{u}) = -\frac{1}{2}\left[1 + \frac{h}{2}\frac{\partial f}{\partial y'}\left(x_i, u_i, \frac{u_{i+1} - u_{i-1}}{2h}\right)\right], \qquad i = 2, 3, \ldots, N$$

$$B_i(\mathbf{u}) = \left[1 + \frac{h^2}{2}\frac{\partial f}{\partial y}\left(x_i, u_i, \frac{u_{i+1} - u_{i-1}}{2h}\right)\right], \qquad i = 1, 2, \ldots, N$$

$$C_i(\mathbf{u}) = -\frac{1}{2}\left[1 - \frac{h}{2}\frac{\partial f}{\partial y'}\left(x_i, u_i, \frac{u_{i+1} - u_{i-1}}{2h}\right)\right], \qquad i = 1, 2, \ldots, N-1$$

and

$$\frac{\partial f}{\partial y'}\left(x_i, u_i, \frac{u_{i+1} - u_{i-1}}{2h}\right) \quad \text{is} \quad \frac{\partial f}{\partial y'}(x_i, y, y')$$

with

$$y \quad \text{evaluated by} \quad u_i$$

and

$$y' \quad \text{evaluated by} \quad \frac{u_{i+1} - u_{i-1}}{2h}$$

In computing $\Phi_1(\mathbf{u})$, $\Phi_N(\mathbf{u})$, $A_N(\mathbf{u})$, $B_1(\mathbf{u})$, $B_N(\mathbf{u})$, and $C_1(\mathbf{u})$, we use $u_0 = \alpha$ and $u_{N+1} = \beta$. Now, with any initial estimate $\mathbf{u}^{[0]}$ of the quantities u_i, $i = 1, 2 \ldots, N$, we define

$$\mathbf{u}^{[k+1]} = \mathbf{u}^{[k]} + \Delta\mathbf{u}^{[k]}, \qquad k = 0, 1, 2, \ldots \tag{2.93}$$

where $\Delta\mathbf{u}^{[k]}$ is the solution of

$$J(\mathbf{u}^{[k]})\Delta\mathbf{u}^{[k]} = -\mathbf{\Phi}(\mathbf{u}^{[k]}), \qquad k = 0, 1, 2, \ldots \tag{2.94}$$

More general boundary conditions than those in (2.89b) are easily incorporated into the difference scheme.

EXAMPLE 5

A class of problems concerning diffusion of oxygen into a cell in which an enzyme-catalyzed reaction occurs has been formulated and studied by means of singular perturbation theory by Murray [7]. The transport equation governing the steady concentration C of some substrate in an enzyme-catalyzed reaction has the general form

$$\nabla(D\nabla C) = g(C)$$

Here D is the molecular diffusion coefficient of the substrate in the medium containing uniformly distributed bacteria and $g(C)$ is proportional to the reaction rate. We consider the case with constant diffusion coefficient D_0 in a spherical cell with a Michaelis-Menten theory reaction rate. In dimensionless variables the diffusion kinetics equation can now be written as

$$Ly = (x^2y')' = x^2 f(y), \qquad 0 < x < 1$$

where

$$x = \frac{r}{R}, \qquad y(x) = \frac{C(r)}{C_0}, \qquad \varepsilon = \left(\frac{D_0 C_0}{nqR^2}\right), \qquad f(y) = \varepsilon^{-1}\frac{y(x)}{y(x)+k}, \qquad k = \frac{k_m}{C_0}$$

Here R is the radius of the cell, C_0 is the constant concentration of the substrate in $r > R$, k_m is the Michaelis constant, q is the maximum rate at which each cell can operate, and n is the number of cells.

Assuming the cell membrane to have infinite permeability, it follows that

$$y(1) = 1$$

Further, from the assumed continuity and symmetry of $y(x)$ with respect to $x = 0$, we must have

$$y'(0) = 0$$

There is no closed-form analytical solution to this problem. Thus, solve this problem using a finite difference method.

