2. Solve the initial-boundary value problems in Exercise 2 on $0 \le x \le 1, 0 \le t \le 1$ by the Finite Difference Method with $h = 0.05$ and k small enough to satisfy the CFL condition. Plot the solution.

3. For the wave equations in Exercise 1, make a table of the approximation and error at $(x, t) = (1/4, 3/4)$ as a function of step sizes $h = ck = 2^{-p}$ for $p = 4, \ldots, 8$.

4. For the wave equations in Exercise 2, make a table of the approximation and error at $(x, t) = (1/4, 3/4)$ as a function of step sizes $h = ck = 2^{-p}$ for $p = 4, \ldots, 8$.

8.3 ELLIPTIC EQUATIONS

The previous sections deal with time-dependent equations. The diffusion equation models the flow of heat as a function of time, and the wave equation follows the motion of a wave. Elliptic equations, the focus of this section, model steady states. For example, the steady-state distribution of heat on a plane region whose boundary is being held at specific temperatures is modeled by an elliptic equation. Since time is usually not a factor in elliptic equations, we will use x and y to denote the independent variables.

DEFINITION 8.6 Let $u(x, y)$ be a twice-differentiable function, and define the **Laplacian** of u as

$$\Delta u = u_{xx} + u_{yy}.$$

For a continuous function $f(x, y)$, the partial differential equation

$$\Delta u(x, y) = f(x, y) \tag{8.37}$$

is called the **Poisson equation**. The Poisson equation with $f(x, y) = 0$ is called the **Laplace equation**. A solution of the Laplace equation is called a **harmonic** function. ❐

Comparing with the normal form (8.1), we compute $B^2 - 4AC < 0$, so the Poisson equation is elliptic. The extra conditions given to pin down a single solution are typically boundary conditions. There are two common types of boundary conditions applied. Dirichlet boundary conditions specify the values of the solution $u(x, y)$ on the boundary ∂R of a region R. Neumann boundary conditions specify values of the directional derivative $\partial u/\partial n$ on the boundary, where n denotes the outward unit normal vector.

► EXAMPLE 8.7 Show that $u(x, y) = x^2 - y^2$ is a solution of the Laplace equation on $[0, 1] \times [0, 1]$ with Dirichlet boundary conditions

$$\begin{aligned} u(x, 0) &= x^2 \\ u(x, 1) &= x^2 - 1 \\ u(0, y) &= -y^2 \\ u(1, y) &= 1 - y^2. \end{aligned}$$

The Laplacian is $\Delta u = u_{xx} + u_{yy} = 2 - 2 = 0$. The boundary conditions are listed for the bottom, top, left, and right of the unit square, respectively, and are easily checked by substitution. ◄

The Poisson and Laplace equations are ubiquitous in classical physics because their solutions represent potential energy. For example, an electric field E is the gradient of an electrostatic potential u, or

$$E = -\nabla u.$$

The gradient of the electric field, in turn, is related to the charge density ρ by Maxwell's equation

$$\nabla E = \frac{\rho}{\epsilon},$$

where ϵ is the electrical permittivity. Putting the two equations together yields

$$\Delta u = \nabla(\nabla u) = -\frac{\rho}{\epsilon},$$

the Poisson equation for the potential u. In the special case of zero charge, the potential satisfies the Laplace equation $\Delta u = 0$.

Many other instances of potential energy are modeled by the Poisson equation. The aerodynamics of airfoils at low speeds, known as incompressible irrotational flow, are a solution of the Laplace equation. The gravitational potential u generated by a distribution of mass density ρ satisfies the Poisson equation

$$\Delta u = 4\pi G\rho,$$

where G denotes the gravitational constant. A steady-state heat distribution, such as the limit of a solution of the heat equation as time $t \to \infty$, is modeled by the Poisson equation. In Reality Check 8, a variant of the Poisson equation is used to model the heat distribution on a cooling fin.

We introduce two methods for solving elliptic equations. The first is a Finite Difference Method that closely follows the development for parabolic and hyperbolic equations. The second generalizes the Finite Element Method for solving boundary value problems in Chapter 7. In most of the elliptic equations we consider, the domain is two-dimensional, which will cause a little extra bookkeeping work.

8.3.1 Finite Difference Method for elliptic equations

We will solve the Poisson equation $\Delta u = f$ on a rectangle $[x_l, x_r] \times [y_b, y_t]$ in the plane, with Dirichlet boundary conditions

$$\begin{aligned} u(x, y_b) &= g_1(x) \\ u(x, y_t) &= g_2(x) \\ u(x_l, y) &= g_3(y) \\ u(x_r, y) &= g_4(y) \end{aligned}$$

A rectangular mesh of points is shown in Figure 8.12(a), using $M = m - 1$ steps in the horizontal direction and $N = n - 1$ steps in the vertical direction. The mesh sizes in the x and y directions are $h = (x_r - x_l)/M$ and $k = (y_t - y_b)/N$, respectively.

A finite difference method involves approximating derivatives by difference quotients. The centered-difference formula (8.4) can be used for both second derivatives in the Laplacian operator. The Poisson equation $\Delta u = f$ has finite difference form

$$\begin{aligned} &\frac{u(x-h, y) - 2u(x, y) + u(x+h, y)}{h^2} + O(h^2) \\ &+ \frac{u(x, y-k) - 2u(x, y) + u(x, y+k)}{k^2} + O(k^2) = f(x, y), \end{aligned}$$

and in terms of the approximate solution $w_{ij} \approx u(x_i, y_j)$ can be written

$$\frac{w_{i-1,j} - 2w_{ij} + w_{i+1,j}}{h^2} + \frac{w_{i,j-1} - 2w_{i,j} + w_{i,j+1}}{k^2} = f(x_i, y_j) \tag{8.38}$$

where $x_i = x_l + (i-1)h$ and $y_j = y_b + (j-1)k$ for $1 \le i \le m$ and $1 \le j \le n$.

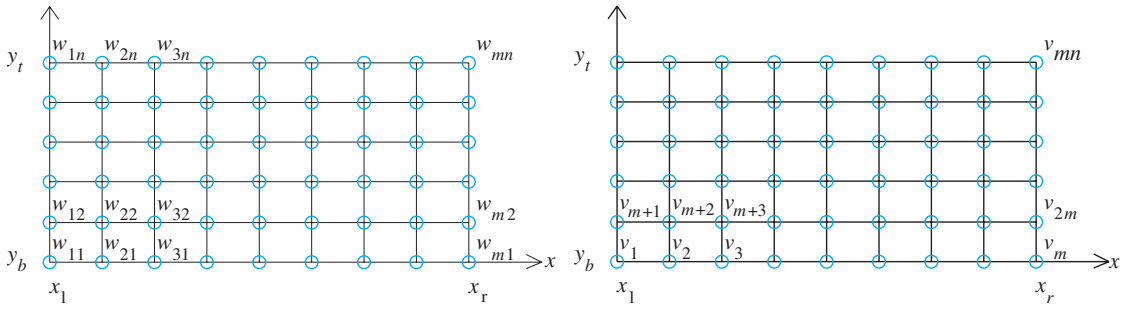

Figure 8.12 Mesh for finite difference solver of Poisson equation with Dirichlet boundary conditions. (a) Original numbering system with double subscripts. (b) Numbering system (8.39) for linear equations, with single subscripts, orders mesh points across rows.

Since the equations in the w_{ij} are linear, we are led to construct a matrix equation to solve for the mn unknowns. This presents a bookkeeping problem: We need to relabel these doubly indexed unknowns into a linear order. Figure 8.12(b) shows an alternative numbering system for the solution values, where we have set

$$v_{i+(j-1)m} = w_{ij}. \tag{8.39}$$

Next, we will construct a matrix A and vector b such that $Av = b$ can be solved for v, and translated back into the solution w on the rectangular grid. Since v is a vector of length mn, A will be an $mn \times mn$ matrix, and each grid point will correspond to its own linear equation.

By definition, the entry A_{pq} is the qth linear coefficient of the pth equation of $Av = b$. For example, (8.38) represents the equation at grid point (i, j), which we call equation number $p = i + (j - 1)m$, according to (8.39). The coefficients of the terms $w_{i-1,j}, w_{ij}, \ldots$ in (8.38) are also numbered according to (8.39), which we collect together in Table 8.1.

x	y	Equation number p
i	j	$i + (j - 1)m$

x	y	Coefficient number q
i	j	$i + (j - 1)m$
$i + 1$	j	$i + 1 + (j - 1)m$
$i - 1$	j	$i - 1 + (j - 1)m$
i	$j + 1$	$i + jm$
i	$j - 1$	$i + (j - 2)m$

Table 8.1 Translation table for two-dimensional domains. The equation at grid point (i, j) is numbered p, and its coefficients are A_{pq} for various q, with p and q given in the right column of the table. The table is simply an illustration of (8.39).

According to Table 8.1, labeling by equation number p and coefficient number q, the matrix entries A_{pq} from (8.38) are

$$A_{i+(j-1)m,i+(j-1)m} = -\frac{2}{h^2} - \frac{2}{k^2} \tag{8.40}$$

$$A_{i+(j-1)m,i+1+(j-1)m} = \frac{1}{h^2}$$

$$A_{i+(j-1)m,i-1+(j-1)m} = \frac{1}{h^2}$$
$$A_{i+(j-1)m,i+jm} = \frac{1}{k^2}$$
$$A_{i+(j-1)m,i+(j-2)m} = \frac{1}{k^2}.$$

The right-hand side of the equation corresponding to (i, j) is

$$b_{i+(j-1)m} = f(x_i, y_j).$$

These entries of A and b hold for the interior points $1 < i < m, 1 < j < n$ of the grid in Figure 8.12.

Each boundary point needs an equation as well. Since we assume Dirichlet boundary conditions, they are quite simple:

Bottom	$w_{ij} = g_1(x_i)$	for	$j = 1, \quad 1 \le i \le m$
Top side	$w_{ij} = g_2(x_i)$	for	$j = n, \quad 1 \le i \le m$
Left side	$w_{ij} = g_3(y_j)$	for	$i = 1, \quad 1 < j < n$
Right side	$w_{ij} = g_4(y_j)$	for	$i = m, \quad 1 < j < n$

The Dirichlet conditions translate via Table 8.1 to

Bottom	$A_{i+(j-1)m,i+(j-1)m} = 1, \quad b_{i+(j-1)m} = g_1(x_i)$	for	$j = 1, \quad 1 \le i \le m$
Top side	$A_{i+(j-1)m,i+(j-1)m} = 1, \quad b_{i+(j-1)m} = g_2(x_i)$	for	$j = n, \quad 1 \le i \le m$
Left side	$A_{i+(j-1)m,i+(j-1)m} = 1, \quad b_{i+(j-1)m} = g_3(y_j)$	for	$i = 1, \quad 1 < j < n$
Right side	$A_{i+(j-1)m,i+(j-1)m} = 1, \quad b_{i+(j-1)m} = g_4(y_j)$	for	$i = m, \quad 1 < j < n$

All other entries of A and b are zero. The linear system $Av = b$ can be solved with appropriate method from Chapter 2. We illustrate this labeling system in the next example.

▶ EXAMPLE 8.8

Apply the Finite Difference Method with $m = n = 5$ to approximate the solution of the Laplace equation $\Delta u = 0$ on $[0, 1] \times [1, 2]$ with the following Dirichlet boundary conditions:

$$u(x, 1) = \ln(x^2 + 1)$$
$$u(x, 2) = \ln(x^2 + 4)$$
$$u(0, y) = 2 \ln y$$
$$u(1, y) = \ln(y^2 + 1).$$

MATLAB code for the Finite Difference Method follows:

```
% Program 8.5 Finite difference solver for 2D Poisson equation
% with Dirichlet boundary conditions on a rectangle
% Input: rectangle domain [xl,xr]x[yb,yt] with MxN space steps
% Output: matrix w holding solution values
% Example usage: w=poisson(0,1,1,2,4,4)
function w=poisson(xl,xr,yb,yt,M,N)
f=@(x,y) 0; % define input function data
g1=@(x) log(x.^2+1); % define boundary values
g2=@(x) log(x.^2+4); % Example 8.8 is shown
g3=@(y) 2*log(y);
g4=@(y) log(y.^2+1);
```

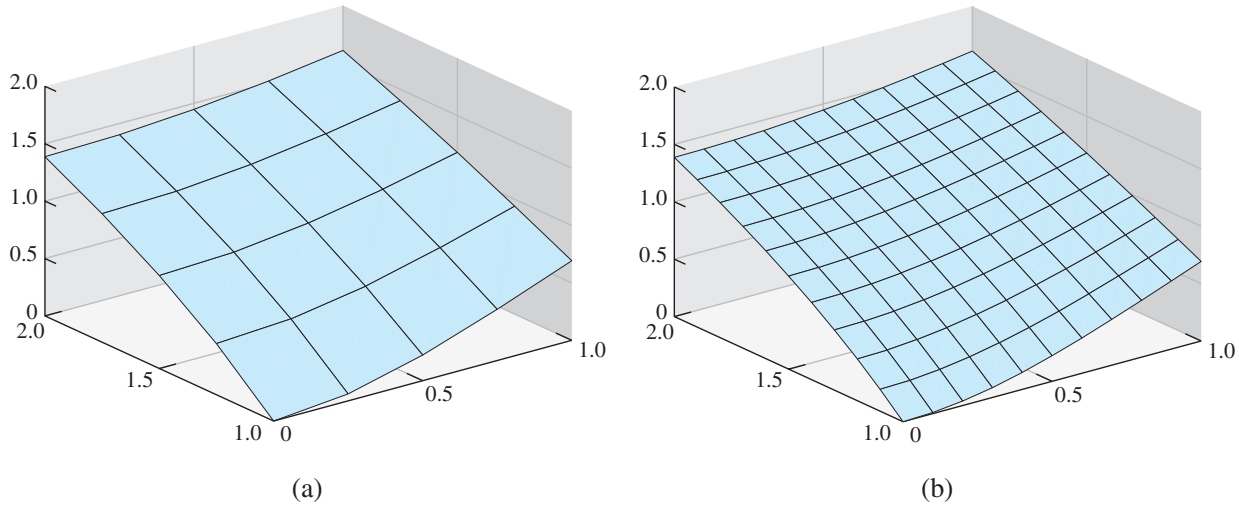

Figure 8.13 Finite Difference Method solution for the elliptic PDE in Example 8.8.
(a) $M = N = 4$, mesh sizes $h = k = 0.25$ (b) $M = N = 10$, mesh sizes $h = k = 0.1$.

```
m=M+1;n=N+1; mn=m*n;
h=(xr-xl)/M;h2=h^2;k=(yt-yb)/N;k2=k^2;
x=xl+(0:M)*h; % set mesh values
y=yb+(0:N)*k;
A=zeros(mn,mn);b=zeros(mn,1);
for i=2:m-1 % interior points
  for j=2:n-1
    A(i+(j-1)*m,i-1+(j-1)*m)=1/h2;A(i+(j-1)*m,i+1+(j-1)*m)=1/h2;
    A(i+(j-1)*m,i+(j-1)*m)=-2/h2-2/k2;
    A(i+(j-1)*m,i+(j-2)*m)=1/k2;A(i+(j-1)*m,i+j*m)=1/k2;
    b(i+(j-1)*m)=f(x(i),y(j));
  end
end
for i=1:m % bottom and top boundary points
  j=1;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g1(x(i));
  j=n;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g2(x(i));
end
for j=2:n-1 % left and right boundary points
  i=1;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g3(y(j));
  i=m;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g4(y(j));
end
v=A\b; % solve for solution in v labeling
w=reshape(v(1:mn),m,n);  %translate from v to w
mesh(x,y,w')
```

We will use the correct solution $u(x, y) = \ln(x^2 + y^2)$ to compare with the approximation at the nine mesh points in the square. Since $m = n = 5$, the mesh sizes are $h = k = 1/4$.

