Chapter 1

Boundary Value Problems for Ordinary Differential Equations

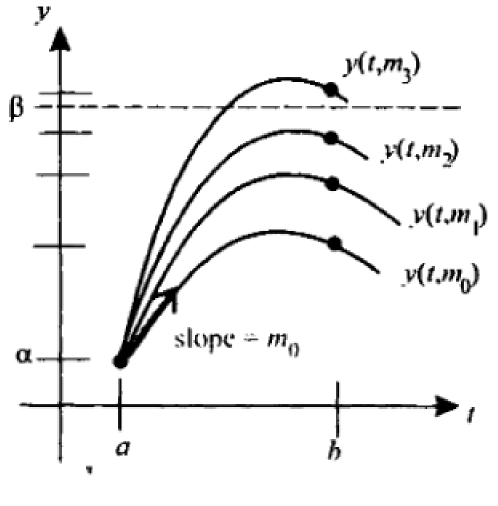
1.1. THE NONLINEAR SHOOTING METHOD

This method will really appear more like we are "shooting" at the solution than was the case with the linear shooting method of the last section. We once again turn to the general BVP (1):

$$\begin{cases} y''(t) = f(t,y,y') & (DE) \\ y(a) = \alpha, y(b) = \beta & (BCs) \end{cases}$$

Recall that when the DE was linear, we obtained the solution of the BVP as a linear combination of two solutions of (just) two specially associated IVPs. In the nonlinear case, we will solve a sequence of related IVPs:

$$(IVP)_k \begin{cases} y''(t) = f(t,y,y') & same\,(DE) \\ y(a) = \alpha, y'(a) = m_k & (ICs) \end{cases} \Rightarrow \quad solution\; y_k(t) \equiv y(t,m_k)$$



Each of the IVPs above is identical, except for the second initial condition $y(a)' = m_k$,where the parameter will be appropriately adjusted ("aimed') at each iteration. We have denoted the solution of $(IVP)_k$ as $y_k(t)$ and , since it depends on m_k, we have introducet the function of two variables $y(t,m_k) \equiv y_k(t)$. The method is roughly illustrated and explained in Figure 10.3.

Figure 1.1: Illustration of the nonlinear shooting method for a BVP: $\begin{cases} y''(t) = f(t,y,y') \\ y(a) = \alpha, y(b) = \beta) \end{cases}$ The initial approximation $y_0(t) = y(t,m_0)$ is the solution of the corresponding IVP having the same DE, the same first condition, and satisfying the initial slope $y_0'(t) = m_0$,obtained numerically by methods of the last chapter. The desired second boundary condition is compared with $y_0(b)$, and ,if neccesery, this process is repeated with adjusted initial slopes $m_1, m_2, \cdots$ until we arrive at a solution that satisfies the second boundary condition(within a desired tolerance).

The only detail left to tend to is the important issue of how best to choose our initial slopes. It turns out to be a bit complicated; indeed, figuring out subsequent initial slopes will require solving an additional IVP. We outline the procedure now, give a specific example, and afterwards give a theoretical explanation of it.

The Nonlinear Shooting Method:

1. Start with an estimate (or guess) for the initial slope of the first IVP $m_0 = y_0'(a)$; a good default is the difference quotient $m_0 = \dfrac{\beta-\alpha}{b-a}$
2. Solve the associated $(IVP)_k \begin{cases} y''(t) = f(t,y,y') \\ y(a) = \alpha, y'(a) = m_k \end{cases}$ $(k=0)$ *on* $a \le t \le b$ using, say, the Runge-Kutta method; denote the solution as $y_k(t) = y(t,m_k) \;\; (k=0)$
3. Check for accuracy by evaluating: $Diff \equiv y(b,m_k) - \beta$. If $|Diff| <$ tolerance, accept $y_k(t) = y(t,m_k)$ as solution to BVP, otherwise update
$$m_{k+1} = m_k - \frac{Diff}{z(b,m_k)},$$
where $z(t,m_k)$ solves the IVP:
$$\begin{cases} z''(t) = zf_y(t,y,y') + z'f_y(t,y,y') \\ z(a) = 0, z'(a) = 1 \end{cases}$$
Increase k and return to step 2 to iterate this procedure.

NOTE: To numerically solve the IVP for z in step 3, we will need to do it in conjunction with the concurrent IVP $y(y_k)$ in step 2 since, in general, the DE of z involves y. Thus, we will have to solve the two IVPs simultaneously by writing them into an equivalent four-dimensional first-order system.

Example 1.1. Numerically solve the BVP: ■

$$\begin{cases} y''(t) = -2(yy' + ty' + y + t) & (DE) \\ y(1) = 0, y(2) = -2 & (BCs) \end{cases}$$

by using the nonlinear shooting method in conjunction with the Runge-Kutta method with step size $h = 0.01$.

(a) Do it first with a tolerance of 0.01. How many "shots" were required? Get MATLAB to display the totality of graphs of the functions $y_k(t) = y(t,m_k)(\text{"}shots\text{"})$ in the same plot with the final one in a different color from the rest.

(b) Next do it for a tolerance of 10^7. How many "shots" were required? This time, using the subplot command, display the plots of the successive difference errors $|y_{k+1}(t) - y_k(t)| fork = 0,1,2,3,\ldots$

SOLUTION: We point out the DE is nonlinear (because of the yy' term) and so the linear shooting method would not be applicable. The associated initial value problems for y are

$$(IVP)_k \begin{cases} y''(t) = f(t,y,y') \equiv -2(yy' + ty' + y + t) \\ y(1) = 0, y'(1) = k_k \end{cases}$$

By introducing the new function $yp(t) = y'(t)$ we can translate this IVP into the following equivalent system:

$$(IVP)_k' \begin{cases} y'(t) = yp, & y(1) = 0 \\ yp'(t) = -2(y(yp) + t(yp) + y + t), & yp(1) = m_k \end{cases}$$

To get the IVP for the auxiliary function z, we compute

$$f_y(t,y,y') = -2y' - 2 \quad f_{y'}(t,y,y') = -2y - 2t,$$

which brings us to the following companion IVP for z:

$$\begin{cases} z''(t) = z(-2y' - 2) + z'(-2y - 2t) \\ z(1) = 0, z'(1) = 1 \end{cases}$$

By introducing the new function $zp(t) = z'(t)$ and combining this IVP with the previous one, we arrive at the following four-dimensional system:

$$\begin{cases} y'(t) = yp, & y(1) = 0 \\ yp'(t) = -2(y(yp) + t(yp) + y + t), & yp(1) = m_k \\ z'(t) = zp, & z(1) = 0 \\ zp'(t) = -2(yp+1)z - 2(y+t)zp, & zp(1) = 1 \end{cases}$$

For the initial slope we use the suggested default $m_0 = \dfrac{y(2)-y(1)}{2-1} = \dfrac{-2-0}{2-1} = -2.$
In turning the problem over to MATLAB, since we plan to make use of the `rksys` routine of the last chapter, we must first construct the vector-valued function corresponding to the right sides of the four-dimensional system above. We do this in the following rather generic way that can be easily mimicked for any other nonlinear shooting problem:

```
function xp=nlshoot(t,x)
xp(1)=x(2);
xp(2)=-2*(x(2)*x(2)+t*x(2)+x(1)+t);
xp(3)=x(4);
xp(4)=-2*(x(2)+1)*x(3)-2*(x(1)+t)*z(4);
```

Note that we have identified $x(1)$ with y, $x(2)$ with yp, $x(3)$ with z and $x(4)$ with zp.

Part (a): We can now perform the desired plots using the following while loop.

```
>>mk=-2; \%initialize\\
>> while 1 \%since 1 is true, loopwill continue to execute
  [t,x]=rksys('nlshoot',1,2,[0 \ mk \ 0 \ 1],0.01);
  y=X(:1); z=X(:3); \%peel off the vectors we need
  Diff=y(101)+2; \%y(101) (MATLAB) corresponds to y(2) (Math)
  if abs(Diff)<0.01
    plot(t,y,'r'), return
    end
  plot(t,y,'b'), \ \ hold on
  n=n+1; \%bump counter up one
  mk=mk-Diff/z(101); \%update slope
end
```

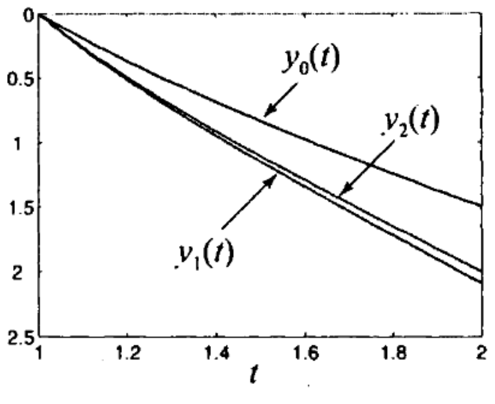


Figure 1.2: Illustration of the nonlinear shooting method applied to the BVP of Example 10.4(a). The successive approx- imations $y_k(t)$ are shown until the value of $y_k(2)$ is within a tolerance of 0.01 to - 2 , at which point the process grinds to a halt. The code is set up to graph the final approximation in red.

The plot shown in Figure 10.4 clearly shows that 3 iterations ("shots") were done: The first was too high, the second too low, and the third about right (within tolerance). Alternatively, by the way that the loop was set up, we could just enter n to query MATLAB to tell us how many iterations were done.

Part (b): We can easily modify the above loop to get the desired information and plots. We leave this as an exercise, but include the plot in Figure 10.5. We point out that in order to use the `subplot` command to get a decent plot, we first found out the number of shots needed and then ran through the loop again with an appropriately dimensioned "subplot" window. We also comment that a plot like the one in part (a) would be not quite so useful here since all approximations from the third onward are essentially indistinguishable using the graphs. This is why we look at successive differences. This is also a good way to check global errors.

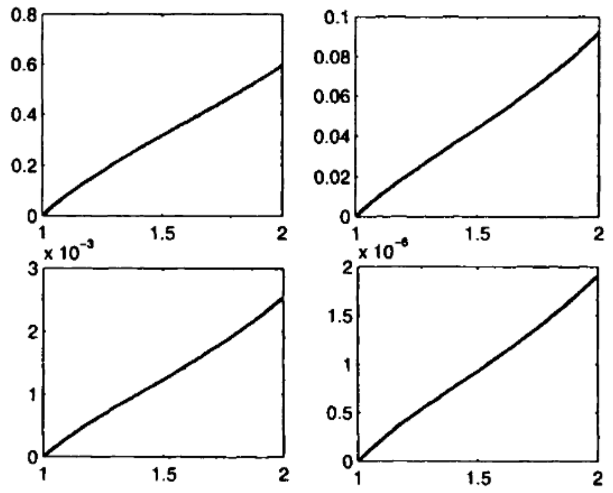


Figure 1.3: These graphs display the successive differences $|y_{k+1}(t) - y_k(t)|$ for nonlinear shooting method in part (b) of Example 10.4; this time the iterations continue until $y_k(2)$ gets within 10^{-7} of the desired value -2 and only 5 approximations ("shots") are needed.

EXERCISE FOR THE READER 1.1.

(a) Write a function M-file that will perform the nonlinear shooting method to numerically solve the BVP (1):

$\begin{cases} y''(t) = f(t,y,y') & (DE) \\ y(a) = \alpha, y(b) = \beta) & (BCs) \end{cases}$ The inputs and outputsshould be as follows:

```
[t, y, nshots] = nonlinshoot(a, alpha, b, beta, f, fy, fyp, tol, hstep)
```

The input and output variables more or less correspond to the data of the problem, but we leave the syntax open (see the suggestion for Exercise for the Reader 10.7). The variable *tol* will provide a stopping criterion for the iterations: $|y(b) - \beta| < tol$.[1] The last output variable is a positive integer giving the number of iterations (shots) that were executed. (b) Test your program by applying it to the two sets of data of Example 10.4. (c) Does Theorem 10.1 tell us anything about solutions for the following BVP?

$$\begin{cases} y''(t) = te^y - \sin(\cos(t))y' & (DE) \\ y(0) = 0, y(6) = 10 & (BCs) \end{cases}$$

Apply the nonlinear shooting method with tolerance $h = 0.01$ to solve this problem. Then repeat with tolerance 10^{-7} . Plot the final numerical solution and record the number of iterations.
Note: It is possible to write this program using MATLAB's Symbolic Toolbox capabilities and in this case the input variables fy and fyp could be dispensed with. The program we write in Appendix B does not use the Symbolic Toolbox; we leave such a construction to the interested reader.

As promised, we will now give a theoretical explanation of what has motivated the nonlinear shooting method. We assume that for any initial slope m, the IVP associated with the BVP (1),

$$\begin{cases} y''(t) = f(t,y,y') & same(DE) \\ y(a) = \alpha, y'(a) = m & (ICs)) \end{cases}$$

always has a unique solution on the time interval $[a,b]$, and we denote it by $y(t,m)$, which is a function of two variables. Our goal is to make m be a root of the equation

$$y(b,m) - \beta = 0 \tag{1.1}$$

In this equation we have held the t-variable of $y(t,m)$ to be fixed at $t = b$ so that the left side of (14) is a function of a single variable (namely m). Assuming it is differentiate, Newton's recursion formula for rootfinding (Chapter 6) suggests that it would be a good idea to define our sequence recursively using the following scheme:

$$m_k = m_{k-1} - \frac{y(b,m_{k-1}) - \beta}{\partial/\partial m\{y(b,m_{k-1})\}} \tag{1.2}$$

To go from one iteration to the next, after having (numerically) found $y(t,m_{k-1})$, the only difficult part of the formula (15) to obtain is the partial derivative $\partial/\partial m\{y(b,m_{k-1})\}$. This can be done (in an at first seemingly roundabout way) by finding an IVP for which the function

[1]Creation of this M-file will require features from MATLAB's Symbolic Toolbox; see Appendix A. Without the symbolic toolbox features, a similar program could be constructed but it would need more input variables, for example, $f_y(t,y,y')$ and $f_{y'}(t,y,y')$

$$z(t) = z(t,m) \equiv \frac{\partial y}{\partial m}(t,m)$$

is a solution and then numerically solving this IVP and evaluating it at $t = b$ to get the needed partial derivative. We can get a DE for $z(t,m)$ by differentiation of the DE for $y(t,m)$ and using the chain rule as follows (wherein we reserve primes (') for differentiations in the t-variable):

$y''(t,m) = f(t, y(t,m), y'(t,m)) \Rightarrow$
$\frac{\partial y''}{\partial m}(t,m) = f_t(t,y,y')\frac{\partial t}{\partial m} + f_y(t,y,y')\frac{\partial y}{\partial m}(t,m) + f_{y'}(t,y,y')\frac{\partial y'}{\partial m}(t,m)$

Since t and m are independent variables, $\partial t/\partial m = 0$ and so the first term on the right vanishes. We now simply replace $\partial t/\partial m(t,m)$ by $z(t,m)$ and the above becomes the following DE for z:

$$z''(t,m) = f_y z + f_{y'} z' \tag{1.3}$$

If we differentiate the initial conditions for $y(t,m)$: $\begin{cases} y(a,m) = \alpha \\ y'(a,m) = m \end{cases}$ we obtain corresponding initial conditions for $z(t,m)$:

$$\begin{cases} z(a,m) = 0 \\ z'(a,m) = 1 \end{cases} \tag{1.4}$$

Replacing the partial derivative in (15) by $z(b, m_{k-1})$ we see at once that (15), (16), and (17) yield the nonlinear shooting algorithm.

EXERCISES 1.1.

1. For each of the linear BVPs in parts (a) through (d) of Exercise 1, Section 10.2, apply the nonlinear shooting method to solve it via the Runge-Kutta method with step size $h = 0.01$ by following the outline below (if possible):

   (i) Write down the associated IVPs both for y and for the auxiliary function z.

   (ii) Translate both IVPs for y and z into a single four-dimensional IVP system of first-order DEs.

   (iii) Use MATLAB to apply the nonlinear shooting method to solve the BVP with a tolerance of 10^4. Display all of the approximations ("shots") in a single plot with the final one being displayed in a different color or plot style.

2. For each of the linear BVPs in parts (a) through (d) of Exercise 2, Section 10.2, repeat the instructions of the last exercise, but change item (iii) to: (iii'): Use MATLAB to apply the nonlinear shooting method to solve the BVP with a tolerance of 10^-6 . How many "shots" were required? Plot the final graph and also in a separate window and using the `subplot` command, get MATLAB to display the plots of the successive differences of the "shots": $|y_{k+1}(t) - y_k(t)|$ for $k = 0, 1, 2, 3, ...$

3. For each of the nonlinear BVPs given, perform the following tasks (if possible):

   (i) Write down the associated IVPs both for » and for the auxiliary function z.

   (ii) Translate both IVPs for y and z into a single four-dimensional IVP system of first-order DEs.

   (iii) Use MATLAB to apply the nonlinear shooting method to solve the BVP with a tolerance of 10^4. Display all of the approximations ("shots") in a single plot with the final one being displayed in a different color or plot style.

   (iv) Along with the BVP, an exact solution $f(t)$ is given; verify that this function actually solves the BVP.

   (v) Using a subplot window if you prefer, plot the errors of each of the successive shots with the exact solution given.

   (a) $\begin{cases} y'' = 12y^{5/3} \\ y(0) = 1, y(2) = 1/27 \end{cases}, \quad f(t) = \frac{1}{(t+1)^3}$

   (b) $\begin{cases} y'' = -[y']^2/y \\ y(0) = 1, y(5) = 4 \end{cases}, \quad (t) = \sqrt{3t+1}$

   (c) $\begin{cases} y'' = y' + 2(y - \ln t)^3 \\ y(1) = 1/2, y(2) = 1/2 + \ln 2 \end{cases}, \quad f(t) = \frac{1}{t} + \ln t$

4. Repeat the instructions for Exercise 3 on the following BVPs.

   (a) $\begin{cases} y'' = y'\cos t - y\sin t \\ y(0) = 1, y(8\pi) = 1 \end{cases}, \quad f(t) = \exp(\sin t)$

(b) $\begin{cases} y'' = y^3 - yy' \\ y(1) = 1/2, y(2) = 1/3 \end{cases}, \; f(t) = \frac{1}{t+1}$

(c) $\begin{cases} '' = y'(\ln t + 1) + y(1 + 1/t) \\ y(1) = 1, y(5) = 3125 \end{cases}, \; f(t) = t^t$

5. Use the nonlinear shooting method to solve the following BVP. Your IVP solver should be Runge-Kutta with step size $h = 0.01$. Your tolerance (for the right BC) should be 0.0001. How many iterations did this take? Plot your solution. Also, what is the value of the solution when $x = 0.4$?

$$\begin{cases} y''(t) = -t(y')^3 \\ y(0) = 0, y(1) = \pi/2 \end{cases}$$

6. *(Physics: Flight of a Well-Hit Baseball)* This problem deals with the flight of a baseball in two dimensions (which we take for convenience as the xy-plane). We consider a ball that is hit so it lands 300 feet from home plate after 3 seconds. Many factors influence the flight of the ball. We assume that the air resistance acts only against the horizontal velocity, and for this particular baseball it is proportional to the horizontal velocity with exponent 1.2. Assuming the of home plate are $(x, y) = (0, 0)$, we let $x(t)$ and $y(t)$ denote the x and y coordinates of the position of the ball t seconds after it is hit. Thus, at time t, the coordinates of the baseball are $(x(t), y(t))$, $0 \leq t \leq 3$. The air resistance assumption and Newton's law from basic physics give the following system of second-order DEs:

$$\begin{cases} x''(t) = -cx(t)^(1.2) \\ y''(t) = -g \end{cases},$$

where for the ball being used the constant $c = 0.44$. The initial position of the ball is $(x(0), y(0)) = (0, 3)$ <feet>. Since the ball lands after 3 seconds we also get $(x(3), y(3)) = (300, 0)$.

(a) Explicitly find the function $y(t)$ just using basic calculus.

(b) Numerically find $x(t)$ by using the shooting method with step size $h = 0.01$ (and implementing the Runge-Kutta method), and sketch a plot of the path of the ball (i.e., of y vs. x). (For this part, you need not print the graph of x vs. t, or give any explicit values for $x(t)$.)

(c) After how many seconds does the ball reach its maximum height? At this time what is the x- coordinate?

(d) With the same hit, how far would the ball have gone (on the x-axis), if, as in the imaginary assumptions of physics courses, there was no air resistance?

7. *(Civil Engineering: Deflection of a Beam)* Use the nonlinear shooting method with $h = 0.01$ in the Runge-Kutta method to solve the exact beam-deflection model BVP:

$$\begin{cases} y''(t)/[1 + (y')^2]^{3/2} = \dfrac{T}{EI}y + \dfrac{wx(x-L)}{2EI} \\ y(0) = 0 = y(L) \end{cases},$$

having the parameters: $L = 50$ feet (length), $T = 300$ lb (tension at ends), $w = 50$ lb/ft (vertical load), $E = 1.2 \times 10^7$ lb/ft^2 (modulus of elasticity), and $I = 4$ ft^4 (central moment of inertia).
(a) Graph the resulting numerical solution, (b) How does the solution compare with that obtained for the corresponding linear approximating BVP of Example 10.3?

1.2. THE FINITE DIFFERENCE METHOD FOR LINEAR BVP'S

The method we present next is philosophically quite different from the shooting methods. It immediately discretizes the BVP by approximating the derivatives with difference quotients. The problem is then translated into a linear system that is easily solved directly. This method will pave the way for the corresponding finite difference methods that we will employ in the next two chapters for solving PDEs. There are analogues of this method for nonlinear BVPs; the discretization is done in the same way but the resulting system of equations will no longer be linear.[2]

All finite difference methods are based on approximating derivatives of a function by certain difference quotients. These difference quotient formulas can always be obtained using Taylor's theorem. We will be needing them only for first and second derivatives, and we now present them in the following lemma. To describe the error bounds, we employ the "big O" notation that was introduced in Section 8.3.

LEMMA 1.1. *(Central Difference Formulas)*

[2]For the nonlinear analogues of the finite difference method, we cite the reference: [BuFa-01] (see Section 11.3 therein.)

(a) If $f(x)$ is a function having a continuous second derivative in the interval $a-h \le x \le a+h$, then we have

$$f'(a) \approx \frac{f(a+h)-f(a-h}{2h}, \tag{1.5}$$

where the error of the approximation is $O(h^2)$.

(b) If, furthermore, f(x) has a continuous fourth derivative throughout $a-h \le x \le a+h$, then we also have the approximation

$$f''(a) \approx \frac{f(a+h)-2f(a)+f(a-h)}{h^2}, \tag{1.6}$$

and the error of this approximation is also $O(h^2)$.

Proof of part (a): Taylor's theorem allows us to write

$$f(a+h) = f(a) + hf'(a) + h^2 f''(a)/2 + O(h^3), \text{and}$$

$$f(a-h) = f(a) - hf'(a) + h^2 f''(a)/2 + O(h^3).$$

Subtracting the second of these equations from the first gives $f(a+h)-f(a-h) = 2hf'(a)+O(h^3)$ and solving this for $f'(a)$ produces (18). We have used the facts that $O(h^3)+O(h^3) = O(h^3)$ and $O(h^3)/h = O(h^2)$. The proof of part (b) is similar and is left as the next exercise for the reader.

EXERCISE FOR THE READER 1.2.

Prove part (b) of Lemma 10.3.

We now explain the finite difference method in more detail. Consider the linear BVP (5):

$$\begin{cases} y''(t) = p(t)y' + q(t)y + r(t) & (DE) \\ y(a) = \alpha, y(b) = \beta & (BCs) \end{cases}$$

Choose a positive integer N, and subdivide the interval $a < t < b$ into N equal subintervals using $N-1$ interior grid values: $t_1 = a+h, t_2 = a+2h, \cdots, t_{N-1} = a+(N-1)h$, where $h = (b-a)/N$. We also write $t_0 = a$ and $t_N = b$; see Figure 10.6. We let,

$$y_i = y(t_i) \ \ (0 \le i \le N),$$

and similarly,

$$p_i = p(t_i),\ q_i = q(t_i),\ r_i = r(t_i)\ (0 \le i \le N).$$

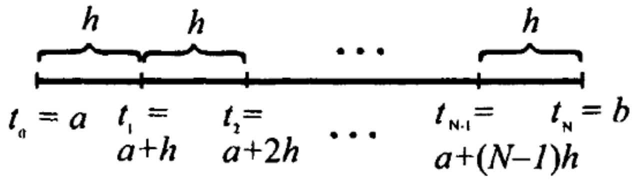


Figure 1.4: Grid value notation for the finite difference method for a BVP.

At each internal grid value $f_i (0 < i < N)$, we approximate the DE (5)

$$y''(t_i) = p(t_i)y'(t_i) + q(t_i)y(t_i) + r(t_i)$$

using the central difference formulas of Lemma 10.3, to obtain the approximation with local truncation error $O(h^2)$:

$$\frac{y_{i+1}-2y_i+y_{i-1}}{h^2} = p_i \frac{y_{i+1}-y_{i-1}}{2h} + q_i y_i + r_i \quad (0 \le i \le N) \tag{1.7}$$

Multiplying by h^2 , and then regrouping, we can rewrite each equation in (20) as:

$$y_{i+1} - 2y_i + y_{i-1} = hp_i(y_{i+1}-y_{i-1})/2 + h^2 q_i y_i + h^2 r_i, \text{or}$$

$$(1+p_i h/2)y_{i-1} - (2+h^2 q_i)y_i + (1-p_i h/2)y_{i+1} = h^2 r_i \quad (0 \le i \le N) \tag{1.8}$$

Since we know from the two BCs of (5) that

$$y_0 = \alpha, \quad y_N = \beta,$$

the equations of (21), form a linear system in the $N-1$ unknowns $y_1, y_2, \cdots, y_{N-1}$, which, when put in matrix form $AY = C$, has

$$A = \begin{bmatrix} -(2+h^2q_1) & 1-p_1h/2 & 0 & \cdots & 0 \\ 1+p_2h/2 & -(2+h^2q_2) & 1-p_2h/2 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 1+p_{N-2}h/2 & -(2+h^2q_{N-2}) & 1-p_{N-2}h/2 \\ 0 & \cdots & 0 & 1+p_{N-1}h/2 & -(2+h^2q_{N-2}) \end{bmatrix}$$

and

$$C = \begin{bmatrix} h^2r_1 - (1+p_1h/2)\alpha \\ h^2r_2 \\ \vdots \\ h^2r_{N-2h} \\ h^2r_{N-1} - (1-p_{N-1}h/2)\beta \end{bmatrix} \tag{1.9}$$

Notice the special form of the coefficient matrix A. Often in finite difference methods and in many other applications, the coefficient matrices that arise are of a similar **banded** form (i.e., nonzero entries lie entirely on a few diagonal bands). This special type of banded matrix is called a **tridiagonal matrix**. Banded matrices are special cases of what are called **sparse** matrices, which are matrices having the majority of the entries being zero. An $n \times n$ tridiagonal matrix has at most $3n-2$ nonzero entries (among its n^2 entries). Since large matrices often eat up a lot of memory with storage, it is often more expedient to deal with sparse matrices of specialized forms using specialized methods. To solve such tridiagonal systems, rather than Gaussian elimination, we will be using the so-called Thomas method, whose algorithm is given below:[3]

PROGRAM 1.1. *The Thomas method for solving tridiagonal systems of the form:*

$$\begin{bmatrix} d_1 & a_1 & 0 & 0 & 0 & \cdots & & 0 & 0 \\ b_2 & d_2 & a_2 & 0 & 0 & \cdots & & 0 & 0 \\ 0 & b_3 & d_3 & a_3 & 0 & & & \vdots & \vdots \\ 0 & 0 & b_4 & d_4 & a_4 & 0 & & & \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & & & \\ 0 & 0 & & & 0 & b_{n-2} & d_{n-2} & a_{n-2} & 0 \\ 0 & 0 & \cdots & & & 0 & b_{n-1} & d_{n-1} & a_{n-1} \\ 0 & 0 & \cdots & & & & 0 & b_n & d_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{bmatrix}.$$

```
function x = thomas(a,d,b,c)
%solves matrix equation Ax=c, where A is a tridiagonal matrix
%inputs: a=upper diagonal of matrix A a(n)=0, d=diagonal of A,
%b=lower diagonal of A, b(1)=0, c=right-hand side of equation
n=length(d);
a(1)=a(1)/d(1);
c(1)=c(1)/d(1);
for i=2:n-1
  denom=d(i)-b(i)*a(i-1);
  if (denom==0), error('zero in denominator'), end
  a(i)=a(i)/denom;
  c(i)=(c(i)-b(i)*c(i-1))/denom;
end
c(n)=(c(n)-b(n)*c(n-1))/(d(n)-b(n)*a(n-1));
x(n)=c(n);
for i=n-1:-1:1
  x(i)=c(i)-a(i)*x(i+1);
end
```

Example 1.2. Use the thomas program above to solve the tridiagonal system: ■

$$\begin{array}{rcrcrcrcl} 2x_1 & - & x_2 & & & & & = & 1 \\ -x_1 & + & 2x_2 & - & x_3 & & & = & 0 \\ & - & x_2 & + & 2x_3 & - & x_4 & = & 0 \\ & & & - & x_3 & + & 2x_4 & = & 1 \end{array}$$

[3]Banded and sparse matrices were studied in Chapter 7; in particular, the Thomas method was introduced in Exercise 9 of Section 7.5

```
>> a=[-1 -1 -1 0]; d=[2 2 2 2]; b=[0 -1 -1 -1]; c=[1 0 0 1];
>> format rat;
>> thomas(a,d,,b,c)
```

ans $\to$ 1 1 1 1 (This answer is easily checked.)

We now make some technical comments on implementing the finite difference method. In order to solve the system $AY = C$, we will need the coefficient matrix A of (22) to be nonsingular. In general this can fail, but the following theorem gives sufficient conditions to guarantee A's invertibility.

THEOREM 1.1.

Suppose that the functions $p(t), q(t),$ and$r(t)$ are continuous on $a \leq t \leq b$ and that $q(t) \geq 0$on this time interval. Then the linear system $AY = C$ where A and C are as in (22) will have a unique solution provided that $h < 2/M$, where $M = max\{|p(t)| : a \leq t \leq b\}$. The hypotheses guarantee that the coefficient matrix A will be **strictly diagonally dominant**, which means that

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|. \tag{1.10}$$

In other words, the absolute value of any diagonal entry dominates the sum of the absolute values of all other entries in its row. Diagonally dominant matrices are always invertible [4], 4 and furthermore, it can be shown that the Thomas algorithm works very well in their presence. There are instances, however, where the Thomas method will fail for tridiagonal nonsingular matrices (e.g., this happens if $a_{11} = 0$), but there are ways to modify the method to deal with such cases. Generally speaking, the linear shooting method is more efficient for solving a linear BVP when the former is coupled with the Runge-Kutta method. This is because the Runge-Kutta method has a local truncation error of $O(h^4)$ while that for the finite difference method is $O(h^2)$. Our main reason for introducing it here is to prime the way for its generalization to solving partial differential equations; the shooting methods do not naturally extend to the setting of PDEs.