SOLUTION

The governing equation is

$$2xy' + x^2y'' = x^2 f(y) \quad \text{or} \quad y'' + \frac{2}{x}y' - f(y) = 0$$

with $y(1) = 1$ and $y'(0) = 0$. With the mesh spacing $h = 1/(N + 1)$ and mesh point $x_i = ih$,

$$\left(1 + \frac{1}{i}\right) u_{i+1} - 2u_i + \left(1 - \frac{1}{i}\right) u_{i-1} - h^2 f(u_i) = 0, \qquad i = 1, 2, \ldots, N$$

with $u_{N+1} = 1.0$. For $x = 0$, the second term in the differential equation is evaluated using L'Hospital's rule:

$$\operatorname*{Lim}_{x \to 0} \left(\frac{y'}{x}\right) = \frac{y''}{1}$$

Therefore, the differential equation becomes

$$3y'' - f(y) = 0$$

at $x = 0$, for which the corresponding difference replacement is

$$u_1 - 2u_0 + u_{-1} - \frac{h^2}{3} f(u_0) = 0$$

Using the boundary condition $y'(0) = 0$ gives

$$u_1 - u_0 - \frac{h^2}{6} f(u_0) = 0$$

The vector $\mathbf{\Phi}(\mathbf{u})$ becomes

$$\mathbf{\Phi}(\mathbf{u}) = \begin{bmatrix} u_1 - u_0 - \frac{h^2}{6} f(u_0) \\ \vdots \\ \left(1 + \frac{1}{i}\right) u_{i+1} - 2u_i + \left(1 - \frac{1}{i}\right) u_{i-1} - h^2 f(u_i) \\ \vdots \\ \left(1 + \frac{1}{N}\right) - 2u_N + \left(1 - \frac{1}{N}\right) u_{N-1} - h^2 f(u_N) \end{bmatrix}$$

and the Jacobian is

$$J(\mathbf{u}) = \begin{bmatrix} B_0 & C_0 & & & & \\ A_1 & B_1 & C_1 & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & C_{N-1} \\ & & & & A_N & B_N \end{bmatrix}$$

where

$$B_0 = -\left(1 + \frac{h^2}{6}\frac{\partial f}{\partial y}(u_0)\right) = -\left(1 + \frac{h^2}{6\varepsilon} \times \frac{k}{(u_0 + k)^2}\right)$$

$$C_0 = 1$$

$$A_i = \left(1 - \frac{1}{i}\right), \qquad i = 1, 2, \ldots, N$$

$$B_i = -\left(2 + \frac{h^2}{\varepsilon} \times \frac{k}{(u_i + k)^2}\right), \qquad i = 1, 2, \ldots, N$$

$$C_i = \left(1 + \frac{1}{i}\right), \qquad i = 1, 2, \ldots, N - 1$$

Therefore, the matrix equation (2.94) for this problem involves a tridiagonal linear system of order $N + 1$.

The numerical results are shown in Table 2.4. For increasing values of N, the approximate solution appears to be converging to a solution. Decreasing the value of TOL below 10^{-6} gave the same results as shown in Table 2.4; thus the differences in the solutions presented are due to the error in the finite difference approximations. These results are consistent with those presented in Chapter 6 of [1].

The results shown in Table 2.4 are easy to interpret from a physical standpoint. Since y represents the dimensionless concentration of a substrate, and since the substrate is consumed by the cell, the value of y can never be negative and should decrease as x decreases (moving from the surface to the center of the cell).

First-Order Systems

In this section we shall consider the general systems of m first-order equations subject to linear two-point boundary conditions:

$$L\mathbf{y} = \mathbf{y}' - \mathbf{f}(x, \mathbf{y}) = \mathbf{0}, \qquad a < x < b \tag{2.95a}$$

$$A\mathbf{y}(a) + B\mathbf{y}(b) = \boldsymbol{\alpha} \tag{2.95b}$$

TABLE 2.4 Results of Example 5, TOL = (−6) on Newton Iteration, $\varepsilon = 0.1$, $k = 0.1$

x	$N = 5$	$N = 10$	$N = 20$	$N = 40$	$N = 80$
0.0	0.283(−1)	0.243(−1)	0.232(−1)	0.229(−1)	0.228(−1)
0.2	0.430(−1)	0.384(−1)	0.372(−1)	0.369(−1)	0.368(−1)
0.4	0.103	0.998(−1)	0.989(−1)	0.987(−1)	0.987(−1)
0.6	0.259	0.257	0.257	0.257	0.257
0.8	0.553	0.552	0.552	0.552	0.552
1.0	1.000	1.000	1.000	1.000	1.000

As before, we take the mesh points on $[a, b]$ as

$$x_i = a + ih, \qquad i = 0, 1, \ldots, N + 1 \qquad h = \frac{b - a}{N + 1} \tag{2.96}$$

Let the m-dimensional vector $\mathbf{u}_i$ approximate the solution $y(x_i)$, and approximate (2.95a) by the system of difference equations

$$L_h \mathbf{u}_i = \frac{\mathbf{u}_i - \mathbf{u}_{i-1}}{h} - \mathbf{f}\left(x_{i-1/2}, \frac{\mathbf{u}_i + \mathbf{u}_{i-1}}{2}\right) = \mathbf{0}, \qquad i = 1, 2, \ldots, N + 1 \tag{2.97a}$$

The boundary conditions are given by

$$A\mathbf{u}_0 + B\mathbf{u}_{N+1} - \alpha = \mathbf{0} \tag{2.97b}$$

The scheme (2.97) is known as the centered-difference method. The nonlinear term in (2.95a) might have been chosen as

$$\tfrac{1}{2}\left[\mathbf{f}(x_i, \mathbf{u}_i) + \mathbf{f}(x_{i-1}, \mathbf{u}_{i-1})\right] \tag{2.98}$$

resulting in the trapezoidal rule.