The solution finds the following nine interior values for u:

$$\begin{array}{lll} w_{24} = 1.1390 & w_{34} = 1.1974 & w_{44} = 1.2878 \\ w_{23} = 0.8376 & w_{33} = 0.9159 & w_{43} = 1.0341 \\ w_{22} = 0.4847 & w_{32} = 0.5944 & w_{42} = 0.7539 \end{array}$$

The approximate solution w_{ij} is plotted in Figure 8.13(a). It compares well with the exact solution $u(x, y) = \ln(x^2 + y^2)$ at the same points:

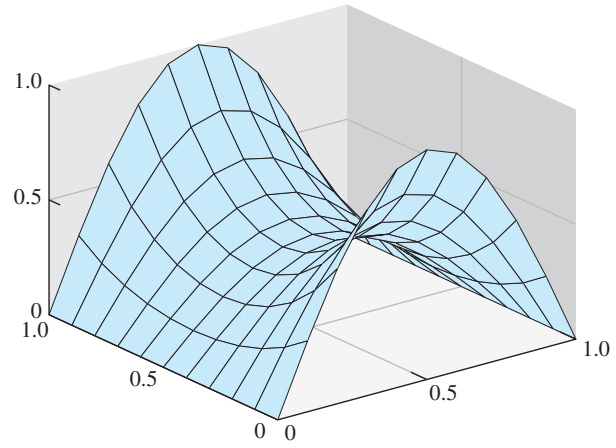

Figure 8.14 Electrostatic potential from the Laplace equation. Boundary conditions set in Example 8.9.

$$u(\tfrac{1}{4},\tfrac{7}{4}) = 1.1394 \quad u(\tfrac{2}{4},\tfrac{7}{4}) = 1.1977 \quad u(\tfrac{3}{4},\tfrac{7}{4}) = 1.2879$$
$$u(\tfrac{1}{4},\tfrac{6}{4}) = 0.8383 \quad u(\tfrac{2}{4},\tfrac{6}{4}) = 0.9163 \quad u(\tfrac{3}{4},\tfrac{6}{4}) = 1.0341$$
$$u(\tfrac{1}{4},\tfrac{5}{4}) = 0.4855 \quad u(\tfrac{2}{4},\tfrac{5}{4}) = 0.5947 \quad u(\tfrac{3}{4},\tfrac{5}{4}) = 0.7538$$

Since second-order finite difference formulas were used, the error of the Finite Difference Method `poisson.m` is second order in h and k. Figure 8.13(b) shows a more accurate approximate solution, for $h = k = 0.1$. The MATLAB code `poisson.m` is written for a rectangular domain, but changes can be made to shift to more general domains. ◀

For another example, we use the Laplace equation to compute a potential.

► **EXAMPLE 8.9** Find the electrostatic potential on the square $[0, 1] \times [0, 1]$, assuming no charge in the interior and assuming the following boundary conditions:

$$u(x, 0) = \sin \pi x$$
$$u(x, 1) = \sin \pi x$$
$$u(0, y) = 0$$
$$u(1, y) = 0.$$

The potential u satisfies the Laplace equation with Dirichlet boundary conditions. Using mesh size $h = k = 0.1$, or $M = N = 10$ in `poisson.m` results in the plot shown in Figure 8.14. ◀

Reality Check 8

Heat distribution on a cooling fin

Heat sinks are used to move excess heat away from the point where it is generated. In this project, the steady-state distribution along a rectangular fin of a heat sink will be modeled. The heat energy will enter the fin along part of one side. The main goal will be to design the dimensions of the fin to keep the temperature within safe tolerances.

The fin shape is a thin rectangular slab, with dimensions $L_x \times L_y$ and width δ cm, where δ is relatively small. Due to the thinness of the slab, we will denote the temperature by $u(x, y)$ and consider it constant along the width dimension.

Heat moves in the following three ways: conduction, convection, and radiation. Conduction refers to the passing of energy between neighboring molecules, perhaps due to

the movement of electrons, while in convection the molecules themselves move. Radiation, the movement of energy through photons, will not be considered here.

Conduction proceeds through a conducting material according to Fourier's first law

$$q = -KA\nabla u, \tag{8.41}$$

where q is heat energy per unit time (measured in watts), A is the cross-sectional area of the material, and ∇u is the gradient of the temperature. The constant K is called the **thermal conductivity** of the material. Convection is ruled by Newton's law of cooling,

$$q = -HA(u - u_b), \tag{8.42}$$

where H is a proportionality constant called the **convective heat transfer coefficient** and u_b is the ambient temperature, or **bulk temperature**, of the surrounding fluid (in this case, air).

The fin is a rectangle $[0, L_x] \times [0, L_y]$ by δ cm in the z direction, as illustrated in Figure 8.15(a). Energy equilibrium in a typical $\Delta x \times \Delta y \times \delta$ box interior to the fin, aligned along the x and y axes, says that the energy entering the box per unit time equals the energy leaving. The heat flux into the box through the two $\Delta y \times \delta$ sides and two $\Delta x \times \delta$ sides is by conduction, and through the two $\Delta x \times \Delta y$ sides is by convection, yielding the steady-state equation

$$\begin{aligned} &-K\Delta y\delta u_x(x, y) + K\Delta y\delta u_x(x + \Delta x, y) - K\Delta x\delta u_y(x, y) \\ &\quad + K\Delta x\delta u_y(x, y + \Delta y) - 2H\Delta x\Delta y u(x, y) = 0. \end{aligned} \tag{8.43}$$

Here, we have set the bulk temperature $u_b = 0$ for convenience; thus, u will denote the difference between the fin temperature and the surroundings.

Dividing through by $\Delta x\Delta y$ gives

$$K\delta\frac{u_x(x + \Delta x, y) - u_x(x, y)}{\Delta x} + K\delta\frac{u_y(x, y + \Delta y) - u_y(x, y)}{\Delta y} = 2Hu(x, y),$$

and in the limit as $\Delta x, \Delta y \to 0$, the elliptic partial differential equation

$$u_{xx} + u_{yy} = \frac{2H}{K\delta}u \tag{8.44}$$

results.

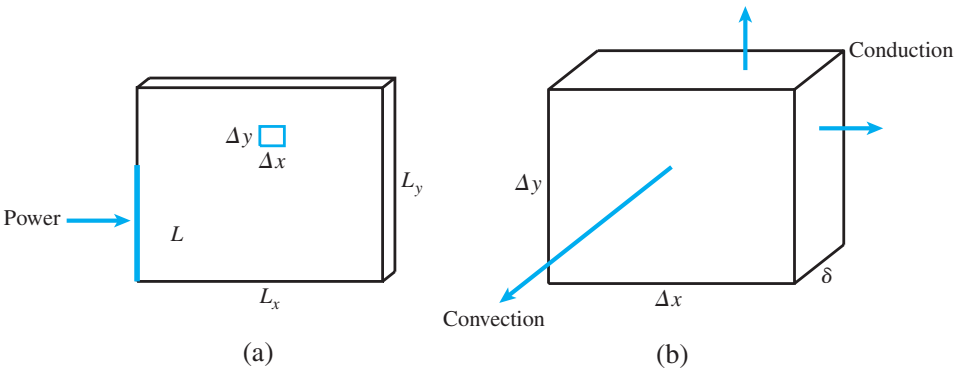

Figure 8.15 Cooling fin in Reality Check 8. (a) Power input occurs along interval $[0, L]$ on left side of fin. (b) Energy transfer in small interior box is by conduction along the x and y directions, and by convection along the air interface.

Similar arguments imply the **convective** boundary condition

$$Ku_{\text{normal}} = Hu$$

where u_{normal} is the partial derivative with respect to the outward normal direction $\vec{n}$. The convective boundary condition is known as a **Robin** boundary condition, one that involves both the function value and its derivative. Finally, we will assume that power enters the fin along one side according to Fourier's law,

$$u_{\text{normal}} = \frac{P}{L\delta K},$$

where P is the total power and L is the length of the input.

On a discrete grid with step sizes h and k, respectively, the finite difference approximation (5.8) can be used to approximate the PDE (8.44) as

$$\frac{u_{i+1,j} - 2u_{ij} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{ij} + u_{i,j-1}}{k^2} = \frac{2H}{K\delta}u_{ij}.$$

This discretization is used for the interior points (x_i, y_j) where $1 < i < m$, $1 < j < n$ for integers m, n. The fin edges obey the Robin conditions using the first derivative approximation

$$f'(x) = \frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h} + O(h^2).$$

To apply this approximation to the fin edges, note that the outward normal direction translates to

$$\begin{aligned}
u_{\text{normal}} &= -u_y \text{ on bottom edge} \\
u_{\text{normal}} &= u_y \text{ on top edge} \\
u_{\text{normal}} &= -u_x \text{ on left edge} \\
u_{\text{normal}} &= u_x \text{ on right edge}
\end{aligned}$$

Second, note that the second-order first derivative approximation above yields

$$\begin{aligned}
u_y &\approx \frac{-3u(x,y) + 4u(x,y+k) - u(x,y+2k)}{2k} \text{ on bottom edge} \\
u_y &\approx \frac{-3u(x,y) + 4u(x,y-k) - u(x,y-2k)}{-2k} \text{ on top edge} \\
u_x &\approx \frac{-3u(x,y) + 4u(x+h,y) - u(x+2h,y)}{2h} \text{ on left edge} \\
u_x &\approx \frac{-3u(x,y) + 4u(x-h,y) - u(x-2h,y)}{-2h} \text{ on right edge}
\end{aligned}$$

Putting both together, the Robin boundary condition leads to the difference equations

$$\begin{aligned}
\frac{-3u_{i1} + 4u_{i2} - u_{i3}}{2k} &= -\frac{H}{K}u_{i1} \text{ on bottom edge} \\
\frac{-3u_{in} + 4u_{i,n-1} - u_{i,n-2}}{2k} &= -\frac{H}{K}u_{in} \text{ on top edge} \\
\frac{-3u_{1j} + 4u_{2j} - u_{3j}}{2h} &= -\frac{H}{K}u_{1j} \text{ on left edge} \\
\frac{-3u_{mj} + 4u_{m-1,j} - u_{m-2,j}}{2h} &= -\frac{H}{K}u_{mj} \text{ on right edge.}
\end{aligned}$$

If we assume that the power enters along the left side of the fin, Fourier's law leads to the equation

$$\frac{-3u_{1j} + 4u_{2j} - u_{3j}}{2h} = -\frac{P}{L\delta K}. \tag{8.45}$$

There are mn equations in the mn unknowns u_{ij}, $1 \le i \le m$, $1 \le j \le n$ to solve.

Assume that the fin is composed of aluminum, whose thermal conductivity is $K = 1.68\,\text{W/cm}\,^\circ\text{C}$ (watts per centimeter-degree Celsius). Assume that the convective heat transfer coefficient is $H = 0.005\,\text{W/cm}^2\,^\circ\text{C}$, and that the room temperature is $u_b = 20\,^\circ\text{C}$.

Suggested activities:

1. Begin with a fin of dimensions 2×2 cm, with 1 mm thickness. Assume that 5W of power is input along the entire left edge, as if the fin were attached to dissipate power from a CPU chip with $L = 2$ cm side length. Solve the PDE (8.44) with $M = N = 10$ steps in the x and y directions. Use the `mesh` command to plot the resulting heat distribution over the xy-plane. What is the maximum temperature of the fin, in °C ?

2. Increase the size of the fin to 4×4 cm. Input 5W of power along the interval $[0, 2]$ on the left side of the fin, as in the previous step. Plot the resulting distribution, and find the maximum temperature. Experiment with increased values of M and N. How much does the solution change?

3. Find the maximum power that can be dissipated by a 4×4 cm fin while keeping the maximum temperature less than 80 °C. Assume that the bulk temperature is 20 °C and the power input is along 2 cm, as in steps 1 and 2.

4. Replace the aluminum fin with a copper fin, with thermal conductivity $K = 3.85\,\text{W/cm}\,^\circ\text{C}$. Find the maximum power that can be dissipated by a 4×4 cm fin with the 2 cm power input in the optimal placement, while keeping the maximum temperature below 80 °C.

5. Plot the maximum power that can be dissipated in step 4 (keeping maximum temperature below 80 degrees) as a function of thermal conductivity, for $1 \le K \le 5\,\text{W/cm}^\circ\text{C}$.

6. Redo step 4 for a water-cooled fin. Assume that water has a convective heat transfer coefficient of $H = 0.1\,\text{W/cm}^2\,^\circ\text{C}$, and that the ambient water temperature is maintained at 20°C.

7. Cut a rectangular notch from the right side of the fin, and redo step 4. Does the notched fin dissipate more, or less, power than the original?

The design of cooling fins for desktop and laptop computers is a fascinating engineering problem. To dissipate ever greater amounts of heat, several fins are needed in a small space, and fans are used to enhance convection near the fin edges. The addition of fans to complicated fin geometry moves the simulation into the realm of computational fluid dynamics, a vital area of modern applied mathematics. ✔

8.3.2 Finite Element Method for elliptic equations

A somewhat more flexible approach to solving partial differential equations arose from the structural engineering community in the mid-20th century. The Finite Element Method converts the differential equation into a variational equivalent called the weak form of the equation, and uses the powerful idea of orthogonality in function spaces to stabilize its calculations. Moreover, the resulting system of linear equations can have considerable symmetry in its structure matrix, even when the underlying geometry is complicated.