EXERCISE FOR THE READER 1.3.

Show that under the conditions of Theorem 10.4, the matrix A of (22) is diagonally dominant.

Example 1.3. Use the finite difference method with $h = 0.1$ to solve the beam-deflection BVP of Example 10.3: ■

$$\begin{cases} y''(x) = \frac{T}{EI}y + \frac{wx(x-L)}{2EI} \\ y(0) = 0 = y(L) \end{cases}, 0 \leq x \leq L,$$

having the parameters L = 50 feet (length), T = 300 lb (tension at ends), w = 50 lbs/ft (vertical load), E = 1.2x$\times 10^7$ lb/ft^2 (modulus of elasticity), and I = 4 ft^4 (central moment of inertia).

(a) Do it first for N = 20 subdivisions to obtain the approximate solution yl and plot its graph.

(b) Redo it for both N = 40, and N = 80 subdivisions, to get approximate solutions y2,and y3.

SOLUTION: Since the scripts are similar, we present only the one for obtaining $y1$ when N = 20. The script is written in such a way as to be easily modified to work for any linear BVP. The graphs of the solutions look identical, so we present only the graph of $y1$, but give plots of the differences $y1 - y2$ and $y2 - y3$.

```
%MATLAB script for finite difference method for above problem.
xa=0; xb=50; n=20; h=(xb-xa)/n;x=h:h:(xb-h);
for i=1:n-1, a(i)=0;end, b=a;
a(1:n-2)=1-p(1:n-2)*h/2; %above diagonal band
d=-(2+h*h*q); %diagonal
b(2:n-1)=1+p(2:n-1)*h/2; %below diagonal band
c(2:n-2)=h*h*r(2:n-2);
c(1)=h*h*r(1)-(1+p(1)*h/2)*ya; c(n-1)=h*h*r(n-1)-(1-p(n-1)*h/2)*yb;
y=thomas(a,d,b,c);
X=(xa x xb);
Y=(ya y yb);
plot(X,Y), grid on
```

[4]Proof: Suppose that A is diagonally dominant. If A were not invertible, then there would exist a nonzero vector x such that $Ax = 0$. Let k be an index so that the absolute value $|x_k| = ||x||_\infty$ is as large as possible (in the norm notation from Section 7.6, this would mean $|x_k| = ||x||_\infty$) Take the kth equation of $Ax = b : \sum_{j=1}^{n} a_{kj}x_j = 0$, divide by x_k, and solve for a_{kk} to get a $a_{kk} = -\sum_{j=1, j\neq k}^{n} a_{kj} \cdot (x_j/x_k)$. Now take absolute values and use the triangle inequality to get $|a_{kk}| \leq \sum_{j=1, j\neq k}^{n} |a_{kj} \cdots (x_j/x_k)| \leq \sum_{\substack{j=1 \\ j\neq k}}^{n}$. Wh at we now have contradicts diagonal dominance of the matrix A, so we have proved that A must indeed be invertible.

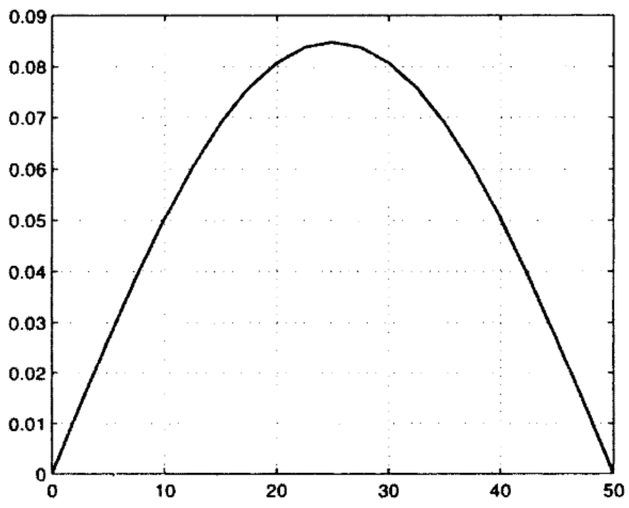


Figure 1.5: Graph of the solution of the beam-deflection problem of Example 10.6 using N=20 subdivisions.

We can now store these x- and y-values as xl and yl by entering:

```
>> xl=X; yl=Y;
```

and next go on to slightly change the script to do N = 40 iterations and then N = 80 iterations and store the corresponding x and y-values as x2, y2 and x3, y3 respectively. You will notice that the graphs look quite identical. To plot the differences: yl - y2 , and y2 - y3 , one must be a bit careful since yl , y2, and y3 all have different lengths. Each has N + 1 components (N - 1 grid points + the two boundary points). Here is one strategy to plot yl - y2 versus x. The grid points for y2 consist of those of yl plus one extra grid point between each adjacent pair of grid points for yl (located at the midpoint). We must reformulate y2, only at the grid values for yl (throwing away the extra ones at the midpoints). Let's call this "trimmed down" version of y2 by y2 trim . To form y2 trim in MATLAB, we could use the following line:

```
>> for i=1:21, y2trim(i)=y2(2*i-1); end %alternatively: y2trim = y2
   (1:2:41)
```

Now we can plot the difference of yl - y2 by simply entering:

```
>> plot (xl,yl-y2trim)
```

In a similar fashion, the following commands give the plot of the difference y2 - y3:

```
>> for i=1:41, y3trim(i)=y3(2*i-1); end
>> plot(xl,y2-y3trim)
```

See Figure 10.8 for both of these error plots. Both scripts finished on the author's computer in less than a second, and the differences are quite small. We leave it to you to see what happens if one continues this by repeatedly doubling the number on subintervals N. N= 160, N=320, N=640,....

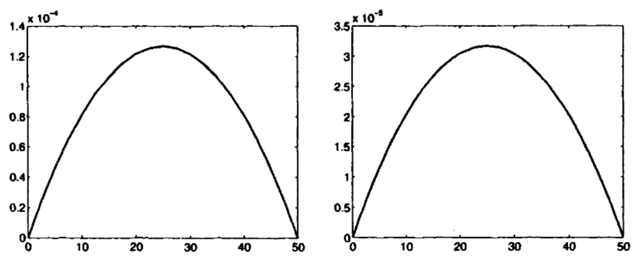


Figure 1.6: Plots of differences of finite difference approximated solutions to the deflected beam problem of Example 10.6. (a) The left graph is of the difference of the N = 20 and N = 40 interior grid point solutions and (b) the right one is the graph of the difference of the N = 40 and N = 80 interior grid point solutions.

EXERCISES 1.2.

1. For each of the linear BVPs of parts (a) through (d) of Exercise 1 of Section 10.2, use the finite difference method with $N = 100$ to solve and then plot the solution. Whenever possible, use the Thomas method to solve the tridiagonal system. If the coefficient matrix fails to be invertible (so that errors will come up with both the Thomas and the Gaussian methods), try bumping N up to 500.

2. For each of the linear BVPs of parts (a) through (d) of Exercise 2 of Section 10.2, use the finite difference method with $N = 100$ to solve and then plot the solution. Whenever possible, use the Thomas method to solve the tridiagonal system. If the coefficient matrix fails to be invertible (so that errors will come up with both the Thomas and the Gaussian methods), try bumping N up to 500.

3. For each of the BVPs and corresponding general solutions for the DEs given in parts (a) through (c) of Exercise 3 of Section 10.2, do the following (if possible): (i) Use the finite difference method with $N = 100$ to solve and store the solution in vectors $t1$, $y1$. (ii) Repeat with $N = 500$ and store the solution in vectors $t2$, $y2$. (iii) Repeat once again with $N = 2500$ and store the solution in the vectors $t3$,$y3$. (iv) Determine the constants in the general solution given so that it solves the given BVP. (v) Plot the four curves in the same graph using different plot colors/styles for each. In situations where graphs are indistinguishable, plot also the errors (differences of approximations with exact solutions). Whenever possible, use the Thomas method to solve the tridiagonal system.

4. Repeat all parts of the previous exercise for each of the BVPs and general solutions given in parts (a) through (c) of Exercise 4 of Section 10.2.

5. A thin rod of length L is insulated along the lateral surface but kept at temperature $T = 0$ at both ends $x = 0$ and $x = L$. The rod has a heat source which is proportional to the temperature at cross-section x with proportionality constant Q, The steady-state temperature function $T(x)$ $0 \le x \le L$ then satisfies the DE:

$$T_{xx} + QT = 0, T = T(x), 0 \le x \le L.$$

(See Section 11.2 for a derivation of more general heat equations.) Solve this DE with the given BC's $T(0) = T(L) = 0$ using the finite difference method with $N = 20$. Repeat with $N = 40, N = 60$, and $N = 120$. Plot these four approximations together (using different plot styles/colors). In cases where two are indistinguishable, plot the corresponding successive differences.
(See Section 11.2 for a derivation of more general heat equations.) Solve this DE with the given BC$'$ s $T(0) = T(L) = 0$ using the finite difference method with $N = 20$. Repeat with $N = 40, N = 60$, and $N = 120$. Plot these four approximations together (using different plot styles/colors). In cases where two are indistinguishable, plot the corresponding successive differences.

6. The general solution of the DE $y'' = -y$ is $y = A\sin t + B\cos t$.

   (a) What restriction (on the parameters A and B) does the condition $y(0) = 0$ place?

   (b) For which values of $L > 0$ does the BVP consisting of the DE and the BC's $y(0) = y(L) = 0$ have a solution (existence)? For such values of L, show that the solution is not unique.

   (c) Use $L = 1$ in the BVP of part (b) and apply the finite difference method with $N = 20$. What happens? Does the Thomas algorithm work? If not, try Gaussian elimination. Is the coefficient matrix nonsingular?

   (d) Repeat part (c) using $L = \pi$.

7. (a) Use Taylor's theorem to establish the following fourth-order central difference formula:

$$f'(a) \approx \frac{-f(a+2h) + 8f(a+h) - 8f(a-h) + f(a-2h)}{12h}$$

   with error $O\left(h^4\right)$, provided that $f^{(5)}(x)$ is continuous in the interval $a - 2h \le x \le a + 2h$.

   (b) In the same fashion, derive the fourth-order central difference formula

$$f''(a) \approx \frac{-f(a+2h) + 16f(a+h) - 30f(a) + 16f(a-h) - f(a-2h)}{12h}$$

   with error $= O\left(h^4\right)$, provided that $f^{(6)}(x)$ is continuous in the interval $a - 2h \le x \le a + 2h$.

1.3. THE RAYLEIGH-RITZ METHOD

Figure 1.7: Johny William Strutt (Lord Rayleigh) (1942-1919), English physicist and mathematician.

The material of this section contains much more theory than a typical section of the text. The ideas contained herein come from an important and very beautiful area of mathematics which blends linear algebra and analysis. It is fair to say that this area gave birth to the subject of functional analysis. Furthermore, the generalization of the Rayleigh-Ritz method to higher dimensions gives rise to the very important finite element method (Chapter 13) for numerical solution of PDEs. As the language in the development will indicate, many of the concepts leading to the Rayleigh-Ritz method are motivated by concepts in physics. Indeed, this was the motivational setting that led to its development. Despite the fact that the RayleighRitz method [5] dates back to the beginning of the twentieth century, it took another half century before the finite element method came to fruition. The basic idea of the Rayleigh-Ritz method is that a boundary value problem can be recast as a certain minimization problem.

Rather than strive for generality, our purpose in this section will be to understand the concepts behind the Rayleigh-Ritz method so we begin by focusing our attention on the following boundary value problem:

$$(\text{BVP})\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases} \tag{1.11}$$

Here $f(x)$ is a continuous function. This problem has, by itself, numerous physical interpretations, as we have seen in previous chapters. As examples we mention the steady-state heat distribution on a thin rod (Chapter 11) with ends maintained at temperature zero, or the deflection of an elastic beam (Section 10.1) whose ends are fixed.

We introduce the **inner product** $\langle u,v\rangle$ for a pair of piecewise continuous bounded functions on $[0,1]$:

$$\langle u,v\rangle = \int_0^1 u(x)v(x)dx. \tag{1.12}$$

Recall that for a function $u(x)$ to be piecewise continuous on $[0,1]$, it means that the domain can be broken up into subintervals: $0 = a_0 < a_1 < \cdots < a_n = 1$ such that $u(x)$ is continuous on each open subinterval (a_{i-1}, a_i). We point out the following simple yet very important properties of this inner product. By linearity of the integral, it immediately follows that the inner product is linear in each variable, i.e.,

$$\begin{aligned} \langle \alpha u_1 + \beta u_2, v\rangle &= \alpha\langle u_1, v\rangle + \beta\langle u_2, v\rangle \\ \langle u, \alpha v_1 + \beta v_2\rangle &= \alpha\langle u, v_1\rangle + \beta\langle u, v_2\rangle\text{'} \end{aligned} \tag{1.13}$$

where the u, u_i, v, v_i denote arbitrary (piecewise continuous bounded) functions and α, β denote arbitrary real numbers. Even clearer is the following symmetry property:

$$\langle u,v\rangle = \langle v,u\rangle \tag{1.14}$$

[5] Despite family attempts to dissuade him from vigorously pursuing a career as a full-time scientist, Lord Rayleigh (who succeeded to the title at age 30) was so intrigued by the mysteries of physics and the power of mathematics, that he made a firm commitment not to let his official diplomatic and social functions interfere too much with his dedication to scientific inquiry. For most of his life, he was financially independent, and this allowed him to set up a personal laboratory in his estate and gave him more time to focus on his research without the distraction of the other duties associated with an academic post. For the periods that he did hold academic posts at Cambridge, he took his duties with utmost conscientiousness and made some very lasting improvements in the university's scientific programs. Lord Rayleigh was a model scientist; his work touched upon and connected many areas (the Rayleigh-Ritz method is a good example inside mathematics) and was extensive (446 publications), and he won numerous prizes and recognitions for his work. Beside his scientific prowess, he was also a kind, modest, and generous man. When he won the Nobel Prize in physics in 1904, he donated his prize money to Cambridge University for the purpose of building more laboratories. In 1902, in his acceptance speech for the National Order of Merit, he stated "... the only merit of which I personally am conscious was that of having pleased myself by my studies, and any results that may be due to my researches were owing to the fact that it has been a pleasure for me to become a physicist."

Walter Ritz (1878-1909) was a Swiss/German mathematician/physicist. After entering the Polytechnic University of Zurich in an engineering program, he found that he was not satisfied with the compromises and lack of rigor in his engineering courses, so he switched to physics. He was a classmate of Albert Einstein. For health reasons, he needed to move away from the humid climate of ZÃijrich, and went on to the University GÃűttingen to complete his studies. There he was influenced by the teachings of David Hubert. Despite his short life and career, he was able to accomplish quite a lot of scientific research. Actually, Lord Rayleigh and Ritz never met. Rayleigh first developed a mathematical method for predicting the first natural frequency of simple structures by minimizing the distributed energy. Ritz subsequently extended the method to solve (numerically) associated displacement and stress functions.

In light of properties (26) and (27), the inner product is said to be a **symmetric bilinear form**. Another property of the inner product is that it is **positive definite**: If $u(x)$ is a piecewise continuous function on $[0,1]$ that is not zero on some open interval (a_{i-1},a_i), then $\langle u,u\rangle > 0$ (see Exercise 17).

We consider the following rather large class of **admissible functions** on $[0,1]$ which obey the boundary conditions of our problem (24) :

$$\mathscr{A} = \{v:[0,1]\to\mathbb{R} : v(x) \text{ is continuous, } v'(x) \text{is piecewise continuous and bounded, and } v(0)=0, v(1)=0\}. \quad (1.15)$$

EXERCISE FOR THE READER 1.4.

Show that the space $\mathscr{A}$ is closed under the operations of addition of functions and scalar multiplication. More precisely, if $v,w\in\mathscr{A}$ and α is any real number, show that the functions $v+w$, and αv also belong to $\mathscr{A}$.
For functions in this class we further define the following functional: $F:\mathscr{A}\to\mathbb{R}$ by the formula:

$$F(v) = \frac{1}{2}\left\langle v',v'\right\rangle - \langle f,v\rangle \quad (1.16)$$

In the setting where (24) models the deflection of an elastic beam, certain physical interpretations can be given to some of these quantities. For a given displacement $v(x)$, the inner product $\langle f,v\rangle$ represents the so-called **load potential** and the term $\frac{1}{2}\langle v',v'\rangle$ represents the internal **elastic energy**. The functional $F(v)$ then represents the **total potential energy**. Using physics it can be proved that the solution of (24) will have minimal total potential energy over all possible admissible functions $v\in\mathscr{A}$. This fact is known as the **Principle of Minimum Potential Energy (MPE)** and we will prove it mathematically in Theorem 10.5 below. Thus, the variational problem which turns out to be equivalent to the boundary value problem (24) is the following:

$$\text{(MPE) Find } u\in\mathscr{A} \text{ satisfying } F(u)\le F(v) \text{ for all } v\in\mathscr{A}. \quad (1.17)$$

Another equivalent, but very different looking problem whose equivalence to the boundary value problem is known in physics as **Principle of Virtual Work (PVW)**, is the following:

$$\text{(PVW) Find } u\in\mathscr{A} \text{ satisfying } \left\langle u',v'\right\rangle = \langle f,v\rangle \text{ for all } v\in\mathscr{A} \quad (1.18)$$

It is quite a surprising fact that the three seemingly different problems (24), (30), and (31) have equivalent solutions. The precise result is stated in the following theorem.

THEOREM 1.2. *(Variational Equivalences of a Boundary Value Problem)*

Suppose that $f(x)$ is any continuous and bounded function on $0<x<1$, and tha $u(x)$ is an admissible function of the class $\mathscr{A}$ defined in (28). Then the followin are equivalent:

(a) The function $u(x)$ is a solution of the (BVP) (24) $\begin{cases} -u''(x)=f(x), 0<x<1 \\ u(0)=0, u(1)=0 \end{cases}$.

(b) The function $u(x)$ is a solution of the (MPE) (30): $F(u)\le F(v)$ for all $v\in\mathscr{A}$.

(c) The function $u(x)$ is a solution of the (PVW) (31) : $\langle u',v'\rangle = \langle f,v\rangle$ for all $v\in\mathscr{A}$.

Furthermore, each of these three problems has unique solutions.

Proof: The proof is rather long, so we break it up into several pieces. The proof that (24) has a unique solution can be accomplished quite easily (see Exercise 22). We point out that Theorem 10.1 does not apply.[6]

Step 1: We first show that (b) implies (c). To this end, suppose that $u(x)$ solves the (MPE), so that $F(u)\le F(v)$ for all $v\in\mathscr{A}$. Letting ε denote any real number, we may conclude that $F(u)\le F(u+\varepsilon v)$, where $v\in\mathscr{A}$ is arbitrary. If we hold the functions u and v fixed, we can view the function on the right $\phi(\varepsilon)\equiv F(u+\varepsilon v)$ as a real-valued function of

[6]A general result shows that existence and uniqueness questions about general BVPs can be reduced to questions about homogeneous BVPs. The following is taken from page 197 of [Sta-79]: **Theorem** For a pair of 2×2 matrices $\mathscr{A}$ and B, and continuous functions $f(x), g(x), h(x)$ on an interval $[a,b]$, the BVP consisting of the DE $y''=h(x)y'+g(x)y+f(x)$ $(y=y(x))$ and the general boundary conditions $A\begin{bmatrix} y(a) \\ y'(a)\end{bmatrix}+B\begin{bmatrix} y(b) \\ y'(b)\end{bmatrix}=\begin{bmatrix}\alpha\\ \beta\end{bmatrix}$ has a unique solution if and only if the corresponding homogeneous problem with $f(x)=0$, and $\alpha,\beta=0$ has only the trivial solution $y(x)=0$. For our special problem (24) we need only take $h(x)=g(x)=0$ to get the DE and $\mathscr{A}=\begin{bmatrix}1&0\\0&0\end{bmatrix}$, $B=\begin{bmatrix}0&0\\1&0\end{bmatrix}$ to get the BC 's. The corresponding homogeneous problem is just: $y''=0$ and $y(0)=y(1)=0$. Integrating this DE and using the BC's easily shows $y(x)=0$ is the only solution. Thus, this theorem implies our problem (24) has a unique solution.

ε. Using bilinearity and then symmetry of the inner product, we may expand this function as follows:

$$\begin{aligned}
\phi(\varepsilon) &\equiv \frac{1}{2}\left\langle (u+\varepsilon v)', (u+\varepsilon v)' \right\rangle - \langle f, u+\varepsilon v\rangle \\
&= \frac{1}{2}\left\langle u'+\varepsilon v', u'+\varepsilon v' \right\rangle - \langle f,u\rangle - \varepsilon\langle f,v\rangle \\
&= \frac{1}{2}\left\langle u',u'\right\rangle + \frac{\varepsilon}{2}\left\langle u',v'\right\rangle + \frac{\varepsilon}{2}\left\langle v',u'\right\rangle + \frac{\varepsilon^2}{2}\left\langle v',v'\right\rangle - \langle f,u\rangle - \varepsilon\langle f,v\rangle \\
&= \frac{1}{2}\left\langle u',u'\right\rangle + \varepsilon\left\langle u',v'\right\rangle + \frac{\varepsilon^2}{2}\left\langle v',v'\right\rangle - \langle f,u\rangle - \varepsilon\langle f,v\rangle.
\end{aligned}$$

Since each of the inner products in the last expression is simply a real number, the function $\phi(\varepsilon)$ is just a second-degree polynomial (in the variable ε). Since we know this function has a minimum value at $\varepsilon = 0$, we must have $\phi'(0) = 0$. Differentiating the last expression for $\phi(\varepsilon)$ in the above expansion, this gives $\langle u',v'\rangle - \langle f,v\rangle = 0$, and since $v \in \mathscr{A}$ was arbitrary, this shows (PVW). *Step 2*: We show that (c) implies (b). So assume that $u(x)$ solves the (PVW), i.e., $\langle u',v'\rangle = \langle f,v\rangle$ for all $v \in \mathscr{A}$. Fix now an admissible function $v \in \mathscr{A}$. Our task is to show that $F(v) \geq F(u)$. Setting $w = v-u$ so $v = u+w$, we may use bilinearity and symmetry as above to write:

$$\begin{aligned}
F(v) &= F(u+w) \\
&= \frac{1}{2}\left\langle u'+w', u'+w'\right\rangle - \langle f, u+w) \\
&= \underbrace{\frac{1}{2}\left\langle u',u'\right\rangle - \langle f,u\rangle}_{=F(u)} + \underbrace{\left\langle u',w'\right\rangle - \langle f,w\rangle}_{=0 \text{ by (PVW)}} + \underbrace{\frac{1}{2}\left\langle w',w'\right\rangle}_{\geq 0} \\
&\geq F(u),
\end{aligned}$$

as desired. *Step 3*: We show that (a) implies (c). We thus assume that the function $u(x)$ solves the BVP (24). From the differential equation $-u''(x) = f(x), 0 < x < 1$, the second derivative of $u(x)$ exists (and is continuous) so it follows that the first derivative $u'(x)$ is continuous (from calculus, differentiability implies continuity). Furthermore, since $f(x)$ is assumed to be bounded, so must be $u'(x)$, and from the boundary conditions stipulated by (24), it follows that $u(x)$ is an admissible function (i.e., $u \in \mathscr{A}$). We now fix an admissible function $v \in$ and proceed to integrate by parts. Doing this and translating into inner products gives:

$$\langle f,v\rangle = \left\langle -u'',v\right\rangle = -\int_0^1 u^n(x)v(x)dx = \underbrace{u'(x)v(x)]_{x=1}^{x=0}}_{=0 \text{ by (BC)}} + \int_0^1 u'(x)v'(x)dx = \left\langle u',v'\right\rangle$$

It follows that $u(x)$ solves the (PVW), as asserted.

Up to this point we have rigorously shown the following implications for solutions of the various three problems:

$$(BVP) \Rightarrow (PVW) \Leftrightarrow (MPE).$$

We will next show that the solutions of (PVW) are unique. From this and what was already proved, it will follow that all three problems have unique solutions.

Step 4: We prove that any two solutions u_1 and u_2, both belonging to $\mathscr{A}$, of the problem (PVW) must be identical. Thus we are assuming that $\langle u_i',v'\rangle = \langle f,v\rangle$ for all $v \in \mathscr{A}(\mathrm{i} = 1,2)$. Our task is to show $u_1 = u_2$. If we use $v = u_1 - u_2 \in \mathscr{A}$, we obtain that: $\left\langle u_1', [u_1-u_2]'\right\rangle = \langle f, u_1-u_2\rangle$ and $\left\langle u_2', [u_1-u_2]'\right\rangle = \langle f, u_1-u_2\rangle$, Subtracting and using linearity gives us: $\left\langle [u_1-u_2]', [u_1-u_2]'\right\rangle = 0$, which translates to $\int_0^1 (u_1'(x) - u_2'(x))^2 dx = 0$. Since the integrand is nonnegative and piecewise continuous, it follows that it must equal zero everywhere on $[0,1]$ except, possibly, at the endpoints of the intervals making up its pieces. We have used positive definiteness of the inner product here. The same is therefore true for $u_1' - u_2' = [u_1-u_2]'$, so it follows that the antiderivative of this latter function must be a constant. Thus we can write $u_1 - u_2 = C$ or $u_1 = u_2 + C$. But the boundary conditions $u_i(0) = 0$ then force $C = 0$ and we can conclude $u_1 = u_2$, as desired.

Step 5: (Final Step) We show that (PVW) implies (BVP). At this point we invoke the fact, mentioned at the outset of this proof, that the (BVP) has a solution $u(x)$ (existence). From what was already proved, this function $u(x)$ is also a solution of (PVW), but from step 4 , the solution of (PVW) is unique. Consequently, any solution of (PVW) really must be the (unique) solution of (BVP), as required. QED

In order to solve the BVP (24), the above theorem allows us to focus our attention on either of the equivalent problems MPE (30) or PVW (31). The finite element method will use one of these two formulations but will replace the very large space $\mathscr{A}$ of admissible functions by a much smaller (finite-dimensional) space in each of the corresponding governing conditions. We begin by partitioning the interval $(0,1)$ into subintervals: $\mathscr{P}: 0 = x_0 < x_1 < \cdots < x_{n+1} = 1$. We denote these intervals by $I_i = (x_i, x_{i+1})$ $(i = 0,1,2,\cdots,n)$ and their lengths by $h_i = x_{i+1} - x_i$. Unlike with finite difference methods, we do not require that these lengths be equal. We define the **mesh size** $\|\mathscr{P}\|$ of this partition as the maximum of the lengths $\max_{0\le i\le n} h_i$. Corresponding to such a partition $\mathscr{P}$ we define the following space of piecewise linear functions:

$$\mathscr{A}(\mathscr{P}) = \{v : [0,1] \to \mathbb{R} : v(x) \text{ is continuous on } [0,1], \text{linear on each } I_i \text{ and } v(0) = 0, v(1) = 0\}.$$

A typical function in this space is depicted in Figure 10.10.

EXERCISE FOR THE READER 1.5.

Show that the space $A(\mathscr{P})$ is closed under the operations of addition of functions and scalar multiplication. More precisely, if $v, w \in \mathscr{A}(\mathscr{P})$ and α is any real number, show that the functions $v + w$, and αv also belong to $\mathscr{A}(\mathscr{P})$.

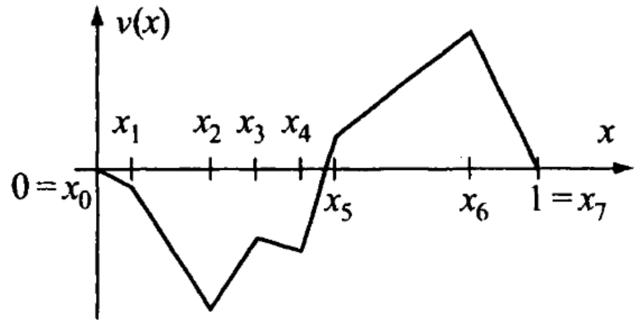


Figure 1.8: llustration of a typical function in the space $\mathscr{A}(\mathscr{P})$.

Notice that a function $v \in \mathscr{A}(\mathscr{P})$ is entirely determined by its values at the interior grid points: $v(x_1), v(x_2), \cdots, v(x_n)$. This follows from linearity and continuity. We need a set of basis functions that can be used to easily describe functions in $\mathscr{A}(\mathscr{P})$. These n functions are usually chosen so that each one equals zero on most of the interval $[0,1]$, so that it will have minimum interaction with other basis functions. [7] One simple set of basis functions meeting this criterion are the so-called **hat functions** $\phi_i(x)(1 \le i \le n)$. Each hat function $\phi_i(x)$ is that member of $\mathscr{A}(\mathscr{P})$ determined by the assignments: $\phi_i(x_i) = 1$ and $\phi_i(x_j) = 0$ for any index $j \ne i$. A typical hat function is shown in Figure 10.11.

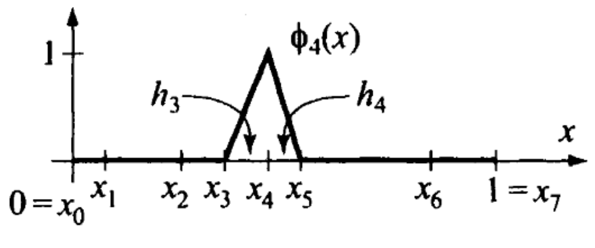


Figure 1.9: A typical hat function for a certain partition of $(0,1)$. Note the (possible) asymmetry.

We observe that any function $v \in \mathscr{A}(\mathscr{P})$ can be expressed in a unique way as a linear combination of the hat functions:

$$v(x) = \sum_{i=1}^{n} v(x_i)\,\phi_i(x). \tag{1.19}$$

(To prove this, just check that both functions agree at each partition point x_i, and then it will follow that they are always equal since both are piecewise linear.) In the language of linear algebra, we say that the n hat functions form a basis for the n-dimensional space $\mathscr{A}(\mathscr{P})$.