On defining the $m(N + 2)$-dimensional vector $\mathbf{U}$ by

$$\mathbf{U} = [\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N+1}]^T \tag{2.99}$$

(2.97) can be written as the system of $m(N + 2)$ equations

$$\mathbf{\Phi}(\mathbf{U}) = \begin{bmatrix} A\mathbf{u}_0 + B\mathbf{u}_{N+1} - \alpha \\ hL_h\mathbf{u}_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ hL_h\mathbf{u}_{N+1} \end{bmatrix} = \mathbf{0} \tag{2.100}$$

With some initial guess, $\mathbf{U}^{[0]}$, we now compute the sequence of $\mathbf{U}^{[k]}$'s by

$$\mathbf{U}^{[k+1]} = \mathbf{U}^{[k]} + \Delta\mathbf{U}^{[k]}, \qquad k = 0, 1, 2, \ldots \tag{2.101}$$

where $\Delta\mathbf{U}^{[k]}$ is the solution of the linear algebraic system

$$\frac{\partial\mathbf{\Phi}(\mathbf{U}^{[k]})}{\partial\mathbf{U}} \Delta\mathbf{U}^{[k]} = -\mathbf{\Phi}(\mathbf{U}^{[k]}) \tag{2.102}$$

One of the advantages of writing a BVP as a first-order system is that variable mesh spacings can be used easily. Let

$$a = x_0 < x_1 < \ldots < x_{N+1} = b \qquad h_i = x_{i+1} - x_i, \qquad h = \max_i h_i \tag{2.103}$$

be a general partition of the interval $[a, b]$.

The approximation for (2.95) using (2.103) with the trapezoidal rule is

$$\Phi(\mathbf{U}) = \begin{bmatrix} A\mathbf{u}_0 + B\mathbf{u}_{N+1} - \alpha \\ h_0 L_h \mathbf{u}_1 \\ \vdots \\ h_N L_h \mathbf{u}_{N+1} \end{bmatrix} = \mathbf{0} \tag{2.104}$$

where

$$h_{i-1} L_h \mathbf{u}_i = \mathbf{u}_i - \mathbf{u}_{i-1} - \frac{h_{i-1}}{2} [\mathbf{f}(x_i, \mathbf{u}_i) + \mathbf{f}(x_{i-1}, \mathbf{u}_{i-1})], \ i = 1, \ldots, N + 1$$

By allowing the mesh to be graded in the region of a sharp gradient, nonuniform meshes can be helpful in solving problems that possess solutions or derivatives that have sharp gradients.

Higher-Order Methods

The difference scheme (2.63) yields an approximation to the solution of (2.52) with an error that is $0(h^2)$. We shall briefly examine two ways in which, with additional calculations, difference schemes may yield higher-order approximations. These error-reduction procedures are Richardson's extrapolation and deferred corrections.

The basis of Richardson's extrapolation is that the error E_i, which is the difference between the approximation and the true solution, can be written as

$$E_i = h^2 a_1(x_i) + h^4 a_2(x_i) + \ldots \tag{2.105}$$

where the functions $a_j(x_i)$ are independent of h. To implement the method, one solves the BVP using successively smaller mesh sizes such that the larger meshes are subsets of the finer ones. For example, solve the BVP twice, with mesh sizes of h and $h/2$. Let the respective solutions be denoted $u_i(h)$ and $u_i(h/2)$. For any point common to both meshes, $x_i = ih = 2i(h/2)$,

$$\begin{aligned} y(x_i) - u_i(h) &= h^2 a_1(x_i) + h^4 a_2(x_i) + \ldots \\ y(x_i) - u_i\left(\frac{h}{2}\right) &= \frac{h^2}{4} a_1(x_i) + \frac{h^4}{16} a_2(x_i) + \ldots \end{aligned} \tag{2.106}$$

Eliminate $a_1(x_i)$ from (2.106) to give

$$y(x_i) = \frac{4u_i\left(\frac{h}{2}\right) - u_i(h)}{3} + 0(h^4) \tag{2.107}$$

Thus an $0(h^4)$ approximation to $y(x)$ on the mesh with spacing h is given by

$$\overline{u}_i = \frac{4}{3} u_i\left(\frac{h}{2}\right) - \frac{1}{3} u_i(h), \qquad i = 0, 1, \ldots, N + 1 \tag{2.108}$$

A further mesh subdivision can be used to produce a solution with error $0(h^6)$, and so on.