We will apply finite elements by using the Galerkin Method, as introduced in Chapter 7 for ordinary differential equation boundary value problems. The method for PDEs follows the same steps, although the bookkeeping requirements are more extensive. Consider the Dirichlet problem for the elliptic equation

$$\begin{aligned} \Delta u + r(x, y)u &= f(x, y) \quad \text{in } R \\ u &= g(x, y) \quad \text{on } S \end{aligned} \tag{8.46}$$

where the solution $u(x, y)$ is defined on a region R in the plane bounded by a piecewise-smooth closed curve S.

We will use an L^2 function space over the region R, as in Chapter 7. Let

$$L^2(R) = \left\{ \text{functions } \phi(x, y) \text{ on } R \;\middle|\; \int\int_R \phi(x, y)^2 \, dx \, dy \text{ exists and is finite} \right\}.$$

Denote by $L_0^2(R)$ the subspace of $L^2(R)$ consisting of functions that are zero on the boundary S of the region R.

The goal will be to minimize the squared error of the elliptic equation in (8.46) by forcing the residual $\Delta u(x, y) + r(x, y)u(x, y) - f(x, y)$ to be orthogonal to a large subspace of $L^2(R)$. Let $\phi_1(x, y), \ldots, \phi_P(x, y)$ be elements of $L^2(R)$. The orthogonality assumption takes the form

$$\int\int_R (\Delta u + ru - f)\phi_p \, dx \, dy = 0,$$

or

$$\int\int_R (\Delta u + ru)\phi_p \, dx dy = \int\int_R f\phi_p \, dx \, dy \tag{8.47}$$

for each $1 \le p \le P$. The form (8.47) is called the **weak form** of the elliptic equation (8.46).

The version of integration by parts needed to apply the Galerkin Method is contained in the following fact:

THEOREM 8.7 **Green's First Identity.** Let R be a bounded region with piecewise smooth boundary S. Let u and v be smooth functions, and let n denote the outward unit normal along the boundary. Then

$$\int\int_R v\Delta u = \int_S v\frac{\partial u}{\partial n} dS - \int\int_R \nabla u \cdot \nabla v. \qquad \blacksquare$$

The directional derivative can be calculated as

$$\frac{\partial u}{\partial n} = \nabla u \cdot (n_x, n_y),$$

where (n_x, n_y) denotes the outward normal unit vector on the boundary S of R. Green's identity applied to the weak form (8.47) yields

$$\int_S \phi_p \frac{\partial u}{\partial n} \, dS \; - \int\int_R (\nabla u \cdot \nabla \phi_p) \, dx \, dy + \int\int_R ru\phi_p \, dx \, dy = \int\int_R f\phi_p \, dx \, dy. \tag{8.48}$$

The essence of the Finite Element Method is to substitute

$$w(x, y) = \sum_{q=1}^{P} v_q \phi_q(x, y) \tag{8.49}$$

for u into the weak form of the partial differential equation, and then determine the unknown constants v_q. Assume for the moment that ϕ_p belongs to $L_0^2(R)$, that is, $\phi_p(S) = 0$. Substituting the form (8.49) into (8.48) results in

$$-\int\int_R \left(\sum_{q=1}^{P} v_q \nabla\phi_q\right) \cdot \nabla\phi_p \, dx\,dy + \int\int_R r\left(\sum_{q=1}^{P} v_q \phi_q\right)\phi_p \, dx\,dy = \int\int_R f\phi_p \, dx\,dy$$

for each ϕ_p in $L_0^2(R)$. Factoring out the constants v_q yields

$$\sum_{q=1}^{P} v_q \left[\int\int_R \nabla\phi_q \cdot \nabla\phi_p \, dx\,dy - \int\int_R r\phi_q\phi_p \, dx\,dy\right] = -\int_R f\phi_p dx\,dy. \tag{8.50}$$

For each ϕ_p belonging to $L_0^2(R)$, we have developed a linear equation in the unknowns $v_1, \ldots, v_P$. In matrix form, the equation is $Av = b$, where the entries of the pth row of A and b are

$$A_{pq} = \int\int_R \nabla\phi_q \cdot \nabla\phi_p \, dx\,dy - \int\int_R r\phi_q\phi_p \, dx\,dy \tag{8.51}$$

and

$$b_p = -\int\int_R f\phi_p \, dx\,dy. \tag{8.52}$$

We are now prepared to choose explicit functions for the finite elements ϕ_p and plan a computation. We follow the lead of Chapter 7 in choosing linear B-splines, piecewise-linear functions of x, y that live on triangles in the plane. For concreteness, let the region R be a rectangle, and form a triangulation with nodes (x_i, y_j) chosen from a rectangular grid. We will reuse the $M \times N$ grid from the previous section, shown in Figure 8.16(a), where we set $m = M + 1$ and $n = N + 1$. As before, we will denote the grid step size in the x and y directions as h and k, respectively. Figure 8.16(b) shows the triangulation of the rectangular region that we will use.

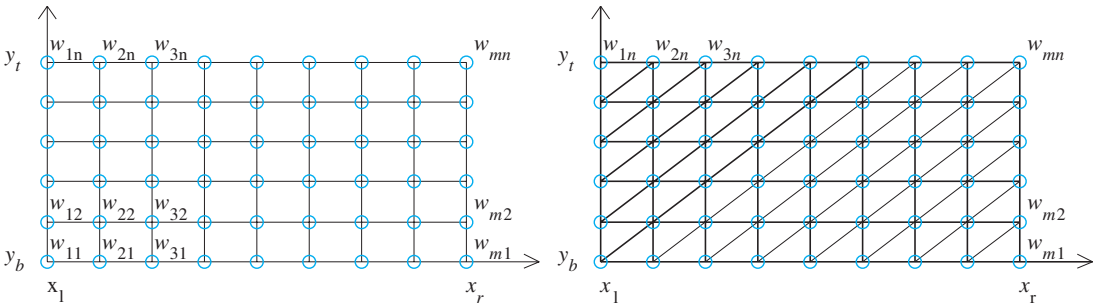

Figure 8.16 Finite element solver of elliptic equation with Dirichlet boundary conditions. (a) Mesh is same as used for finite difference solver. (b) A possible triangulation of the region. Each interior point is a vertex of six different triangles.

Our choice of finite element functions ϕ_p from $L^2(R)$ will be the $P = mn$ piecewise-linear functions, each of which takes the value 1 at one grid point in Figure 8.16(a) and zero at the other $mn - 1$ grid points. In other words, $\phi_1, \ldots, \phi_{mn}$ are determined by the equality $\phi_{i+(j-1)m}(x_i, y_j) = 1$ and $\phi_{i+(j-1)m}(x_{i'}, y_{j'}) = 0$ for all other grid points $(x_{i'}, y_{j'})$, while being linear on each triangle in Figure 8.16(b). We are once again using the numbering system of Table 8.1, on page 400. Each $\phi_p(x, y)$ is differentiable, except along the triangle edges, and is therefore a Riemann-integrable function belonging to $L^2(R)$. Note that for every nonboundary point (x_i, y_j) of the rectangle R, $\phi_{i+(j-1)m}$ belongs to $L_0^2(R)$. Moreover, due to assumption (8.49), they satisfy

$$w(x_i, y_j) = \sum_{i=1}^{m} \sum_{j=1}^{n} v_{i+(j-1)m} \phi_{i+(j-1)m}(x_i, y_j) = v_{i+(j-1)m}$$

for $i = 1, \ldots, m$, $j = 1, \ldots, n$. Therefore, the approximation w to the correct solution u at (x_i, v_j) will be directly available once the system $Av = b$ is solved. This convenience is the reason B-splines are a good choice for finite element functions.

It remains to calculate the matrix entries (8.51) and (8.52) and solve $Av = b$. To calculate these entries, we gather a few facts about B-splines in the plane. The integrals of the piecewise-linear functions are easily approximated by the two-dimensional Midpoint Rule. Define the **barycenter** of a region in the plane as the point $(\overline{x}, \overline{y})$ where

$$\overline{x} = \frac{\int\int_R x \, dx\, dy}{\int\int_R 1 \, dx\, dy}, \quad \overline{y} = \frac{\int\int_R y \, dx\, dy}{\int\int_R 1 \, dx\, dy}.$$

If R is a triangle with vertices $(x_1, y_1), (x_2, y_2), (x_3, y_3)$, then the barycenter is (see Exercise 8)

$$\overline{x} = \frac{x_1 + x_2 + x_3}{3}, \quad \overline{y} = \frac{y_1 + y_2 + y_3}{3}.$$

LEMMA 8.8 The average value of a linear function $L(x, y)$ on a plane region R is $L(\overline{x}, \overline{y})$, the value at the barycenter. In other words, $\int\int_R L(x, y)\, dx\, dy = L(\overline{x}, \overline{y}) \cdot$ area (R). ■

Proof. Let $L(x, y) = a + bx + cy$. Then

$$\begin{aligned} \int\int_R L(x, y)\, dx\, dy &= \int\int_R (a + bx + cy)\, dx\, dy \\ &= a \int\int_R dx\, dy + b \int\int_R x \, dx\, dy + c \int\int_R y \, dx\, dy \\ &= \text{area } (R) \cdot (a + b\overline{x} + c\overline{y}). \end{aligned}$$

❐

Lemma 8.8 leads to a generalization of the Midpoint Rule of Chapter 5 that is useful for approximating the entries of (8.51) and (8.52). Taylor's Theorem for functions of two variables says that

$$\begin{aligned} f(x, y) &= f(\overline{x}, \overline{y}) + \frac{\partial f}{\partial x}(\overline{x}, \overline{y})(x - \overline{x}) + \frac{\partial f}{\partial y}(\overline{x}, \overline{y})(y - \overline{y}) \\ &\quad + O((x - \overline{x})^2, (x - \overline{x})(y - \overline{y}), (y - \overline{y})^2) \\ &= L(x, y) + O((x - \overline{x})^2, (x - \overline{x})(y - \overline{y}), (y - \overline{y})^2). \end{aligned}$$

Therefore,

$$\int\int_R f(x,y)\,dx\,dy = \int\int_R L(x,y)\,dx\,dy + \int\int_R O((x-\overline{x})^2,(x-\overline{x})(y-\overline{y}),(y-\overline{y})^2)\,dx\,dy$$
$$= \text{area}\,(R)\cdot L(\overline{x},\overline{y}) + O(h^4) = \text{area}\,(R)\cdot f(\overline{x},\overline{y}) + O(h^4),$$

where h is the **diameter** of R, the largest distance between two points of R, and where we have used Lemma 8.8. This is the Midpoint Rule in two dimensions.

Midpoint Rule in two dimensions

$$\int\int_R f(x,y)\,dx\,dy = \text{area}\,(R)\cdot f(\overline{x},\overline{y}) + O(h^4), \tag{8.53}$$

where $(\overline{x},\overline{y})$ is the barycenter of the bounded region R and $h = \text{diam}(R)$.

The Midpoint Rule shows that to apply the Finite Element Method with $O(h^2)$ convergence, we need to only approximate the integrals in (8.51) and (8.52) by evaluating integrands at triangle barycenters. For the B-spline functions ϕ_p, this is particularly easy. Proofs of the next two lemmas are deferred to Exercises 9 and 10.

LEMMA 8.9 Let $\phi(x,y)$ be a linear function on the triangle T with vertices $(x_1,y_1),(x_2,y_2),(x_3,y_3)$, satisfying $\phi(x_1,y_1)=1, \phi(x_2,y_2)=0$, and $\phi(x_3,y_3)=0$. Then $\phi(\overline{x},\overline{y}) = 1/3$. ■

LEMMA 8.10 Let $\phi_1(x,y)$ and $\phi_2(x,y)$ be the linear functions on the triangle T with vertices $(x_1,y_1),(x_2,y_2)$, and (x_3,y_3), satisfying $\phi_1(x_1,y_1)=1, \phi_1(x_2,y_2)=0$, $\phi_1(x_3,y_3)=0$, $\phi_2(x_1,y_1)=0, \phi_2(x_2,y_2)=1$, and $\phi_2(x_3,y_3)=0$. Let $f(x,y)$ be a twice-differentiable function. Set

$$d = \det\begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix}.$$

Then

(a) the triangle T has area $|d|/2$

(b) $\nabla\phi_1(x,y) = \left(\dfrac{y_2-y_3}{d}, \dfrac{x_3-x_2}{d}\right)$

(c) $\int\int_T \nabla\phi_1\cdot\nabla\phi_1\,dx\,dy = \dfrac{(x_2-x_3)^2+(y_2-y_3)^2}{2|d|}$

(d) $\int\int_T \nabla\phi_1\cdot\nabla\phi_2\,dx\,dy = \dfrac{-(x_1-x_3)(x_2-x_3)-(y_1-y_3)(y_2-y_3)}{2|d|}$

(e) $\int\int_T f\phi_1\phi_2\,dx\,dy = f(\overline{x},\overline{y})|d|/18 + O(h^4) = \int\int_T f\phi_1^2\,dx\,dy$

(f) $\int\int_T f\phi_1\,dx\,dy = f(\overline{x},\overline{y})|d|/6 + O(h^4)$

where $(\overline{x},\overline{y})$ is the barycenter of T and $h = \text{diam}(T)$. ■

We can now calculate the matrix entries of A. Consider a vertex (x_i,y_j) that is not on the boundary S of the rectangle. Then $\phi_{i+(j-1)m}$ belongs to $L_0^2(R)$, and according to (8.51) with $p = q = i+(j-1)m$, the matrix entry $A_{i+(j-1)m,i+(j-1)m}$ is composed of two integrals. The integrands are zero outside of the six triangles shown in Figure 8.17.

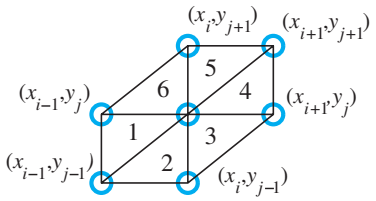

Figure 8.17 Detail of the (i,j) interior point from Figure 8.16(b). Each interior point (x_i, y_j) is surrounded by six triangles, numbered as shown. The B-spline function $\phi_{i+(j-1)m}$ is linear, takes the value 1 at the center, and is zero outside of these six triangles.