The equations of the hat functions are as follows:

$$\phi_i(x) = \begin{cases} 0, & \text{if } 0 \le x \le x_{i-1} \text{ or } x_{i+1} \le x \le 1, \\ \frac{x - x_{i-1}}{h_{i-1}}, & \text{if } x_{i-1} \le x \le x_i, \\ \frac{x_{i+1} - x}{h_i}, & \text{if } x_i \le x \le x_{i+1}. \end{cases} \tag{1.20}$$

[7]General Rayleigh-Ritz methods result from using any set of linearly independent functions which are continuous, piecewise differentiable and satisfy the required boundary conditions as a set of "basis functions."

The **Rayleigh-Ritz method** for approximating the BVP (24) is to solve the following finite-dimensional version (discretization) of it:

$$\text{Find } u \in \mathscr{A}(\mathscr{P}) \text{ satisfying } F(u) \le F(v) \text{ for all } v \in \mathscr{A}(\mathscr{P}). \tag{1.21}$$

Note that the Rayleigh-Ritz problem (34) is obtained by the corresponding (MPE) problem (30) simply by replacing $\mathscr{A}$ by $\mathscr{A}(\mathscr{P})$. We will proceed now to discuss the special Rayleigh-Ritz method for our BVP (2) using the hat functions $\phi_i(x)$ of (33). Different basis functions and, more generally, different finite dimensional spaces give rise to different versions of the Rayleigh-Ritz method. Implementations using such hat functions are often referred to as the **(piecewise) linear Rayleigh-Ritz method**. Since, as in (32), any function in $\mathscr{A}(\mathscr{P})$ can be written as $\sum c_i\phi_i$, making use of bilinearity, we may write:

$$\begin{aligned} F\left(\sum c_i\phi_i\right) &= \frac{1}{2}\left\langle \left[\sum c_i\phi_i\right]', \left[\sum c_i\phi_i\right]'\right\rangle - \left\langle f, \sum c_i\phi_i\right\rangle \\ &= \frac{1}{2}\left\langle \sum c_i\phi_i', \sum c_j\phi_j'\right\rangle - \sum c_i \langle f, \phi_i\rangle \\ &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} c_ic_j\left\langle \phi_i', \phi_j'\right\rangle - \sum c_i\langle f,\phi_i\rangle \end{aligned} \tag{1.22}$$

The above expression can be viewed as a (quadratic) function of the variable $(c_1, c_2, \cdots, c_n) \in \mathbb{R}^n$. We can locate its minimum by setting each of the partial derivatives equal to zero. Using the product rule, we can compute as follows:

$$\frac{\partial}{\partial c_k}F\left(\sum c_i\phi_i\right) = 0 \Rightarrow \frac{1}{2}\sum_{j=1}^{n} c_j\left\langle \phi_k', \phi_j'\right\rangle + \frac{1}{2}\sum_{i=1}^{n} c_i\left\langle \phi_i', \phi_k'\right\rangle = \langle f, \phi_k\rangle \, (1 \le k \le n).$$

Now using symmetry of the inner product, we can combine the two summations on the left into one:

$$\sum_{j=1}^{n}\left\langle \phi_k', \phi_j'\right\rangle c_j = \langle f, \phi_k\rangle \, (1 \le k \le n) \tag{1.23}$$

We abbreviate this linear system as

$$Ac = b, \tag{1.24}$$

where $A = [a_{ij}] = \left[\left\langle \phi_i', \phi_j'\right\rangle\right]$ is the so-called $(n \times n)$ **stiffness matrix**, and b is the so called $(n \times 1)$ **load vector**: $[b_j] = \left[\langle f, \phi_j\rangle\right]$. The terminology comes from the model of (24) for an elastic beam.

To compute the entries of the stiffness matrix: $\left\langle \phi_i', \phi_j'\right\rangle = \int_0^1 \phi_i'(x)\phi_j'(x)dx$, we first observe that, from the properties of the hat functions, $\phi_i'(x)\phi_j'(x) = 0$ unless i and j are equal or are adjacent indices. Thus, the stiffness matrix is both symmetric and tridiagonal. To compute the nonzero entries of A, there are just two cases. We use (33) for the computations:

$$\left\langle \phi_i', \phi_i'\right\rangle = \int_{x_{i-1}}^{x_{i+1}} \left[\phi_i'(x)\right]^2 dx = \int_{x_{i-1}}^{x_i} [1/h_{i-1}]^2 dx + \int_{x_i}^{x_{i+1}} [1/h_i]^2 dx = \frac{1}{h_{i-1}} + \frac{1}{h_i}, \tag{1.25}$$

$$\left\langle \phi_i', \phi_{i+1}'\right\rangle = \int_{x_i}^{x_{i+1}} \phi_i'(x)\phi_{i+1}'(x)dx = \int_{x_i}^{x_{i+1}} \left[\frac{-1}{h_i}\right]\left[\frac{1}{h_i}\right] dx = \frac{-1}{h_i}. \tag{1.26}$$

EXERCISE FOR THE READER 1.6.

(a) Show that the stiffness matrix A is positive definite (i.e., show that for any $n \times 1$ vector c, we have $c\prime Ac \ge 0$ with equality if and only if c is the zero vector). [8]

(b) Show that in case all grid spaces are equal, (i.e., $h_i = \|\|\mathscr{P}\|\|$ for all i), the stiffness matrix for linear system of the linear Rayleigh-Ritz (FEM) is a constant multiple of the coefficient matrix for the finite difference method introduced in Section 10.3. How do the linear systems compare?

As a general rule for Rayleigh-Ritz methods (and finite element methods for PDEs), it is usually a good idea to place more nodes where the (known) coefficient functions in the problem undergo the most activity. Adaptive methods can be developed in which successive refinements are used to see where to place additional nodes. We are ready to give a numerical example of the Rayleigh-Ritz method. In order to be able to get a check on errors, the following theorem will be useful:

THEOREM 1.3. *(Error Estimate for Rayleigh-Ritz Approximations)*

[8]Some general facts about positive definite matrices are that they are nonsingular and, if symmetric, their eigenvalues are all positive. The latter is, in fact, an equivalent definition (see, e.g., Section 8.4 of [HoKu-71] for proofs and more information on positive definite matrices). In particular, the stiffness matrix is nonsingular, so the Rayleigh-Ritz method leads to a unique solution.

Let $u_{\mathscr{P}}(x)$ denote the (piecewise) linear Rayleigh-Ritz approximation corresponding to a partition $\mathscr{P}$ of $[0,1]$ of the BVP (24) : $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$, where $f(x)$ is a continuous function. The following error estimate holds for each $x, 0 \le x \le 1$:

$$|u_{\mathscr{P}}(x) - u(x)| \le \frac{\|\mathscr{P}\|^2}{2} \max_{0 \le x \le 1} |f(x)| \tag{1.27}$$

The proof of this theorem involves some nice ideas from analysis; an outline is left to Exercises 18-21 (see also the note preceding Exercise 17). In fact, in this setting it is even true that $u_{\mathscr{P}}(x_i) = u(x_i)$ at each grid point and thus $u_{\mathscr{T}}$ is really the piecewise linear interpolant of u with respect to the partition $\mathscr{T}$; see Exercise 21.

Example 1.4. Consider the (BVP) (24) $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$ with [9]

$$f(x) = \sin\left[\operatorname{sign}(x - .5)\exp\left(\frac{1}{4|x-.5|^{1.05} + .3}\right)\right] \cdot$$
$$\exp\left(\frac{1}{4|x-.5|^{1.2} + .2} - 100(x - .5)^2\right).$$

■

(a) Use the Rayleigh-Ritz method with $n = 50$ equally spaced interior grid values to solve this BVP and plot the resulting approximation.

(b) Solve the problem again with the Rayleigh-Ritz method and $n = 50$ interior grid values, but this time deploy a higher concentration of grid points where the inhomogeneity $f(x)$ is more oscillatory.

(c) Use Theorem 10.6 to find a (uniform) grid size that will guarantee that the Rayleigh-Ritz solution will be visually (without zooms) identical to the exact solution and compare both solutions of (a) and (b) with this more accurate solution.

SOLUTION: Since the BVP (24) is rather specialized, we will not bother writing here an M-file to perform the Rayleigh-Ritz method. Instead, we will go through each part directly, using MATLAB whenever convenient.

Part (a): Here we have $h_i = \|50\mathscr{P}\| = 1/51$ for each i, so that from the calculations above, the stiffness matrix is given by:

$$A = 51\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & 0 & \\ & -1 & 2 & -1 & \\ & 0 & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{bmatrix}$$

The entries of the load vector can be computed using MATLAB's integrator `quad`. The resulting system is then stored and solved using the Thomas algorithm. Note that in the case of equal grid spaces, the hat functions become symmetric and formula (33) for them can be abbreviated as (we set $h = \|\mathscr{P}\|$)

$$\phi_i(x) = \begin{cases} \frac{h-|x-x_i|}{h} = 1 - \frac{|x-x_i|}{h}, & \text{if } x_{i-1} \le x \le x_{i+1}, \\ 0, & \text{otherwise.} \end{cases}$$

(Verify this!) As the integrals required for the load vector entries are related, a loop will be used to compute them. The integrals will depend on a parameter. One way to compute such parameter-dependent integrals is to declare the parameter variables as global variables.

global var →	Inside the definition of a function M-file having var as a variable, this command declares this variable to be a global variable. Recall that by default, all variables appearing in an M-file are local variables. Should also be used in the command window before invoking such an M-file.

The use of this strategy is demonstrated in the remainder of this example.

The coefficients of the load vector are given by: $(1 \le j \le 50)$

$$b_i = \langle f, \phi_i \rangle = \int_{x_{i-1}}^{x_{i+1}} f(x)\phi_i(x)dx = \int_{x_{i-1}}^{x_{i+1}} \left[1 - \frac{|x-x_i|}{h}\right] f(x)dx$$
$$= \int_{x_{i+1}} [1 - 51\,|x - x_i|]\, f(x)dx.$$

[9]We use the notation of the "sign function" (whose MATLAB counterpart has the same name): sign(x) $= 1$, if $x > 0, 0$, if $x = 0$, and -1, if $x < 0$.

The integrands depend on the parameter x_i, so we will first create an M-file for them using xi, which we declare as a global variable, to represent x_i [10].

```
function y = frayritz10_7(x)
global xi;
y=(1-51*abs (x-xi)). *sin(sign(x-.5).*exp(1./(4*abs(x-.5).^1.05+.3)))...
    .*exp(1./(4*abs(x-.5).^1.2+.2)-100*(x-.5).^2);
```

The load coefficients can now be created as follows: First declare our global variable and create the vector x of grid points. We remind the reader that vector indices must be positive integers so $x(1)$ represents x_0 and so on.

```
>> global xi;
>> for i=1:52
  x(i)=(i-1)/51;
end
```

With our M-file, the load coefficients are now easily created with the following loop. Notice that we have used `quadl` rather than `quad`. This former integrator works in the same syntax as `quad`, but uses a refined adaptive technique. It takes a bit more time to use but gives more accurate results.

```
>> for i=2:51
  xi=x(i);
  b(i)=quadl('frayritz10_7',x(i-1),x(i+1));
end
```

We have kept the indices consistent with those of the vector JC, but consequently we have created a vector with one extra component $b(1) = 0$. This component must be left out when we go on to solve the linear system. In order to solve the linear system, we will use the `thomas` M-file, which will solve our tridiagonal system quite efficiently. We must create the appropriate vectors to meet the syntax of this M-file:

```
>> d=2*ones(50,1)*51; %diagonal of stiffness matrix A
>> da=-1*ones(50,1)*51; da(51)=0; %superdiagonal (above)
>> db=-1*ones(50,1)*51; db(1)=0; %subdiagonal (below)
>> c=thomas(da,d,db,b(2:51));
```

As explained earlier, the values of the solution vector c are precisely the values of the numerical Rayleigh-Ritz solution at the interior grid points $x_1, x_2, \cdots, x_{50} = (x(24), x(25), \ldots, x(51))$. To plot the entire graph of c versus x, we need to augment the vector c to have first and last components which equal zero (from the boundary conditions). With this being done below, the resulting numerical plot is shown in Figure 10.12

```
>> c = [0 c 0];
>> plot(x,c,'b-o')
```

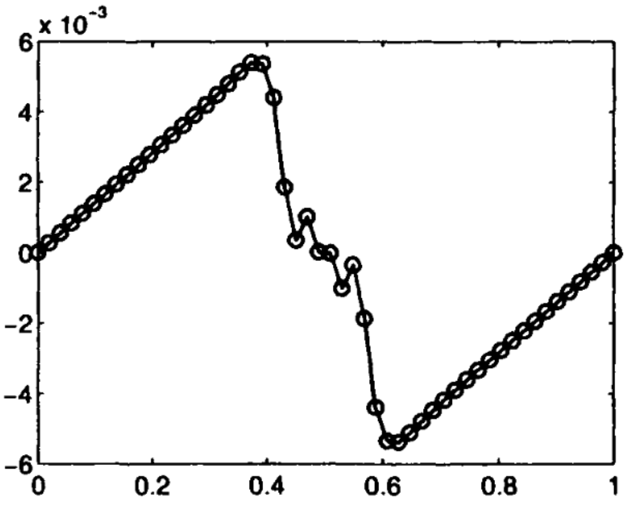


Figure 1.10: Rayleigh-Ritz solution of the BVP in Example 10.7 using 50 equally spaced interior grid points. The grid points/values are shown with (blue) circles.

Part (b): The right-hand side of the DE $-u''(x) = f(x)$ has the graph shown in Figure 10.13.

[10] A syntax note: If, after creating and storing this M-file we were to enter `frayritz10_7(24)`, the output would be "[]" (the empty vector), a reasonable answer since we have not yet defined `xi`. If we first entered a value for `xi`, say `xi=2` and reentered the above command, however, we would still get the empty vector as output. It is essential to first declare `xi` as a global variable in the command window (even though this was already done in the M-file). If this is done, and `xi=2` is reentered, then entering `frayritz10_7(24)` would finally produce an answer (ans =4.7321)).

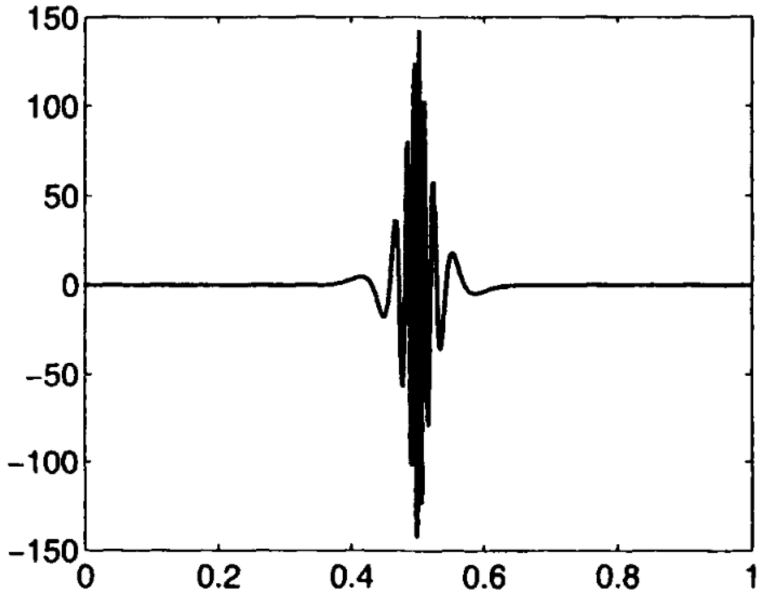


Figure 1.11: Graph of the right-hand side $f(x)$ of the DE $-u''(x) = f(x)$ of Example 10.7.

From Figure 10.13, we see that the inhomogeneity $f(x)$ is most oscillatory approximately on the interval $[0.35, 0.65]$ and elsewhere is rather tame. With this perspective, it would seem that any grid that is uniformly highly dense would give rise to much wasted computation on the long intervals of inactivity. Motivated by Figure 10.13, we propose the following deployment of the 50 interior grid points.

Put 6 in each of the intervals $[0, 0.35)$ and $(0.65, 1]$, and put the remaining 40 in $[0.35, 0.65]$. We stipulate that the grid points in each of these intervals be uniformly spaced but this is by no means necessary (the Rayleigh-Ritz method is totally flexible). The linspace command will make the construction of these grid values particularly straightforward:

```
>> x2(1:7)=linspace(0,0.35,7);
>> x2 (7:46)=linspace(0.35,0.65,40);
>> x2(46:52)=linspace(0.65,1,7);
```

Since the grid is no longer uniform, we need to construct a vector for the h_i:

```
>> for i=1:51, h(i)=x2(i+1)-x2(i); end
```

It is left to construct the load vector b. By (33) the coefficients are

$$b_i = \langle f, \phi_j \rangle = \int_{x_{i-1}}^{x_{i+1}} f(x)\phi_i(x)dx = \int_{x_{i-1}}^{x_i} \frac{x - x_{i-1}}{h_{i-1}} f(x)dx + \int_{x_i}^{x_{i+1}} \frac{x_{i+1} - x}{h_i} f(x)dx$$

Employing the strategy used in part (a), we need here a pair of M-files for the two respective integrands:

```
function y = frayritz10_7a(x)
global xim;
global him;
y=(x-xim)./him.*sin(sign(x-.5).*exp(1./(4*abs(x-... .5).^1.05+.3))).*exp(
   1./(4*abs(x-.5).^1.2+.2)-100*(x-.5).^2);
function y = frayritz10_7b(x)
global xip;
global hi;
y=(xip-x),/hi.*sin(sign(x-.5).*exp(1./(4*abs(x-... .5) .^1.05+.3))).*exp(
   1./(4*abs(x-.5).^1.2+.2)-100*(x-.5).^2);
```

The load vector is now easily constructed, and the linear tridiagonal system can be assembled and solved as before:

```
>> global xim him xip hi;
>> for i=2:51;
   xip=x2(i+1); xim=x2(i-1); hi=h(i); him=h(i-1);
   b2(i)=quadl('frayritz18_1a', x(i-1), x(i))+... quadl('frayritz18_1b f ,
      x(i), x(i+1));
end
>> for i=1:51, h(i)=x(i+1)-x(i); end
>> for i=2:51, d2(i)=1/h(i-1)+1/h(i); end
>> %main diagonal will be d(Â£:51).
>> for i=2:50, da2(i)=-1/h(i); end
>> da2(51)=0; %superdiagonal will be da(2:51).
>> for i=2:50, db2(i)=-1 h(i-1); end
>> %subdiagonal will be db(1:50)
>> c2=thomas(da2(2:51),d2(2:51),db2(1:50),b2(2:51));
```

The commands needed to plot this solution are just as in part (a), and those commands produce the plot shown in Figure 10.14(a). Part (c): From Figure 10.12, we see that the amplitude of the solution is roughly 6e-3. Theorem 10.6 gives maximum bound for the error to be $\frac{\|\mathscr{P}\|^2}{2}\max_{0\le x\le 1}|f(x)|$. Setting this expression to be $6e-3/100$ (so the maximum error will be less than about $1/100$ of the amplitude), using 150 for $\max_{0\le x\le 1}|f(x)|$ (from Figure 10.13), and solving for $\|\mathscr{P}\|$ gives roughly $1e-4$, so that if we use 10,000 interior grid points, the Rayleigh-Ritz solution should have the desired accuracy. The construction and plotting of this solution is done just as in part (a), except that instances of 50 or 51 , etc. should be changed to 10,000 or 10,001 , etc. The resulting graph is compared with the two obtained in parts (a) and (b) in Figure 10.14.

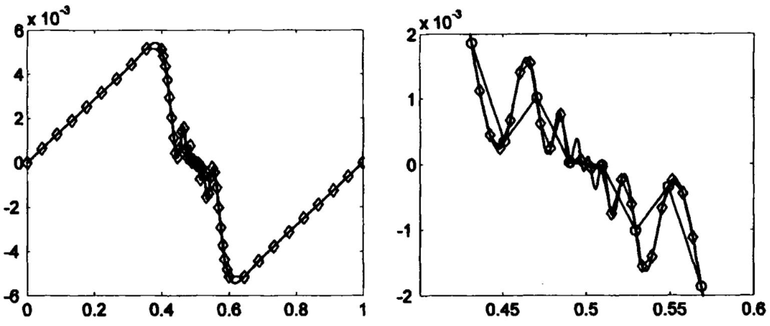


Figure 1.12: (a) (left) Rayleigh-Ritz solution obtained for Example 10.7(b) shown with diamonds, along with the exact solution in black. The grid used is nonuniform with more grid points (diamonds) deployed in the areas where the inhomogeneity is most active, (b) (right) Zoomed-in comparison of the Rayleigh-Ritz solutions in part (a) (circles) and part (b) (diamonds) with the exact solution (smooth curve) of Example 10.7. Note the surprising fact that the Rayleigh-Ritz solutions are exactly equal to the solution at the respective grid points, and hence the Rayleigh-Ritz solutions turn out simply to be the piecewise linear interpolants of the actual solution with respect to the associated grids (see Exercise 21 for a proof). This theorem will no longer hold in higher dimensions or even for more complicated single-variable BVPs.

We now turn to the **Galerkin method** [11] for approximating the solution of the BVP (24). In the piecewise linear setting with the finite-dimensional space $\mathscr{A}(\mathscr{P})$ in place of the space $\mathscr{A}$ of all admissible functions, this method solves the discrete analogue of the Principle of Virtual Work (31):

$$\text{Find} u \in \mathscr{A}(\mathscr{P}) \text{satisfying} \langle u', v' \rangle = \langle f, v \rangle \text{for all} v \in \mathscr{A}(\mathscr{P}). \tag{1.28}$$

In light of the bilinearity of the inner product, it is enough to check (41) for the function v running through the n (basis) functions: $\{\phi_k\}_{k=1}^n$. The discrete problem is thus to determine the coefficients $(c_1, c_2, \cdots, c_n) \in \mathbb{R}^n$ of the function $u = \sum_{j=1}^n c_j\phi_j \in \mathscr{A}(\mathscr{P})$ such that $\left\langle (\sum c_i\phi_i)', \phi_k' \right\rangle = \langle f, \phi_k \rangle$ $(1 \le k \le n)$.

Using bilinearity, these equations become $\sum_{i=1}^n c_j \left\langle \phi_j', \phi_k' \right\rangle = \langle f, \phi_k \rangle$, which is precisely (36). Thus, for the (BVP) (24), the Rayleigh-Ritz and Galerkin methods coincide, and this is true for any choice of basis functions (not necessarily the piecewise linear basis functions). The next exercise for the reader will use a set of basis functions that does not depend on any particular partition, but rather comes from the so-called eigenfunctions of the BVP.[12]

EXERCISE FOR THE READER 1.7.

Apply the Galerkin method to re-solve the BVP of Example 10.7 using the following 50 basis functions: $\varphi_k(x) = \sin(k\pi x), k = 1, 2, \cdots, 50$. Compare the accuracy with that obtained in part (a) of Example 10.7.

[11]Like the works of Rayleigh and Ritz, the work of Russian engineer/mathematician Boris Grigorievich Galerkin (1871-1945) was motivated by physical problems. It is fair to characterize Galerkin as an applied mathematician of the purest sense. He worked many years as an engineer before his first publication at the relatively late age of 38 on longitudinal curvature. The paper was a significant extension of work of Euler and it was applied in the construction of bridges and building frames. His continued interest in structural mechanics led him to the discovery in 1915 of his most notable contribution to mathematics, what is known today as the Galerkin method. He subsequently took on some academic posts in St. Petersburg, which was the de facto mathematical capital of Russia at the time. His interests in consulting with industry and in the relevant mathematics continued until his death. In 1937 he published a pivotal treatise on thin elastic plates.

[12]For the BVP $-u'' = f(x), u(0) = u(1) = 0$, the associated **eigenfunctions** are nontrivial solutions of the BVP $-u'' = \lambda u, u(0) = u(1) = 0$ for some $\lambda > 0$. It can be shown that the totality of these eigenfunctions is as follows: $u_k(x) = \sin(k\pi x), k = 1, 2, \cdots$ (see Exercise 24). The eigenfunctions are pairwise orthogonal: $\langle u_k, u_\ell \rangle = \delta_{k\ell}/2$ (where $\delta_{k\ell}$ denotes the Kronecker delta equaling 1 if the indices are equal, otherwise equaling 0), as are their derivatives (Exercise 24). Moreover, the eigenfunctions have the remarkable property that any function u satisfying the same boundary conditions and satisfying reasonable regularity assumptions (say if $u \in \mathscr{A}$) can be expressed as an infinite series of these eigenfunctions: $u(x) = \sum_{k=1}^\infty c_k u_k(x)$. In particular, solutions of such inhomogeneous BVPs have such eigenfunction expansions. Such eigenfunction expansion theory of ODE BVPs falls under the name of Sturm-Liouville theory. The analog for PDE BVPs is the theory of Fourier series. Both of these analytical techniques are covered extensively in many theoretically or analytically oriented textbooks. For references we cite [Str-92] and [Sni-99]. All of these properties make finite subsets of these eigenfunctions seem like very reasonable candidates for Rayleigh-Ritz and Galerkin methods; these types of Rayleigh-Ritz methods are often referred to as spectral methods.

For general BVPs, the Rayleigh-Ritz and Galerkin methods often, but not always, coincide. For this reason the nomenclature sometimes refers to the "RayleighRitz-Galerkin method." Both methods have been developed for a great many BVPs. The formulation of the Rayleigh-Ritz method in general is a bit more involved since it entails the determination of the appropriate functional for the analogue of Theorem 10.5 to be valid. Such problems usually fall under the classical area of the *calculus of variations*. We now present a brief outline for the Rayleigh-Ritz method for solving the following more general BVP whose DE will be a prototype for the elliptic PDE problems we shall contemplate Chapter 13. All of what follows in this outline can backed up theoretically with techniques similar to those used earlier to deal with the simpler problem (24); some of the details will be left to the exercises.

$$\text{(BVP)}\begin{cases} -(p(x)u'(x))' + q(x)u(x) = f(x), & 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases} \tag{1.29}$$

Different boundary conditions and more general equations can be dealt with using modified functionals. The next exercise for the reader, however, shows how BVPs with general Dirichlet BC's can be reduced to (42) using a simple change of variables. The exercises will examine some further modifications.

EXERCISE FOR THE READER 1.8.

(a) Show that the following BVP,

$$\text{(BVP)}\begin{cases} -(p(x)w'(x))' + q(x)w(x) = f(x), & 0 < x < 1 \\ w(0) = \alpha, w(1) = \beta \end{cases} \tag{1.30a}$$

can be reduced to the form (42) by making the following change of variables/ function:

$$u(x) = w(x) - (1-x)\alpha - \beta x$$

(b) Show that the following BVP,

$$\text{(BVP)}\begin{cases} -(p(t)w'(t))' + q(t)w(t) = f(t), a < t < b \\ w(a) = \alpha, w(b) = \beta \end{cases} \tag{1.30b}$$

can be reduced to $(42a)$ by making the following change of variables/function:

$$x = (t-a)/(b-a).$$

The analogue for Theorem 10.5 (for the Rayleigh-Ritz formulation) is the following theorem whose complete proof can be found in Section 7.2 in [Sch-73].

THEOREM 1.4. *(Rayleigh-Ritz Principle for a One-Dimensional BVP*

In the BVP (42):

$$\begin{cases} -(p(x)u'(x))' + q(x)u(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$$

suppose that the (known) functions $p(x), q(x)$ and $f(x)$ are continuous and additionally that $p(x)$ is differentiable on the open interval $I = [0,1]$ [13]. Also assume that $p(x) > 0$ and $q(x) \geq 0$ throughout I. Under these assumptions, the BVP has a unique solution which coincides with the unique minimizer of the functional

$$F(v) = \int_0^1 \left[p(x)\left(v'(x)\right)^2 + q(x)(v(x))^2 - 2f(x)v(x)\right] dx, \tag{1.31}$$

over the set of admissible functions

$\mathscr{A} = \{v : [0,1] \to \mathbb{R} : v(x)$ is continuous, $v'(x)$ is piecewise continuous and bounded, and $v(0) = 0, v(1) = 0\}$. We remind the reader that without these hypotheses, the BVP (42), in general, may have no solutionâĂŤsee Exercise 12 of Section 10.2 or Exercise 24 of this section.