For some problems Richardson's extrapolation is useful, but in general, the method of deferred corrections, which is described next, has proven to be somewhat superior [8].

The method of deferred corrections was introduced by Fox [9], and has since been modified and extended by Pereyra [10–12]. Here, we will outline Pereyra's method since it is used in software described in the next section.

Pereyra requires the BVP to be in the following form:

$$\begin{gathered} \mathbf{y}' = \mathbf{f}(x, \mathbf{y}), \qquad a < x < b \\ \mathbf{g}(\mathbf{y}(a), \mathbf{y}(b)) = \mathbf{0} \end{gathered} \tag{2.109}$$

and uses the trapezoidal rule approximation

$$\mathbf{u}_{i+1} - \mathbf{u}_i - \tfrac{1}{2} h \left[\mathbf{f}(x_i, \mathbf{u}_i) + \mathbf{f}(x_{i+1}, \mathbf{u}_{i+1})\right] = h\mathbf{T}(\mathbf{u}_{i+1/2}) \tag{2.110}$$

where $\mathbf{T}(\mathbf{u}_{i+1/2})$ is the truncation error. Next, Pereyra writes the truncation error in terms of higher-order derivatives

$$\mathbf{T}(\mathbf{u}_{i+1/2}) = -\sum_{s=1}^{q} \left[\alpha_s h^{2s} \mathbf{f}^{(2s)}_{i+1/2}\right] + 0(h^{2s+2}) \tag{2.111}$$

where

$$\mathbf{f}^{(2s)}_{i+1/2} = \frac{d^{2s}}{dx^{2s}} \mathbf{f}(x_{i+1/2}, \mathbf{u}_{i+1/2})$$

$$\alpha_s = \frac{s}{2^{2s-1}(2s + 1)(2s!)}$$

$$q = \text{number of terms in series (sets the desired accuracy)}$$

The first approximation $\mathbf{u}_i^{[1]}$ is obtained by solving

$$\begin{gathered} \mathbf{u}_{i+1}^{[1]} - \mathbf{u}_i^{[1]} - \tfrac{1}{2} h[\mathbf{f}(x_i, \mathbf{u}_i^{[1]}) + \mathbf{f}(x_{i+1}, \mathbf{u}_{i+1}^{[1]})] = \mathbf{0} \\ i = 0, 1, \ldots, N \\ \mathbf{g}(\mathbf{u}_0^{[1]}, \mathbf{u}_{N+1}^{[1]}) = \mathbf{0} \end{gathered} \tag{2.112}$$

where the truncation error is ignored. This approximation differs from the true solution by $0(h^2)$.

The process proceeds as follows. An approximate solution $\mathbf{u}^{[k]}$ [differs from the true solution by terms of order $0(h^{2k})$] can be obtained from:

$$\begin{gathered} \mathbf{u}_{i+1}^{[k]} - \mathbf{u}_i^{[k]} - \tfrac{1}{2} h \left[\mathbf{f}(x_i, \mathbf{u}_i^{[k]}) + \mathbf{f}(x_{i+1}, \mathbf{u}_{i+1}^{[k]})\right] = h\mathbf{T}^{[k-1]}(\mathbf{u}_{i+1/2}^{[k-1]}) \\ i = 0, 1, \ldots, N \\ \mathbf{g}(\mathbf{u}_0^{[k]}, \mathbf{u}_{N+1}^{[k]}) = \mathbf{0}) \end{gathered} \tag{2.113}$$

where

$$\mathbb{T}^{[k-1]} = \mathbb{T} \quad \text{with} \quad q = k - 1$$

In each step of (2.113), the nonlinear algebraic equations are solved by Newton's method with a convergence tolerance of less than $0(h^{2k})$. Therefore, using (2.112) gives $\mathbf{u}_i^{[1]}$ $(0(h^2))$, which can be used in (2.113) to give $\mathbf{u}_i^{[2]}$ $(0(h^4))$. Successive iterations of (2.113) with increasing k can give even higher-order accurate approximations.