The triangles have horizontal and vertical sides h and k, respectively. For the first integral, summing from triangle 1 to triangle 6, respectively, we can use Lemma 8.10(c) to sum the six contributions

$$\frac{k^2}{2hk} + \frac{h^2}{2hk} + \frac{h^2+k^2}{2hk} + \frac{k^2}{2hk} + \frac{h^2}{2hk} + \frac{h^2+k^2}{2hk} = \frac{2(h^2+k^2)}{hk}. \tag{8.54}$$

For the second integral of (8.51), we use Lemma 8.10(e). Again, the integrals are zero except for the six triangles shown. The barycenters of the six triangles are

$$\begin{aligned}
B_1 &= (x_i - \frac{2}{3}h, y_j - \frac{1}{3}k) \\
B_2 &= (x_i - \frac{1}{3}h, y_j - \frac{2}{3}k) \\
B_3 &= (x_i + \frac{1}{3}h, y_j - \frac{1}{3}k) \\
B_4 &= (x_i + \frac{2}{3}h, y_j + \frac{1}{3}k) \\
B_5 &= (x_i + \frac{1}{3}h, y_j + \frac{2}{3}k) \\
B_6 &= (x_i - \frac{1}{3}h, y_j + \frac{1}{3}k).
\end{aligned} \tag{8.55}$$

The second integral contributes $-(hk/18)[r(B_1) + r(B_2) + r(B_3) + r(B_4) + r(B_5) + r(B_6)]$, and so summing up (8.54) and (8.55),

$$\begin{aligned}
A_{i+(j-1)m,i+(j-1)m} = \frac{2(h^2+k^2)}{hk} - \frac{hk}{18}[&r(B_1) + r(B_2) + r(B_3) \\
&+ r(B_4) + r(B_5) + r(B_6)].
\end{aligned} \tag{8.56}$$

Similar usage of Lemma 8.10 (see Exercise 12) shows that

$$\begin{aligned}
A_{i+(j-1)m,i-1+(j-1)m} &= -\frac{k}{h} - \frac{hk}{18}[r(B_6) + r(B_1)] \\
A_{i+(j-1)m,i-1+(j-2)m} &= -\frac{hk}{18}[r(B_1) + r(B_2)] \\
A_{i+(j-1)m,i+(j-2)m} &= -\frac{h}{k} - \frac{hk}{18}[r(B_2) + r(B_3)] \\
A_{i+(j-1)m,i+1+(j-1)m} &= -\frac{k}{h} - \frac{hk}{18}[r(B_3) + r(B_4)] \\
A_{i+(j-1)m,i+1+jm} &= -\frac{hk}{18}[r(B_4) + r(B_5)]
\end{aligned}$$

$$A_{i+(j-1)m,i+jm} = -\frac{h}{k} - \frac{hk}{18}[r(B_5) + r(B_6)] \tag{8.57}$$

Calculating the entries b_p makes use of Lemma 8.10(f), which implies that for $p = i + (j-1)m$,

$$b_{i+(j-1)m} = -\frac{hk}{6}[f(B_1) + f(B_2) + f(B_3) + f(B_4) + f(B_5) + f(B_6)]. \tag{8.58}$$

For finite element functions on the boundary, $\phi_{i+(j-1)m}$ does not belong to $L_0^2(R)$, and the equations

$$\begin{aligned} A_{i+(j-1)m,i+(j-1)m} &= 1 \\ b_{i+(j-1)m} &= g(x_i, y_j) \end{aligned} \tag{8.59}$$

will be used to guarantee the Dirichlet boundary condition $v_{i+(j-1)m} = g(x_i, y_j)$, where (x_i, y_j) is a boundary point.

With these formulas, it is straightforward to build a MATLAB implementation of the finite element solver on a rectangle with Dirichlet boundary conditions. The program consists of setting up the matrix A and vector b using (8.56) – (8.59), and solving $Av = b$. Although the MATLAB backslash operation is used in the code, for real applications it might be replaced by a sparse solver as in Chapter 2.

```
% Program 8.6 Finite element solver for 2D PDE
% with Dirichlet boundary conditions on a rectangle
% Input: rectangle domain [xl,xr]x[yb,yt] with MxN space steps
% Output: matrix w holding solution values
% Example usage: w=poissonfem(0,1,1,2,4,4)
function w=poissonfem(xl,xr,yb,yt,M,N)
f=@(x,y) 0; % define input function data
r=@(x,y) 0;
g1=@(x) log(x.^2+1); % define boundary values on bottom
g2=@(x) log(x.^2+4); % top
g3=@(y) 2*log(y);    % left side
g4=@(y) log(y.^2+1); % right side
m=M+1; n=N+1; mn=m*n;
h=(xr-xl)/M; h2=h^2; k=(yt-yb)/N; k2=k^2; hk=h*k;
x=xl+(0:M)*h; % set mesh values
y=yb+(0:N)*k;
A=zeros(mn,mn); b=zeros(mn,1);
for i=2:m-1 % interior points
 for j=2:n-1
  rsum=r(x(i)-2*h/3,y(j)-k/3)+r(x(i)-h/3,y(j)-2*k/3)...
       +r(x(i)+h/3,y(j)-k/3);
  rsum=rsum+r(x(i)+2*h/3,y(j)+k/3)+r(x(i)+h/3,y(j)+2*k/3)...
       +r(x(i)-h/3,y(j)+k/3);
  A(i+(j-1)*m,i+(j-1)*m)=2*(h2+k2)/(hk)-hk*rsum/18;
  A(i+(j-1)*m,i-1+(j-1)*m)=-k/h-hk*(r(x(i)-h/3,y(j)+k/3)...
                           +r(x(i)-2*h/3,y(j)-k/3))/18;
  A(i+(j-1)*m,i-1+(j-2)*m)=-hk*(r(x(i)-2*h/3,y(j)-k/3)...
                           +r(x(i)-h/3,y(j)-2*k/3))/18;
  A(i+(j-1)*m,i+(j-2)*m)=-h/k-hk*(r(x(i)-h/3,y(j)-2*k/3)...
                         +r(x(i)+h/3,y(j)-k/3))/18;
  A(i+(j-1)*m,i+1+(j-1)*m)=-k/h-hk*(r(x(i)+h/3,y(j)-k/3)...
                           +r(x(i)+2*h/3,y(j)+k/3))/18;
```

```
  A(i+(j-1)*m,i+1+j*m)=-hk*(r(x(i)+2*h/3,y(j)+k/3)...
                          +r(x(i)+h/3,y(j)+2*k/3))/18;
  A(i+(j-1)*m,i+j*m)=-h/k-hk*(r(x(i)+h/3,y(j)+2*k/3)...
                        +r(x(i)-h/3,y(j)+k/3))/18;
  fsum=f(x(i)-2*h/3,y(j)-k/3)+f(x(i)-h/3,y(j)-2*k/3)...
       +f(x(i)+h/3,y(j)-k/3);
  fsum=fsum+f(x(i)+2*h/3,y(j)+k/3)+f(x(i)+h/3,y(j)+2*k/3)...
       +f(x(i)-h/3,y(j)+k/3);
  b(i+(j-1)*m)=-h*k*fsum/6;
 end
end
for i=1:m % boundary points
  j=1;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g1(x(i));
  j=n;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g2(x(i));
end
for j=2:n-1
  i=1;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g3(y(j));
  i=m;A(i+(j-1)*m,i+(j-1)*m)=1;b(i+(j-1)*m)=g4(y(j));
end
v=A\b; % solve for solution in v numbering
w=reshape(v(1:mn),m,n);
mesh(x,y,w')
```

▶ EXAMPLE 8.10 Apply the Finite Element Method with $M = N = 4$ to approximate the solution of the Laplace equation $\Delta u = 0$ on $[0, 1] \times [1, 2]$ with the Dirichlet boundary conditions:

$$\begin{aligned} u(x, 1) &= \ln(x^2 + 1) \\ u(x, 2) &= \ln(x^2 + 4) \\ u(0, y) &= 2 \ln y \\ u(1, y) &= \ln(y^2 + 1) \end{aligned}$$

Since $M = N = 4$, there is a $mn \times mn$ linear system to solve. Sixteen of the 25 equations are evaluation of the boundary conditions. Solving $Av = b$ yields

$$\begin{array}{lll} w_{24} = 1.1390 & w_{34} = 1.1974 & w_{44} = 1.2878 \\ w_{23} = 0.8376 & w_{33} = 0.9159 & w_{43} = 1.0341 \\ w_{22} = 0.4847 & w_{32} = 0.5944 & w_{42} = 0.7539 \end{array}$$

agreeing with the results in Example 8.8. ◀

▶ EXAMPLE 8.11 Apply the Finite Element Method with $M = N = 16$ to approximate the solution of the elliptic Dirichlet problem

$$\begin{cases} \Delta u + 4\pi^2 u = 2 \sin 2\pi y \\ u(x, 0) = 0 \text{ for } 0 \le x \le 1 \\ u(x, 1) = 0 \text{ for } 0 \le x \le 1 \\ u(0, y) = 0 \text{ for } 0 \le y \le 1 \\ u(1, y) = \sin 2\pi y \text{ for } 0 \le y \le 1 \end{cases}$$

We define $r(x, y) = 4\pi^2$ and $f(x, y) = 2 \sin 2\pi y$. Since $m = n = 17$, the grid is 17×17, meaning that the matrix A is 289×289. The solution is computed approximately within a maximum error of about 0.023, compared with the correct solution $u(x, y) = x^2 \sin 2\pi y$. The approximate solution w is shown in Figure 8.18. ◀

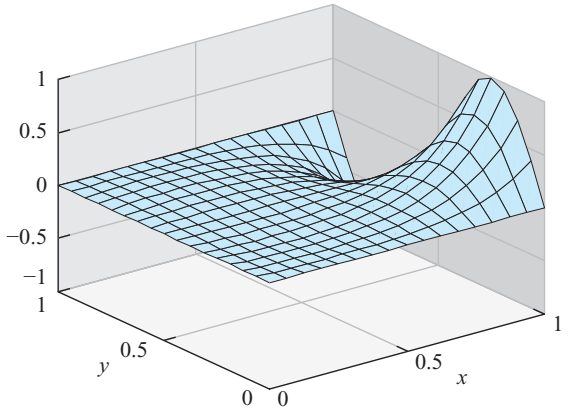

Figure 8.18 Finite element solution of Example 8.11. Maximum error on $[0, 1] \times [0, 1]$ is 0.023.

8.3 Exercises

1. Show that $u(x, y) = \ln(x^2 + y^2)$ is a solution to the Laplace equation with Dirichlet boundary conditions of Example 8.8.

2. Show that (a) $u(x, y) = x^2 y - 1/3\ y^3$ and (b) $u(x, y) = 1/6\, x^4 - x^2 y^2 + 1/6\ y^4$ are harmonic functions.

3. Prove that the functions (a) $u(x, y) = e^{-\pi y} \sin \pi x$, (b) $u(x, y) = \sinh \pi x \sin \pi y$ are solutions of the Laplace equation with the specified boundary conditions:

$$\text{(a)} \quad \begin{cases} u(x,0) = \sin \pi x \text{ for } 0 \le x \le 1 \\ u(x,1) = e^{-\pi} \sin \pi x \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = 0 \text{ for } 0 \le y \le 1 \end{cases} \qquad \text{(b)} \quad \begin{cases} u(x,0) = 0 \text{ for } 0 \le x \le 1 \\ u(x,1) = 0 \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = \sinh \pi \sin \pi y \text{ for } 0 \le y \le 1 \end{cases}$$

4. Prove that the functions (a) $u(x, y) = e^{-xy}$, (b) $u(x, y) = (x^2 + y^2)^{3/2}$ are solutions of the specified Poisson equation with the given boundary conditions:

$$\text{(a)} \quad \begin{cases} \Delta u = e^{-xy}(x^2 + y^2) \\ u(x,0) = 1 \text{ for } 0 \le x \le 1 \\ u(x,1) = e^{-x} \text{ for } 0 \le x \le 1 \\ u(0,y) = 1 \text{ for } 0 \le y \le 1 \\ u(1,y) = e^{-y} \text{ for } 0 \le y \le 1 \end{cases} \qquad \text{(b)} \quad \begin{cases} \Delta u = 9\sqrt{x^2 + y^2} \\ u(x,0) = x^3 \text{ for } 0 \le x \le 1 \\ u(x,1) = (1 + x^2)^{3/2} \text{ for } 0 \le x \le 1 \\ u(0,y) = y^3 \text{ for } 0 \le y \le 1 \\ u(1,y) = (1 + y^2)^{3/2} \text{ for } 0 \le y \le 1 \end{cases}$$

5. Prove that the functions (a) $u(x, y) = \sin \frac{\pi}{2}\, xy$, (b) $u(x, y) = e^{xy}$ are solutions of the specified elliptic equation with the given Dirichlet boundary conditions:

$$\text{(a)} \quad \begin{cases} \Delta u + \frac{\pi^2}{4}(x^2 + y^2)u = 0 \\ u(x,0) = 0 \text{ for } 0 \le x \le 1 \\ u(x,1) = \sin \frac{\pi}{2} x \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = \sin \frac{\pi}{2} y \text{ for } 0 \le y \le 1 \end{cases} \qquad \text{(b)} \quad \begin{cases} \Delta u = (x^2 + y^2)u \\ u(x,0) = 1 \text{ for } 0 \le x \le 1 \\ u(x,1) = e^x \text{ for } 0 \le x \le 1 \\ u(0,y) = 1 \text{ for } 0 \le y \le 1 \\ u(1,y) = e^y \text{ for } 0 \le y \le 1 \end{cases}$$

6. Prove that the functions (a) $u(x, y) = e^{x+2y}$, (b) $u(x, y) = y/x$ are solutions of the specified elliptic equation with the given Dirichlet boundary conditions:

(a)
$$\begin{cases} \Delta u = 5u \\ u(x,0) = e^x \text{ for } 0 \le x \le 1 \\ u(x,1) = e^{x+2} \text{ for } 0 \le x \le 1 \\ u(0,y) = e^{2y} \text{ for } 0 \le y \le 1 \\ u(1,y) = e^{2y+1} \text{ for } 0 \le y \le 1 \end{cases} \qquad \text{(b)} \quad \begin{cases} \Delta u = \dfrac{2u}{x^2} \\ u(x,0) = 0 \text{ for } 1 \le x \le 2 \\ u(x,1) = 1/x \text{ for } 1 \le x \le 2 \\ u(1,y) = y \text{ for } 0 \le y \le 1 \\ u(2,y) = y/2 \text{ for } 0 \le y \le 1 \end{cases}$$

7. Prove that the functions (a) $u(x, y) = x^2 + y^2$, (b) $u(x, y) = y^2/x$ are solutions of the specified elliptic equation with the given Dirichlet boundary conditions:

(a)
$$\begin{cases} \Delta u + \dfrac{u}{x^2+y^2} = 5 \\ u(x,1) = x^2 + 1 \text{ for } 1 \le x \le 2 \\ u(x,2) = x^2 + 4 \text{ for } 1 \le x \le 2 \\ u(1,y) = y^2 + 1 \text{ for } 1 \le y \le 2 \\ u(2,y) = y^2 + 4 \text{ for } 1 \le y \le 2 \end{cases} \qquad \text{(b)} \quad \begin{cases} \Delta u - \dfrac{2u}{x^2} = \dfrac{2}{x} \\ u(x,0) = 0 \text{ for } 1 \le x \le 2 \\ u(x,2) = 4/x \text{ for } 1 \le x \le 2 \\ u(1,y) = y^2 \text{ for } 0 \le y \le 2 \\ u(2,y) = y^2/2 \text{ for } 0 \le y \le 2 \end{cases}$$

8. Show that the barycenter of a triangle with vertices $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ is $\overline{x} = (x_1 + x_2 + x_3)/3$, $\overline{y} = (y_1 + y_2 + y_3)/3$.