If we use spaces $\mathscr{A}(\mathscr{P})$ of admissible functions spanned by the hat-functions determined by a partition $\mathscr{P}$ of $[0,1]$, the Rayleigh-Ritz method seeks to minimize the functional F evaluated at a typical element $v(x) = \sum_{i=1}^n c_i\phi_i(x)$ of $\mathscr{A}(\mathscr{D})$ (see (32)). A similar computation to what was given above (Exercise 12) will show that if we substitute this function into (43) and set each of the partial derivatives (with respect to the parameters $c_i (1 \leq i \leq n)$) equal to zero, we obtain the $n \times n$ linear system

$$Ac = b,$$

[13] Actually, the theorem still works under weaker conditions stated in [Sch-73]. The most natural setting for the Rayleigh-Ritz method is in the context of Sobolev functions. This topic is rather advanced, however, so we fix our ideas on the classical formulation. The interested reader may also consult the references [StFi-73] and [AxBa-84] for more sophisticated treatments on the subject.

where the $n \times n$ **stiffness matrix** $A = [a_{ij}]$ has coefficients given by:

$$a_{ij} = \int_0^1 \left[p(x)\phi_i'(x)\phi_j'(x) + q(x)\phi_i(x)\phi_j(x)\right] dx \tag{1.32}$$

and the $n \times 1$ **load vector** b has entries given by:

$$b_j = \int_0^1 [f(x)\phi_j(x)]\, dx \tag{1.33}$$

As before, the stiffness matrix is clearly a tridiagonal symmetric matrix that can be shown to be positive definite. Thus there will be a unique solution of the linear system and so the method will always produce Rayleigh-Ritz approximations. There is also an error estimate analogous to Theorem 10.6 which states roughly that $\left|u_g(x) - u(x)\right| \leq C\,\|\mathscr{P}\|^2 \max_{0\leq x\leq 1}|f(x)|$. Thus, we get the same type of error estimate (proportional to $\|\mathscr{P}\|^2$) as we had in the simpler introductory BVP. The proportionality constant C will of course depend on the data $p(x)$ and $q(x)$, but not on $u(x)$ or $f(x)$ (see [StFi-73] for details). The tridiagonal coefficients of the stiffness matrix in (44) can be simplified, using (33), as was done previously, to obtain (cf. with (38), (39)):

$$\begin{aligned} a_{ii} &= \frac{1}{h_{i-1}^2}\int_{x_{i-1}}^{x_i} p(x)dx + \frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} p(x)dx \\ &+ \frac{1}{h_{i-1}^2}\int_{x_{i-1}}^{x_i} (x-x_{i-1})^2 q(x)dx + \frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 q(x)dx, \text{ for } 1 \leq i \leq n \end{aligned} \tag{1.34}$$

$$a_{i,i+1} = \frac{-1}{h_i^2}\int_{x_i}^{x_{i+1}} p(x)dx + \frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)(x-x_i)\,q(x)dx, \quad \text{for } 1 \leq i < n \tag{1.35}$$

Evaluation of the integrals in (44) through (47) can be a time-consuming process in cases of fine partitions. In such cases where the coefficient functions p, q, and f are not too wildly behaved, it is a good idea to replace each of these functions by their piecewise linear approximation (piecewise linear splines) in the integrals. By Exercise 13(b), the local errors of such approximations are $O\left(h_i^2\right)$ on each of the corresponding intervals, provided that the function is $\mathscr{C}^2$, and this in turn implies $O\left(h_i^3\right)$ estimates for each of the integrals. We do one such approximation for the fourth integral in (46); the rest are done in a similar fashion and are left as Exercise 13(a). The piecewise linear approximation $Q(x)$ to $q(x)$ relative to the partition $\mathscr{P}$ of $[0,1]$ can be expressed quite simply using the hat functions as follows:

$$Q(x) = q(x_i)\,\phi_i(x) + q(x_{i+1})\,\phi_{i+1}(x), \quad x \in [x_i, x_{i+1}].$$

Replacing this approximation for $q(x)$ in the last integral of (46) leads us to the following estimate:

$$\begin{aligned} &\frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 q(x)dx \\ &\qquad\approx \frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 \left[q(x_i)\,\phi_i(x) + q(x_{i+1})\,\phi_{i+1}(x)\right] dx \\ &\qquad= \frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 \left[q(x_i)\,\frac{x_{i+1}-x}{h_i} + q(x_{i+1})\,\frac{x-x_i}{h_i}\right] dx \\ &\qquad= \frac{q(x_i)}{h_i^3}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^3\, dx + \frac{q(x_{i+1})}{h_i^3}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 (x-x_i)\, dx. \end{aligned}$$

The two latter integrals are easily evaluated explicitly, for example:

$$\begin{aligned} \int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 (x-x_i)\, dx \underset{\substack{\text{Subst.} \\ u=x_{i+1}-x}}{=} \int_{h_i}^{0} u^2 (h_i-u)(-du) &= \int_{h_i}^{0} \left[u^3 - h_i u^2\right] du \\ &= \left[\frac{u^4}{4} - h_i\frac{u^3}{3}\right]_{h_i}^{0} = \frac{h_i^4}{12}. \end{aligned}$$

In a similar fashion we find that $\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^3\, dx = \frac{h_i^4}{4}$. Putting these into the above estimate gives us that:

$$\frac{1}{h_i^2}\int_{x_i}^{x_{i+1}} (x_{i+1}-x)^2 q(x)dx \approx \frac{h_i}{12}\left[3q(x_i) + q(x_{i+1})\right].$$

Similar treatments for the remaining integrals appearing in (46) through (47) (see Exercise 13(a)) result in the following estimates:

$$\begin{aligned} a_{ii} \approx &\frac{1}{2h_{i-1}}\left[p(x_{i-1}) + p(x_i)\right] + \frac{1}{2h_i}\left[p(x_i) + p(x_{i+1})\right] \\ &+ \frac{h_i}{12}\left[q(x_{i-1}) + 3q(x_i)\right] + \frac{h_i}{12}\left[3q(x_i) + q(x_{i+1})\right] \end{aligned} \tag{1.36}$$

for $1 \le i \le n$, and

$$a_{i,i+1} \approx -\frac{1}{2h_i}\left[p(x_i)+p(x_{i+1})\right]+\frac{h_i}{12}\left[q(x_i)+q(x_{i+1})\right] \tag{1.37}$$

for $1 \le i < n$. In the same fashion, the load vector coefficients are estimated as follows:

$$b_j \approx \frac{h_{j-1}}{6}\left[f(x_{j-1})+2f(x_j)\right]+\frac{h_j}{6}\left[2f(x_j)+f(x_{j+1})\right], \tag{1.38}$$

for $1 \le j \le n$.

Our next example will compare performance speed and accuracy of both of the above implementations of the Rayleigh-Ritz method for a specific BVP. It is possible to solve this BVP explicitly, so we will be able to make accurate estimates for the error. The explicit solution, however, is quite a mess. It can be derived using standard methods in differential equations (undetermined coefficients). To avoid having to even write it down, we use MATLAB's Symbolic Toolbox to compute the explicit solution but suppress its output.

Example 1.5. Consider the following problem:

■

$$\text{(BVP)}\begin{cases} -u''(x)+6u(x)=e^{10x}\cos(12x) & 0<x<1 \\ u(0)=0, u(1)=0 \end{cases}$$

(a) Use the Rayleigh-Ritz method with $n = 500$ equally spaced interior grid values to solve this BVP and plot the resulting approximation. Keep a record of the computing time it takes to determine the load vector and stiffness matrix coefficients.

(b) Repeat part (a), this time invoking the approximations (48) thru (50) for the integrals appearing in the Rayleigh-Ritz method.

(c) Compare both solutions of (a) and (b) with the exact solution as obtained using MATLAB's Symbolic Toolbox.

SOLUTION: The BVP given indeed fits the template of (42) with $p(x) = 1$, $q(x) = 6$, and $f(x) = e^{10x}\cos(12x)$.

Part (a): Here we have $h_i = \|\mathscr{P}\| = 1/501$ for each i, so we must compute the tridiagonal entries of the 501×501 stiffness matrix along with the 501 load coefficients using (45)-(47). The computations are done in a similar fashion to how the load vector coefficients were done in Example 10.7. Of the $1+4+2=7$ integrals appearing in (45) through (47), $1+2+1=4$ of them will need the "global variable" strategy in conjunction with MATLAB's numerical integrator `quadl`. The remaining three integrals have constant integrand ($p(x) = 1$) and so will be done directly. For (45) we use the fact that since the spacing is uniform, we have $\phi_i(x) = 1-|x-x_i|/\|\mathscr{P}\| = 1-501|x-x_i|$ for $x_{i-1} \le x \le x_{i+1}$. Actually, because $p(x)$ and $q(x)$ are constant functions for this problem, the approximations (48) and (49) are indeed exact. Nonetheless, we will proceed to use the quadl integrator for these integrals so as to give a good impression of the extra expense in bringing in a more sophisticated tool. For the global variables x_{i-1}, x_i, x_{i+1} we will use the MATLAB notation: `xim, xi, xip` (p for plus, *m* for minus). The four needed M-files are as follows.

```
function y = frayritz10_8load(x)
global xi;
y=(1-501*abs(xi-x)).*exp(10*x).*cos(12*x);

function y = frayritz10_8stiff1(x)
global xim;
y=(x-xim).^2*6;

function y = frayritz10_8stiff2(x)
global xip;
y=(x-xip).^2*6;

function y = frayritz10_8stiff3(x)
global xi xip;
y=(xip-x).*(x-xi)*6
```

Note that all of the intervals of integration have length $h = \|\mathscr{P}\| = 1/501$, so that each of the integrals in (46) and (47) with integrand p equals (since $p(x) = 1$) $\|\mathscr{P}\| = 1/501$. With these M-files stored, the following loop will use (45) thru (47) to construct the needed coefficients:

```
>> x=linspace(0,1,502); h=1/501; global xi xim xip;
>> tic, for i=2:501
xi=x(i); xim=x(i-1); xip=x(i+1);
b(i)=quadl('frayritz10_8load',xim,xip);
d(i)=2/h+1/h^2*quadl('frayritz10_8stiff1,/xim/xi)...
+1/h^2*quadl('frayritz10_8stiff2', xi, xip)
```

```
%d(2:501) is diagonal of stiffness matrix
da(i)=-1/h+1/h"2*quadl('frayritz10_8stiff3',xi,xip);
%da(2:501) is superdiagonal of stiffness matrix (above),
%once we set da(501)=0 (after loop).
end, toc
```

→ elapsed_time = 4.0360(seconds)

```
>> db(3:501)=da(2:500);
>> db(2)=0; da(501)=0;
>> %db is subdiagonal of stiffness matrix (below)
```

As usual, we needed to properly format the sub/superdiagonals for input into the Thomas algorithm, which we apply next.

```
>> c1=thomas(da(2:501),d(2:501),db(2:501),b(2:501));
c1=[0 c1 0] ;
plot(x,c1)
```

The resulting plot, which as we will see turns out to be visually indistinguishable from that of the exact solution, is shown in Figure 10.15.

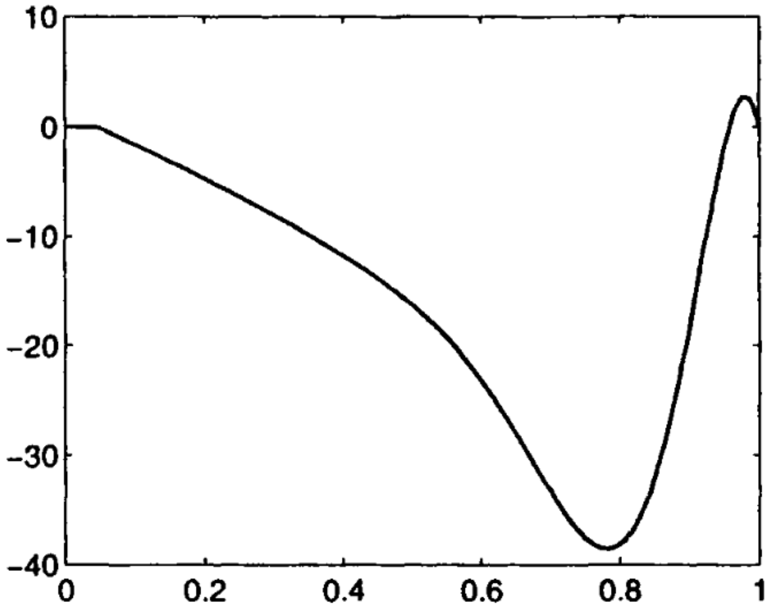


Figure 1.13: Plot of the solution of the BVP of Example 10.8.

Part (b): Using the estimates (48) through (50), it will be quite a simple (and quick) task to collect the needed coefficients. This can be accomplished with the following loop:

```
>> p=ones(502,1); q=6*p; x=linspace(0,1,502); f=exp(10*x).*cos(12*x);
>> h=1/501; %uniform step size
```

```
>> tic, for i=2:501
d(i)=1/(2*h)*(p(i-1)+2*p(i)+p(i+1))+h/12*(q(i-1)+6*q(i)+q(i+1) ;
%d(2:501) is diagonal of stiffness matrix
da(i)=-1/(2*h)*(p(i)+p(i+1))+h/12*(q(i)+q(i+1));
da(501)=0;
%da(2:501) is superdiagonal od stiffnes matrix (above)
db(i)=-1/(2*h)*(p(i-1)+p(i))+h/12*(q(i-1)+q(i));
db(2)=0;
%db(2:501) is subdiagonal of stiffness matrix (below)
b(i)=h/6*(f(i-1)+4*f(i)+f(i+1)) ;
% b(2:501) is load vector
end, toc
```

→ elapsed_time = 0.0500(seconds)

```
>> c2=thomas(da(2:501),d(2:501),db(2:501),b(2:501));
>> c2=[0 c2 0] ;
>> plot(x,c2)
```

The resulting plot is visually indistinguishable from the one in part (a), shown in Figure 10.15.

Part (c): The BVP is rather special in that an explicit solution can be written down. Labeling the symbolic solution as `yexact` and suppressing the long output, we can create it in a MATLAB session (provided the symbolic toolbox or student edition is being used) with the following command.

```
>> yexact=dsolve('-D2y+6*y=exp(10*t)*cos(12*t)', 'y(0)=0', 'y(1)=0');
```

We next create two vectors for the appropriate time values and corresponding values of the exact solution. We will need to use the `double` command along with the subs commands introduced earlier to convert the symbolic numbers to floating point format. The data is then plotted and the result is shown in Figure 10.15.

```
>> t=linspace(0,1,502);
>> Yexact=subs(yexact,t);
>> plot(t,Yexact)
```

With the variables from parts (a) and (b) still remaining in our workspace, we can easily obtain plots of the errors of the numerical solutions in those parts with the following commands. The two plots are shown in Figure 10.16.

```
>> plot(x,abs(c1-Yexact))
>> plot(x,abs(c2-Yexact))
```

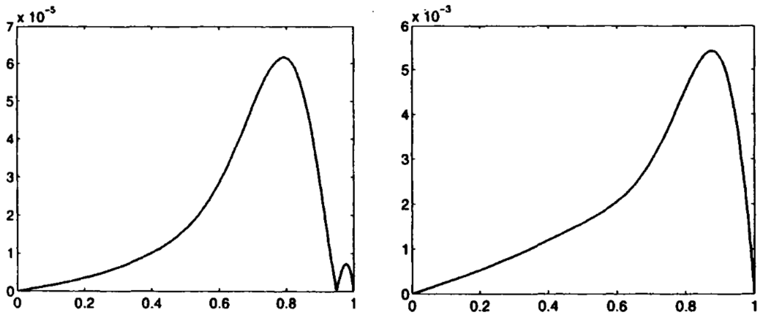


Figure 1.14: Plots for the errors of the two numerical solutions obtained in parts (a) (left) and (b) (right) of Example 10.8.

EXERCISE FOR THE READER 1.9.

(a) Write an M-file called `rayritz` having the following input/output variables: `[x,u]=rayritz(p,q,f,n)`. The program will implement the piecewise linear Rayleigh-Ritz method with (48) through (50) to solve the BVP (42) :

$$\begin{cases} -(p(x)u'(x))' + q(x)u(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$$

The first three inputs `p`, `q`, and `f` can represent the coefficient functions in the DE of (42), and the last input variable `n` denotes the number of interior grid points to use. A uniform grid is assumed. The output variables will be the domain and range vectors for the numerical solution.

(b) Starting with $n = 99$ interior grid points ($h = 1/100$), use this program to get a numerical solution $y1$ of the BVP in Example 10.8, then use 199 grid points ($h = 1/200$), getting a corresponding solution $y2$, and find the maximum absolute difference of the computed solutions on the common domain values. Now cut the gap in half again with $n = 399$, and get a corresponding solution $y3$ and look at the maximum absolute difference with the vector $y2$ at common domain points. Continue this process until the maximum absolute difference is less than 5×10^{-5}. Now (if you have access to the Symbolic Toolbox) compute the actual maximum error of this final solution compared to the exact solution in Example 10.8.

The Rayleigh-Ritz approximations we have obtained were all piecewise continuous but not differentiable. The versatility of the Rayleigh-Ritz method allows us, in fact, to use any sets of linearly independent functions as basis functions. The catch is that the resulting stiffness matrix should be reasonably well conditioned. Some different sets of basis functions will be examined in the exercises; see Exercise 6 for a problematic situation. The hat basis functions we used resulted in numerical approximations that were piecewise continuous but not differentiable. This lack of differentiability can be overcome by the use of more elaborate basis functions. One popular scheme is to use cubic splines for the basis functions; a typical one is shown in Figure 10.17.

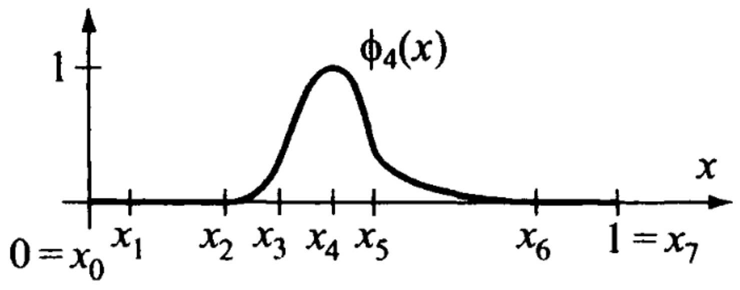


Figure 1.15: A cubic spline basis function. Unlike the piecewise linear hat functions of Figure 10.11, such basis functions are typically nonzero at three node points.

Each cubic spline basis function will have two continuous derivatives and thus so will the numerical approximations (since they are linear combinations of basis functions). The price we will need to pay for this extra smoothness in the numerical solutions is that the resulting stiffness matrix will typically have seven nonzero diagonals, rather than three, and the coefficients will be more complicated to compute. We proceed to outline an implementation of such cubic spline basis functions in the Rayleigh-Ritz method.

We restrict to the case of uniform grids and begin by defining the basic cubic spline function from which all other spline basis functions can be defined. This basic spline, which we denote by $BS(x)$, will be defined using the five nodes: $x_0=-2, x_1=-1, x_2=0, x_3=1$, and $x_4=2$ by the following requirements:

1. On each interval $[x_i,x_{i+1}]$ $(i=0,1,2,3), BS(x)$ will be a polynomial of degree at most three.
2. $BS(x), BS'(x)$, and $BS''(x)$ are each continuous at the node interfaces $x=x_1,x_2,x_3$.
3. $BS(\pm 2)=0, BS(0)=1$ (interpolation requirements).
4. $BS'(x)$, and $BS''(x)$ both equal zero at the endpoint nodes $x=x_0,x_4$.

EXERCISE FOR THE READER 1.10.

(a) Show that the conditions (i) through (iv) above uniquely determine the function $BS(x)$ to be an even function $(BS(-x)=BS(x))$ in $\mathscr{C}^2(\mathbb{R})$ and specified by the following formula:

$$BS(x)=\begin{cases}\left[(2-x)^3-4(1-x)^3\right]/4, & \text{if } x\in[0,1],\\ (2-x)^3/4, & \text{if } x\in(1,2],\\ 0, & \text{if } x>2,\\ BS(-x), & \text{if } x<0.\end{cases} \tag{1.39}$$

(b) Get MATLAB to plot this function. Using the basic spline function $BS(x)$, we can define our basis $\{\phi_i(x)\}_{i=1}^n$ functions for the BVP (42) on $[0,1]$ corresponding to a uniform grid $0=x_0<x_1<\cdots<x_n<x_{n+1}=1$ with mesh size $h=x_{i+1}-x_i=1/(n+1)$. These functions are specified by the formulas below:

$$\phi_i(x)=\begin{cases}BS\left(\frac{x-h}{h}\right)-BS\left(\frac{x+h}{h}\right), & \text{if } i=1\\ BS\left(\frac{x-ih}{h}\right), & \text{if } i=2,3,\cdots,n-1\\ BS\left(\frac{x-nh}{h}\right)-BS\left(\frac{x-(n+2)h}{h}\right), & \text{if } i=n.\end{cases} \tag{1.40}$$

EXERCISE FOR THE READER 1.11.

(a) Show that the basis functions $\{\phi_i(x)\}_{i=1}^n$, as specified in (52), form a linearly independent set of functions. Also, show that on each interval (x_i,x_{i+1}), ϕ_i is a polynomial of degree at most three and that ϕ_i,ϕ_i',ϕ_i'' are continuous at the endpoints x_i,x_{i+1}. Show that $\phi_i(x_i)=1$, $\phi_i(x_j)=0$ if $|i-j|\geq 2$ or $j=0$ if $i=1$, or $j=n+1$ if $i=n$, and $\phi_i(x)=0$ if there is such an x_j that lies between x_i and x.
(b) Using the value $n=5$, get MATLAB to plot each of the five corresponding hat functions.
In order to implement these basis functions in the method, we will need to compute their derivatives. By the chain rule, these can be easily computed in terms of BS'(x), which by simple computation is as specified below:

$$BS'(x)=\begin{cases}\frac{3}{4}\left[4(1-x)^2-(2-x)^2\right], & \text{if } x\in[0,1],\\ -\frac{3}{4}(2-x)^2, & \text{if } x\in(1,2],\\ 0, & \text{if } x>2,\\ -BS'(-x), & \text{if } x<0.\end{cases} \tag{1.41}$$

Each of the "BS(● " expressions in (52) will have, by the chain rule, derivative equal to $BS'(\bullet)/h$. Note also that since $\phi_i(x)$ and $\phi_i'(x)$ equal zero outside the interval $[x_{i-2}, x_{i+2}]$, it follows that the stiffness matrix entries $a_{ij} = \int_0^1 \left[p(x)\phi_i'(x)\phi_j'(x) + q(x)\phi_i(x)\phi_j(x)\right] dx$ (from (44)) will be zero if $|i-j| > 3$, and from this it follows that the stiffness matrix will be a banded matrix with (at most) seven bands. With these observations, it is a simple matter to incorporate the cubic spline Rayleigh-Ritz method into a MATLAB program. Examples will be left to the exercises. We close this section with a result on errors of the cubic spline Rayleigh-Ritz method, which shows it is often worth the extra work required over the basic piecewise linear scheme.

THEOREM 1.5. *(Errors in Cubic Spline Rayleigh-Ritz Approximations)*

Suppose that the exact solution of the BVP (42)

$$\begin{cases} -(p(x)u'(x))' + q(x)u(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$$

is $\mathscr{C}^4([0,1])$ and the data $p(x), q(x)$, and $f(x)$ satisfy the assumptions of Theorem 10.7. If $u_{\mathscr{P}}$ is the cubic spline Rayleigh-Ritz approximation for this problem corresponding to a partition $\mathscr{P}$ of $[0,1]$, then we have the following error estimate:

$$|u_{\mathscr{P}}(x) - u(x)| \le C\|\mathscr{P}\|^3 \max_{0\le x\le 1} \left|u^{(4)}(x)\right| \text{ for each } x \text{ in } [0,1] \tag{1.42}$$

For a proof of this theorem we refer to Section 7.5 of [StBu-92]. The key point is that the error estimate is proportional to $\|\mathscr{P}\|^3$, which is superior to the $\|\mathscr{P}\|^2$ estimates for the piecewise linear Rayleigh-Ritz method and for the finite difference method.

EXERCISES 1.3.

1. For each of the following BVPs, perform the following tasks.

   (i) Use the piecewise linear Rayleigh-Ritz method with $n = 50$ equally spaced grid values to solve the BVP numerically and plot the results.

   (ii) Repeat part (i) with $n = 200$ equally spaced grid points.

   (iii) Repeat part (i) with $n = 500$ equally spaced grid points.

   In each part, first perform all integrals directly, and then repeat using the approximations (48)-(50) as needed. Compare performance times. When it is possible to compute the exact solution using the symbolic toolbox, or if one is given, plot the errors of each approximation obtained.

   (a) $(DE)u'' = e^{8x-[2(x-1)]^2}\cos\left(e^{8x}\right)(BC)u(0) = u(1) = 0$

   (b) $(DE)\left(e^{-3x}u'\right)' - e^{-3x}u = 3\pi\cos(\pi x), (BC)u(0) = u(1) = 0; u_{\text{exact}}(x) = e^{3x}\cos(\pi x)$

   (c) $(DE)(2u')' + 12u = x^3, (BC)u(0) = u(1) = 0; u_{\text{exact}}(x) = \left(x^3 - x\right)/12$

2. Repeat the instructions of Exercise 1 for each of the following BVPs:

   (a) $(DE)u'' = \cos(2x) + \sin(16x)/8, (BC)u(0) = u(1) = 0$

   (b) $(DE)\left(\left(1+x^2\right)u'\right)' = 2, (BC)u(0) = u(1) = 0; u_{\text{exact}}(x) = \ln\left(x^2+1\right)$

   (c) $\begin{cases} (DE) - u'' + 400u = -400\cos^2(\pi x) - 2\pi^2\cos(2\pi x) \\ (BC)u(0) = u(1) = 0 \end{cases}$

   $u_{\text{exact}}(x) = e^{20x}/\left(e^{20}+1\right) + e^{-20x}/\left(e^{-20}+1\right) - \cos^2(\pi x)$

3. Repeat each part of Exercise 1 for each of the BVPs given, but this time choose the indicated number of interior nodes randomly, using the `rand` function.

4. Repeat each part of Exercise 2 for each of the BVPs given, but this time choose the indicated number of nodes according to the properties of the coefficient and right-hand-side data.

5. For each of the BVPs given below, use the piecewise linear Rayleigh-Ritz method in conjunction with the method of Exercise for the Reader 10.15 to numerically solve the BVP according to each of the following node deployments:

   (i) Use $n = 50$ equally spaced interior nodes. Repeat with each of $n = 200$ and $n = 500$.

   (ii) Use $n = 250$ randomly chosen interior nodes. Repeat with each of $n = 200$ and $n = 500$.

   (iii) Use $n = 250$ nodes deployed (without equal spaces) in a way that seems reasonable from given data. Repeat with each of $n = 200$ and $n = 500$.

Whenever possible, graph the errors of each of these approximations.

(a) The beam-deflection problem of Example 10.3.

(b) The BVP of Exercise 3(a) of Section 10.2.

(c) The BVP of Exercise 3(c) of Section 10.2.

6. *(A Problematic Choice of Basis Functions)* Consider applying the Rayleigh-Ritz method to our model problem (24) $\left\{\begin{array}{l} -u''(x)=f(x), 0<x<1 \\ u(0)=0, u(1)=0 \end{array}\right.$ using the following bases: $\{\phi_i(x)\}_{i=1}^n$ where $\phi_i(x)=x^i(1-x)$.

(a) Use the Rayleigh-Ritz method with this basis and $n=50$ to re-solve the (BVP) (24) of Example 10.7. How does the solution compare with the "exact" solution found in part (c) ofthat example?

(b) Try to repeat using $n=500$ basis functions. What happens?

(c) Show that $\langle\phi_i', \phi_j'\rangle = \frac{(i+1)(j+1)}{i+j+1} + \frac{(i+2)(j+2)}{i+j+3} - \frac{(i+1)(j+2)+(i+2)(j+1)}{i+j+2}$, for any $i, j>0$.

(d) Make a plot of the condition numbers (use cond (A)) of the $n\times n$ stiffness matrix A as a function of n as n ranges from 2 to 100 . Recall (Chapter 7) that large condition numbers make linear systems difficult to solve.

(e) How would matters change if we instead used $\phi_i(x)=x^i$ as our basis functions?

7. Repeat each part of Exercise 2 for each of the BVPs given, but this time adapt the Rayleigh-Ritz method using the basis functions $\varphi_k(x)=\sin(k\pi x), k=1,2,\cdots,n$ of Exercise for the Reader 10.14.

8. Consider once again the (BVP) $\left\{\begin{array}{ll} -u''(x)+6u(x)=e^{10x}\cos(12x) & 0<x<1 \\ u(0)=0, u(1)=0 \end{array}\right.$ of the Example 10.8.

(a) Use the cubic spline Rayleigh-Ritz method with $n=500$ equally spaced interior grid values to solve this BVP and plot the resulting approximation. Keep a record of the computing time it takes to determine the load vector and stiffness matrix coefficients.

(b) Graph the error of the numerical solution by using the exact solution as in the last example.

9. Repeat each part of Exercise 2 for each of the BVPs given, but this time use the cubic spline Rayleigh-Ritz method. Compare the results (and errors, when possible) with the numerical solutions obtained in Exercise 2.

10. *(Natural Boundary Conditions)* This exercise will show how to develop the Galerkin method for BVPs with non-Dirichlet boundary conditions on the following model problem:

$$\left\{\begin{array}{l} -u''(x)=f(x), 0<x<1 \\ u(0)=0, u'(1)=0 \end{array}\right. .$$

For this problem we use the following for our admissible functions:

$$\mathscr{A}_1=\{v:[0,1]\to\mathbb{R}: v(x) \text{ is continuous},$$

$v'(x)$ is piecewise continuous and bounded, and $v(0)=0\}$. Notice the only difference with this and the class (28) of the model problem (24) considered in the text is that this class has one less requirement: the condition $v(l)=0$ is no longer essential.

(a) Use the DE and integrate by parts (as in step 3 of the proof of Theorem 10.5) to show that any solution of the BVP satisfies the corresponding PVW: $\langle u', v'\rangle = \langle f, v\rangle$ for all $v\in\mathscr{A}$ converse is also true and so the PVW is equivalent to the BVP just as in Theorem 10.5. This gives rise to a Galerkin method for numerically solving the BVP, given any basis of a finitedimensional subspace of A. The fact that no boundary condition is required at $x=1$ for admissible functions in this method has motivated the terminology of a **natural boundary condition** at $x=1$ as opposed to an **essential boundary condition** like the one at $x=0$. It is quite surprising that even though the natural boundary conditions force no conditions on the admissible functions, the solution of the PVW will automatically satisfy them.

(b) (Piecewise Linear Galerkin Method) Given a partition $\mathscr{P}$ of $[0,1]$, we let

$$\mathscr{A}_1(\mathscr{P})=\{v:[0,1]\to\mathbb{R}: v(x) \text{ is continuous on } [0,1], \text{ linear on each } I_i \text{ and } v(0)=0\}.$$

The hat functions $\phi_i(x)(1\le i\le n)$ need one more function to be added to form a basis of $\mathscr{A}_1(\mathscr{P})$. The function $\phi_{n+1}(x)\in\mathscr{A}_1(\mathscr{P})$ defined by $\phi_{n+1}(x_j)=0, (j=0,1,\cdots,n)$ and $\phi_{n+1}(1)=1$ will do the job. By substituting a linear combination of these $\sum_{i=1}^{n+1} c_i\phi_i(x)$ into the PVW, set up a linear system for the resulting Galerkin method.

(c) Apply the method using $n=50$ equally spaced interior nodes to the BVP in case $f(x)=e^{2x}\cos(\pi x)$. Compute the error by comparing with the exact solution (obtainable with the symbolic toolbox).

(d) Repeat part (c) with $n = 200$.

(e) What can be said in general about the stiffness matrix for this method (e.g., is it invertible, symmetric, positive definite)?

11. (Natural Boundary Conditions)Parts (a) through (c): Go through each part of Exercise 10 for the BVP $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u'(0) = 0, u(1) = 0 \end{cases}$, making changes where needed.

    (f) What happens if we try to develop a similar method when both boundary conditions are natural: $u'(0) = 0, u'(1) = 0$?

12. Complete the justification of the approximations (48) through (50).

13. Suppose that $b - a = h$ and that $p(x)$ is a function on $[a,b]$ whose second derivative is continuous on $[a,b]$ (i.e., $p(x)$ is in the space $\mathscr{C}^2([a,b])$). Let $p_\ell(x)$ be the linear function which agrees with $p(x)$ at $x = a$ and $x = b$. Show that for any x in $[a,b]$, we have $|p_\ell(x) - p(x)| = O(h^2)$. Next use this to show that $\left|\int_a^b p_\ell(x) - p(x)dx\right| = O(h^3)$

    Suggestion: For Part (b), use the mean value theorem from calculus to find a number c in $[a,b]$ for which $p'(c) = p'_\ell(c)$. Next use Taylor's theorem to write $p(x) = p_\ell(x) + p''(z_x)(x-c)^2/2$ for any x in $[a,b]$, where z_x is an number between x and c. From this we get that $|p(x) - p_\ell(x)| \le \max_{a \le z \le b} |p''(z)| (x-c)^2/2$ and the assertions readily follow.