MATHEMATICAL SOFTWARE

The available software that is based on the methods of this chapter is not as extensive as in the case of IVPs. A subroutine for solving a BVP will be designed in a manner similar to that outlined for IVPs in Chapter 1 except for the fact that the routines are much more specialized because of the complexity of solving BVPs. The software discussed below requires the BVPs to be posed as first-order systems (usually allows for simpler algorithms). A typical calling sequence could be

```
CALL DRIVE (FUNC, DFUNC, BOUND, A, B, U, TOL)
```

where

FUNC = user-written subroutine for evaluating $\mathbf{f}(x, \mathbf{y})$

DFUNC = user-written subroutine for evaluating the Jacobian of $\mathbf{f}(x, \mathbf{y})$

BOUND = user-written subroutine for evaluating the boundary conditions and, if necessary, the Jacobian of the boundary conditions

A = left boundary point

B = right boundary point

U = on input contains initial guess of solution vector, and on output contains the approximate solution

TOL = an error tolerance

This is a simplified calling sequence, and more elaborate ones are actually used in commercial routines.

The subroutine DRIVE must contain algorithms that:

1. Implement the numerical technique
2. Adapt the mesh-size (or redistribute the mesh spacing in the case of nonuniform meshes)
3. Calculate the error so to implement step (2) such that the error does not surpass TOL

Implicit within these steps are the subtleties involved in executing the various techniques, e.g., the position of the orthonormalization points when using superposition.

Each of the major mathematical software libraries—IMSL, NAG, and HARWELL—contains routines for solving BVPs. IMSL contains a shooting routine and a modified version of DD04AD (to be described below) that uses a variable-order finite difference method combined with deferred corrections. HARWELL possesses a multiple shooting code and DD04AD. The NAG library includes various shooting codes and also contains a modified version of DD04AD. Software other than that of IMSL, HARWELL, and NAG that is available is listed in Table 2.5. From this table and the routines given in the main libraries, one can see that the software for solving BVPs uses the techniques that are outlined in this chapter.

We illustrate the use of BVP software packages by solving a fluid mechanics problem. The following codes were used in this study:

1. HARWELL, DD03AD (multiple shooting)
2. HARWELL, DD04AD

Notice we have chosen a shooting and a finite difference code. The third major method, superposition, was not used in this study. The example problem is nonlinear and would thus require the use of SUPORQ if superposition is to be included in this study. At the time of this writing SUPORQ is difficult to implement and requires intimate knowledge of the code for effective utilization. Therefore, it was excluded from this study. DD03AD and DD04AD will now be described in more detail.

DD03AD [18]

This program is the multiple-shooting code of the Harwell library. In this algorithm, the interval is subdivided and "shooting" occurs in both directions. The boundary-value problem must be specified as an initial-value problem with the code or the user supplying the initial conditions. Also, the partitioning of the interval can be user-supplied or performed by the code. A tolerance parameter (TOL) controls the accuracy in meeting the continuity conditions at the

TABLE 2.5 BVP Codes

Name	Method Implemented	Reference
BOUNDS	Multiple shooting	[13,14]
SHOOT1	Shooting with separated boundary conditions	[15]
SHOOT2	Same as SHOOT1 with more general boundary conditions	[15]
MSHOOT	Mutliple shooting	[15]
SUPORT	Superposition (linear problems only)	[4]
SUPORQ	Superposition with quasilinearization	[16]

matching points [see (2.35)]. This type of code takes advantage of the highly developed software available for IVPs (uses a fourth-order Runge-Kutta algorithm [19]).

DD04AD [17, 20]

This code was written by Lentini and Pereyra and is described in detail in [20]. Also, an earlier version of the code is discussed in [17]. The code implements the trapezoidal approximation, and the resulting algebraic system is solved by a modified Newton method. The user is permitted to specify an initial interval partition (which does not need to be uniform), or the code provides a coarse, equispaced one. The user may also specify an initial estimate for the solution (the default being zero). Deferred corrections is used to increase accuracy and to calculate error estimates. An error tolerance (TOL) is provided by the user. Additional mesh points are automatically added to the initial partition with the aim of reducing error to the user-specified level, and also with the aim of equidistribution of error throughout the interval [17]. The new mesh points are always added between the existing mesh points. For example, if x_j and x_{j+1} are initial mesh points, then if m mesh points t_i, $i = 1, 2, \ldots, m$, are required to be inserted into $[x_j, x_{j+1}]$, they are placed such that

$$x_j < t_1 < \ldots < t_m < x_{j+1} \tag{2.114}$$

where

$$t_1 = \frac{t_2 - x_j}{2}, \ldots, t_i = \frac{t_{i+1} - t_{i-1}}{2}, \ldots, t_m = \frac{x_{j+1} - t_{m-1}}{2}$$

The approximate solution is given as a discrete function at the points of the final mesh.