9. Prove Lemma 8.9.

10. Prove Lemma 8.10.

11. Derive the barycenter coordinates of (8.55).

12. Derive the matrix entries in (8.57).

13. Show that the Laplace equation $\Delta T = 0$ on the rectangle $[0, L] \times [0, H]$ with Dirichlet boundary conditions $T = T_0$ on the three sides $x = 0, x = L$, and $y = 0$, and $T = T_1$ on the side $y = H$ has solution

$$T(x,y) = T_0 + \sum_{k=0}^{\infty} C_k \sin \frac{(2k+1)\pi x}{L} \sinh \frac{(2k+1)\pi y}{L}$$

where

$$C_k = \frac{4(T_1 - T_0)}{(2k+1)\pi \sinh \frac{(2k+1)\pi H}{L}}.$$

8.3 Computer Problems

1. Solve the Laplace equation problems in Exercise 3 on $0 \le x \le 1, 0 \le y \le 1$ by the Finite Difference Method with $h = k = 0.1$. Use MATLAB's `mesh` command to plot the solution.

2. Solve the Poisson equation problems in Exercise 4 on $0 \le x \le 1, 0 \le y \le 1$ by the Finite Difference Method with $h = k = 0.1$. Plot the solution.

3. Use the Finite Difference Method with $h = k = 0.1$ to approximate the electrostatic potential on the square $0 \le x, y \le 1$ from the Laplace equation with the specified boundary conditions. Plot the solution.

(a)
$$\begin{cases} u(x,0) = 0 \text{ for } 0 \le x \le 1 \\ u(x,1) = \sin \pi x \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = 0 \text{ for } 0 \le y \le 1 \end{cases} \qquad \text{(b)} \quad \begin{cases} u(x,0) = \sin \frac{\pi}{2} x \text{ for } 0 \le x \le 1 \\ u(x,1) = \cos \frac{\pi}{2} x \text{ for } 0 \le x \le 1 \\ u(0,y) = \sin \frac{\pi}{2} y \text{ for } 0 \le y \le 1 \\ u(1,y) = \cos \frac{\pi}{2} y \text{ for } 0 \le y \le 1 \end{cases}$$

4. Use the Finite Difference Method with $h = k = 0.1$ to approximate the electrostatic potential on the square $0 \le x, y \le 1$ from the Laplace equation with the specified boundary conditions. Plot the solution.

(a)
$$\begin{cases} u(x,0) = 0 \text{ for } 0 \le x \le 1 \\ u(x,1) = x^3 \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = y^2 \text{ for } 0 \le y \le 1 \end{cases}$$

(b)
$$\begin{cases} u(x,0) = 0 \text{ for } 0 \le x \le 1 \\ u(x,1) = x \sin \frac{\pi}{2} x \text{ for } 0 \le x \le 1 \\ u(0,y) = 0 \text{ for } 0 \le y \le 1 \\ u(1,y) = y \text{ for } 0 \le y \le 1 \end{cases}$$

5. Hydrostatic pressure can be expressed as the hydraulic head, defined as the equivalent height u of a column of water exerting that pressure. In an underground reservoir, steady-state groundwater flow satisfies the Laplace equation $\Delta u = 0$. Assume that the reservoir has dimensions 2 km×1 km, and water table heights

$$\begin{cases} u(x,0) = 0.01 \text{ for } 0 \le x \le 2 \\ u(x,1) = 0.01 + 0.003x \text{ for } 0 \le x \le 2 \\ u(0,y) = 0.01 \text{ for } 0 \le y \le 1 \\ u(1,y) = 0.01 + 0.006y^2 \text{ for } 0 \le y \le 1 \end{cases}$$

on the reservoir boundary, in kilometers. Compute the head $u(1, 1/2)$ at the center of the reservoir.

6. The steady-state temperature u on a heated copper plate satisfies the Poisson equation

$$\Delta u = -\frac{D(x,y)}{K},$$

where $D(x, y)$ is the power density at (x, y) and K is the thermal conductivity. Assume that the plate is the shape of the rectangle $[0, 4] \times [0, 2]$ cm whose boundary is kept at a constant 30°C, and that power is generated at the constant rate $D(x, y) = 5$ watts/cm^3. The thermal conductivity of copper is $K = 3.85$ watts/cm°C. (a) Plot the temperature distribution on the plate. (b) Find the temperature at the center point $(x, y) = (2, 1)$.

7. For the Laplace equations in Exercise 3, make a table of the finite difference approximation and error at $(x, y) = (1/4, 3/4)$ as a function of step sizes $h = k = 2^{-p}$ for $p = 2, \ldots, 5$.

8. For the Poisson equations in Exercise 4, make a table of the finite difference approximation and error at $(x, y) = (1/4, 3/4)$ as a function of step sizes $h = k = 2^{-p}$ for $p = 2, \ldots, 5$.

9. Solve the Laplace equation problems in Exercise 3 on $0 \le x \le 1, 0 \le y \le 1$ by the Finite Element Method with $h = k = 0.1$. Use MATLAB's `mesh` command to plot the solution.

10. Solve the Poisson equation problems in Exercise 4 on $0 \le x \le 1, 0 \le y \le 1$ by the Finite Element Method with $h = k = 0.1$. Plot the solution.

11. Solve the elliptic partial differential equations in Exercise 5 by the Finite Element Method with $h = k = 0.1$. Plot the solution.

12. Solve the elliptic partial differential equations in Exercise 6 by the Finite Element Method with $h = k = 1/16$. Plot the solution.

13. Solve the elliptic partial differential equations in Exercise 7 by the Finite Element Method with $h = k = 1/16$. Plot the solution.

14. Solve the elliptic partial differential equations with Dirichlet boundary conditions by the Finite Element Method with $h = k = 0.1$. Plot the solution.

(a)
$$\begin{cases} \Delta u + \sin \pi x y = (x^2 + y^2)u \\ u(x, 0) = 0 \text{ for } 0 \le x \le 1 \\ u(x, 1) = 0 \text{ for } 0 \le x \le 1 \\ u(0, y) = 0 \text{ for } 0 \le y \le 1 \\ u(1, y) = 0 \text{ for } 0 \le y \le 1 \end{cases}$$

(b)
$$\begin{cases} \Delta u + (\sin \pi x y)u = e^{2xy} \\ u(x, 0) = 0 \text{ for } 0 \le x \le 1 \\ u(x, 1) = 0 \text{ for } 0 \le x \le 1 \\ u(0, y) = 0 \text{ for } 0 \le y \le 1 \\ u(1, y) = 0 \text{ for } 0 \le y \le 1 \end{cases}$$

15. For the elliptic equations in Exercise 5, make a table of the Finite Element Method approximation and error at $(x, y) = (1/4, 3/4)$ as a function of step sizes $h = k = 2^{-p}$ for $p = 2, \ldots, 5$.

16. For the elliptic equations in Exercise 6, make a log–log plot of the maximum error of the Finite Element Method as a function of step sizes $h = k = 2^{-p}$ for $p = 2, \ldots, 6$.

17. For the elliptic equations in Exercise 7, make a log–log plot of the maximum error of the Finite Element Method as a function of step sizes $h = k = 2^{-p}$ for $p = 2, \ldots, 6$.

18. Solve the Laplace equation with Dirichlet boundary conditions from Exercise 13 on $[0, 1] \times [0, 1]$ with $T_0 = 0$ and $T_1 = 10$ using (a) a finite difference approximation and (b) the Finite Element Method. Make log–log plots of the error at particular locations in the rectangle as a function of step sizes $h = k = 2^{-p}$ for p as large as possible. Explain any simplifications you are making to evaluate the correct solution at those locations.

8.4 NONLINEAR PARTIAL DIFFERENTIAL EQUATIONS

In the previous sections of this chapter, finite difference and finite element methods have been analyzed and applied to linear PDEs. For the nonlinear case, an extra wrinkle is necessary to make our previous methods appropriate.

To make matters concrete, we will focus on the implicit Backward Difference Method of Section 8.1 and its application to nonlinear diffusion equations. Similar changes can be applied to any of the methods we have studied to make them available for use on nonlinear equations.

8.4.1 Implicit Newton solver

We illustrate the approach with a typical nonlinear example

$$u_t + uu_x = Du_{xx}, \tag{8.60}$$

known as **Burgers' equation**. The equation is nonlinear due to the product term uu_x. This elliptic equation, named after J.M. Burgers (1895–1981), is a simplified model of fluid flow. When the diffusion coefficient $D = 0$, it is called the inviscid Burgers' equation. Setting $D > 0$ corresponds to adding viscosity to the model.

This diffusion equation will be discretized in the same way as the heat equation in Section 8.1. Consider the grid of points as shown in Figure 8.1. We will denote the approximate solution at (x_i, t_j) by w_{ij}. Let M and N be the total number of steps in the x and t directions, and let $h = (b - a)/M$ and $k = T/N$ be the step sizes in the x and t directions. Applying backward differences to u_t and central differences to the other terms yields

$$\frac{w_{ij} - w_{i,j-1}}{k} + w_{ij}\left(\frac{w_{i+1,j} - w_{i-1,j}}{2h}\right) = \frac{D}{h^2}(w_{i+1,j} - 2w_{ij} + w_{i-1,j}),$$

or

$$w_{ij} + \frac{k}{2h}w_{ij}(w_{i+1,j} - w_{i-1,j}) - \sigma(w_{i+1,j} - 2w_{ij} + w_{i-1,j}) - w_{i,j-1} = 0 \quad (8.61)$$

where we have set $\sigma = Dk/h^2$. Note that due to the quadratic terms in the w variables, we cannot directly solve for $w_{i+1,j}, w_{ij}, w_{i-1,j}$, explicitly or implicitly. Therefore, we call on Multivariate Newton's Method from Chapter 2 to do the solving.

To clarify our implementation, denote the unknowns in (8.61) by $z_i = w_{ij}$. At time step j, we are trying to solve the equations

$$F_i(z_1, \ldots, z_m) = z_i + \frac{k}{2h}z_i(z_{i+1} - z_{i-1}) - \sigma(z_{i+1} - 2z_i + z_{i-1}) - w_{i,j-1} = 0 \quad (8.62)$$

for the m unknowns $z_1, \ldots, z_m$. Note that the last term $w_{i,j-1}$ is known from the previous time step, and is treated as a known quantity.

The first and last equations will be replaced by appropriate boundary conditions. For example, in the case of Burgers' equation with Dirichlet boundary conditions

$$\begin{cases} u_t + uu_x = Du_{xx} \\ u(x,0) = f(x) \text{ for } x_l \le x \le x_r \\ u(x_l,t) = l(t) \text{ for all } t \ge 0 \\ u(x_r,t) = r(t) \text{ for all } t \ge 0, \end{cases} \quad (8.63)$$

we will add the equations

$$\begin{aligned} F_1(z_1, \ldots, z_m) &= z_1 - l(t_j) = 0 \\ F_m(z_1, \ldots, z_m) &= z_m - r(t_j) = 0. \end{aligned} \quad (8.64)$$

Now there are m nonlinear algebraic equations in m unknowns.

To apply Multivariate Newton's Method, we must compute the Jacobian $DF(\vec{z}) = \partial\vec{F}/\partial\vec{z}$, which according to (8.62) and (8.64) will have the tridiagonal form

$$\begin{bmatrix} 1 & 0 & \cdots & & & \\ -\sigma - \frac{kz_2}{2h} & 1 + 2\sigma + \frac{k(z_3 - z_1)}{2h} & -\sigma + \frac{kz_2}{2h} & & & \\ & -\sigma - \frac{kz_3}{2h} & 1 + 2\sigma + \frac{k(z_4 - z_2)}{2h} & -\sigma + \frac{kz_3}{2h} & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\sigma - \frac{kz_{m-1}}{2h} & 1 + 2\sigma + \frac{k(z_m - z_{m-2})}{2h} & -\sigma + \frac{kz_{m-1}}{2h} \\ & & & \cdots & 0 & 1 \end{bmatrix}.$$

The top and bottom rows of DF will in general depend on boundary conditions. Once DF has been constructed, we solve for the $z_i = w_{ij}$ by the Multivariate Newton iteration

$$\vec{z}^{K+1} = \vec{z}^K - DF(\vec{z}^K)^{-1}F(\vec{z}^K). \quad (8.65)$$

▶ **EXAMPLE 8.12** Use the Backward Difference Equation with Newton iteration to solve Burgers' equation

$$\begin{cases} u_t + uu_x = Du_{xx} \\ u(x,0) = \dfrac{2D\beta\pi \sin \pi x}{\alpha + \beta \cos \pi x} \text{ for } 0 \le x \le 1 \\ u(0,t) = 0 \text{ for all } t \ge 0 \\ u(1,t) = 0 \text{ for all } t \ge 0. \end{cases} \tag{8.66}$$