14. Derive the Galerkin method for the BVP (41) : $\begin{cases} -(p(x)u'(x))' + q(x)u(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$ by mimicking step 3 of the proof of Theorem 10.5. Does the method agree with the Rayleigh-Ritz method?

15. Suppose that $g(x)$ is a continuous function on $[0,1]$ that satisfies $\int_0^1 g(x)v(x)dx = 0$ for every $v \in \mathscr{A}$. Prove that $g(x) \equiv 0$ by providing more details to the following outline.

    Sketch of Proof: Suppose that $g(x_0) > 0$ for some $x_0 \in (0,1)$. Then by continuity, $g(x) > 0$ for all x in some interval $(x_0 - h, x_0 + h)$. Let $v(x)$ be a hat function with $v(x_0) = 1$, $v(x_0 \pm h) = 0$. Show that $\int_0^1 g(x)v(x)dx > 0$ and this hat function is admissible. This contradiction shows that we cannot have such an $x_0 \in (0,1)$. Conclude similarly that there is no $x_0 \in (0,1)$ for which $g(x_0) < 0$.

16. The proof of Theorem 10.5 made use of one external theorem stating the existence of a solution of the (BVP) (24). In this exercise, you are to follow an outline to prove that part (c) of this theorem implies part (a) by using only the assumption that $u''(x)$ exists and is piecewise continuous and bounded whenever $u(x)$ is a solution of (PVW).

    Write (PVW) in the form $\int_0^1 u'(x)v'(x)dx = \int_0^1 f(x)v(x)dx \quad (v \in \mathscr{A})$. Fix a function $v \in \mathscr{A}$ integrate this by parts to obtain $\int_0^1 [u''(x) + f(x)]\, v(x)dx = 0$. Now use Exercise 16 to show the differential equation (24) must hold.

NOTE: Much error analysis for Rayleigh-Ritz methods depends on certain integral inequalities. A prototypical inequality of this sort is the so-called **Cauchy-Bunyakowski-Schwarz (CBS)** inequality, which reads as follows:

$$|\langle u,v\rangle| \le |\langle u,u\rangle|^{1/2} |\langle v,v\rangle|^{1/2}, \tag{1.43}$$

valid for any functions u, v for which the inner product (25) is defined. In integral form (see (25)) the CBS inequality becomes:

$$\left|\int_0^1 u(x)v(x)dx\right| \le \left(\int_0^1 u(x)^2 dx\right)^{1/2} \left(\int_0^1 v(x)^2 dx\right)^{1/2}.$$[14]

[14]The CBS inequality is a good example of an important mathematical result whose history is often subject to political bias. Cauchy was the first to discover a discrete version (for sums) of the inequality. Bunykowski was the first to discover, in 1859, the integral version of the CBS inequality as written above. Schwarz generalized Bunykowski's result some 25 years later to general inner products. Subsequent mathematical literature from each of the three countries usually attributes any version (from sums to general inner products) of the CBS inequality solely to their mathematical contributor. Thus, in French literature it is usually called Cauchy's inequality, etc. All three of these individuals were eminent mathematical figures in their respective countries. Cauchy began work in 1810 as a civil engineer, but his passion for mathematics kept him trying hard to land positions in mathematics. After numerous attempts he finally got one five years later. Cauchy's output was amazingâĂŤhis complete works spanned over practically all areas of mathematics and filled 27 volumes. His textbooks were used for many years in most French universities. Despite his keen mathematical abilities, however, his strong religious positions and often criticism for his contemporaries made it difficult for him to retain desirable positions. Bunyakowski actually earned his doctorate under Cauchy in Paris in 1825. He then went to St. Petersburg where he spent most of his mathematical career. Schwarz originally entered what is now known as the Technical University of Berlin with the intention of earning a degree in chemistry. This school had the top German mathematics department at the time and the lessons of his mathematics teachers (including the famous analyst Karl Weierstrass (1815-1897)) led him to switch his major and eventually earn a doctorate in mathematics. Schwarz had a remarkable potential in blending analytical and geometrical methods that led him to discover many important results. After he took over Weierstrass's professorial position in 1892, however, he had already begun shifting his focus away from research being his main priority and his output decreased to a less remarkable level. At about this time, the main mathematics institute in Germany shifted from Berlin to GÃűttingen.

Figure 1.16(a): Augustin Louis Cauchy (1789-1857), French mathematician

Figure 1.16(b): Viktor Yakovlevich Bunyakowsky (1804-1889), Russian mathematician.

Figure 1.16(c): Hermann Amandus Schwarz (1843-1921), German mathematician.

In manipulations with such integrals, it is often convenient to introduce the norm notation: $\|u\| = |\langle u,u\rangle|^{1/2} = \left(\int_0^1 u(x)^2 dx\right)^{1/2}$. Using this notation the Cauchy-Bunyakowski-Schwarz inequality takes on the more elegant form:

$$|\langle u,v\rangle| \leq \|u\|\|v\|.$$

For a further discussion of such concepts and, in particular, a proof of the Cauchy-BunyakowskiSchwarz inequality, we refer the reader to any good book on analysis, for example [Ros-96] or [Rud64]. The next few exercises will give examples of such uses of the CBS inequality.

17. Show that the function norm defined above satisfies the three vector norm axioms (see Chapter 7, equations (36A-C)). For simplicity, assume, in your proofs, that all functions are continuous on $[0,1]$.

    (a) $\|u\| \geq 0, \|u\| = 0$ if and only if $u(x) = 0$ for all x in $[0,1]$.

    (b) $\|cu\| = |c|\|u\,|$, for any scalar c.

    (c) $\|u+v\| \leq \|u\| + \|v\|$ (triangle inequality).

    Suggestions: For an idea for part (a), see Exercise 15. For part (c) use the CBS inequality.
    Note: If we allow more general functions, such as piecewise continuous functions in some $\mathscr{A}(\mathscr{P})$, then we have to change the condition in part (a) to $u(x) = 0$ for all x in $[0,1]$ except for a possible finite set of exceptions.

18. *(Rayleigh-Ritz Approximations Have Errors of Minimum Internal Elastic Energy)* Recall that the internal elastic energy of an (admissible) function v was defined to be $(1/2)\langle v',v'\rangle = (1/2)\int_0^1 (v'(x))^2 dx$. Follow the outline below to prove the following useful and interesting error estimate which shows that among all admissible (piecewise linear) admissible functions, the Rayleigh-Ritz approximation to the solution of the BVP (24) $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$ is the best possible approximation if errors are measured by internal elastic energies. That is, if $u(x)$ is the solution of (24) and $u_{\mathscr{P}}(x)$ is the (piecewise linear) Rayleigh-Ritz approximation, both corresponding to a partition $\mathscr{P}$ of $[0,1]$, then we have:

$$\left\|(u-u_{\mathscr{P}})'\right\| \leq \|(u-v)' \text{ for all } v \in \mathscr{A}(\mathscr{P}). \tag{1.44}$$

    (a) Show that $\left\langle (u-u_g)', v\right\rangle = 0$ for any $v \in \mathscr{A}(\mathscr{P})$ by using the principle of virtual work..

    (b) For any $v \in \mathscr{A}(\mathscr{P})$, note that $w \equiv u_{\mathscr{P}} - v \in \mathscr{A}(\mathscr{P})$. Use (55) to write

$$\left\langle (u-u_{\mathscr{P}})', (u-u_{\mathscr{P}})'\right\rangle = \left\langle (u-u_{\mathscr{P}})', (u-v)'\right\rangle.$$

    (c) Next, use the CBS inequality to obtain (56).

19. *(Comparison of Solution with Linear Interpolant)*

    (a) Let $\bar{u}_{\mathscr{P}} \in \mathscr{A}(\mathscr{P})$ be the (piecewise) **linear interpolant** of the solution u of (24), i.e., $v(x_i) = u(x_i)$ at each partition point (and $\bar{u}_{\mathscr{P}}$ is linear between partition points). Let x be any number in $[0,1]$ that lies between two partition points of $\mathscr{P}: x_j < x < x_{j+1}$. Use the mean value theorem from calculus to show why we can write

$$\bar{u}_{\mathscr{P}}(x) = u(c) + (x-c)u'(x) \text{ for some fixed number } c, x_j < c < x_{j+1}.$$

    (So c depends only on j, but not on the particular value of x.)

(b) In the notation of part (a) use Taylor's theorem and only the assumption that u has a continuous second derivative to show that for any $x, x_j < x < x_{j+1}$, we have

$$|u(x) - \tilde{u}_{\mathscr{P}}(x)| \leq \frac{h_j^2}{2} \max_{x_j < x < x_{j+1}} |u''(x)|.$$

Next use the differential equation of (24) to translate this estimate to the form

$$|u(x) - \tilde{u}_{\mathscr{P}}(x)| \leq \frac{\|\mathscr{P}\|^2}{2} \max_{x_j < x < x_{j+1}} |f(x)|,$$

and that this is valid for all x in $[0,1]$.

(c) By applying a similar analysis as done in parts (a) and (b) except now on the derivatives of the above two functions, obtain the following estimate for all x in $[0,1]$ (except the partition points at which $\tilde{u}'_{\mathscr{P}}(x)$ may not exist):

$$|u'(x) - \tilde{u}'_{\mathscr{P}}(x)| \leq \|\mathscr{P}\| \max_{0 \leq x \leq 1} |f(x)|.$$

20. *(Error Estimate for Rayleigh-Ritz Approximation)* This exercise will provide an outline for using the estimates of the preceding exercises to obtain an estimate for the error of the RayleighRitz approximation. We will show that if $u_{\mathscr{P}}(x)$ is the Rayleigh-Ritz approximation corresponding to a partition $\mathscr{P}$ of $[0,1]$ of the BVP (24): $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$, where $f(x)$ is a continuous function, then we have the following error estimate valid for all $x, 0 \leq x \leq 1$: $|u_{\mathscr{P}}(x) - u(x)| \leq \|\mathscr{P}\| \max_{0 \leq x \leq 1} |f(x)|$.

(a) Use (54) with v taken to be the linear interpolant $\tilde{u}_{gp}$ of Exercise 19, and then use the results of Exercise 19 to justify the following string of inequalities:

$$\|(u - u_{\mathscr{P}})'\| \leq \|(u - \tilde{u}_{\mathscr{P}})'\| = \left(\int_0^1 \left[(u - \tilde{u}_{\mathscr{P}})'(x) \right]^2 dx \right)^{1/2}$$
$$\leq \left(\int_0^1 \left[\|\mathscr{P}\| \max_{0 \leq x \leq 1} |f(x)| \right]^2 dx \right)^{1/2} \leq \|\mathscr{P}\| \max_{0 \leq x \leq 1} |f(x)|$$

(b) Since $u_{\mathscr{P}}(0) = u(0) = 0$, we can write $u(x) - u_{\mathscr{P}}(x) = \int_0^x (u - u_{\mathscr{P}})'(x) dx$. Use this and the CBS inequality to justify the following string of inequalities, thereby completing the proof of Theorem 10.6.

$$|u(x) - u_{\mathscr{P}}(x)| \leq \langle (u - \tilde{u}_{\mathscr{P}})', 1 \rangle \leq \|(u - \tilde{u}_{\mathscr{P}})'\| \cdot \|1\| \leq \|\mathscr{P}\| \max_{0 \leq x \leq 1} |f(x)|$$

21. *(Refined Error Estimate for Rayleigh-Ritz Approximation Using Green's Functions)* In this exercise we will show that when the Rayleigh-Ritz method is to solve the BVP (24): $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$, where $f(x)$ is a continuous function for some partition 5 of $[0,1]$, then the Rayleigh-Ritz solution $u_{\mathscr{P}}$ actually coincides with the linear interpolant $\tilde{u}_{\mathscr{P}}$ of Exercise 19. From this it will follow from the estimate of part (b) of Exercise 19 that the estimate of Theorem 10.6 is valid. The key is the introduction of so-called **Green's functions** for the BVP. For each interior node x_i of $\mathscr{P}$ the following Green's function:

$$G_i(x) = \begin{cases} (1 - x_i)x, & \text{for } 0 \leq x \leq x_i \\ x_i(1 - x), & \text{for } x_i \leq x \leq 1 \end{cases}$$

(a) Show that $G_i(x) \in \mathscr{A}(\mathscr{P})$ and that for any $v \in \mathscr{A}$, we have: $\langle v', G_i' \rangle = v(x_i)$.

(b) Take $v = u - u_{\mathscr{P}}$ in part (a) and apply a result from one of the preceding exercises to show that $v(x_i) = 0$ and hence $u = u_{\mathscr{P}}$, as desired.

22. Show directly that the BVP (24) $\begin{cases} -u''(x) = f(x), 0 < x < 1 \\ u(0) = 0, u(1) = 0 \end{cases}$ has a unique solution.

Suggestion: Integrate the DE once to get $u'(x) = \int_0^x -f(t) dt$ and once more to get $u(x) = \int_0^x \int_0^t -f(s) ds dt + C$

23. Using the direct approach to solving the BVP (24) suggested in the preceding exercise, set up and execute a MATLAB code for solving the BVP in Example 10.7 once again. Arrange your parameters so that the total error of your numerical solution is no more than 10^4.

Suggestion: You may wish to try some different approaches using MATLAB's built-in integrator in conjunction with Simpson's rule or the trapezoidal rule (see any standard calculus textbook or [BuFa-01]) and perhaps even the Symbolic Toolbox if you have access to it.

24. (a) Verify that the general solution of the DE $-u'' = \lambda u$ for $\lambda > 0$ is given by $u = C\cos(\sqrt{\lambda}x) + D\sin(\sqrt{\lambda}x)$ for arbitrary constants C and D.

    (b) Show that if we also require the boundary conditions $u(0) = u(1) = 0$, then the resulting BVP will only have nontrivial solutions if $\lambda = (k\pi)^2, k = 1, 2, \cdots$ and these (eigenfunctions) are $u_k(x) = \sin(k\pi x)$.

    (c) Prove the orthogonality relations for the eigenfunctions: $\langle u_k, u_\ell \rangle = \delta_{k\ell}/2$, where $\delta_{k\ell}$ denotes the Kronecker delta.

    (d) Prove the following orthogonality relations for the derivatives of the eigenfunctions: $\left(u'_k, u'_i\right) = k\ell\pi^2\delta_{kl}/2$.

Chapter 2

Introduction to Partial Differential Equations

2.1. THREE-DIMENSIONAL GRAPHICS WITH MATLAB

As mentioned in Chapter 8 , **partial differential equations (PDEs)** are differential equations where the unknown function (solution) is a function of several variables (and so the derivatives will be partial derivatives). The subject of PDEs is probably the most vast branch of mathematics and we will be focusing mostly on PDEs with two independent variables. There are two main reasons for this restriction. First, many of the key aspects of the theory and numerical methods in partial differential equations are well represented in PDEs having two independent variables. Second, once we obtain (numerical) solutions of such PDE, they will be functions of two variables and we will be able to graph them using MATLAB's three-dimensional capabilities. Graphs are not feasible if there are more than two independent variables, as such graphs would require at least four dimensions. Most of our PDEs will arise from physical models, and it will be convenient to use two different sets of independent variables: either x (space) and t (time) or x and y (two space variables). The solutions of such PDEs will be functions of two variables: $z = f(x,t)$ or $z = f(x,y)$, and often the best way to understand such a function is by a graph. Such a graph will require three dimensions and it is customary to have the dependent variable's axis be vertical (just like for functions of one variable), so the two independent variables will have their axes span a twodimensional plane that must protrude out of the paper (or screen) on which it is graphed. The graph of such a function will be a surface in the three-dimensional xyz-plane.

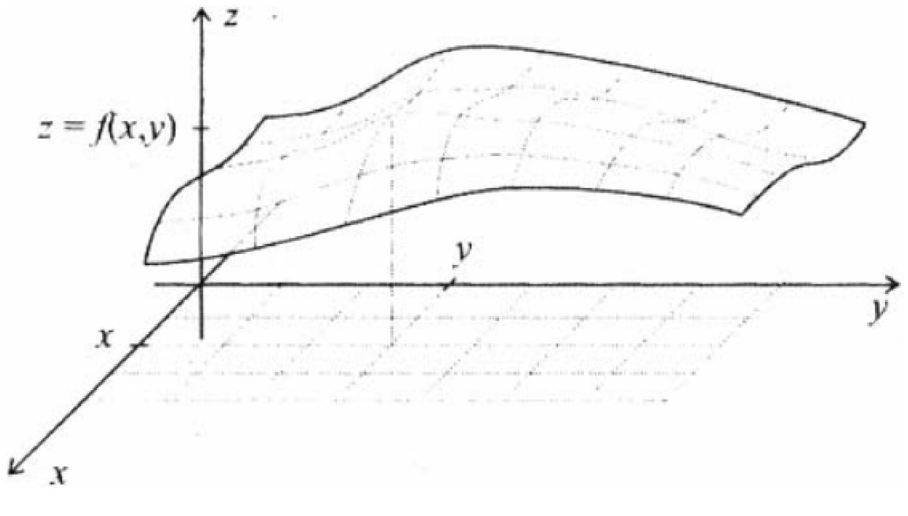


Figure 2.1: Mesh net graph of a function $f(x,y)$ of two variables.

In Figure 11.1, we have graphed a surface $z = f(x,y)$ with a mesh type of graph where images of equally spaced parallel lines of the form $x = c$ and $y = d$ under the function $f(x,y)$ are shown. In MATLAB, there are numerous ways of plotting and viewing the graph of a function of two variables. As with two-dimensional plots, the axis range can be specified. But to help with the third dimension, there are also other useful features such as viewing perspective (from where should we view the surface?), lighting, shading, and mesh grid lines. Color ranges can be used to vary with the height of the dependent variable. Often it is necessary to experiment with different versions of the same graph to find the best rendition of it for a particular purpose. The simplest way to understand how MATLAB does a plot of a function $z = f(x,y)$ is to consider the so-called mesh grid plots. These look a lot like the one in Figure 11.1, except only the points which are actually plotted are the grid points where one of the equally spaced horizontal lines $x = c$ meets a corresponding vertical line $y = d$-see Figure 11.2. Once the values of $f(x,y)$ are computed at these grid points, adjacent plotted points are connected by straight line segments.

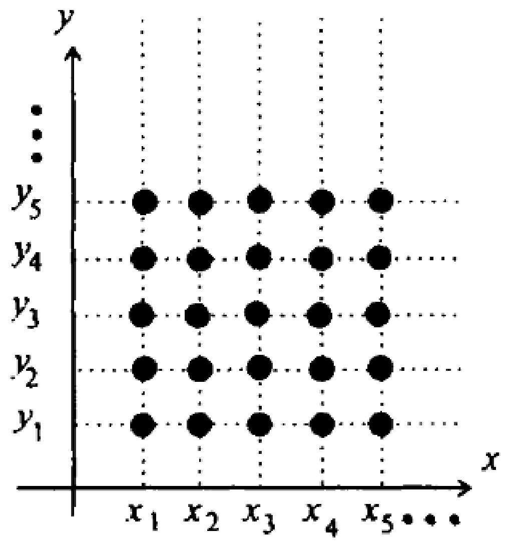


Figure 2.2: When MATLAB plots a function $z = f(x,y)$ of two variables, it will evaluate the function only for grid points (x,y) (the solid dots). Resolution can be improved by refining the grids (on both x-and y-axes). MATLAB has a function that will build the matrices of grid points from the corresponding vectors $x = (x_1, x_2, \cdots x_n)$ and $y = (y_1, y_2, \cdots y_m)$.

As with all things numerical in MATLAB, three-dimensional plots in MATLAB will be obtained from appropriate matrices. Once the x and y vectors (determining x - and y-coordinates of grid points) have been entered, we need to construct matrices X and Y of all of the x - and y-coordinates for the grid points. If x has n elements and y has m elements then X and Y will be of size $n \times m$. MATLAB has a convenient function `meshgrid` that will construct these meshed matrices for us.

$[X,Y] =$ meshgrid $(x,y) \rightarrow$	For input vectors $x = (x_1, x_2, \cdots x_n)$ and $y = (y_1, y_2, \cdots y_m)$, meshgrid outputs two matrices X and Y each of size $n \times m$, which will be the corresponding x - and y-coordinates of all the points in the grid determined by the vectors x and y.

The next example will help us to understand how `meshgrid` is used to obtain plots of functions $z = f(x,y)$ in MATLAB. Since the three-dimensional plot functions and their options are best explained and illustrated by examples, we use the next example as a venue for illustrating several ways to plot surfaces and change the graphs. Help menus will introduce other options and features.

Example 2.1. ■

(a) Starting with the vectors $x = y = (-3, -2, \ldots, 2, 3)$, use meshgrid to create two corresponding matrices of grid points for these vectors and then obtain a meshgrid plot of the surface

$$z = f(x,y) = \frac{3}{1+x^2+y^2} - \frac{10}{4+(x+1.5)^2+y^2}$$

(b) Use vectors with 50 equally spaced elements over the same x-and y-ranges to obtain finer plots of this function and display plots used by several different MATLAB plotting tools.

SOLUTION: Part (a): Here are the MATLAB commands.

```
>> x=-3:3; %x-values of grid
>> y=x;    %y-values of grid
>> [X, Y]=meshgrid(x,y) %creates matrices of grid point coordinate
```

```
X =                                     Y =

   -3  -2  -1   0   1   2   3              -3  -3  -3  -3  -3  -3  -3
   -3  -2  -1   0   1   2   3              -2  -2  -2  -2  -2  -2  -2
   -3  -2  -1   0   1   2   3              -1  -1  -1  -1  -1  -1  -1
   -3  -2  -1   0   1   2   3               0   0   0   0   0   0   0
   -3  -2  -1   0   1   2   3               1   1   1   1   1   1   1
   -3  -2  -1   0   1   2   3               2   2   2   2   2   2   2
   -3  -2  -1   0   1   2   3               3   3   3   3   3   3   3
```

Corresponding entries in these grid matrices pair to give us (x,y) grid coordinates. The matrices are constructed in a way that MATLAB's plotting functions will be able to interpret, associate corresponding z-coordinates, and produce the plots.

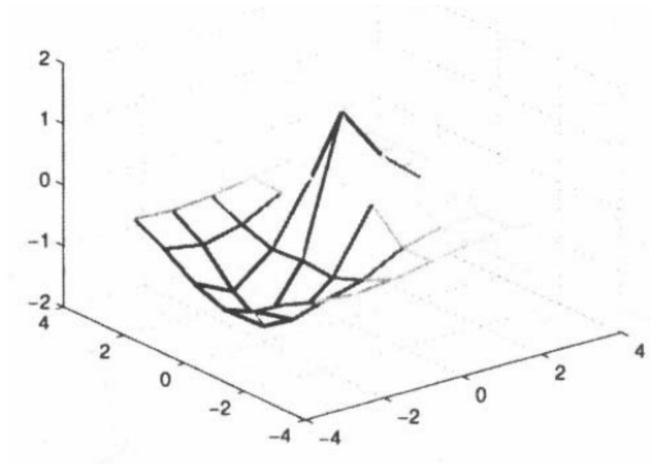
```
>> Z=3./(1+X.^2+Y.^2)-10./(4+(X+1.5).^2+Y.^2); %construct
>> %corresponding matrix Z of z-coordinates for grid points
>> mesh(X,Y,Z) %produces the desired mesh grid plot.
```

The plot is shown in Figure 11.3 on the left.

Part (b): We refine the x - and y-values of the grid, reconstruct the meshgrid matrices and the corresponding Z-matrix, and then use `mesh` to get the higherresolution plot on the right of Figure 11.3.

```
>> x=linspace(-3,3,50); y=x;
>> [X,Y]=meshgrid(x,y);
>> Z=3./(1+X.^2+Y.^2)-10./(4+(X+1.5).^2+Y.^2);
>> mesh(X,Y,Z)
```

CAUTION: If you just type mesh (Z) you will get the same surface graph; however, the numbers on the x-and y-axes will be the vector indices (so in this case will range from 1 to 50). It is, however, acceptable to use the original vectors x and y : `mesh` (x,y,z), rather than the mesh matrices for x and y.



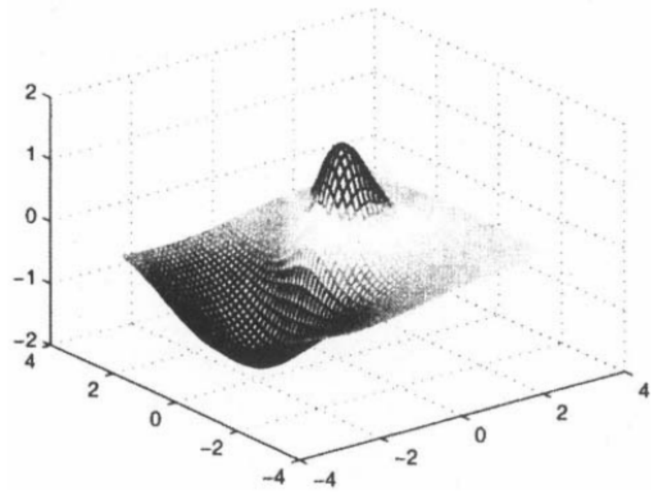


Figure 2.3: Two MATLAB meshgrid plots of the function

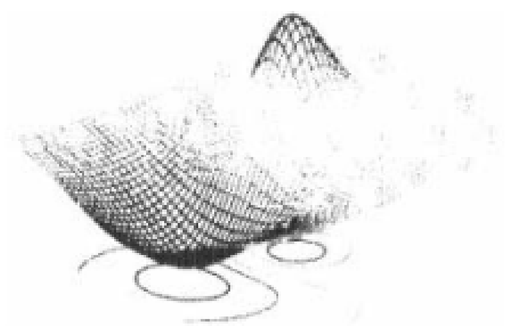
$z=\frac{3}{1+x^2+y^2}-\frac{10}{4+(x+1.5)^2+y^2}$

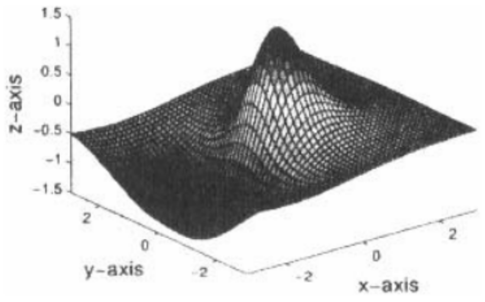
over the square $-3 \leq x,y \leq 3$. The first used only seven x - and y-grid coordinates, while the second used 50 . The main plotting command was mesh (X,Y,Z); see Example II.I. We point out some defaults in these plots: Colors are used with red for the highest y-values and blue for the lowest, this will come in handy for temperature plots. Also from our perspective, hidden parts of the mesh are not visible, and grid lines are shown.radar chart[1]

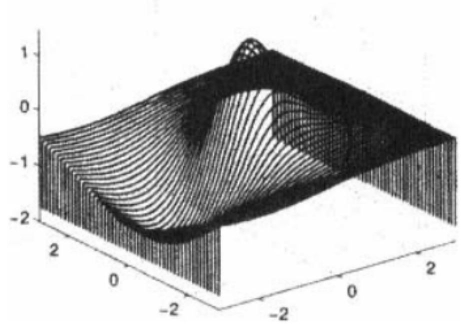
TABLE 11.1
Catalogue of several other useful plotting commands. In each we assume that `x, y, X, Y,` and `Z` have already been constructed as in Example 11.1.

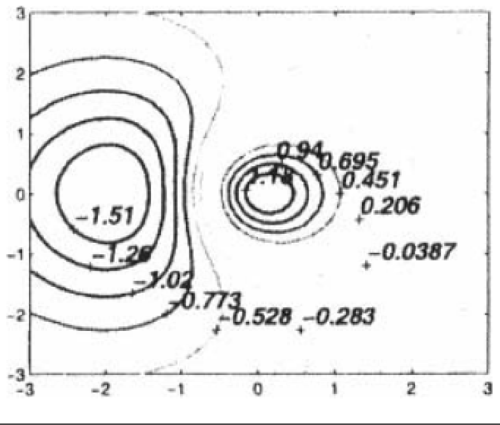
[1]To remove the grid lines simply enter `grid off`.

MATLAB Plotting Function	Syntax and Options	Resulting Graph

`mesh ->` (mesh plot with underlying contour lines)	`>> meshc(X,Y,Z)`
	`>> axis('off')`
	`%supresses axes (and grids)`
		

`surf ->` (surface plot = 1 mesh plot with squares between mesh lines filled in)	`>> surf(X,Y,Z)`
	`>> grid off %leaves axes in but takes out all the extra grid lines; note that unlike with mesh plots, the mesh lines are now all black.`
	`>> xlabel('x-axis'), ylabel('y-axis'), zlabel('z-axis') % labels work like in 2 dim. plots.`
		

`waterfall ->` (surface plot using only y-grid lines; these are extended over edges of plot)	`>> waterfall(X,Y,Z)`
	`>> hidden off %%allows to see through all parts of surface`
	`>> colormap([0 0 0]) % changes plot to black. In general the input vector [r g b] of 'colormap' has r measuring intensity of red, g intensity of green and b intenstity of blue. All are numbers between 0 and 1.`
		

`contour ->` (two-dimensional contour plot for surface)	`>> c=contour(x,y,Z,12);`
	`%we can specify the number of contour lines we want (here 12). Setting "c=" to this commands allow us to label contours with their z-values`
	`>>clabel(c,'manual')`
	`%allows us to manually choose which contours to label using the mouse.`
		

Other plotting commands include `surfc(x,y,z)` (plots like surf and adds contour lines), `surfl(x,y,Z,lvec)` (surface plot with lighting; light source located at the vector `lvec`), and `contour3(x,y,z,n)` (plots threedimensional contour plots, the positive integer n is number of contour levels drawn). More can be learned about using these commands and their associated options by experimenting and reading on-line help menus. You can get a good general summary of MATLAB's 3D graphing tools (including some useful preprogrammed colormaps) by entering `help graph3d`

Once a three-dimensional plot has been created, it is often useful to view it from different perspectives. This can be accomplished using the `view` command:

view (azimuth,elevation) $\to$	Resets viewing angles for three-dimensional plots. **The azimuth angle** α and the **elevation angle** ε are measured as shown below in Figure 11.4. These angles are measured in degrees. The default values are $\alpha = -37.5$ and $\varepsilon = 30^2$.