Example Problem

The following BVP arises in the study of the behavior of a thin sheet of viscous liquid emerging from a slot at the base of a converging channel in connection with a method of lacquer application known as "curtain coating" [21]:

$$\frac{d^2y}{dx^2} - \frac{1}{y}\left(\frac{dy}{dx}\right)^2 - y\frac{dy}{dx} + 1 = 0 \tag{2.115}$$

The function y is the dimensionless velocity of the sheet, and x is the dimensionless distance from the slot. Appropriate boundary conditions are [22]:

$$y = y_0 \quad \text{at} \quad x = 0 \tag{2.116}$$

$$\frac{dy}{dx} \rightarrow (2x)^{-1/2} \quad \text{at} \quad \text{sufficiently large } x$$

In [22] (2.115) was solved using a reverse shooting procedure subject to the boundary conditions

$$y = 0.325 \quad \text{at} \quad x = 0 \tag{2.117}$$

and

$$\frac{dy}{dx} = (2x)^{-1/2} \quad \text{at} \quad x = x_R$$

The choice of $x_R = 50$ was found by experimentation to be optimum in the sense that it was large enough for (2.116) to be "sufficiently valid." For smaller values of x_R, the values of y at zero were found to have a variation of as much as 8%.

We now study this problem using DD03AD and DD04AD. The results are shown in Table 2.6. DD03AD produced approximate solutions only when a large number of shooting points were employed. Decreasing TOL from 10^{-4} to 10^{-6} when using DD03AD did not affect the results, but increasing the number of shooting points resulted in drastic changes in the solution. Notice that the boundary condition at $x = 0$ is never met when using DD03AD, even when using a large number of shooting points (SP = 360). Davis and Fairweather [23] studied this problem, and their results are shown in Table 2.6 for comparison. DD04AD was able to produce the same results as Davis and Fairweather in significantly less execution time than DD03AD.

We have surveyed the types of BVP software but have not attempted to make any sophisticated comparisons. This is because in the author's opinion, based upon the work already carried out on IVP solvers, there is no sensible basis for comparing BVP software.

Like IVP software, BVP codes are not infallible. If you obtain spurious results from a BVP code, you should be able to rationalize your data with the aid of the code's documentation and the material presented in this chapter.

TABLE 2.6 Results of Eq. (2.115) with (2.117) and $x_R = 5.0$

	DD03AD	DD03AD	DD03AD	DD04AD	Reference [23]
	TOL = 10^{-4},	TOL = 10^{-6},	TOL = 10^{-6},		
x	SP = 80†	SP = 80	SP = 320	TOL = 10^{-4}	TOL = 10^{-4}
0.0	0.3071	0.3071	0.3205	0.3250	0.3250
1.0	0.9115	0.9115	0.9253	0.9299	0.9299
2.0	0.1462(1)	0.1462(1)	0.1474(1)	0.1477(1)	0.1477(1)
3.0	0.1931(1)	0.1931(1)	0.1941(1)	0.1945(1)	0.1945(1)
4.0	0.2340(1)	0.2340(1)	0.2349(1)	0.2349(1)	0.2349(1)
5.0	0.2737(1)	0.2737(1)	0.2743(1)	0.2701(1)	0.2701(1)
E.T.R.‡	3.75	4.09	14.86	1.0	—

† SP = number of "shooting" points.

‡ E.T.R. = Execution time ratio = $\dfrac{\text{execution time}}{\text{execution time of DD04AD with TOL} = 10^{-4}}$.

PROBLEMS

1. Consider the BVP

$$\begin{aligned} y'' + r(x)y &= f(x), \qquad a < x < b \\ y(a) &= \alpha \\ y(b) &= \beta \end{aligned}$$

Show that for a uniform mesh, the integration method gives the same result as Eq. (2.64).

2. Refer to Example 4. If

$$S(r) = S_0\left[1 + b\left(\frac{r}{r_f}\right)^2\right]$$

solve the governing differential equation to obtain the temperature profile in the core and the cladding. Compare your results with the analytical solution given on page 304 of [24]. Let $k_c = 0.64$ cal/(s·cm·K), $k_f = 0.066$ cal/(s·cm·K), $T_0 = 500$ K, $r_c = \frac{1}{2}$ in, and $r_f = \frac{3}{8}$ in.