MATLAB code for the Dirichlet boundary condition version of our Newton solver follows, where we have set $\alpha = 5, \beta = 4$. The program uses three Newton iterations for each time step. For typical problems, this will be sufficient, but more may be needed for difficult cases. Note that Gaussian elimination or equivalent is carried out in the Newton iteration; as usual, no explicit matrix inversion is needed.

```
% Program 8.7  Implicit Newton solver for Burgers equation
% input: space interval [xl,xr], time interval [tb,te],
%        number of space steps M, number of time steps N
% output: solution w
% Example usage: w=burgers(0,1,0,2,20,40)
function w=burgers(xl,xr,tb,te,M,N)
alf=5;bet=4;D=.05;
f=@(x) 2*D*bet*pi*sin(pi*x)./(alf+bet*cos(pi*x));
l=@(t) 0*t;
r=@(t) 0*t;
h=(xr-xl)/M; k=(te-tb)/N; m=M+1; n=N;
sigma=D*k/(h*h);
w(:,1)=f(xl+(0:M)*h)';              % initial conditions
w1=w;
for j=1:n
  for it=1:3                        % Newton iteration
    DF1=zeros(m,m);DF2=zeros(m,m);
    DF1=diag(1+2*sigma*ones(m,1))+diag(-sigma*ones(m-1,1),1);
    DF1=DF1+diag(-sigma*ones(m-1,1),-1);
    DF2=diag([0;k*w1(3:m)/(2*h);0])-diag([0;k*w1(1:(m-2))/(2*h);0]);
    DF2=DF2+diag([0;k*w1(2:m-1)/(2*h)],1)...
        -diag([k*w1(2:m-1)/(2*h);0],-1);
    DF=DF1+DF2;
    F=-w(:,j)+(DF1+DF2/2)*w1;     % Using Lemma 8.11
    DF(1,:)=[1 zeros(1,m-1)];     % Dirichlet conditions for DF
    DF(m,:)=[zeros(1,m-1) 1];
    F(1)=w1(1)-l(j);F(m)=w1(m)-r(j);  % Dirichlet conditions for F
    w1=w1-DF\F;
  end
  w(:,j+1)=w1;
end
x=xl+(0:M)*h;t=tb+(0:n)*k;
mesh(x,t,w')                        % 3-D plot of solution w
```

The code is a straightforward implementation of the Newton iteration (8.65), along with a convenient fact about homogeneous polynomials. Consider, for example, the polynomial $P(x_1, x_2, x_3) = x_1x_2x_3^2 + x_1^4$, which is called homogeneous of degree 4, since it consists entirely of degree 4 terms in x_1, x_2, x_3. The partial derivatives of P with respect to the three variables are contained in the gradient

$$\nabla P = (x_2x_3^2 + 4x_1^3, x_1x_3^2, 2x_1x_2x_3).$$

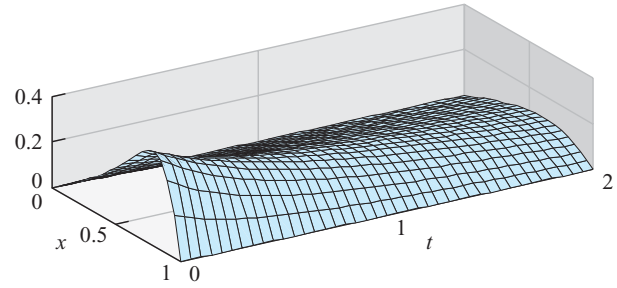

Figure 8.19 Approximate solution to Burgers' equation (8.66). Homogeneous Dirichlet boundary conditions are assumed, with step sizes $h = k = 0.05$.

The remarkable fact is that we can recover P by multiplying the gradient by the vector of variables, with a extra multiple of 4:

$$\nabla P \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = (x_2 x_3^2 + 4x_1^3)x_1 + x_1 x_3^2 x_2 + 2x_1 x_2 x_3 x_3 = 4x_1 x_2 x_3^2 + 4x_1^4 = 4P.$$

In general, define the polynomial $P(x_1, \ldots, x_m)$ to be **homogeneous** of degree d if

$$P(cx_1, \ldots, cx_m) = c^d P(x_1, \ldots, x_m) \tag{8.67}$$

for all c.

LEMMA 8.11 Let $P(x_1, \ldots, x_m)$ be a homogeneous polynomial of degree d. Then

$$\nabla P \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} = dP.$$

■

Proof. Differentiating (8.67) with respect to c yields

$$x_1 P_{x_1}(cx_1, \ldots, cx_m) + \ldots + x_m P_{x_m}(cx_1, \ldots, cx_m) = dc^{d-1} P(x_1, \ldots, x_m)$$

using the multivariable chain rule. Evaluating at $c = 1$ results in the desired conclusion. ❐

Using this fact allows us to write code very compactly for partial differential equations with polynomial terms, as long as we group terms of the same degree together. Note how the matrix `DF1` in Program 8.7 collects derivatives of degree 1 terms of `F`; `DF2` collects derivatives of degree 2 terms. Then we can define the Jacobian matrix `DF` as the sum of derivatives of degree 1 and 2 terms, and essentially for free, define the function `F` as the sum of degree 0, 1, and 2 terms. Lemma 8.11 is used to identify the degree d terms of `F` as gradient times variables, divided by d. The added convenience of this simplification will be even more welcome when we proceed to more difficult problems.

For certain boundary conditions, an explicit solution for Burgers' equation is known. The solution to the Dirichlet problem (8.66) is

$$u(x,t) = \frac{2D\beta\pi e^{-D\pi^2 t} \sin \pi x}{\alpha + \beta e^{-D\pi^2 t} \cos \pi x}. \tag{8.68}$$

We can use the exact solution to measure the accuracy of our approximation method, as a function of the step sizes h and k. Using the parameters $\alpha = 5$, $\beta = 4$, and the diffusion coefficient $D = 0.05$, we find the errors at $x = 1/2$ after one time unit are as follows:

h	k	$u(0.5, 1)$	$w(0.5, 1)$	error
0.01	0.04	0.153435	0.154624	0.001189
0.01	0.02	0.153435	0.154044	0.000609
0.01	0.01	0.153435	0.153749	0.000314

We see the roughly first-order decrease in error as a function of time step size k, as expected with the implicit Backward Difference Method. ◀

Another interesting category of nonlinear PDEs is comprised of **reaction-diffusion equations**. A fundamental example of a nonlinear reaction-diffusion equation is due to the evolutionary biologist and geneticist R.A. Fisher (1890–1962), a successor of Darwin who helped create the foundations of modern statistics. The equation was originally derived to model how genes propagate. The general form of **Fisher's equation** is

$$u_t = Du_{xx} + f(u), \tag{8.69}$$

where $f(u)$ is a polynomial in u. The reaction part of the equation is the function f; the diffusion part is Du_{xx}. If homogeneous Neumann boundary conditions are used, the constant, or equilibrium state $u(x, t) \equiv C$ is a solution whenever $f(C) = 0$. The equilibrium state turns out to be stable if $f'(C) < 0$, meaning that nearby solutions tend toward the equilibrium state.

► EXAMPLE 8.13 Use the Backward Difference Equation with Newton iteration to solve Fisher's equation with homogeneous Neumann boundary conditions

$$\begin{cases} u_t = Du_{xx} + u(1-u) \\ u(x, 0) = \sin \pi x \text{ for } 0 \le x \le 1 \\ u_x(0, t) = 0 \text{ for all } t \ge 0 \\ u_x(1, t) = 0 \text{ for all } t \ge 0. \end{cases} \tag{8.70}$$

Note that $f(u) = u(1-u)$, implying that $f'(u) = 1 - 2u$. The equilibrium $u = 0$ satisfies $f'(0) = 1$, and the other equilibrium solution $u = 1$ satisfies $f'(1) = -1$. Therefore, solutions are likely to tend toward the equilibrium $u = 1$.

The discretization retraces the derivation that was carried out for Burgers' equation:

$$\frac{w_{ij} - w_{i,j-1}}{k} = \frac{D}{h^2}(w_{i+1,j} - 2w_{ij} + w_{i-1,j}) + w_{ij}(1 - w_{ij}),$$

or

$$(1 + 2\sigma - k(1 - w_{ij}))w_{ij} - \sigma(w_{i+1,j} + w_{i-1,j}) - w_{i,j-1} = 0. \tag{8.71}$$

This results in the nonlinear equations

$$F_i(z_1, \ldots, z_m) = (1 + 2\sigma - k(1 - z_i))z_i - \sigma(z_{i+1} + z_{i-1}) - w_{i,j-1} = 0 \tag{8.72}$$

to solve for the $z_i = w_{ij}$ at the jth time step. The first and last equations will establish the Neumann boundary conditions:

$$\begin{aligned} F_1(z_1, \ldots, z_m) &= (-3z_0 + 4z_1 - z_2)/(2h) = 0 \\ F_m(z_1, \ldots, z_m) &= (-z_{m-2} + 4z_{m-1} - 3z_m)/(-2h) = 0 \end{aligned}$$

The Jacobian DF has the form

$$\begin{bmatrix} -3 & 4 & -1 & & & \\ -\sigma & 1+2\sigma-k+2kz_2 & -\sigma & & & \\ & -\sigma & 1+2\sigma-k+2kz_3 & -\sigma & & \\ & & \ddots & \ddots & \ddots & \\ & & & -\sigma & 1+2\sigma-k+2kz_{m-1} & -\sigma \\ & & & -1 & 4 & -3 \end{bmatrix}.$$

After altering the function F and Jacobian DF, the Newton iteration implemented in Program 8.7 can be used to solve Fisher's equation. Lemma 8.11 can be used to separate the degree 1 and 2 parts of DF. Neumann boundary conditions are also applied, as shown in the code fragment below:

```
DF1=diag(1-k+2*sigma*ones(m,1))+diag(-sigma*ones(m-1,1),1);
DF1=DF1+diag(-sigma*ones(m-1,1),-1);
DF2=diag(2*k*w1);
DF=DF1+DF2;
F=-w(:,j)+(DF1+DF2/2)*w1;
DF(1,:)=[-3 4 -1 zeros(1,m-3)];F(1)=DF(1,:)*w1;
DF(m,:)=[zeros(1,m-3) -1 4 -3];F(m)=DF(m,:)*w1;
```

Figure 8.20 shows approximate solutions of Fisher's equation with $D = 1$ that demonstrate the tendency to relax to the attracting equilibrium $u(x,t) \equiv 1$. Of course, $u(x,t) \equiv 0$ is also a solution of (8.69) with $f(u) = u(1-u)$, and will be found by the initial data $u(x,0) = 0$. Almost any other initial data, however, will eventually approach $u = 1$ as t increases. ◀

While Example 8.13 covers the original equation considered by Fisher, there are many generalized versions for other choices of the polynomial $f(u)$. See the Computer Problems for more explorations into this reaction-diffusion equation. Next, we will investigate a higher-dimensional version of Fisher's equation.

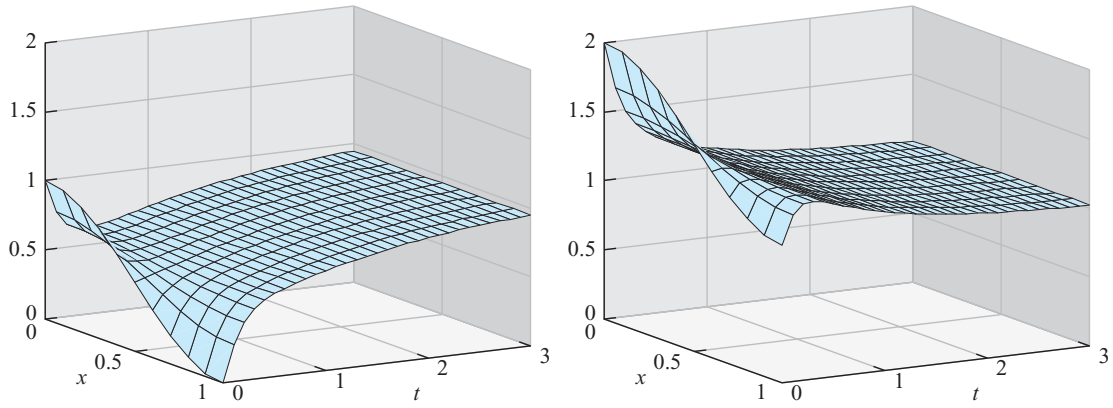

Figure 8.20 Two solutions to Fisher's equation. Both solutions tend toward the equilibrium solution $u(x,t) = 1$ as t increases. (a) Initial condition $u(x,0) = 0.5 + 0.5\cos\pi x$. (b) Initial condition $u(x,0) = 1.5 + 0.5\cos\pi x$. Homogeneous Neumann boundary conditions are assumed, with step sizes $h = k = 0.1$.

8.4.2 Nonlinear equations in two space dimensions

Solving partial differential equations with two-dimensional domains requires us to combine techniques from previous sections. The implicit Backward Difference Method with Newton iteration will handle the nonlinearity, and we will need to apply the accordion-style coordinates of Table 8.1 to do the bookkeeping for the two-dimensional domain.

We begin by extending Fisher's equation from one space dimension to two.

► **EXAMPLE 8.14** Apply the Backward Difference Method with Newton's iteration to Fisher's equation on the unit square $[0, 1] \times [0, 1]$:

$$\begin{cases} u_t = D\Delta u + u(1-u) \\ u(x, y, 0) = 2 + \cos \pi x \cos \pi y \text{ for } 0 \le x, y \le 1 \\ u_{\vec{n}}(x, y, t) = 0 \text{ on rectangle boundary, for all } t \ge 0. \end{cases} \tag{8.73}$$

Here D is the diffusion coefficient, and $u_{\vec{n}}$ denotes the directional derivative in the outward normal direction. We are assuming Neumann, or no-flux, boundary conditions on the rectangle boundary.