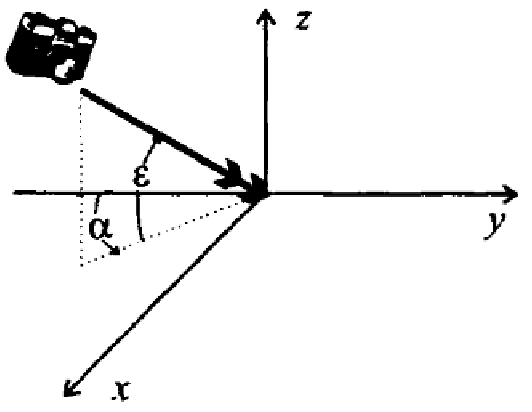


Figure 2.4: Measurement of the azimuth angle α and the elevation angle ε for use of the MATLAB command `view(azimuth, elevation)` to change perspectives on three-dimensional plots.

EXERCISE FOR THE READER 2.1.

(a) Plot the function $z = \sin x \sin y \exp\left(-\sqrt{x^2+y^2}/4\right)$ over the set $-5 \le x, y \le 5$ using 30 grid values for each of x and y and the `surf` command. Add labels to each of the three axes and set the grid off. Obtain views from three additional and significantly different viewing angles. Redo these plots with `mesh`.

(b) Repeat part (a) for the "mountain pass function" $z = \sin(y + \cos x)$.

MATLAB's three-dimensional graphics capabilities are quite vast and we will not even begin to talk about things like lighting and graphs of other types of surfaces (manifolds), as such a thorough treatment will not be needed for what we will be doing with PDEs. In closing this section, we mention two more useful plotting commands, although they are more relevant for our past work on three- dimensional orbits rather than for our subsequent work on PDEs.

plot $3(x,y,z) \to$	Plots the curve in three-dimensional space stored by the vectors x, y, and z.

comet $3(x,y,z) \to$	Plots the curve in three-dimensional space stored by the vectors $x, y,$ and z in an animated fashion from start to finish. Final result is same as plot 3.

This command allows us to plot orbits for three-dimensional first-order systems of ordinary differential equation which were dealt with in the last chapter or any three-dimensional parametric equations. We give an example of the latter.

Example 2.2. ■

Use both the `plot3` and `comet3` commands to plot the following decaying oscillating helix:

$$\begin{cases} x(t) = e^{-t/4}\cos(4t) \\ y(t) = e^{-t/4}\sin(4t) \quad \text{on the interval } 0 \le t \le 6\pi \\ z(t) = \sin(t/2) + 1 \end{cases}$$

Next, use the `view` command to get a view from the top (i.e., the xy-plane projection of the curve).

SOLUTION:

[2]The "3D Rotate" button on the MATLAB graphics window can be also used to change the viewing perspective.

```
>> t=0:.005:6*pi;
>> x=exp(-t/4).*cos(4*t); y=exp(-t/4).*sin(4*t); z=sin(t/2)+1;
>> comet3(x,y,z)
>> plot3(x,y,z)
>> xlabel('x'), ylabel('y'), zlabel('z')
>> view(0,90)
```

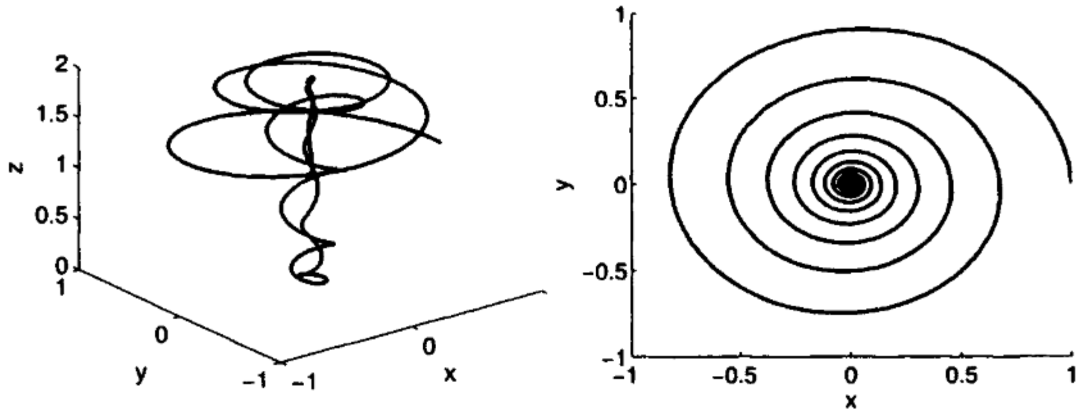


Figure 2.5: On the left, a three-dimensional plot of the oscillating decaying helix of Example 11.2; the right shows the top view of this decaying helix, which turns out to be just a spiral.

EXERCISES 2.1.

1. (i) Using the mesh plotting tool and also using 50 grid values for both x - and y-coordinates, plot each of the following functions $z = f(x,y)$ over the indicated region. Use the `colormap[0 0 0]` so the display will be in black/white, and turn off the grid. Experiment with the three additional view settings: $[0,90],[90,-45],[135,45]$. (ii) Repeat these plots using 500 grid values for the x-and y-coordinates.

   (a) $z = x^2 + y^2, \quad -5 \le x,y \le 5$
   (b) $z = 4x/\left(x^2+y^2+1\right), \quad -4 \le x,y \le 4$
   (c) $z = \frac{xy\left(y^2-x^2\right)}{x^2+y^2}, \quad -3 \le x \le 3$
   (d) $z = \exp\left(-x^2/2\right) + \exp\left(-2y^2\right), \quad -3 \le x,y \le 3$

2. (i) Using the surfc plotting tool and also using 50 grid values for both x - and y-coordinat plot each of the following functions $z = f(x,y)$ over the indicated region. Turn off the ax Experiment with the three additional view settings: $[135-45],[135,0],[135,25]$. Repeat these plots using 500 grid values for the x - and y-coordinates.

   (a) $z = \sin x \sin y, \quad -2\pi \le x,y \le 2\pi$
   (b) $z = \sin xy, \quad -2\pi \le x,y \le 2\pi$
   (c) $z = \cos x \cos y \exp\left(-\sqrt{x^2+y^2}/4\right), \quad -2\pi \le x,y \le 2\pi$
   (d) $z = 1/[1+\sqrt{|y-\sin x|}+\sqrt{|x+\sin y|}], \quad -16 \le x,y \le 16$

3. (i) Using the waterfall plotting tool with the colormap[127/255|212/255] ("aquamarine") and with 50 grid values for both x-and y-coordinates, plot each of the following functions $z = f(x,y)$ over the indicated region. Turn off the grid and hidden defaults, and label the $x-, y-$, and $z-$ axes as such. Experiment with the three additional view settings: $[0,30],[0,60],[0,90]$. (ii) Repeat these plots using 500 grid values for the x - and y-coordinates.

   (a) $z = 20 - 2x^2 - 3y^2, \quad -2 \le x,y \le 2$
   (b) $z = x^2 - y^2, \quad -2.5 \le x,y \le 2.5$
   (c) $z = \sin\sqrt{x^2+y^2}, \quad -2\pi \le x,y \le 2\pi$
   (d) $z = \cos(x+y)\cos(3x-y) + \cos(x-y)\sin(x+3y)\exp\left(-\left[x^2+y^2\right]/8\right), \quad -\pi \le x,y \le \pi$

4. Redo tasks (i) and (ii) for each part (a) through (d) of Exercise 3 using the `contour3` plotting tool, but this time use the default colormap.

NOTE: For analyzing three-dimensional graphics it is often useful to change the axes range of an existing plot (so to focus in on a particular portion). The `axis` command for two-dimensional plots extends naturally to this setting:

`axis ([xmin xmax ymin ymax zmin zmax])` $\rightarrow$	Restricts a given plot to the range defined by $x\min \le x \le x\max, y\min \le y \le y\max, z\min \le z \le z\max$.

5. Obtaining first a good (or several good) surface plots of the function $z = -2y/\left(x^2+y^2+1\right)$ over the region $-4 \le x,y \le 4$ (using a surface plotting tool and settings of your choice), and then making use of the `axis` and/or `contour` commands (repeatedly), find the x-, y - and $z-$ coordinates of the maximum value of the function on this region. Your answer should be accurate to within a tolerance of 0.01 in each coordinate. Repeat for the minimum value.

6. Obtaining first a good (or several good) surface plots of the function $z = \cos(x+y)\cos(3x-y) + \cos(x-y)\sin(x+3y)\exp\left(-\left[x^2+y^2\right]/8\right)$, over the region $-\pi \le x,y \le \pi$ (using a surface plotting tool and settings of your choice), and then making use of the `axis` and/or `contour` commands (repeatedly), find the x,y - and z-coordinates of the maximum value of the function on this region. Your answer should be accurate to within a tolerance of 0.001 in each coordinate.

7. Use `plot3` and `comet3` to graph each of the following space curves. Experiment with the `view` command to choose a "nice" view for your printout. Use the option `axis('equal')` to prevent axis distortion.

   (a) $x(t) = \frac{\cos t}{\sqrt{1+\sin^2 2t}}, y(t) = \frac{\sin t}{\sqrt{1+\sin^2 2t}}, \quad z(t) = \frac{\sin 2t}{\sqrt{1+\sin^2 2t}}, 0 \le t \le 50$

   (b) $x(t) = \frac{\cos 3t}{\sqrt{1+\sin^2 2t}}, y(t) = \frac{\sin 3t}{\sqrt{1+\sin^2 2t}}, z(t) = \frac{\sin 2t}{\sqrt{1+\sin^2 2t}}, 0 \le t \le 50$

   (c) $x(t) = \frac{\cos 11t}{\sqrt{1+\sin^2 5t}}, y(t) = \frac{\sin 11t}{\sqrt{1+\sin^2 5t}}, z(t) = \frac{\sin 5t}{\sqrt{1+\sin^2 5t}}, 0 \le t \le 100$

   (d) $x(t) = \frac{\cos(\sqrt{2}t)}{\sqrt{1+\sin^2 t}}, y(t) = \frac{\sin(\sqrt{2}t)}{\sqrt{1+\sin^2 t}}, z(t) = \frac{\sin t}{\sqrt{1+\sin^2 t}}, 0 \le t \le 100$

   If you have done each of the above parts, point out some similarities and differences in the above plots. Which are periodic? Do you have any general comments/conjectures to make about parametric equations related to these?

8. Use `plot3` to plot the solution of the Lorenz strange attractor of Example 9.7 in Section 9.4. Experiment with the view command to get one (or more) "nice" plots to include with your printout.

9. Redo Exercise 5 of Section 9.4 (the Rossler band), but now plot only the orbits in three dimensions using both `comet3` and `plot3`. Experiment with the `view` command to choose a "nice" view for your printout.

10. The following ODE system has a very interesting orbit; an analysis of which is done in the article [Lan-84].

$$\begin{cases} x'(t) = (z-0.7)x - 3.5y, & x(0) = 0.1 \\ y'(t) = 3.5x + (z-0.7)y, & y(0) = 0.03 \\ z'(t) = 0.6 + z - 0.33z^3 - \left(x^2+y^2\right)(1+0.25z), & z(0) = 0.001 \end{cases}$$

   Use the Runge-Kutta method with step size $h = 0.01$ and `plot3` to graph the solution of the above system on the time range $0 \le t \le 100$. Repeat with step size $h = 0.005$. Use the `axis(equal)` command to remove any axis scale distortions.

11. Often, a three-dimensional surface plot is desired over a different shape than a rectangle, for example, over a circle or over an ellipse. This can be done using one of the surface plotting tools in MATLAB, but one needs to use polar coordinates to create the appropriate meshgrid matrices. For example, suppose that we wanted to plot the paraboloid $z = 1 - x^2 - y^2$ over the disk $x^2 + y^2 \le 4$. (a) Follow the following outline to create this plot:

   (i) Create vectors for r and theta:
   ```
   >>r=linspace(0,4,16); theta=linspace(0,2*pi,20);
   ```

   (ii) Create corresponding mesh matrices X and Y for these polar coordinates:
   ```
   >> X=r'*cos(theta); Y=r'*sin(theta);
   ```

   Note: Convince yourself that these will be mesh matrices.
   Optional: To see the meshgrid, type:
   ```
   >>Z=zeros(size(X)); mesh(X,Y,Z)
   ```

   (iii) Create the Z-matrix and plot as usual:
   ```
   >> Z=1-X.^2-Y.^2; mesh(X,Y,Z)
   ```

   Try some other options (like `hidden off`).

   (b) Plot the graph of $z = \sqrt{\cos\left(x^2+2y^2\right)}$ over the disk $x^2+y^2 \le 2$.
   (c) Plot the graph of the paraboloid $z = 1 - x^2 - 3y^2$ over the ellipse $x^2 + 3y^2 \le 1$.
   (d) Plot the graph of the function in part (d) of Exercise 3 over the disk $(x-\pi/2)^2 + (y-\pi/2)^2 \le 1$

2.2. EXAMPLES AND CONCEPTS OF PARTIAL DIFFERENTIAL EQUATIONS

We begin our discussion with a natural problem about heat conduction. Consider a rod of length L that is insulated on the outer boundary but perhaps not at the ends.

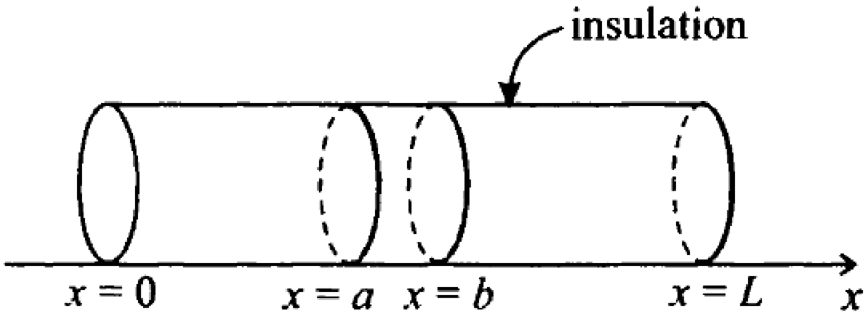


Figure 2.6: A one-dimensional rod of length L that is insulated along its cylindrical surface but not necessarily on the flat edges at $x=0$ and $x=L$.

For this problem we are interested in the temperature of the rod as a function of the time t and the position x along the x-axis. We assume that the rod is very thin so that the temperature is constant for a given time throughout the cross-section. We call this function $u(x,t)$. The basic (and very intuitive) physical principle that we will use here, Fourier's Law, states that heat flows from hot places to cold places at a rate proportional to the temperature gradient. We will make this more precise shortly. The substance that the rod is made out of is relevant for knowing how quickly heat is transferred along the rod. To quantify this, we introduce

$c=$ the **specific heat** of the substance of the rod $=$ the heat energy needed to raise 1 unit of mass 1 unit of temperature.

The specific heat is a chemical/molecular property of the substance. Many partial differential equations in the natural sciences are based on a conservation principle, and the equation we will derive for $u(x,t)$ will be a good example of this. We let A denote the cross-sectional area of the rod and ρ denote the mass density (= mass per unit volume) of the rod. For values $x=a<x=b$ along the rod, we introduce

$$\text{Q(t)} = Q_{[a,b]}(t) = \text{the heat energy along the rod} \underset{\text{from } x=a \text{ to } x=b}{=} \int_a^b u(x,t)c\rho A dx. \tag{2.1}$$

By the conservation of energy, we have

$$\frac{dQ}{dt} = \text{flux term} + \text{source term} \tag{2.2}$$

Letting

$F(x,t)=$ heat flux function $=$ heat energy passing through cross section at x per unit area per unit time in the positive x-direction,

empirical physical laws imply that

$$F(x,t) = -\kappa \frac{\partial u}{\partial x}(x,t),$$

where $0<\kappa$ is the **heat conductivity** of the material of the rod. Higher conductivities mean substances are better conductors of heat. For example, the specific heat of copper is about three times that of lead but less than half that of aluminum. We may now write,

$$\begin{aligned} \text{flux term } &= -A[F(b,t)-F(a,t)] \\ &= A\left[\kappa u_x(b,t) - \kappa u_x(a,t)\right] \\ &= \int_a^b A\left[\partial/\partial x\{\kappa u_x(x,t)\}\right]dx \end{aligned}$$

where we used the fundamental theorem of calculus to write the last integral. Now, provided that u and u_t are continuous we may differentiate (1) under the integral sign (see any book on advanced calculus, e.g., [Rud-64]):

$$\frac{d}{dt}\left(\int_a^b u(x,t)c\rho A dx\right) = \int_a^b u_t(x,t)c\rho A dx.$$

Combining this with (2) and the above expression for the flux term gives:

$$\int_a^b [c\rho u_t(x,t) - \partial/\partial x\{\kappa u_x(x,t)\}]Adx = \text{ source term.} \tag{2.3}$$

If there are no internal heat sources within the rod, then the integral in (3) must vanish for any $a,b; 0 \le a < b \le L$. If the integrand is continuous, it follows (from the fundamental theorem of calculus) that the integrand must vanish identically. If κ is constant, then this leads us to the PDE:

$$u_t = ku_{xx}, \quad u = u(x,t),$$

Where $k = \kappa/c\rho$ is called the **diffusivity** of the material. This equation is called the one-dimensional **heat equation** and it is also known as the one-dimensional **diffusion equation**. The reason for the latter terminology is that it more generally models the spatial (x) and temporal (t) spread of many different phenomena which obey a similar flux principle where higher concentrations spread to neighboring areas of lower concentrations at a rate proportional to the gradient. Indeed the diffusion equation has been used to successfully model spreads of populations from the molecular level (diseases) to larger-scale models of plants and animals and also the spread of other chemicals, gases, and drugs.

$$\int_a^b [c\rho u_t(x,t) - \partial/\partial x\{\kappa u_x(x,t)\}]Adx = \text{ source term.} \tag{2.4}$$

If there are no internal heat sources within the rod, then the integral in (3) must vanish for any $a,b; 0 \le a < b \le L$. If the integrand is continuous, it follows (from the fundamental theorem of calculus) that the integrand must vanish identically. If κ is constant, then this leads us to the PDE:

$$u_t = ku_{xx}, \quad u = u(x,t),$$

Where $k = \kappa/c\rho$ is called the **diffusivity** of the material. This equation is called the one-dimensional **heat equation** and it is also known as the one-dimensional **diffusion equation**. The reason for the latter terminology is that it more generally models the spatial (x) and temporal (t) spread of many different phenomena which obey a similar flux principle where higher concentrations spread to neighboring areas of lower concentrations at a rate proportional to the gradient. Indeed the diffusion equation has been used to successfully model spreads of populations from the molecular level (diseases) to larger-scale models of plants and animals and also the spread of other chemicals, gases, and drugs. We will return to the one-dimensional heat equation shortly, but we first give some extensions of it and related PDEs. If the rod in Figure 11.5 has internal heat sources (e.g., chemical catalysts or electronic components), then we put

$$q_1(x,t,u) = \text{rate of production of heat energy per unit time per unit volume at position } x. \tag{2.5}$$

Note that $q_1(x,t,u) > 0$ means there is a heat source at cross-section x and $q_1(x,t,u) < 0$ means there is a heat sink at cross section x. This function gives rise to the "source term" in (3) to be $\int_a^b q_1(t,x,u(x,t))Adx$ and so now (3) can be rewritten as

$$\int_t^b [c\rho u_t(x,t) - \partial/\partial x\{\kappa u_x(x,t)\} - q_1(x,t,u)]Adx = 0.$$

As before, if the integrand is continuous and κ is constant, we obtain the following PDE, known as the one-dimensional **heat equation with source term**

$$u_t = ku_{xx} + q, \quad u = u(x,t), \tag{2.6}$$

where $q = q_1/c\rho$. A similar derivation would work also in two or three space dimensions to give us the corresponding two- and three-dimensional heat equations with or without source terms. For example, the **two-dimensional heat/diffusion equation** looks like:

$$\mathrm{u_t} = k\left(u_{xx} + u_{yy}\right), \quad u = u(x,y,t), \tag{2.7}$$

and similarly the three-dimensional analogue is

$$\mathrm{u_t} = k\left(u_{xx} + u_{yy} + u_z\right), \quad u = u(x,y,z,t). \tag{2.8}$$

These PDEs model the spread of heat in a flat plate or a solid three-dimensional region and also population and chemical diffusions. If we are modeling a population that grows according the Malthusian or logistical model and also diffuses, we can model the temporal and spatial population distribution by the diffusion equation with the source term (ru for Malthusian growth or $ru(1-u/K)$ for logistical growth).

The common expression appearing in the right side of each of the heat/diffusion equations $((4),(7)$, and $(8))$, obtained by adding all of the nonmixed second-order spatial partial derivatives of the function u (or just the second derivative if u has one space variable), is a very important differential operator known as the **Laplace operator** or the **Laplacian**, and is denoted Δu. It is named after the French mathematician Pierre Simon de Laplace. [3] Using the Laplace operator, all three of the heat equations ((4), (7), and (8)) can be expressed in same appealing form:

$$u_t = k\Delta u \tag{2.9}$$

[3]Laplace and Lagrange worked on many of the problems of mechanics introduced by Euler and were among the first mathematicians to success-

Figure 2.7(a): Pierre Simon de Laplace (1749-1827), French mathematician and astronomer.

Figure 2.7(b): Joseph Louis Lagrange (1736-1813), French/Italian mathematician.

If we consider **steady-state** heat distributions, for which time is no longer a relevant independent variable, we get the so-called **Laplace equation**

$$\Delta u = 0, \tag{2.10}$$

which could have any number of space variables. The Laplace equation arises in many applications which, apart from celestial mechanics, include electromagnetism, fluid mechanics, and atomic physics. In fact, the study of the Laplace equation alone is such a fertile area of mathematics that it has its own name, "potential theory," and this field has developed into a major area of mathematics.

To introduce some concepts, we retum now to the one-dimensional heat equation (4) (or (9)) for the heat distribution on the rod in Figure 11.5. Just like with ordinary differential equations, a PDE will in general have infinitely many solutions. In fact, the variety of solutions for a PDE is usually much greater than that for an ODE, so much so that it is often not feasible to write down the general solution of a PDE explicitly. The next example gives a sample supply of solutions for the one-dimensional heat equation.

Example 2.3. ■

Show that each of the following functions $u(x,t)$ solves the heat equation (4) $u_t = ku_{xx}$, where $k > 0$ is a constant. Here a and b can be any real numbers.

(a) $u(x,t) = ax + b$ (c) $u(x,t) = e^{-kt}\left[ae^{\sqrt{k}x} + be^{-\sqrt{k}x}\right]$

(b) $u(x,t) = e^{-kt}[a\cos(\sqrt{k}x) + b\sin(\sqrt{k}x)]$ (d) $u(x,t) = \exp\left[-(x-a)^2/4kt\right]/\sqrt{t}$

SOLUTION: Each part is just a computation and comparison of partia derivatives, so we do only part (d) (the most complicated one) and leave the rest a an exercise. Writing u as $\exp\left[-(x-a)^2t^{-1}/4k\right]\cdot t^{-1/2}$, the chain and product rule give that

$$\begin{aligned} u_t =& \exp\left[-(x-a)^2t^{-1}/4k\right](x-a)^2t^{-2}/4k\cdot t^{-1/2} \\ &- \exp\left[-(x-a)^2t^{-1}/4k\right]\cdot(-1/2)t^{-3/2} \\ =& \exp\left[-(x-a)^2t^{-1}/4k\right]\cdot t^{-3/2}\left\{(x-a)^2t^{-1}/4k + 1/2\right\} \end{aligned}$$

In the same fashion,

$$\begin{aligned} u_{xx} =& \partial/\partial x\{u_x\} = \partial/\partial x\left\{\exp\left[-(x-a)^2/4kt\right]\cdot t^{-1/2}\cdot(-2)(x-a)/4kt\right\} \\ =& \exp\left[-(x-a)^2/4kt\right]\cdot t^{-1/2}\cdot 4(x-a)^2/(4kt)^2 \\ &+ \exp\left[-(x-a)^2/4kt\right]\cdot t^{-1/2}\cdot(-2)/4kt \\ =& \exp\left[-(x-a)^2/4kt\right]\cdot t^{-3/2}\left\{(x-a)^2t^{-1}/4k^2 - 1/2k\right\} \end{aligned}$$

fully develop theories of partial differential equations motivated by solving such celestial problems. Laplace wrote a massive five-volume treatise, *MÃľcanique CÃľleste*, which contained a plethora of results. Laplace had unfortunately neglected to credit any of the theory in his work to any other scientists and this created some problems for him. At one point, Napoleon had criticized him for not even having credited God as the creator of the universe in his definitive work (something that was expected to be done in all significant scientific works during this era). Laplace wittily replied with the now-famous retort, "Sir, I had no need for this hypothesis." Actually in his treatise, Laplace systematically developed potential theory which has applications in many other fields apart from mechanics. Ironically, the Laplace equation was actually first introduced by Lagrange, but Laplace went so far with his own extensions that the equation has been named in his honor.

Lagrange grew up and was educated in Turin, Italy and later accepted, by a personal offer from King Frederick the Great, a position at the Berlin Academy (which Euler vacated when he went to Saint Petersburg). After 20 years at this post, Lagrange moved to Paris. In his famous work, *MÃľcanique Analytique*, Lagrange unified the theory of mechanics in a way that made analysis much easier and "coordinate free." Using this approach he was able to do significant work on the "three-body problem" which won him, in 1764, the Grand Prize of the French Academy of Sciences. Lagrange is also credited for having invented the metric system. After his first wife died, he was remarried at age 56 to a teenage daughter of a friend of his and he became very conscious of his health. He never drank alcohol and became a vegetarian. To honor his scientific accomplishments, Lagrange was buried in the majestic PanthÃľon (along with literary greats Voltaire, Victor Hugo and ÃĽmil Zola) in Paris.

Comparing these two expressions we clearly have $u_t = ku_{xx}$ as desired.

Another important property of the heat/diffusion equation is that it is **linear**, as is the Laplace equation. The definition of linear a PDE is similar to that for an ODE; it is tantamount to the requirement that each term involving the unknown function (or one of its partial derivatives) must have only one such appearance of the function to the first power multiplied by some function of the independent variables (i.e., terms like $\sin u, u^2, uu_x, 4/u_{xx}$ are not allowed). Since we will mostly be working with second-order PDEs with two independent variables, we write down the most general linear PDE of this form (with independent variables $\boldsymbol{x}$ and y):

$$\begin{aligned} a(x,y)u_{xx} + b(x,y)u_{xy} + c(x,y)u_{yy} + d(x,y)u_x \\ + e(x,y)u_y + f(x,y)u = q(x,y) \end{aligned} \tag{2.11}$$

Here $u = u(x,y)$ is the unknown function and a,b,c,d,e,f,q are known functions of the independent variables x and y. (Of course for the heat/diffusion equation, we would replace y with t). If the function $q(x,y)$ is zero, then the linear equation (11) is further said to be **homogeneous**. Linear equations are important because they are often more amenable to numerical methods (recall the similar situation with BVPs of Chapter 10). Also, as we had seen in Chapter 10 for linear homogeneous ODEs, linear homogenous PDEs also satisfy the following **superposition principle**.

THEOREM 2.1. *(Superposition Principle):*[4]

If $u_1, u_2, \cdots$ are solutions of a linear homogenous PDE and $c_1, c_2, \cdots$ are constants, then $c_1u_1 + c_2u_2 + \cdots$ is also a solution of the PDE.

Sketch of Proof: We outline the proof only for the case of the second-order equation (11) with $q(x,y) = 0$ (the proof in the general case is just more writing but uses the same ideas). Furthermore, since the main ideas were already encountered in Section 10.1, we will leave similar parts of this proof as exercises. Let's rewrite the left side of (11) as the operator $L[u]$, thus (11) can be written as $L[u] = 0$. The main idea is to show that $L[u]$ is a **linear operator** in u; this means that the following two conditions hold for functions u_1 and u_2 which have second partial derivatives (so L can be computed) and for a constant c_1 :

$$\text{(i) } \mathbf{L}[u_1 + u_2] = \mathbf{L}[u_1] + \mathbf{L}[u_2], \text{ and } \quad \text{(ii) } \mathrm{L}[c_1u_1] = c_1\mathbf{L}[u_1].$$

We will leave the proofs of (i) and (ii) as exercises. From (i) and (ii) we can easily get the superposition principle; for example for a two-term sum, supposing that u_1 and u_2 are solutions of $L[u] = 0$, we have,

$$L[c_1u_1 + c_2u_2] \underset{\substack{\nwarrow \\ \text{by(i)}}}{=} L[c_1u_1] + L[c_2u_2] \underset{\substack{\nwarrow \\ \text{by(ii)}}}{=} c_1L[u_1] + c_2L[u_2] \underset{\substack{\nwarrow \\ \text{since } u_1, u_2 \text{ are solutions}}}{=} c_10 + c_20 = 0$$

which proves that $c_1u_1 + c_2u_2$ is also a solution. This proves the superposition principle with two terms. The general case now follows by induction.

EXERCISE FOR THE READER 2.2.

Prove that the operator $L[u]$ defined by the left side of (11) is a linear operator, i.e., that it satisfies (i) $L[u_1 + u_2] = \mathbf{L}[u_1] + L[u_2]$, and (ii) $L[c_1u_1] = c_1L[u_1]$.

Second-order PDEs are usually classified into three major types: elliptic, parabolic, and hyperbolic. Many common methods and concepts can be developed for all second-order (even nonlinear) PDEs in one of these three classes and, furthermore, PDEs of different types usually have some significant differences. We give the classification for the second-order linear PDEs of form (11). The classifications are entirely in terms of the coefficients a,b,c of the highest (second) order derivatives of the unknown function u : The PDE (11)

$$a(x,y)u_{xx} + b(x,y)u_{xy} + c(x,y)u_{yy} + d(x,y)u_x + e(x,y)u_y + f(x,y)u = q(x,y)$$

is said to be **elliptic** if $b^2 - 4ac < 0$,
parabolic if $b^2 - 4ac = 0$,
and **hyperbolic** if $b^2 - 4ac > 0$.

Generally speaking, elliptic PDEs describe physical processes that are in a steady-state and so do not depend on time, parabolic PDEs describe physical processes (such as diffusion of heat or a gas) which evolve toward a steady-state equilibrium, and hyperbolic PDEs describe time-dependent physical processes (such as motion of waves) which are not tending

[4]The sum in this theorem is intended as a finite sum; however, with extra hypotheses the superposition remains true for certain infinite sums. This leads to Fourier series solutions of linear PDEs, which is an important topic in many theoretical PDE courses. Actually solving a PDE problem in terms of Fourier series still leaves the numerical problem of evaluating the (infinite) Fourier series to within a tolerated maximum error. Although this could be worked into MATLAB routines, there are more effective numerical schemes, so we will not go further with this approach.

to settle into a steady-state. These terms are used throughout the subject of PDEs and have formulations for higher-order as well as nonlinear PDEs. The type of a linear PDE is entirely determined by looking at the so-called **discriminant** of the coefficients, $b^2 - 4ac$. At each point, (x,y), the PDE (11) will be of exactly one of these three types; however, it is possible for the type to change when (x,y) lies in different regions in the plane.