3.* Axial conduction and diffusion in an adiabatic tubular reactor can be described by [2]:

$$\frac{1}{\text{Pe}}\frac{d^2C}{dx^2} - \frac{dC}{dx} - R(C, T) = 0$$

$$\frac{1}{\text{Bo}}\frac{d^2T}{dx^2} - \frac{dT}{dx} - \beta R(C, T) = 0$$

with

$$\left.\begin{aligned} \frac{1}{\text{Pe}}\frac{dC}{dx} &= C - 1 \\ \frac{1}{\text{Bo}}\frac{dT}{dx} &= T - 1 \end{aligned}\right\} \quad \text{at} \quad x = 0$$

and

$$\left.\begin{aligned} \frac{dC}{dx} &= 0 \\ \frac{dT}{dx} &= 0 \end{aligned}\right\} \quad \text{at} \quad x = 0$$

Calculate the dimensionless concentration C and temperature T profiles for $\beta = -0.05$, $\text{Pe} = \text{Bo} = 10$, $E = 18$, and $R(C, T) = 4C \exp[E(1 - 1/T)]$.

4.* Refer to Example 5. In many reactions the diffusion coefficient is a function of the substrate concentration. The diffusion coefficient can be of form [1]:

$$\frac{D(y)}{D_0} = 1 + \frac{\lambda}{(y + k_2)^2}$$

Computations are of interest for $\lambda = k_2 = 10^{-2}$, with ε and k as in Example 5. Solve the transport equation using $D(y)$ instead of D for the parameter choice stated above. Next, let $\lambda = 0$ and show that your results compare with those of Table 2.4.

5.* The reactivity behavior of porous catalyst particles subject to both internal mass concentration gradients as well as temperature gradients can be studied with the aid of the following material and energy balances:

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} = \phi^2 y \exp\left[\gamma\left(1 - \frac{1}{T}\right)\right]$$

$$\frac{d^2T}{dx^2} + \frac{2}{x}\frac{dT}{dx} = -\beta\phi^2 y \exp\left[\gamma\left(1 - \frac{1}{T}\right)\right]$$

with

$$\frac{dT}{dx} = \frac{dy}{dx} = 0 \quad \text{at} \quad x = 0$$

$$T = y = 1 \quad \text{at} \quad x = 1$$

where

y = dimensionless concentration
T = dimensionless temperature
x = dimensionless radial coordiante (spherical geometry)
ϕ = Thiele modulus (first-order reaction rate)
γ = Arrhenius number
β = Prater number

These equations can be combined into a single equation such that

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} = \phi^2 y \exp\left[\frac{\gamma\beta(1 - y)}{1 + \beta(1 - y)}\right]$$

with

$$\frac{dy}{dx} = 0 \quad \text{at} \quad x = 0$$

$$y = 1 \quad \text{at} \quad x = 1$$

For $\gamma = 30$, $\beta = 0.4$, and $\phi = 0.3$, Weisz and Hicks [25] found three solutions to the above equation using a shooting method. Calculate the dimensionless concentration and temperature profiles of the three solutions.

Hint: **Try various initial guesses.**

REFERENCES

1. Keller, H. B., *Numerical Methods for Two-Point Boundary-Value Problems,* Blaisdell, New York (1968).
2. Carberry, J. J., *Chemical and Catalytic Reaction Engineering,* McGraw-Hill, New York (1976).
3. Deuflhard, P., "Recent Advances in Multiple Shooting Techniques," in *Computational Techniques for Ordinary Differential Equations,* I. Gladwell and D. K. Sayers (eds.), Academic, London (1980).
4. Scott, M. R., and H. A. Watts, SUPORT—A Computer Code for Two-Point Boundary-Value Problems via Orthonormalization, SAND75-0198, Sandia Laboratories, Albuquerque, N. Mex. (1975).
5. Scott, M. R., and H. A. Watts, "Computational Solutions of Linear Two-Point Boundary Value Problems via Orthonormalization," SIAM J. Numer. Anal., *14,* 40 (1977).
6. Varga, R. S., *Matrix Iterative Analysis,* Prentice-Hall, Englewood Cliffs, N.J. (1962).
7. Murray, J. D., "A Simple Method for Obtaining Approximate Solutions for a Large Class of Diffusion-Kinetic Enzyme Problems," Math. Biosci., *2* (1968).
8. Fox, L., "Numerical Methods for Boundary-Value Problems," in *Computational Techniques for Ordinary Differential Equations,* I. Gladwell and D. K. Sayers (eds.), Academic, London (1980).
9. Fox, L., "Some Improvements in the Use of Relaxation Methods for the Solution of Ordinary and Partial Differential Equations," Proc. R. Soc. A, *190,* 31 (1947).
10. Pereyra, V., "The Difference Correction Method for Non-Linear Two-Point Boundary Problems of Class M," Rev. Union Mat. Argent. *22,* 184 (1965).
11. Pereyra, V., "High Order Finite Difference Solution of Differential Equations," STAN-CS-73-348, Computer Science Dept., Stanford Univ., Stanford, Calif. (1973).
12. Keller, H. B., and V. Pereyra, "Difference Methods and Deferred Corrections for Ordinary Boundary Value Problems," SIAM J. Numer. Anal., *16,* 241 (1979).