In this section, the two discretization subscripts will represent the two space coordinates x and y, and we will use superscripts to denote time steps. Assuming M steps in the x direction and N steps in the y direction, we will define step sizes $h = (x_r - x_l)/M$ and $k = (y_t - y_b)/N$. The discretized equations at nonboundary grid points, for $1 < i < m = M + 1, 1 < j < n = N + 1$, are

$$\begin{aligned} \frac{w_{ij}^t - w_{ij}^{t-\Delta t}}{\Delta t} = {} & \frac{D}{h^2}(w_{i+1,j}^t - 2w_{ij}^t + w_{i-1,j}^t) + \frac{D}{k^2}(w_{i,j+1}^t - 2w_{ij}^t + w_{i,j-1}^t) \\ & + w_{ij}^t(1 - w_{ij}^t), \end{aligned} \tag{8.74}$$

which can be rearranged to the form $F_{ij}(w^t) = 0$, or

$$\begin{aligned} & \left(\frac{1}{\Delta t} + \frac{2D}{h^2} + \frac{2D}{k^2} - 1\right) w_{ij}^t - \frac{D}{h^2} w_{i+1.j}^t - \frac{D}{h^2} w_{i-1.j}^t - \frac{D}{k^2} w_{i.j+1}^t - \frac{D}{k^2} w_{i.j-1}^t \\ & + (w_{ij}^t)^2 - \frac{w_{ij}^{t-\Delta t}}{\Delta t} = 0 \end{aligned} \tag{8.75}$$

We need to solve the F_{ij} equations implicitly. The equations are nonlinear, so Newton's method will be used as it was for the one-dimensional version of Fisher's equation. Since the domain is now two-dimensional, we need to recall the alternative coordinate system (8.39)

$$v_{i+(j-1)m} = w_{ij},$$

illustrated in Table 8.1. There will be mn equations F_{ij}, and in the v coordinates, (8.75) represents the equation numbered $i + (j-1)m$. The Jacobian matrix DF will have size $mn \times mn$. Using Table 8.1 to translate to the v coordinates, we get the Jacobian matrix entries

$$\begin{aligned} DF_{i+(j-1)m,i+(j-1)m} &= \left(\frac{1}{\Delta t} + \frac{2D}{h^2} + \frac{2D}{k^2} - 1\right) + 2w_{ij} \\ DF_{i+(j-1)m,i+1+(j-1)m} &= -\frac{D}{h^2} \\ DF_{i+(j-1)m,i-1+(j-1)m} &= -\frac{D}{h^2} \end{aligned}$$

$$DF_{i+(j-1)m,i+jm} = -\frac{D}{k^2}$$

$$DF_{i+(j-1)m,i+(j-2)m} = -\frac{D}{k^2}$$

for the interior points of the grid. The outside points of the grid are governed by the homogenous Neumann boundary conditions

Bottom	$(3w_{ij} - 4w_{i,j+1} + w_{i,j+2})/(2k) = 0$	for $j = 1, 1 \le i \le m$
Top side	$(3w_{ij} - 4w_{i,j-1} + w_{i,j-2})/(2k) = 0$	for $j = n, 1 \le i \le m$
Left side	$(3w_{ij} - 4w_{i+1,j} + w_{i+2,j})/(2h) = 0$	for $i = 1, 1 < j < n$
Right side	$(3w_{ij} - 4w_{i-1,j} + w_{i-2,j})/(2h) = 0$	for $i = m, 1 < j < n$

The Neumann conditions translate via Table 8.1 to

Bottom $DF_{i+(j-1)m,i+(j-1)m} = 3,\ DF_{i+(j-1)m,i+jm} = -4,\ DF_{i+(j-1)m,i+(j+1)m} = 1,$
$b_{i+(j-1)m} = 0$ for $j = 1, 1 \le i \le m$

Top $DF_{i+(j-1)m,i+(j-1)m} = 3,\ DF_{i+(j-1)m,i+(j-2)m} = -4,\ DF_{i+(j-1)m,i+(j-3)m} = 1,$
$b_{i+(j-1)m} = 0$ for $j = n, 1 \le i \le m$

Left $DF_{i+(j-1)m,i+(j-1)m} = 3,\ DF_{i+(j-1)m,i+1+(j-1)m} = -4,$
$DF_{i+(j-1)m,i+2+(j-1)m} = 1,$
$b_{i+(j-1)m} = 0$ for $i = 1, 1 < j < n$

Right $DF_{i+(j-1)m,i+(j-1)m} = 3,\ DF_{i+(j-1)m,i-1+(j-1)m} = -4,$
$DF_{i+(j-1)m,i-2+(j-1)m} = 1,$
$b_{i+(j-1)m} = 0$ for $i = m, 1 < j < n$

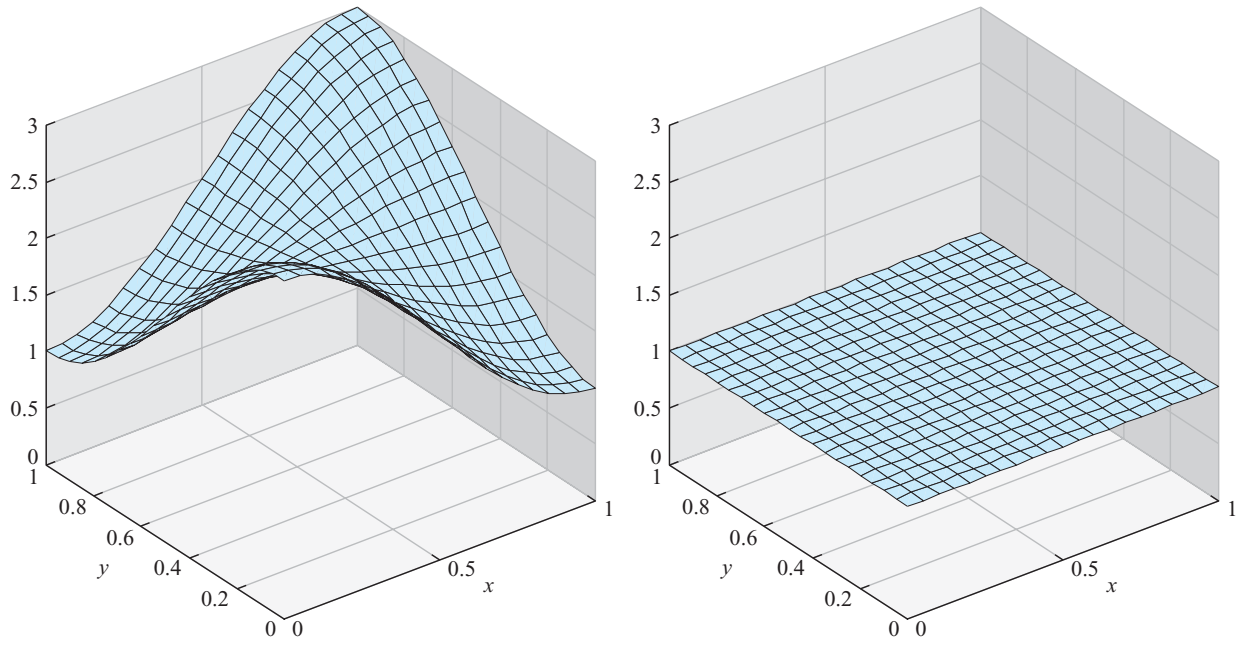

Figure 8.21 Fisher's equation with Neumann boundary conditions on a two-dimensional domain. The solution tends toward the equilibrium solution $u(x, y, t) = 1$ as t increases. (a) The initial condition $u(x, y, 0) = 2 + \cos \pi x \cos \pi y$. (b) Approximate solution after 5 time units. Step sizes $h = k = \Delta t = 0.05$.

The Newton iteration is carried out in the following program. Note that Lemma 8.11 has been used to divide the contributions to DF into degree 1 and degree 2 terms.

```
% Program 8.8 Backward difference method with Newton iteration
%             for Fisher's equation with two-dim domain
% input: space region [xl xr]x[yb yt], time interval [tb te],
% M,N space steps in x and y directions, tsteps time steps
% output: solution mesh [x,y,w]
% Example usage: [x,y,w]=fisher2d(0,1,0,1,0,5,20,20,100);
function [x,y,w]=fisher2d(xl,xr,yb,yt,tb,te,M,N,tsteps)
f=@(x,y) 2+cos(pi*x).*cos(pi*y)
delt=(te-tb)/tsteps;
D=1;
m=M+1;n=N+1;mn=m*n;
h=(xr-xl)/M;k=(yt-yb)/N;
x=linspace(xl,xr,m);y=linspace(yb,yt,n);
for i=1:m          %Define initial u
  for j=1:n
      w(i,j)=f(x(i),y(j));
  end
end
for tstep=1:tsteps
 v=[reshape(w,mn,1)];
 wold=w;
 for it=1:3
   b=zeros(mn,1);DF1=zeros(mn,mn);DF2=zeros(mn,mn);
   for i=2:m-1
     for j=2:n-1
       DF1(i+(j-1)*m,i-1+(j-1)*m)=-D/h^2;
       DF1(i+(j-1)*m,i+1+(j-1)*m)=-D/h^2;
       DF1(i+(j-1)*m,i+(j-1)*m)= 2*D/h^2+2*D/k^2-1+1/(1*delt);
       DF1(i+(j-1)*m,i+(j-2)*m)=-D/k^2;DF1(i+(j-1)*m,i+j*m)=-D/k^2;
       b(i+(j-1)*m)=-wold(i,j)/(1*delt);
       DF2(i+(j-1)*m,i+(j-1)*m)=2*w(i,j);
     end
   end
   for i=1:m     % bottom and top
     j=1; DF1(i+(j-1)*m,i+(j-1)*m)=3;
     DF1(i+(j-1)*m,i+j*m)=-4;DF1(i+(j-1)*m,i+(j+1)*m)=1;
     j=n; DF1(i+(j-1)*m,i+(j-1)*m)=3;
     DF1(i+(j-1)*m,i+(j-2)*m)=-4;DF1(i+(j-1)*m,i+(j-3)*m)=1;
   end
   for j=2:n-1   % left and right
     i=1; DF1(i+(j-1)*m,i+(j-1)*m)=3;
     DF1(i+(j-1)*m,i+1+(j-1)*m)=-4;DF1(i+(j-1)*m,i+2+(j-1)*m)=1;
     i=m; DF1(i+(j-1)*m,i+(j-1)*m)=3;
     DF1(i+(j-1)*m,i-1+(j-1)*m)=-4;DF1(i+(j-1)*m,i-2+(j-1)*m)=1;
   end
   DF=DF1+DF2;
   F=(DF1+DF2/2)*v+b;
   v=v-DF\F;
   w=reshape(v(1:mn),m,n);
 end
 mesh(x,y,w');axis([xl xr yb yt tb te]);
 xlabel('x');ylabel('y');drawnow
end
```

The dynamical behavior of the two-dimensional Fisher's equation is similar to that of the one-dimensional version in Figure 8.20, where we saw convergence to the stable equilibrium solution at $u(x,t) = 1$. Figure 8.21(a) shows the initial data $f(x, y) = 2 + \cos \pi x \cos \pi y$. The solution after $t = 5$ time units is shown in Figure 8.21(b). The solution relaxes quickly toward the stable equilibrium at $u(x, y, t) = 1$. ◀

The mathematician Alan Turing (1912–1954), in a landmark paper (Turing [1952]), proposed a possible explanation for many shapes and structures found in biology. Certain reaction-diffusion equations that model chemical concentrations gave rise to interesting spatial patterns, including stripes and hexagonal shapes. These were seen as a stunning example of emergent order in nature, and are now known as **Turing patterns**.

Turing found that just by adding a diffusive term to a model of a stable chemical reaction, he could cause stable, spatially constant equilibriums, such as the one in Figure 8.21(b), to become unstable. This so-called **Turing instability** causes a transition in which patterns evolve into a new, spatially varying steady-state solution. Of course, this is the opposite of the effect of diffusion we have seen so far, of averaging or smoothing initial conditions over time.

An interesting example of a Turing instability is found in the **Brusselator model**, proposed by the Belgian chemist I. Prigogine in the late 1960's. The model consists of two coupled PDEs, each representing one species of a two-species chemical reaction.

▶ EXAMPLE 8.15

Apply the Backward Difference Method with Newton's iteration to the Brusselator equation with homogeneous Neumann boundary conditions on the square $[0, 40] \times [0, 40]$:

$$\begin{cases} p_t = D_p \Delta p + p^2 q + C - (K+1)p \\ q_t = D_q \Delta q - p^2 q + Kp \\ \\ p(x, y, 0) = C + 0.1 \quad \text{for } 0 \le x, y \le 40 \\ q(x, y, 0) = K/C + 0.2 \quad \text{for } 0 \le x, y \le 40 \\ u_{\vec{n}}(x, y, t) = 0 \text{ on rectangle boundary, for all } t \ge 0. \end{cases} \tag{8.76}$$

The system of two coupled equations has variables p, q, two diffusion coefficients $D_p, D_q > 0$, and two other parameters $C, K > 0$. According to Exercise 5, the Brusselator has an equilibrium solution at $p \equiv C, q \equiv K/C$. It is known that the equilibrium is stable for small values of the parameter K, and that a Turing instability is encountered when

$$K > \left(1 + C\sqrt{\frac{D_p}{D_q}}\right)^2. \tag{8.77}$$

The discretized equations at the interior grid points, for $1 < i < m, 1 < j < n$, are

$$\begin{aligned} &\frac{p_{ij}^t - p_{ij}^{t-\Delta t}}{\Delta t} - \frac{D_p}{h^2}(p_{i+1,j}^t - 2p_{ij}^t + p_{i-1,j}^t) - \frac{D_p}{k^2}(p_{i,j+1}^t - 2p_{ij}^t + p_{i,j-1}^t) \\ &\qquad - (p_{ij}^t)^2 q_{ij}^t - C + (K+1)p_{ij}^t = 0 \\ &\frac{q_{ij}^t - q_{ij}^{t-\Delta t}}{\Delta t} - \frac{D_q}{h^2}(q_{i+1,j}^t - 2q_{ij}^t + q_{i-1,j}^t) - \frac{D_q}{k^2}(q_{i,j+1}^t - 2q_{ij}^t + q_{i,j-1}^t) \\ &\qquad + (p_{ij}^t)^2 q_{ij}^t - Kp_{ij}^t = 0 \end{aligned}$$

This is the first example we have encountered with two coupled variables, p and q. The alternative coordinate vector v will have length $2mn$, and (8.39) will be extended to

$$\begin{aligned} v_{i+(j-1)m} &= p_{ij} \quad \text{for } 1 \le i \le m, 1 \le j \le n \\ v_{mn+i+(j-1)m} &= q_{ij} \quad \text{for } 1 \le i \le m, 1 \le j \le n. \end{aligned} \tag{8.78}$$