Example 2.4. ■

Show that the one-dimensional heat/diffusion equation $u_t = ku_{xx}$ is a parabolic PDE, that Laplace's equation $\Delta u = u_{xx} + u_{yy} = 0$ is elliptic, and that the one-dimensional **wave equation** $u_{tt} = c^2 u_{xx}$ is hyperbolic.

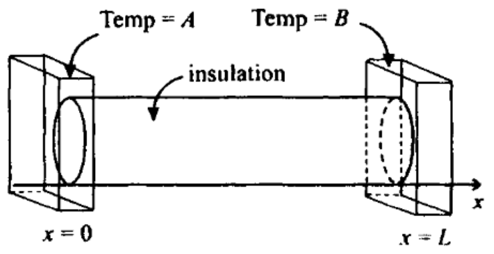
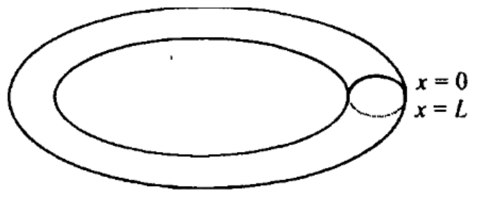
SOLUTION: If we put each of these three equations in the form (11), we see that for the heat/diffusion equation, $a = k, b = c = 0$, so $b^2 - 4ac = 0$ and thus it is parabolic. For Laplace's equation $a = c = 1, b = 0$, thus $b^2 - 4ac = -4 < 0$ so it is elliptic. Finally, for the wave equation, $a = 1, b = 0, c = -1$, so $b^2 - 4ac = 4 > 0$ and it thus is hyperbolic.

We will say more about the wave and hyperbolic equations later. We note that the Tricomi PDE $yu_{xx} + u_{yy} = 0$ has $a = y, b = 0, c = 1$ and so the discriminant $b^2 - 4ac = -4y$ shows the Tricomi PDE to be hyperbolic when $y < 0$, parabolic when $y = 0$, and elliptic when $y > 0$.

In order to specify a unique solution for a PDE, certain auxiliary conditions must also be specified. The acceptable types of auxiliary conditions that will result in existence and uniqueness theorems are varied and different for each type of PDE. If the type of auxiliary conditions given with a certain PDE will always result in existence and uniqueness of solutions for the PDE problem, we say that the PDE problem is **well-posed**. If the auxiliary conditions are either too demanding (so no solution will exist-nonexistence) or too lax (so many solutions existnonuniqueness), then we say that the PDE problem is **ill-posed**. Most problems that arise in applications are well posed. We proceed now to give some auxiliary conditions that will result in well-posed problems for the one-dimensional heat/diffusion equation (with or without source term) and then for the twodimensional Laplace equation.

We begin with the heat equation and the model being the temperature of the rod in Figure 11.5. In order to know the temperature function $u(x,t)$ at all times $t \geq 0$ and at all cross sections $x, 0 \leq x \leq L$, we will firstly need to know the temperature distribution on the rod at time $t = 0$ (initial temperature distribution): this is the function $u(x,0) = f(x)$ (a function of one variable). Also, we will need to know what is going on at the ends of the rod. We will call this information the **boundary conditions (BCs)**. There are a number of acceptable (and physically significant) boundary conditions which give rise (along with the heat PDE and initial temperature distribution) to well-posed problems. Table 11.2 gives some of the more important ones:

TABLE 11.2: Boundary conditions for the one-dimensional heat equation $u_t = ku_{xx}$, which, when given along with the initial temperature distribution $u(x,0) = f(x)$, result in well-posed problems.

Type of BC	Mathematical Equations	Illustration
Constant temperatures at the boundary, maintained by temperature reservoirs (heaters/coolers) at each boundary.	$u(0,t) = A$ and $u(L,t) = B$ for all $t \geq 0$	 Temp = A Temp = B insulation x $x = 0$ $x = L$
Insulated boundaries	$u_x(0,t) = 0$ and $u_x(L,t) = 0$ for all $t \geq 0$	Same picture as above, but ends are insulators rather than temperature reservoirs.
Periodic boundary conditions. This is valid, inparticular, when the rod is a loop.	$u(0,t) = u(L,t)$ and $u_x(0,t) = u_x(L,t)$ for all $t \geq 0$	

To give some acceptable boundary conditions for the two-dimensional Laplace equation we will again refer to the model of $u(x,y)$ being a steady-state temperature distribution (time independent) of some thin plate in the twodimensional xy-plane. We will denote the region as D and for simplicity here we let D be the rectangle $D=\{(x,y): \quad 0\le x\le a, 0\le y\le b\}$. The most common boundary conditions that result in a well-posed problem are where the temperatures on the boundary are specified. This type of boundary condition is often called a **Dirichlet boundary condition** and is illustrated in Figure 11.8. What this means is that if the temperature is steady-state on a plate (does not change with time) and we know the temperature on the edges of the plate, then the temperature will be completely determined at every point inside the plate. This can be proved and is made plausible by physical or thermodynamic principles. Another common problem results from the Laplace equation on a region with the boundary being insulated. For the rectangle of Figure 11.8, this would correspond to the following boundary conditions: right edge: $u_x(a,y)=0$; top edge: $u_y(x,b)=0$; left edge: $u_x(0,y)=0$; and bottom edge: $u_y(x,0)=0$. Such boundary conditions are often called **Neumann boundary conditions**. The solutions of such Neumann problems are not unique, since any constant function can be added to a solution to yield a different solution.

Look at Figure 11.8 and convince yourself of why each of these conditions corresponds to zero temperature gradients across the boundary edges. In fact, it is even permissible to stipulate that some parts of the boundary be given Dirichlet conditions and others be given Neumann conditions. Even when the interfaces of the adjacent parts of the boundary have discontinuities (breaks) in the boundary conditions, we will still have a well posed problem. This could correspond, for example, to some parts of the boundary being insulated and others being kept at certain temperatures. Under very general circumstances, the well-posed problems that we have specified to the heat and Laplace equations also give well-posed problems for general parabolic and elliptic equations respectively.

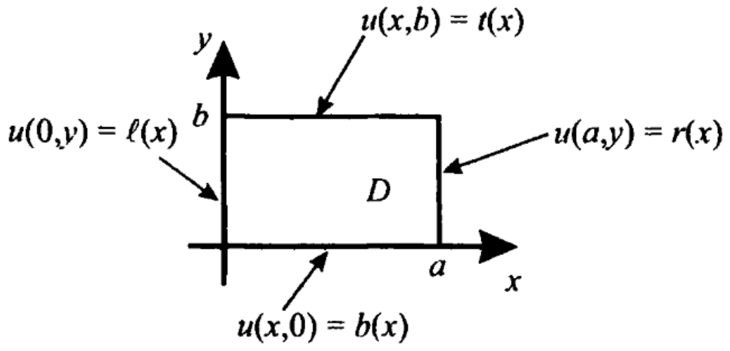


Figure 2.8: Dirichlet boundary conditions for the two-dimensional Laplace equation $u_{xx}+u_{yy}=0$ are specified on each of the four sides making up the boundary of a rectangular domain in the plane. If the four functions specified on each edge are continuous, then they give a well-posed problem for the Laplace equation.

Solutions of the Laplace equation $\Delta u=0$ (in any number of space dimensions) are known as **harmonic functions**. They include an incredibly vast collection of functions and are the subject of the very fruitful field of **potential theory**. Many aspects of potential theory have natural extensions to elliptic equations (even for nonlinear PDE). The most complete general reference on this is [GiTr-83], which is quite an advanced textbook. For example, any complex analytic function (the subject of the field of complex variables) will have harmonic functions as its real and imaginary parts. A fundamental result of potential theory is that harmonic functions satisfy the **maximum principle**, which roughly states that the maximum (or minimum) value of a harmonic function on any given (closed) bounded region in the space variables (like a rectangle) must be attained on the boundary and, furthermore, if the maximum is attained also at a point inside the boundary of the region then the harmonic function has to be a constant. In particular, applied to our steady-state temperature model for the rectangle, this says that the hottest spot on the rectangle must be on one of the edges. Again, this can be corroborated from thermodynamic principles.

EXERCISE FOR THE READER 11.3: Which of the following functions harmonic?

(a) $u(x,y,z)=1+x+2y+3z$ (c) $u(x,y)=x^2-y^2$
(b) $u(x,y)=x^2+y^2$ (d) $u(x,y)=\log(r),\ r=x^2+y^2$

Since elliptic equations are the best understood and behaved, we will begin our numerical methods for solving PDEs with elliptic equations and this will be done in the next section. In the next chapter, we will give related methods for parabolic and hyperbolic equations and so, in particular, we will hold off until the next chapter to give some general comments about the wave equation and hyperbolic equations.

EXERCISES 2.2.

1. For each of the PDEs given below, indicate its order and state whether it is a linear PDE. For those which are second-order linear and have two independent variables, state the type. If the type changes for different values of the independent variables, indicate precisely how the type varies with the independent variables.

(a) $u_x + tu_t = e^t, u = u(x,t)$ (c) $u_{xxyy} - u_{zzzz} = 0, u = u(x,y,z)$
(b) $u_{xx} - 2u_n + u_{tx} + t^2u_t = u, u = u(x,t)$ (d) $\Delta(\Delta u) = 0, u = u(x,y)$

Note: The operator on the left side of the PDE of (d) is called the **biharmonic operator** and its solutions are called **biharmonic functions**.

2. Repeat Exercise 1 for each of the following PDEs:

(a) $u_x + e^u = t, u = u(x,t)$ (c) $u_{xx} + u_{xx}u_{yy} = 0, u = u(x,y)$
(b) $u_{xx} + u_{yy} = u_{zz}/\left(1+t^2\right), u = u(x,y,z)$ (d) $u_{xx} = (u_{yy} + u_y + u)\sin x, u = u(x,y)$

3. Determine which of the following functions solve the PDE $\Delta u = 2u$:

(a) $\sin x \cos y$ (c) $\exp(\sqrt{3}x)\sin x$
(b) $\exp(x+y)$ (d) $\arctan\left(x+y^2\right)y$

4. Determine which of the following functions solve the wave equation $u_t = u_{\mathrm{x}}$:

(a) $x + 2t + 3$ (c) $\exp(x+2t) - \exp(2x-t)$
(b) $\exp(x+t) - \exp(x-t)$ (d) $f(x+t), f =$ any twice differentiable function.

5. TRUE or FALSE?: If $u(x,y)$ and $v(x,y)$ are harmonic functions, then so is $u(x,y) + v(x,y)$. Either indicate why it is true or give a counterexample if it is false.

6. TRUE or FALSE?: If $u(x,y)$ is a harmonic function, then so is $u(x,y)^2$. Either indicate why it is true or give a counterexample if it is false.

7. The function $u(x,t) = \exp\left[-(x-a)^2/4t\right]/\sqrt{4\pi t}$, which is a solution of the one-dimensional heat equation $u_1 = u_{xx}$ (this was seen in Example 11.3; here $k = 1$ and we are multiplying the function given there by a constant, so it will still satisfy the heat equation), is called the **fundamental solution** of the heat equation. It corresponds to the solution of the heat problem on a very long rod (mathematically think of an infinite rod) with the initial heat distribution being a very hot heat blast focused all at the section $x = a$. To get some idea of how this temperature distribution changes with time, do the following:
(a) Using $a = 1$, get MATLAB to plot snapshots of the temperature distribution at the following times: $t = 0.05, t = .1, t = .2, t = 1, t = 2, t = 10, t = 20$. Based on the graphs, what seems to be happening as time advances?
(b) For each of the above values of t, use the MATLAB function quad to integrate the snapshot at time t temperature distribution, which will be a function of x (impossible to integrate explicitly). Each of these integrals is improper (since the x-interval is unbounded), but integrate on a large enough interval so that the improper integral will be adequately approximated. What do these integrals seem to be doing? What is your interpretation?

NOTE: The problem of a very long rod for heat equation mentioned in the last exercise effectivel removes the issue of the boundaries of the rod. It turns out that this problem is well posed as long as a initial temperature distribution $f(x) = u(x,0)$ is given which is continuous and decays to zero a $x \to \infty$. Furthermore, the resulting solution of this problem can be expressed as an integral involvin the fundamental solution as follows:

$$u(x,t) = \int_{-\infty}^{\infty} f(s)\exp\left[-(x-s)^2/4t\right]/\sqrt{4\pi t}ds$$

8. Given the initial temperature distribution $u(x,0) = f(x)$ on a very long rod where

$$f(x) = \begin{cases} 1, & \text{if } |x| \le 1 \\ 2-|x|, & \text{if } 1 < |x| \le 2, \\ 0, & \text{if } |x| > 2 \end{cases}$$

(a) get MATLAB to plot the initial temperature distribution $u(x,0)$ along with the following temperature distribution profiles $u(x,0.1), u(x,0.2), u(x,1), u(x,2), u(x,10)$ all over the x-range $-10 \le x \le 10$. You can put them all in single plot (with different colors/styles) or use a subplot.
(b) Get MATLAB to plot a surface plot of the temperature function $u(x,t)$ over the range $-10 \le x \le 10, 0 \le t \le 10$. Suggestion: Use the integral formula of the above note and make creative use of MATLAB's quad function.

9. Repeat both parts of Exercise 8, using instead the function

$$f(x) = \begin{cases} 5(1-|x+5|), & \text{if } -6 \le x \le -4 \\ 10(1-|x-5|), & \text{if } 4 \le x \le 6 \\ 0, & \text{otherwise} \end{cases}$$

10. In this exercise you will establish a more elaborate version of the superposition principle of Theorem 11.1. Suppose that $L[u]$ denotes the left side of equation (11), i.e.,

$$L[u]=a(x,y)u_{xx}+b(x,y)u_{xy}+c(x,y)u_{yy}+d(x,y)u_x+e(x,y)u_y+f(x,y)u,$$

that $u_1=u_1(x,y)$ solves the PDE $L[u]=q_1(x,y), u_2$ solves $L[u]=q_2(x,y)$, and so on and suppose that $c_1,c_2,\cdots$ are constants. Show that the function $c_1u_1+c_2u_2+\cdots$ solves the PDE $L[u]=c_1q_1+c_2q_2+\cdots$ (where the sums are assumed finite).

2.3. FINITE DIFFERENCE METHODS FOR ELLIPTIC EQUATIONS

We will be focusing attention mainly on variations of the **Poisson equation**

$$\Delta u=q(x,y), u=u(x,y). \tag{2.12}$$

This linear elliptic equation specializes to the Laplace equation in case $q=0$. Finite difference methods in general work very nicely for boundary value problems given on domains of "nice" shape, the ideal example being a rectangle. In Chapter 13 we will give a different method, called the finite element method, that works better in oddly shaped domains. The finite difference methods in this section will be based on the following central difference formula for approximating second derivatives:

$$f''(x)=\frac{f(x+h)-2f(x)+f(x-h)}{h^2}+O\left(h^2\right) \tag{2.13}$$

We proved this formula (and others like it) in Section 10.4 using Taylor's theorem (see Lemma 10.3). Recall the "big O " notation means that the error of the approximation is less than a constant times h^2. The idea of the finite difference method for a rectangular domain can be briefly described as follows. We form a grid of points inside the rectangle, with N equally spaced x-coordinates and M equally spaced y-coordinates. We replace each of the partial derivatives in the PDE (12) by central difference quotient obtained using (13) where the terms in the quotient come from adjacent grid points. This gives a (very large) linear system of $N\cdot M$ unknowns (being the approximate values of our solution at the grid points). The boundary conditions will be enough to make the linear system well posed. We label the grid points in an efficient manner so as to make the resulting matrix for the linear system a banded matrix. We then use an efficient matrix equation solver (which may take advantage of the special form of the matrix) and solve the linear system to get an approximation of the solution to the BVP. Rather than explain the method more completely in a general fashion, we will introduce it by going through some specific examples involving Dirichlet BCs. The next section will delve into other boundary conditions for elliptic problems. Similar methods can be developed for parabolic and hyperbolic BVPs, but because stability is more often a problem for such BVPs, we put them off until the next chapter.

Example 2.5. ■

Use the finite difference method with $N=4$ interior grid values on the x-axis and $M=9$ interior grid values on the y-axis, to solve the following steady-state temperature distribution problem:

$$\begin{cases}\text{(PDE)} & \Delta u=0, \quad u=u(x,y) \quad 0<x<0.5, 0<y<1\\ \text{(BC)} & u(x,1)=4\equiv t(x), u(x,0)=16x^2\equiv b(x)\\ & u(.5,y)=4\equiv r(y), u(0,y)=4y\equiv \ell(y)\end{cases}$$

Create a surface plot of the resulting approximation to the solution.

SOLUTION: In this problem we are given the temperature reading of all of the edges of a rectangular plate and need to find the temperature at all of the interior points. Figure 11.9 summarizes graphically the given boundary conditions.

Since we will be using $N=4$ grid points equally spaced inside the x-interval [0, .5], this means that there will be a spacing of $h=(.5-0)/(N+1)=.5/5=.1$. Similarly, there will be a spacing of $k=(1-0)/(M+1)=1/10=1$ between the $M=9$ grid point inserted inside the y-interval $[0,1]$. We label the x-grid points as $x_1,x_2,\cdots,x_N$ and for convenience add in the two extra endpoint grid values x_0 (left endpoint) and x_{N+1} (right endpoint), Thus the x-grid values are:

$$x_0=0,x_1=.1(=h),x_2=.2(=2h),x_3=.3(=3h),x_4=.4(=4h),x_5=.5(=5h)$$

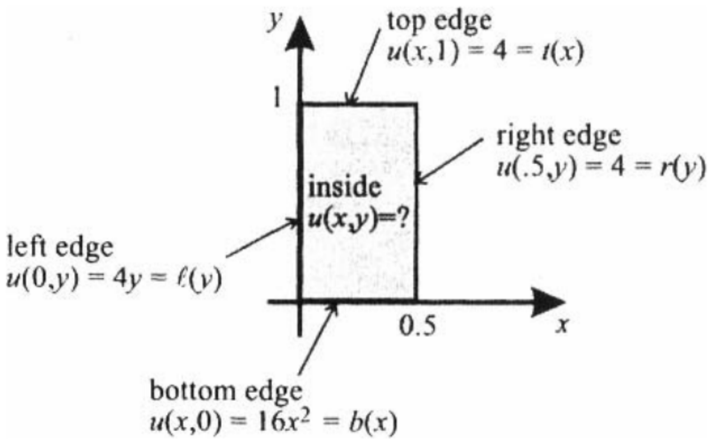


Figure 2.9: Illustration of the steady-state heat problem of Example 11.5. We are given the temperatures on each of the edges of the gray rectangle and we must solve for the temperatures at all of the points on the inside of the rectangle.

Doing the same for the y-grid values, we get the $11y$-grid values:

$$y_0 = 0, y_1 = .1(=k), y_2 = .2(=2k), \cdots, x_{10} = 1(=10k).$$

Our goal is to approximate the values $u(x_i, y_j)$ where both x_i and y_j are interior grid values (the boundary conditions give these values when either x_i or y_j is an endpoint grid value). For convenience, we employ the shorthand notation:

$$u_{i,j} = u(x_i, y_j).$$

Because of the equal spacing of the x-grid values ($h =$ gap between adjacent x_i 's), we can use the central difference formula (13) to write:

$$u_{xx}(x_i, y_j) \approx \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{2.14}$$

where x_i and y_j are allowed to be any interior grid values. Here of course $h^2 = 0.01$, but we wanted to record this formula in general form for future reference. In the same fashion, we obtain the analogous approximation for the second y-partial derivatives, valid when y_j is any interior grid value:

$$u_{yy}(x_i, y_j) \approx \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{k^2} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2}. \tag{2.15}$$

If we substitute these approximations into the Laplace PDE $\Delta u = 0$ of the example, and multiply through by $h^2 = k^2$, we arrive at the following system of linear equations:

$$4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = 0, \quad (1 \le i \le 4, 1 \le j \le 9), \tag{2.16}$$

This is a system of $M \cdot N = 4 \cdot 9 = 36$ equations in 36 unknowns. It is helpful to realize that each of the 36 equations in (16) involves values associated with the cross-shaped part of the grid shown in Figure 11.10, called the **(computational) stencil** for the finite difference method. At this point the only problem that remains is to solve this large linear system.

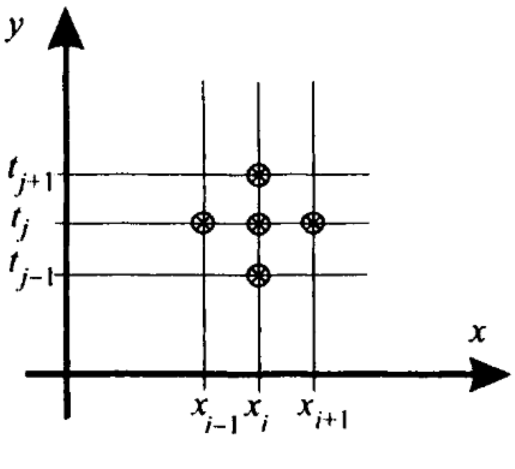
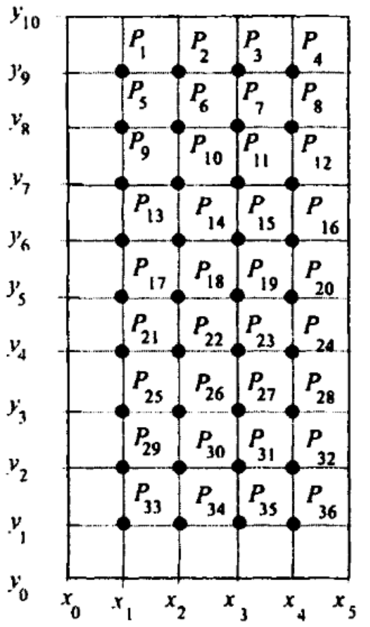


Figure 2.10: Illustration of the stencil for the linear equation (16)[5] These grid values form a cross centered at the $u_{i,j}$-grid value. Although the central grid value $u_{i,j}$ will be at a point interior to the rectangle, one or two of the other four grid values may be from boundary points.

[5]This particular stencil is often referred to as the *(standard) five-point stencil for the Laplacian*.

We will next put it in matrix form, but there are many ways to do this, depending on which order we choose to list the interior grid points (each interior grid point corresponds to one of the unknown 36 variables in our linear system). It is advantageous to number the interior grid points in a way that makes the resulting coefficient matrix look as simple as possible. One rather effective way to do this is to label the grid points in "reading order" starting at the upper left, proceeding right to the end of the first row of interior grid points, then moving down to the next row and continuing. This numbering scheme is illustrated in Figure 11.11. In order to write down a matrix equation for the linear system represented by the equations (16), we introduce the following notation for the variables of the system and the corresponding interior grid points gotten by following the labeling scheme of Figure 11.11:



y_{10}
y_9
y_8
y_7
y_6
y_5
y_4
y_3
y_2
y_1
y_0
x_0 x_1 x_2 x_3 x_4 x_5
P_1 P_2 P_3 P_4 … P_{36}

Figure 2.11: A generic and (as it tums out) very good way to label interior grid points in the finite difference method for solving elliptic PDEs.

$$P_k = (x_i, y_j), \quad U_k = u(P_k) \quad (1 \le k \le 36) \tag{2.17}$$

In general, the following relationship exists between the index k and the indices i and j (Exercise 7):

$$k = i + N(M - j) \tag{2.18}$$

For us, $N = 4$ and $M = 9$, so this becomes

$$k = i + 4(9 - j)$$

The reader should now convince himself or herself of this using Figure 11.11. With this indexing scheme, the coefficient matrix A of the linear system,

$$A \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_{35} \\ U_{36} \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{35} \\ c_{36} \end{bmatrix} \text{ or } AU = C$$

will always be diagonally banded with at most $2N + 1$ bands (thus for us in this example, at most 9 bands). To get the matrix A and the numbers $c_1, c_2, \cdots, c_{36}$ on the right column matrix C, we rewrite enough of the 36 equations in (16) to discover the patterns. We do this now, using the new variables U_k of (17) for the unknowns (and leave these on the left side) Recall that $u_{i,j}$ is known if either $i = 0$ or 5 or $j = 0$ or 10 . **TABLE 11.3**: An abbreviated list of the 36 linear equations in the 36 unknowns $U_1, U_2, \cdots, U_{36}$ arising from the finite difference method for the elliptic PDE of Example 11.5. Each row represents one of the equations (16) using the notation (17) for the unknowns on the left sides of the equation and putting the known (boundary values) on the right side.

Interior vertex	Linear equation [unknowns] $=$ [knowns]
P_1	$4U_1 - U_2 - U_5 = u_{0,9} + u_{1,10}$
P_2	$4U_2 - U_3 - U_1 - U_6 = u_{2,10}$
P_3	$4U_3 - U_4 - U_2 - U_7 = u_{3,10}$
P_4	$4U_4 - U_3 - U_8 = u_{5,9} + u_{4,10}$
P_5	$4U_5 - U_6 - U_9 - U_1 = u_{0,8}$
P_6	$4U_6 - U_7 - U_5 - U_{10} - U_2 = 0$
P_7	$4U_7 - U_8 - U_6 - U_{11} - U_3 = 0$
P_8	$4U_8 - U_7 - U_{12} - U_4 = u_{5,8}$
$\vdots$	$\vdots$
P_{33}	$4U_{33} - U_{34} - U_{29} = u_{0,1} + u_{1,0}$
P_{34}	$4U_{34} - U_{35} - U_{33} - U_{30} = u_{2,0}$
P_{35}	$4U_{35} - U_{36} - U_{34} - U_{31} = u_{3,0}$
P_{36}	$4U_{36} - U_{35} - U_{32} = u_{5,1} + u_{4,0}$

From the above equations we see that A has the following appealing "diagonally banded form":

$$A=\left[\begin{array}{ccccccccccc}
4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & \cdots & & \\
-1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & \cdots & & \\
0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & \cdots & & \\
0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & \ddots & & \\
-1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & \ddots & & \\
0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & \ddots & & \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\
 & & & & \ddots & -1 & 0 & 0 & -1 & 4 & -1 \\
 & & & & & 0 & -1 & 0 & 0 & -1 & 4
\end{array}\right],$$

i.e., A has a band of 4 's down the main diagonal, two more diagonals of -1 's which are 4 diagonals above/below the main diagonal, and finally two more diagonal bands, directly above and below the main diagonal, which repeat the pattern $-1,-1,-1,0$ (with the last zero not appearing).

For simplicity in describing the entries of C, we introduce the following vectors of known boundary values (written in general form, for us, $N=4, M=9$)

$$\begin{aligned}
L&=[\ell(y_0),\ell(y_1),\cdots \ell(y_{M+1})]=[u_{0,0},u_{0,1},\cdots,u_{0,M+1}]\\
R&=[r(y_0),r(y_1),\cdots r(y_{M+1})]=[u_{N+1,0},u_{N+1,1},\cdots,u_{N+1,M+1}]\\
B&=[b(x_0),b(x_1),\cdots b(x_{N+1})]=[u_{0,0},u_{1,0},\cdots,u_{N+1,0}]\\
T&=[t(x_0),t(x_1),\cdots t(x_{N+1})]=[u_{0,M+1},u_{1,M+1},\cdots,u_{N+1,M+1}].
\end{aligned}\tag{2.19}$$

By looking once more at the equations in Table 11.3, and using (19), we arrive at the following expression for (the transpose of) C. Please note, we are starting to use MATLAB's index notation for vectors, so, for example, $L(10)$ will mean the 10th component of the vector L that is $\ell(y_9)$ (since $L(1)=\ell(y_0)$).

$$\begin{aligned}
C' =&[L(10)+T(2),T(3),T(4),R(10)+T(5),L(9),0,0,R(9),L(8),0,0,R(8),\\
&L(7),0,0,R(7),L(6),\cdots 0,0,R(6),L(5),0,0,R(5),L(4),0,0,R(4)\\
&L(3),0,0,R(3),L(2)+B(2),B(3),B(4),R(2)+B(5)]
\end{aligned}$$

The reader is advised to convince himself or herself that this vector is correct by contemplating Figure 11.11. The column matrix C has a bit more elusive of a pattern. The beginning and end of C are a bit misleading, but the pattern of the middle terms is quite simple. The careful reader will perhaps be able to guess at what the patterns of A and C will look like in the general case (the true test would be to be able to write a MATLAB M-file that is able to do all of the above for a more general problem, not just for this specific example).

We are now ready to turn things over to MATLAB. We basically have to get MATLAB to solve the matrix equation $AU=C$ for U and then put the values of U together with the given boundary values to get the approximations for the solution on a meshgrid. Then it will be easy to plot the function. Because of the special form of A, we will be able to enter this 36×36 matrix (all 1296 entries of it) rather quickly if we take advantage of MATLAB's array operations. Let us begin by initially constructing a 36×36 matrix with 4 's down the main diagonal and then add on the remaining diagonals:

```
>> A=4*eye(36);
```

We now key in the four vectors that represent the nontrivial bands above and below the diagonal (the four bands are 1 and 4 units above and below the main diagonal).

Because of its repetitive nature, $al =$ the vector one unit above the main diagonal is easy to enter:

```
>> a1rep=[0 -1 -1 -1];
>> a1=[-1 -1 -1];
>> for i=1:8, 1=[a1 a1rep]; end
```

To check if a vector is the correct size, we can always type the size command:

```
>> size(a1);
```

$\rightarrow$ **ans = 1 35**

The vectors a4 and b4 (the same) are even more simple:

```
>> a4=-1*ones(1,32);
```

Now that we have the band vectors keyed in, the `diag` command can easily put them where we want them to be:

```
>> A=A+diag(a 1,-1)+diag(a1,1)+diag(a4,-4)+diag(a4,4);
>> x=0:.1:.5; y=0:.1:1;
>> L=4*y; B=16*x.*x; %enter left and bottom sdge boundary values
>> T=4*ones(1,6); % enter top edge boundary values
>> R=4*ones(1,11); % enter right edge  boundary valuses
```

Now that the vectors L,R,T,B are keyed in we can enter the vector C. Notice that since we want C to be a column vector, we type the transpose symbol after it (1) that changes it from a 1×36 row vector to a 36×1 column vector.

$$\begin{aligned}C' =&[L(10)+T(2),T(3),T(4),R(10)+T(5),L(9),0,0,R(9),L(8),0,0,\\&R(8),L(7),0,0,R(7),L(6),\cdots 0,0,R(6),L(5),0,0,R(5),L(4),0,0,\\&R(4),L(3),0,0,R(3),L(2)+B(2),B(3),B(4),R(2)+B(5)]\end{aligned}$$

```
>> C=[L(10)+T(2) T(3) T(4) R(10)+T(5) L(9) 0 0  R(9) L(8) 0 0 R(8) ...
L(7) 0 0 R(7) L(6) 0 0 R(6) L(5) 0 0 R(5) L(4) 0 0 R(4) L(3) 0 0 ...
R(3) L(2)+B(2) B(3) B(4) R(2)+B(5)]';
```

The left divide operation will effectively solve our matrix equation:

```
>> U=A\C;
```

In order to get MATLAB to plot our solution, we need to form a matrix Z which is 11×6 and has all of the computed values for u that we just computed, as well as the boundary values that were given in the problem (and they should be placed in the corresponding spots in the matrix Z to where they are supposed to appear on the grid). We first contruct a 10×5 matrix Z that contains the just-computed boundary values (given in the vector U) in their correct places.