13. Bulirsch, R., "Multiple Shooting Codes," in *Codes for Boundary-Value Problems in Ordinary Differential Equations,* Lecture Notes in Computer Science, *76,* Springer-Verlag, Berlin (1979).

14. Deuflhard, P., "Nonlinear Equation Solvers in Boundary-Value Codes," Rep. TUM-MATH-7812. Institut fur Mathematik, Universitat Munchen (1979).

15. Scott, M. R. and H. A. Watts, "A Systematized Collection of Codes for Solving Two-Point Boundary-Value Problems," *Numerical Methods for Differential Systems,* L. Lapidus and W. E. Schiesser (eds.), Academic, New York (1976).

16. Scott, M. R., and H. A. Watts, "Computational Solution of Nonlinear Two-Point Boundary Value Problems," Rep. SAND 77-0091, Sandia Laboratories, Albuquerque, N. Mex. (1977).

17. Lentini, M., and V. Pereyra, "An Adaptive Finite Difference Solver for Nonlinear Two-Point Boundary Problems with Mild Boundary Layers," SIAM J. Numer. Anal., *14,* 91 (1977).

18. England, R., "A Program for the Solution of Boundary Value Problems for Systems of Ordinary Differential Equations," Culham Lab., Abingdon: Tech. Rep. CLM-PDM 3/73 (1976).

19. England, R., "Error Estimates for Runge-Kutta Type Solutions to Systems of Ordinary Differential Equations," Comput. J., *12,* 166 (1969).

20. Pereyra, V., "PASVA3—An Adaptive Finite-Difference FORTRAN Program for First-Order Nonlinear Ordinary Boundary Problems," in *Codes for Boundary-Value Problems in Ordinary Differential Equations,* Lecture Notes in Computer Science, *76,* Springer-Verlag, Berlin (1979).

21. Brown, D. R., "A Study of the Behavior of a Thin Sheet of Moving Liquid," J. Fluid Mech., *10,* 297 (1961).

22. Salariya, A. K., "Numerical Solution of a Differential Equation in Fluid Mechanics," Comput. Methods Appl. Mech. Eng., *21,* 211 (1980).

23. Davis, M., and G. Fairweather, "On the Use of Spline Collocation for Boundary Value Problems Arising in Chemical Engineering," Comput. Methods Appl. Mech. Eng., *28,* 179 (1981).

24. Bird, R. B., W. E. Stewart, and E. L. Lightfoot, *Transport Phenomena,* Wiley, New York (1960).

25. Weisz, P. B., and J. S. Hicks, "The Behavior of Porous Catalyst Particles in View of Internal Mass and Heat Diffusion Effects," Chem. Eng. Sci., *17,* 265 (1962).

BIBLIOGRAPHY

For additional or more detailed information concerning boundary-value problems, see the following:

Aziz, A. Z. (ed.), *Numerical Solutions of Boundary-Value Problems for Ordinary Differential Equations,* Academic, New York (1975).

Childs, B., M. Scott, J. W. Daniel, E. Denman, and P. Nelson (eds.), *Codes for Boundary-Value Problems in Ordinary Differential Equations,* Lecture Notes in Computer Science, Volume 76, Springer-Verlag, Berlin (1979).

Fox, L., *The Numerical Solution of Two-Point Boundary-Value Problems in Ordinary Differential Equations,* (1957).

Gladwell, I., and D. K. Sayers (eds.), *Computational Techniques for Ordinary Differential Equations,* Academic, London (1980).

Isaacson, E., and H. B. Keller, *Analysis of Numerical Methods,* Wiley, New York (1966).

Keller, H. B., *Numerical Methods for Two-Point Boundary-Value Problems,* Blaisdell, New York (1968).

Russell, R. D., *Numerical Solution of Boundary Value Problems,* Lecture Notes, Universidad Central de Venezuela, Publication 79-06, Caracas (1979).

Varga, R. S., *Matrix Iterative Analysis,* Prentice-Hall, Englewood Cliffs, N.J. (1962).