The Neumann boundary conditions are essentially the same as Example 8.14, now for each variable p and q. Note that there are degree 1 and degree 3 terms to differentiate for the Jacobian DF. Using Table 8.1 expanded in a straightforward way to cover two variables, and Lemma 8.11, we arrive at the following MATLAB code:

```
% Program 8.9 Backward difference method with Newton iteration
%             for the Brusselator
% input: space region [xl,xr]x[yb,yt], time interval [tb,te],
%   M,N space steps in x and y directions, tsteps time steps
% output: solution mesh [x,y,w]
% Example usage: [x,y,p,q]=brusselator(0,40,0,40,0,20,40,40,20);
function [x,y,p,q]=brusselator(xl,xr,yb,yt,tb,te,M,N,tsteps)
Dp=1;Dq=8;C=4.5;K=9;
fp=@(x,y) C+0.1;
fq=@(x,y) K/C+0.2;
delt=(te-tb)/tsteps;
m=M+1;n=N+1;mn=m*n;mn2=2*mn;
h=(xr-xl)/M;k=(yt-yb)/N;
x=linspace(xl,xr,m);y=linspace(yb,yt,n);
for i=1:m          %Define initial conditions
  for j=1:n
    p(i,j)=fp(x(i),y(j));
    q(i,j)=fq(x(i),y(j));
  end
end
for tstep=1:tsteps
  v=[reshape(p,mn,1);reshape(q,mn,1)];
  pold=p;qold=q;
  for it=1:3
    DF1=zeros(mn2,mn2);DF3=zeros(mn2,mn2);
    b=zeros(mn2,1);
    for i=2:m-1
      for j=2:n-1
        DF1(i+(j-1)*m,i-1+(j-1)*m)=-Dp/h^2;
        DF1(i+(j-1)*m,i+(j-1)*m)= Dp*(2/h^2+2/k^2)+K+1+1/(1*delt);
        DF1(i+(j-1)*m,i+1+(j-1)*m)=-Dp/h^2;
        DF1(i+(j-1)*m,i+(j-2)*m)=-Dp/k^2;
        DF1(i+(j-1)*m,i+j*m)=-Dp/k^2;
        b(i+(j-1)*m)=-pold(i,j)/(1*delt)-C;
        DF1(mn+i+(j-1)*m,mn+i-1+(j-1)*m)=-Dq/h^2;
        DF1(mn+i+(j-1)*m,mn+i+(j-1)*m)= Dq*(2/h^2+2/k^2)+1/(1*delt);
        DF1(mn+i+(j-1)*m,mn+i+1+(j-1)*m)=-Dq/h^2;
        DF1(mn+i+(j-1)*m,mn+i+(j-2)*m)=-Dq/k^2;
        DF1(mn+i+(j-1)*m,mn+i+j*m)=-Dq/k^2;
        DF1(mn+i+(j-1)*m,i+(j-1)*m)=-K;
        DF3(i+(j-1)*m,i+(j-1)*m)=-2*p(i,j)*q(i,j);
        DF3(i+(j-1)*m,mn+i+(j-1)*m)=-p(i,j)^2;
        DF3(mn+i+(j-1)*m,i+(j-1)*m)=2*p(i,j)*q(i,j);
        DF3(mn+i+(j-1)*m,mn+i+(j-1)*m)=p(i,j)^2;
        b(mn+i+(j-1)*m)=-qold(i,j)/(1*delt);
      end
    end
```

```
    for i=1:m   % bottom and top Neumann conditions
      j=1;DF1(i+(j-1)*m,i+(j-1)*m)=3;
      DF1(i+(j-1)*m,i+j*m)=-4;
      DF1(i+(j-1)*m,i+(j+1)*m)=1;
      j=n;DF1(i+(j-1)*m,i+(j-1)*m)=3;
      DF1(i+(j-1)*m,i+(j-2)*m)=-4;
      DF1(i+(j-1)*m,i+(j-3)*m)=1;
      j=1;DF1(mn+i+(j-1)*m,mn+i+(j-1)*m)=3;
      DF1(mn+i+(j-1)*m,mn+i+j*m)=-4;
      DF1(mn+i+(j-1)*m,mn+i+(j+1)*m)=1;
      j=n;DF1(mn+i+(j-1)*m,mn+i+(j-1)*m)=3;
      DF1(mn+i+(j-1)*m,mn+i+(j-2)*m)=-4;
      DF1(mn+i+(j-1)*m,mn+i+(j-3)*m)=1;
    end
    for j=2:n-1   %left and right Neumann conditions
      i=1;DF1(i+(j-1)*m,i+(j-1)*m)=3;
      DF1(i+(j-1)*m,i+1+(j-1)*m)=-4;
      DF1(i+(j-1)*m,i+2+(j-1)*m)=1;
      i=m;DF1(i+(j-1)*m,i+(j-1)*m)=3;
      DF1(i+(j-1)*m,i-1+(j-1)*m)=-4;
      DF1(i+(j-1)*m,i-2+(j-1)*m)=1;
      i=1;DF1(mn+i+(j-1)*m,mn+i+(j-1)*m)=3;
      DF1(mn+i+(j-1)*m,mn+i+1+(j-1)*m)=-4;
      DF1(mn+i+(j-1)*m,mn+i+2+(j-1)*m)=1;
      i=m;DF1(mn+i+(j-1)*m,mn+i+(j-1)*m)=3;
      DF1(mn+i+(j-1)*m,mn+i-1+(j-1)*m)=-4;
      DF1(mn+i+(j-1)*m,mn+i-2+(j-1)*m)=1;
    end
    DF=DF1+DF3;
    F=(DF1+DF3/3)*v+b;
    v=v-DF\F;
    p=reshape(v(1:mn),m,n);q=reshape(v(mn+1:mn2),m,n);
  end
  contour(x,y,p');drawnow;
end
```

Figure 8.22 shows contour plots of solutions of the Brusselator. In a contour plot, the closed curves trace level sets of the variable $p(x, y)$. In models, p and q represent chemical concentrations which self-organize into the varied patterns shown in the plots. ◀

Reaction-diffusion equations with a Turing instability are routinely used to model pattern formation in biology, including butterfly wing patterns, animal coat markings, fish and shell pigmentation, and many other examples. Turing patterns have been found experimentally in chemical reactions such as the CIMA (chlorite-iodide-malonic acid) starch reaction. Models for glycolysis and the Gray-Scott equations for chemical reactions are closely related to the Brusselator.

The use of reaction-diffusion equations to study pattern formation is just one direction among several of great contemporary interest. Nonlinear partial differential equations are used to model a variety of temporal and spatial phenomena throughout engineering and the sciences. Another important class of problems is described by the Navier-Stokes equations, which represent incompressible fluid flow. Navier-Stokes is used to model phenomena as diverse as film coatings, lubrication, blood dynamics in arteries, air flow over an airplane wing and the turbulence of stellar gas. Improving finite difference and finite element solvers for linear and nonlinear partial differential equations stands as one of the most active areas of research in computational science.

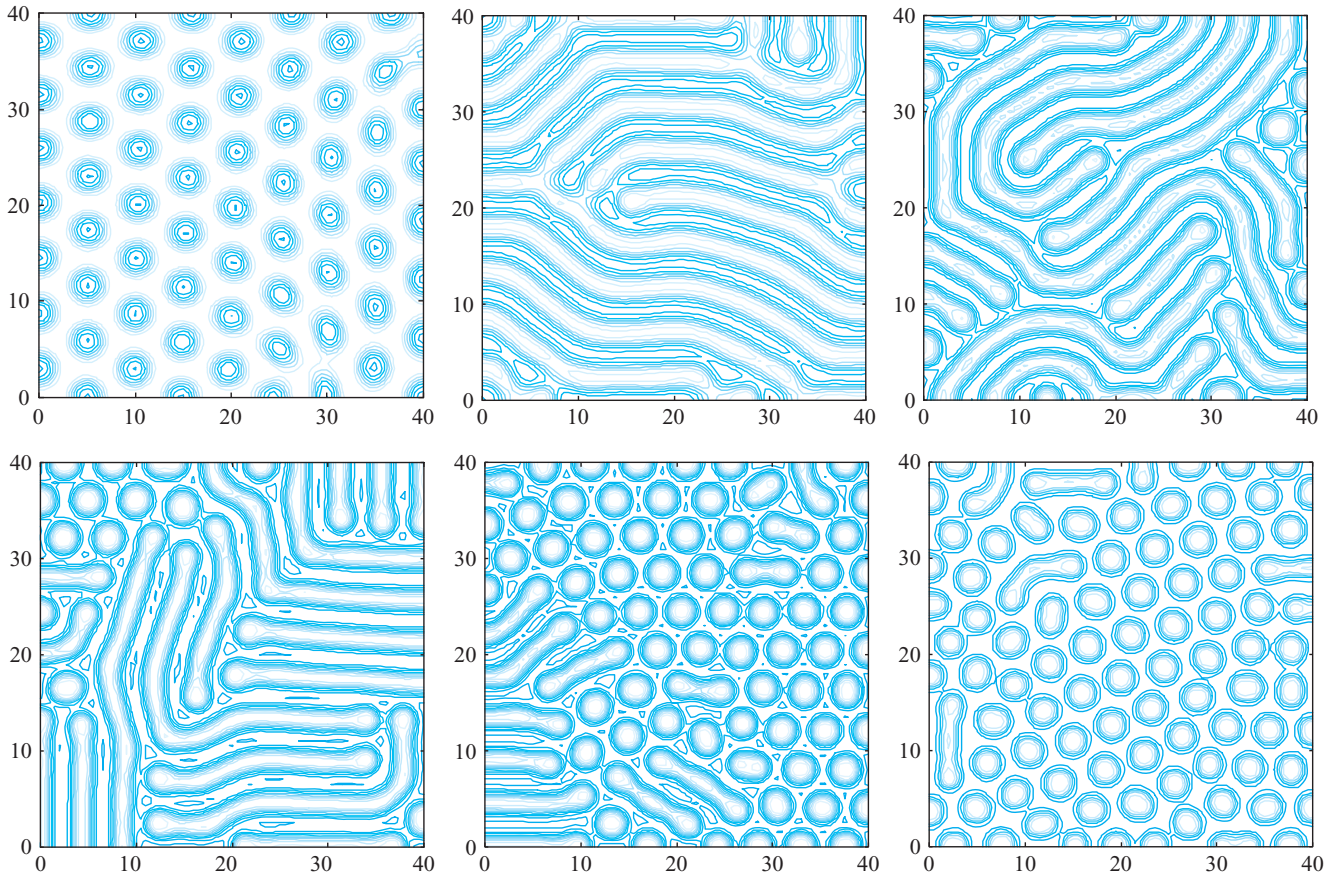

Figure 8.22 Pattern formation in the Brusselator. Contour plots of solutions $p(x, y)$ at $t = 2000$ show Turing patterns. Parameters are $D_p = 1, D_q = 8, C = 4.5$ and (a) $K = 7$ (b) $K = 8$ (c) $K = 9$ (d) $K = 10$ (e) $K = 11$ (f) $K = 12$. Settings for the finite differences are $h = k = 0.5, \Delta t = 1$.

8.4 Exercises

1. Show that for any constant c, the function $u(x, t) = c$ is an equilibrium solution of Burgers' equation $u_t + uu_x = Du_{xx}$.
2. Show that over an interval $[x_l, x_r]$ not containing 0, the function $u(x, t) = x^{-1}$ is a time-invariant solution of the Burgers' equation $u_t + uu_x = -\frac{1}{2}u_{xx}$.
3. Show that the function $u(x, t)$ in (8.68) is a solution of the Burgers' equation with Dirichlet boundary conditions (8.66).
4. Find all stable equilibrium solutions of Fisher's equation (8.69) when $f(u) = u(u - 1)(2 - u)$.
5. Show that the Brusselator has an equilibrium solution at $p \equiv C, q \equiv K/C$.
6. For parameter settings $D_p = 1, D_q = 8, C = 4.5$ of the Brusselator, for what values of K is the equilibrium solution $p \equiv C, q \equiv K/C$ stable? See Computer Problems 5 and 6.

8.4 Computer Problems

1. Solve Burgers' equation (8.63) on $[0, 1]$ with initial condition $f(x) = \sin 2\pi x$ and boundary conditions $l(t) = r(t) = 0$, using step sizes (a) $h = k = 0.1$ and (b) $h = k = 0.02$. Plot the

approximate solutions for $0 \le t \le 1$. Which equilibrium solution does the solution approach as time increases?

2. Solve Burgers' equation on the interval [0, 1] with homogeneous Dirichlet boundary conditions and the initial condition given in (8.66) with parameters $\alpha = 4, \beta = 3$, and $D = 0.2$. Plot the approximate solution using step sizes $h = 0.01, k = 1/16$, and make a log–log plot of the approximation error at $x = 1/2, t = 1$ as a function of k for $k = 2^{-p}$, $p = 4, \ldots, 8$.

3. Solve Fisher's equation (8.69) with $f(u) = u(u-1)(2-u)$ and homogeneous Neumann boundary conditions, using initial condition (a) $f(x) = 1/2 + \cos 2\pi x$ (b) $f(x) = 3/2 - \cos 2\pi x$. Plot the approximate solution for $0 \le t \le 2$ for step sizes $h = k = 0.05$. Which equilibrium solution does the solution approach as time increases?

4. Solve Fisher's equation with $f(u) = u(u-1)(2-u)$ on a two-dimensional space domain. Assume homogeneous Neumann boundary conditions, and the initial conditions of (8.73). Plot the approximate solution for integer times $t = 0, \ldots, 5$ for step sizes $h = k = 0.05$ and $\Delta t = 0.05$. Which equilibrium solution does the solution approach as time increases?

5. Solve the Brusselator equations for $D_p = 1$, $D_q = 8, C = 4.5$ and (a) $K = 4$ (b) $K = 5$ (c) $K = 6$ (d) $K = 6.5$. Using homogeneous Neumann boundary conditions and initial conditions $p(x, y, 0) = 1 + \cos \pi x \cos \pi y, q(x, y, 0) = 2 + \cos 2\pi x \cos 2\pi y$, estimate the least value T for which $|p(x, y, t) - C| < 0.01$ for all $t > T$.

6. Plot contour plots of solutions $p(x, y, 2000)$ of the Brusselator for $D_p = 1$, $D_q = 8, C = 4.5$ and $K = 7.2, 7.4, 7.6$, and 7.8. Use step sizes $h = k = 0.5, \Delta t = 1$. These plots fill in the range between Figure 8.22.

Software and Further Reading

There is a rich literature on partial differential equations and their applications to science and engineering. Recent textbooks with an applied viewpoint include Haberman [2004], Logan [1994], Evans [2002], Strauss [1992], and Gockenbach [2002]. Many textbooks provide deeper information about numerical methods for PDEs, such as finite difference and finite element methods, including Strikwerda [1989], Lapidus and Pinder [1982], Hall and Porsching [1990], and Morton and Mayers [1996]. Brenner and Scott [1994], Ames [1992], Strang and Fix [1973] are primarily directed toward the Finite Element Method.

MATLAB's PDE toolbox is highly recommended. It has become extremely popular as a companion in PDE and engineering mathematics courses. Maple has an analogous package called PDEtools. Several stand-alone software packages have been developed for numerical PDEs, for general use or targeting special problems. ELLPACK (Rice and Boisvert [1984]) and PLTMG (Bank [1998]) are freely available packages for solving elliptic partial differential equations in general regions of the plane. Both are available at Netlib.

Finite Element Method software includes freeware FEAST (Finite Element and Solution Tools), FreeFEM, and PETSc (Portable Extensible Toolkit for Scientific Computing) and commercial software COMSOL, NASTRAN, and DIFFPACK, among many others. The IMSL contains the routine DFPS2H for solving the Poisson equation on a rectangle, and DFPS3H on a three-dimensional box. These methods are based on finite differences.

The NAG library contains several routines for finite difference and finite element methods. The program D03EAF solves the Laplace equation in two dimensions by means of an integral equation method; D03EEF uses a seven-point finite difference formula and handles many types of boundary conditions. The routines D03PCF and D03PFF handle parabolic and hyperbolic equations, respectively.