We start off by forming a matrix Z of zeros of the correct size.

```
>> Z=zeros(4,9);
```

Actually the correct size of the matrix will be 9×4, but we will soon take a transpose to give it the correct size. There is a useful command in MATLAB for slipping in the entries of a vector U (say of size 1×36) into a matrix a whose dimensions multiply to the size of the vector (say Z):

```
>> Z(:)=U; Z=Z';
```

The reason we needed to take the transpose of Z (apart from correcting the size of Z) is because of the order that the entries of u were slipped into Z (top-to-bottom first, rather than left-to-right first). The following example illustrates this:

```
>> E=zeros(3,2); v=[1 2 3 4 5 6]; E(:)=v;
```

$$\rightarrow \mathrm{E} = \begin{matrix}1 & 4\\ 2 & 5\\ 3 & 6\end{matrix}$$

Next, the following commands will put the correct boundary values on the top and bottom of Z:

```
>> Z=[T(2:5); Z; B(2: 5)];
```

Note that Z will now be an 11×4 matrix; what is left to do is put the left and right boundary values on the left and right sides of Z.
Before we put the left boundary values on, it will be convenient to reverse the order of the vector L, which we can easily do by creating a new vector `Lrev` as follows:

```
>> for i=1:11, Lrev(i)=L(12-i); end
```

For R this need not be done since the components of R are all constants. We can now get the final 11×6 vector by pasting Lrev to the left of Z and R to the right; this can be achieved in MATLAB with the command:

```
>> Z=[Lrev;  Z'; R]';
```

It is a good idea to check the matrix Z (which should represent the temperatures at the grid points):

```
>> Z      4.0000 4.0000 4.0000 4.0000 4.0000 4.0000
          3.6000 3.6782 3.7571 3.8371 3.9182 4.0000
          3.2000 3.3558 3.5131 3.6731 3.8358 4.0000
          2.8000 3.0317 3.2665 3.5065 3.7517 4.0000
          2.4000 2.7044 3.0147 3.3347 3.6644 4.0000
          2.0000 2.3711 2.7533 3.1533 3.5711 4.0000
          1.6000 2.0268 2.4741 2.9541 3.4668 4.0000
          1.2000 1.6620 2.1623 2.7223 3.3420 4.0000
          0.8000 1.2588 1.7907 2.4307 3.1788 4.0000
          0.4000 0.7825 1.3111 2.0311 2.9425 4.0000
             0   0.1600 0.6400 1.4400 2.5600 4.0000
```

Things look pretty good (cf. Figure 11.9). We can next use the MATLAB command surf, to give us a graph of the surface (for the solution of our heat problem). For a technical reason in the syntax of the surf and other 3D plotting commands,[6] we need to use *yrev*, the reverse vector of *y*, for the *y*-axis vector.

```
>> for i=1:11, yrev(i)=y(12-i); end
>> surf(x, yrev, z), xlabel('x'), ylabel('y')
```

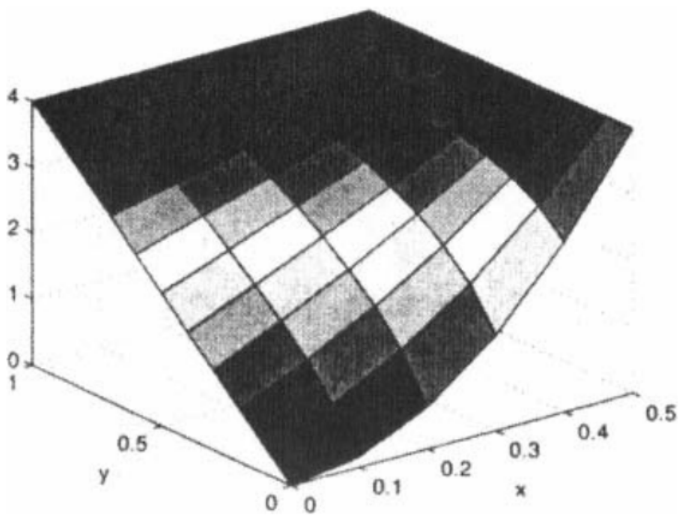


Figure 2.12: Plot of the heat function solution of Example 11.5. MATLAB will kindly color the "hot parts" of the surface red and the "cold parts" blue, as if it knew we were solving a heat equation. This comes from the default `colormap` setting.

With the example behind us, we now turn to make some general comments on numerically solving Poisson's equation $(12)\ \Delta u = q(x,y)$ on a rectangular domain. The general method just needs a few technical adjustments, but the main ideas were all covered in the above example, so we indicate only the few extra changes that might be needed in general and then we comment on writing a MATLAB Mfile to automate this procedure.

In general, the gap between x-grid values (h) need not be the same as the gap between y-grid values (k). (In the last example we had $h = k = 0.1$.) Also the function $q(x,y)$ need not be zero (as in our example). In this general case, letting $q_{i,j} = q(x_i, y_j)$, the central difference approximations applied to the differential equation $\Delta u = q(x,y)$ give the linear system:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = q_{i,j}.$$

In general the gaps h and k will be small, so to keep the terms from getting too large in this system, we multiply the equation by h^2 (alternatively k^2) and regroup to obtain (for $1 \le i \le M, 1 \le j \le N$)

$$2\left(\frac{h^2}{k^2} + 1\right) u_{i,j} - u_{i+1,j} - u_{i-1,j} - \left(\frac{h^2}{k^2}\right) u_{i,j+1} - \left(\frac{h^2}{k^2}\right) u_{i,j-1} = -h^2 q_{i,j} \tag{2.20}$$

We point out that when the x-mesh size h equals the y-mesh size k, (20) simplifies considerably to

$$4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = -h^2 q_{i,j} \tag{2.21}$$

whose left side is identical with that of (16). Apart from these small differences, the procedure given in the example is very much the same as it would be to solve a general Poisson BVP on rectangle. In the next exercise for the reader, we will test the method out on a problem where the exact solution is known.

EXERCISE FOR THE READER 11.4: (a) Use the finite difference method with $N = 4$ interior grid values on the x-axis and $M = 4$ interior grid values on the y-axis, to solve the following steady-state temperature distribution problem:

$$\begin{cases} \text{(PDE)} & \Delta u = \left(4 - \pi^2\right) e^{-2y} \sin \pi x, u = u(x,y) \quad 0 \le x \le 1, 0 \le y \le 1 \\ \text{(BC)} & u(x,1) = e^{-2} \sin \pi x \equiv t(x), u(x,0) = \sin \pi x \equiv b(x) \\ & u(1,y) = 0 \equiv r(y), u(0,y) = 0 \equiv \ell(y) \end{cases}$$

Afterwards, graph the resulting approximation to the solution.
(b) Check that $u(x,y) = e^{-2y} \sin \pi x$ solves the above BVP and thus (since it is well posed) is the unique solution. Compare the values of this exact solution with those of the approximation you obtained in part (a). Among all interior grid points find both the maximum error and the maximum relative error of the numerical solution.

[6] Recall from Example 11.1 that when a meshgrid is set up, the x-coordinates are listed in the usual order but the y-coordinates go in backwards order. When we use the finite difference method, we are setting up a meshgrid for the u-values in the usual order; thus, to get correct results with `surf`, we need to feed in the y-vector in reverse order.

(c) Redo parts (a) and (b), this time doubling N and M to be 8 .

In principle, the method described in this section for solving Poisson's equation works well for grids having up to 500 or so internal grid points. Also, if grid gaps are cut in half, we can compare the approximation with the refined grid with the approximation using the original grid (at the original internal grid points) and this can be repeated until the maximum errors (computed as in part (b) of the preceding exercise for the reader) become less than our tolerance for error so as to get a desired approximation with a confidence measurement. When the number of grid points gets larger than several hundred, the memory and storage of the matrix A starts to become a serious issue and it is better to solve the matrix equation $AU = C$ using different approaches that take advantage of the special form.

The key issue here is that for very large numbers of grid points, the percentage of entries of A which are nonzero is very small (it is less than $5n/n^2 = 5/n$); recall that such matrices are called sparse. Since our A has such a special form there are specialized algorithms (cf. the Thomas algorithm used Section 10.4) that avoid having to store the whole matrix A in its memory and solve the system much more expediently. For example, if we had 1000 internal grid points, then A would be a matrix with a million entries! But only about 5000 of these would be nonzero and it starts to become a good idea to take advantage of this structure. Most all of the matrices that arise the numerical methods that we develop for PDEs will be sparse. Readers can increase the size of problems that can be solved by utilizing sparse matrix manipulations in MATLAB as well as iterative methods. These topics were covered in Section 7.7. The same ideas work also for three (and higher)dimensional elliptic equations but it is of course a bit more of an abstract problem to visualize the three-dimensional rectangle of internal grid points.

EXERCISE FOR THE READER 11.5: (a) Write a MATLAB function M-file called poissonsol ver having the following syntax:

```
[xgrid, ygrid, zsol] = poissonsolver(g,a,b,c,d,h)
```

which will solve the Poisson equation $\Delta u = q(x,y)$ on the rectangle $a \le x \le b, c \le y \le d$ with Dirichlet boundary conditions $u = 0$ on the boundary. It should use a common mesh size h and have three output variables, two vectors `xgrid`, `ygrid` and a matrix `Zsol`. More precisely:

Inputs: `q` (inline or M-file function for $q(x,y)$, may be the zero function), `a, b, c, d, h`. The step size must divide evenly into both length and width of the rectangle. Outputs: `xgrid, ygrid, Zsol`. From these outputs you should be able to plug them into `surf` to get a graph of the numerical solution.

Note, the `xgrid` vector will look like $[a, a+h$ step, $a+2h$ step, $\ldots, b]$, and similarly for `ygrid`.

(b) Run your program to solve the Dirichlet problem $\Delta u = \sin(2\pi y)\left\{4\pi^2\left(x-x^3\right) + 6x\right\}$ on the unit square: $0 \le x, y \le 1$. First do it with a step size $h = 0.1$, and then repeat with $h = 0.02$. Compare these numerical solutions with the exact solution to this problem given by $u(x,y) = \left(x^3 - x\right)\sin(2\pi y)$. Plot surface graphs of the errors.

Finite difference methods are, in general, not so well suited for problems on domains that are not rectilinear. For Dirichlet boundary conditions on elliptic PDEs, however, finite difference methods can sometimes be used if we approximate the domain using a grid of squares. The general idea is illustrated in Figure 11.13.

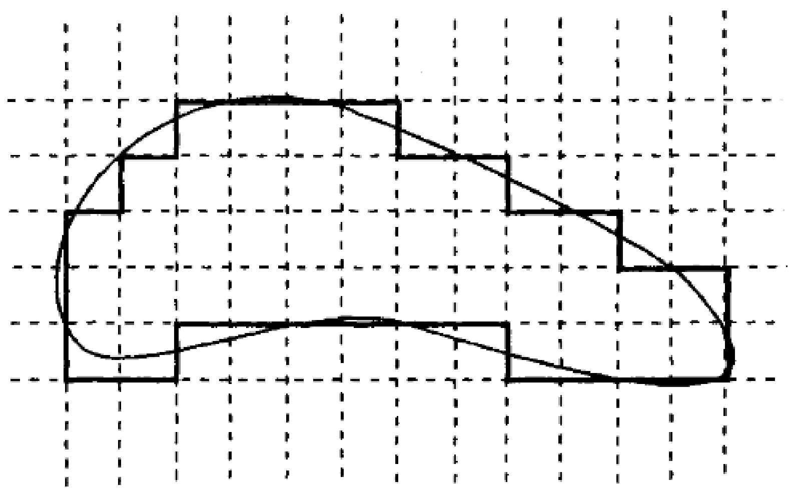


Figure 2.13: Approximation of a domain using a rectangular grid. Under certain regularity assumptions the solutions of Dirichlet problems for elliptic PDEs on arbitrary domains (curved) can be approximated by the corresponding solutions on the approximating rectilinear domain (segmented).

The details of such a scheme are not particularly difficult, but the creation of a generalcodes would be a laborious task. Such techniques work reasonably well for Dirichet problems[7] but not for boundary value problems where the BCs involve other sorts of boundary conditions (eg., Neumann or Robin). The finite element method of Chapter 13 , however, is able to handle all sorts of boundary conditions and domains. Thus, we will forgo a general development, and simply outline this scheme for a particular nonrectangular domain.

EXAMPLE 11.6: Consider the problem of finding the steady-state heat distribution on the right isosceles triangular domain shown in Figure 11.14, with the boundary temperatures as indicated in the figure. Assume the short sidelengths equal one. Use the finite difference method with $h = k = 0$. I to solve the problem numerically, and plot the solution.

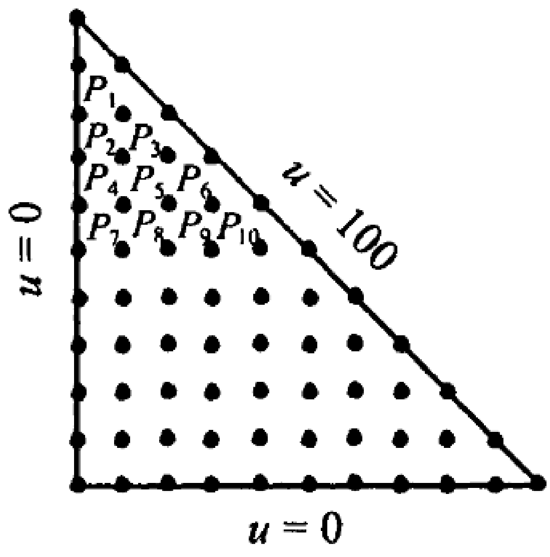


Figure 2.14: A finite difference grid for the domain of Example 11.6. The temperature distribution given is discontinuous at the upper and right comers, but these vertices will not enter into the linear system for the finite difference method. The interior nodes are labeled using the "reading order" (10 of 36 are shown).

SOLUTION: Being able to choose equal step sizes in the x - and y-directions helps a lot here since this distributes nodes nicely along the boundary. The number of interior nodes is $1+2+\ldots+8 = 36$. We label these as $P_1, P_2, \cdots, P_{36}$ using the "reading order" scheme of the last example, and invoke the analogues of all of the other relevant notations of that example. For the resulting 36 linear equations corresponding to (16), each interior node P_k give rises to an equation of the form:

$$4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = 0,$$

where $P_k = (x_i, y_j)$ corresponds to $u_{i,j}$, and the known values need to be moved to the right side. By looking at the figure, we see that the right sides of these equations will thus be either 200 (if $k = 1, 3, 6, 10, 15, 21, 28, 36$) or 0 (all other nodes). In Table 11.4, we give a few of the resulting equations in the unknowns $U_k = u(P_k)$. Although the coefficient matrix A of the linear system $AU = C$ could be easily entered by (brute) inspection, we choose to construct it instead with the following loop. The loop starts with a 36×36 diagonal matrix A and then places the -1's in appropriate places. Unlike the brute force construction, this loop easily generalizes to finer grids.

```
>> A=diag(4*ones(1,36));
border =[0 1 3 6 10 15 21 28 36];
for k=2:length(border)
  if k>2, pregap=right-left+1; end
  left=border(k-1)+1; right=border(k);
  if k<length(border)
    postgap=right-left $+1$;
  end
  for i=left:right
    if i<right %has right neighbor and top neighbor
      A(i,[i+1  i-pregap])=-1;
    end
    if i>left %has left neighbor
      A(i, i-1)=-1;
    end
    if k<length(border) %has bottom neighbor
      A(i, i+postgap)=-1;
    end
```

[7] What is required is that the domain be bounded by a finite number of smooth curves, and that the coefficient of the elliptic equation are continuous functions defined in some tube about the boundary. General theorems will then show that as the mesh size gets sufficiently small, the solution of the Dirichlet problem on the approximating domain (as in Figure 11.13) will converge to the actual solution.

```
  end
end
```

TABLE 11.4: An abbreviated list of the 36 linear equations for the finite difference method of Example 11.6.

Interior vertex	Linear equation
P_1	$4U_1 - U_2 = 200$
P_2	$4U_2 - U_4 - U_1 - U_3 = 0$
P_3	$4U_3 - U_5 - U_2 = 200$
P_4	$4U_4 - U_2 - U_7 - U_5 = 0$
P_5	$4U_5 - U_3 - U_4 - U_6 - U_8 = 0$
P_6	$4U_6 - U_5 - U_9 = 200$
P_7	$4U_7 - U_4 - U_8 - U_{11} = 0$
P_8	$4U_8 - U_5 - U_7 - U_9 - U_{12} = 0$
$\vdots$	$\vdots$
P_{33}	$4U_{33} - U_{34} - U_{32} - U_{26} = 0$
P_{34}	$4U_{34} - U_{35} - U_{33} - U_{27} = 0$
P_{35}	$4U_{35} - U_{36} - U_{34} - U_{28} = 0$
P_{36}	$4U_{36} - U_{35} = 200$

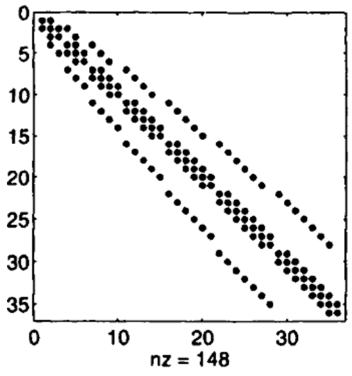


Figure 2.15: Spy plot of the coefficient matrix A for Example 11.6. The $nz = 148$ indicates the number of nonzero entries of the 1296 entries of A. We know the diagonal entries all equal 4 and all other nonzero entries equal -1. Such spy plots make the general structure of such matrices quite evident.

Since we know the numerical values of all entries of the matrix (diagonal entries $= 4$, off diagonal entries $= 0$ or -1), rather than printing the matrix A, a more practical way to view it would be using MATLAB's spy function:

spy(A) $\rightarrow$	Produces a graphic indicating the placement of the nonzero entries of a matrix A.

Thus the command spy (A) produces the plot in Figure 11.15.

From what was observed above, the right-side vector C can be easily constructed as follows:

```
>> C=zeros(36,1);
>> C(border(2:length(border)))=200;
```

and the system can be solved:

```
>> U=A/C;
```

We now can use the values of U to fill in the values of a matrix Z of the numerical values of the solution of the boundary value problem. We first form a 64×64 matrix for the interior grid points, temporarily putting $Z = 100$ for the values above the main diagonal.[8]

```
>> Z=100*ones(8) ;
>> count=1;
>> for i=1:8
  gap=border(i+1)-count+1;
  Z(i,1:gap)=U(count:(count+gap-1))';
  count=count+gap;
end
```

Next we enlarge this matrix to a 11×11 matrix which contains the given boundary values. As a compromise, we set $Z = 50$ at the interfaces of the discontinuous boundary data.

```
Z=[100*ones(1,8);100*ones(1,8);Z;zeros(1,8)) ;
Z=[[50 zeros(1,10)]', Z, [100*ones(1,10) 0 ]', [100*ones(1,10) 50]']
```

[8]This is simply a reasonable convention since we need to fill in a whole square matrix of values, even though the present problem only has actual values corresponding to the lower-left triangular part of the matrix.

In order to get surface plot on just the triangle, we will redefine the entries of Z that are off the triangle as nan, except for those nodes which are adjacent to two nodes on the slanted side of the triangle. For these latter nodes, take the average of the values at the two neighboring nodes on the slanted edge.

nan $\rightarrow$	When stored as an entry of a matrix or vector, nan (meaning: not a number) will produce a hole in any plots of this vector at the corresponding location. This is useful for three-dimensional plots over nonrectangular domains.

```
>> for i=1:11
     if i<11
       Z(i,i+1)=(Z(i,i)+Z(i+1,i+1))/2 ;
     end
     for j=i+2:11
       Z(i,j)=nan;
     end
end
```

Let us now check the matrix Z:
$\rightarrow$ **Z**

50	75	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0	100	100	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0	60.38	100	100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0	41.52	74.90	100	100	NaN	NaN	NaN	NaN	NaN	NaN
0	30.81	58.07	80.88	100	100	NaN	NaN	NaN	NaN	NaN
0	23.66	45.68	65.44	83.26	100	100	NaN	NaN	NaN	NaN
0	18.14	35.55	51.95	67.61	83.26	100	100	NaN	NaN	NaN
0	13.36	26.44	39.19	51.95	65.44	80.88	100	100	NaN	NaN
0	8.86	17.65	26.44	35.55	45.68	58.07	74.90	100	100	NaN
0	4.43	8.86	13.36	18.14	23.66	30.81	41.52	60.38	100	75
0	0	0	0	0	0	0	0	0	0	50

Notice the symmetry of this numerical data. This should be the case because of the symmetry of the given temperature distribution.

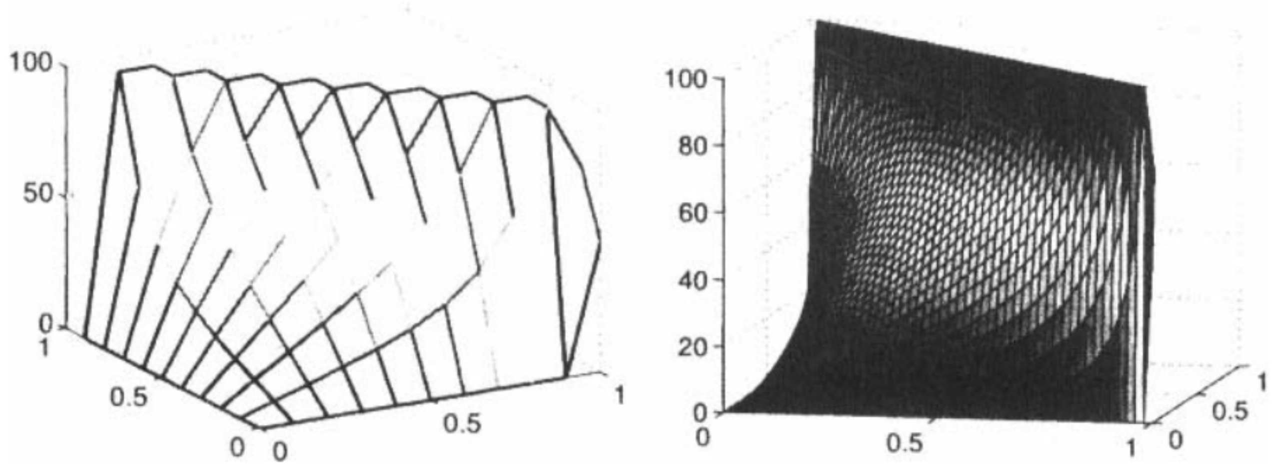


Figure 2.16: (a) (left) The mesh plot of the solution to the heat problem of Example 11.6. (b) (right) A surface plot of the same problem using a finer grid (Exercise for the Reader 11.6).

EXERCISE FOR THE READER 11.6: (a) Write a MATLAB function M-file that is designed precisely to solve the Dirichlet problem for the Laplace equation $\Delta u = 0$ on the special triangulular domain with vertices $(0,0),(1,0)$, and $(0,1)$. The syntax should be as follows:

```
[Z,x,y]=triangledirichletsolver(n,leftdata, bottomdata, slantdata)
```

where, `n`= the common number of interior grid values on the x - and y-axes $(h = k)$, and the remaining three input variables are vectors having $n+2$ components giving the boundary data on the three faces of the triangle: `leftdata` gives the boundary values at the nodes on the left face read from top to bottom, `bottomdata` gives the boundary values at the nodes on the bottom face read from left to right, and `slantdata` gives the boundary data at the nodes on the slanted face read from top to bottom. The output variables are: Z an $(n+2)\times(n+2)$ matrix of the values of the solution at the corresponding grid values of the triangle: the first column of z should thus be the vector `leftdata`, the main

diagonal the vector `slantdata`, etc. The entries of the matrix Z above the main diagonal should be NaN's. The last two output variables x and y should simply be the $(n+2)$ vectors giving the x - and y-grid values; however, y should be `surf(x,y,z)` will give a plot of the numerical solution.
(b) Test the program on the BVP of Example 11.6 but with $n = 49$ and obtain a graphic of the numerical solution as in Figure 11.16 b. Next, use the data to obtain an isotherm (contour) plot as in Figure 11.17.

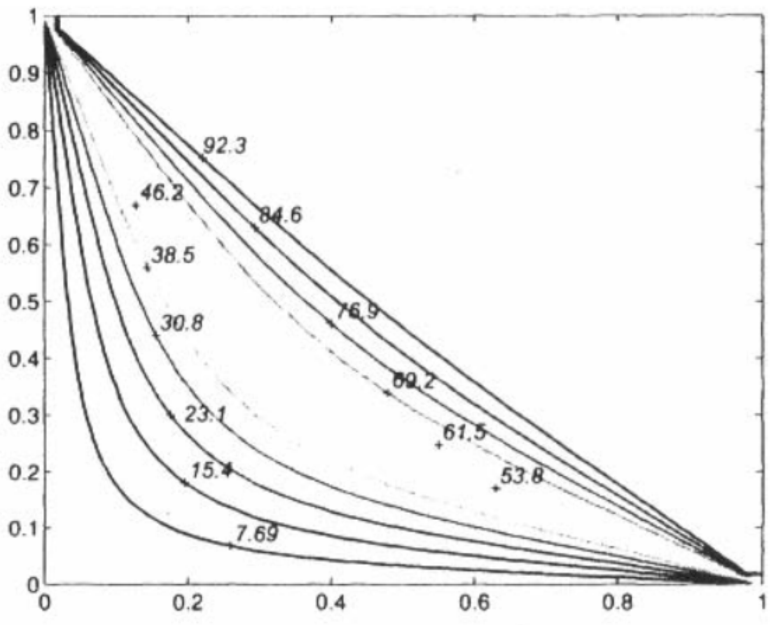


Figure 2.17: Isotherms (lines of constant temperature) for the heat problem in Example 11.6. This plot was obtained in Exercise for the Reader 11.6.

We close this section with some theoretical comments about the finite difference method applied to Dirichlet problems. From a purely linear algebraic perspective, it is not at all clear that a finite difference scheme will have a solution (i.e., if the coefficient matrix will be nonsingular). This turns out to be the case if we apply the method to the Poisson PDE on any rectilinear domain on which the boundary data is specified. We state this as our next theorem:

THEOREM 11.2: *(Existence and Uniqueness of the Finite Difference Method for Dirichlet Problems)* Suppose that the domain D is bounded, connected and rectilinear[9] and that the finite difference method is used to solve the Dirichlet problem for the Poisson equation:

$$\begin{cases} \Delta u = f(x,y) & \text{on } D \\ u = g(x,y) & \text{on } \partial D \end{cases} \qquad u = u(x,y)$$

where that data $f(x,y)$ and $g(x,y)$ are arbitrary functions (not necessarily continuous). If any grid (with $h = x$-step and $k = y$-step not necessarily equal) is used for which each corner point of the boundary of D is a node, then the finite difference method will produce a unique solution.

Proof: We give the proof for the case in which $h = k$, since the ideas present themselves most elegantly in this case. The general case will be left to the exercises. First we assume that $f(x,y) = 0$ (i.e., the Laplace equation). So we can think of the BVP as a steady-state heat distribution problem. Recall that harmonic functions (solutions of $\Delta u = 0$) satisfy the maximum principle: If $u(x,y)$ attains a maximum (or minimum) in the interior of a domain (as opposed to a boundary point), then $u(x,y)$ must be a constant function. We will show that finite difference solutions to the Laplace equation also have this important property. Indeed the finite difference scheme for the Laplace equation (16) $4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = 0$, when rewritten as

$$u_{i,j} = \frac{1}{4}\left[u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}\right],$$

can be thought of as saying that the value of the finite difference solution at an interior point will be the average of the values of the finite difference solution at the four neighbors (right, left, top, bottom; see Figure 11.10). It follows that if this finite difference solution were to have a maximum at some interior point, then each of the four neighbors would share this maximum value (if just one was less, then so would the average and hence $u_{i,j}$). We then apply this same argument to each of the neighbor nodes that are interior nodes. By the connectedness assumption, if we continue to repeat this

[9]This means that the boundary of the domain is made up of vertical and horizontal line segments only. Such domains are allowed to have holes (e.g., an L-shaped region with some rectangular holes punched out). The connectedness assumption simply means (informally) that the domain is all in one piece. More technically, it means that any two points in the domain can be joined by an arc which lies entirely in the domain. This assumption is not at all restrictive since BVPs on nonconnected domains can be broken into separate problems on connected domains.

argument, eventually all the nodes of the domain will be accounted for and shown to have the same maximum value. This proves the maximum principle for finite difference solutions.

A slight modification of this proof will prove the analogous minimum principle: If a finite difference solution attains a minimum value at an interior point, then the finite difference solution must be a constant. From these principles, it is easy to show that finite difference solutions are unique for the Poisson equation. Indeed, if $u_{i,j}$ and $v_{i,j}$ were both finite difference solutions to the above Poisson BVP, then $w_{i,j} \equiv u_{i,j} - v_{i,j}$ would be a finite difference solution of the the Laplace equation $\Delta u = 0$ and have zero boundary data (since the boundary data of $u_{i,j}$ and $v_{i,j}$ are the same). By the maximum and minimum principles, it follows that $w_{i,j} \equiv 0$ (i.e., $u_{i,j} \equiv v_{i,j}$); this proves uniqueness. Next, any finite difference solution $u_{i,j}$ is a solution of a linear system of N equations and N unknowns ($N =$ total number of interior nodes). For such a linear system (with square coefficient matrix), existence of a solution is equivalent to uniqueness of solutions (and to the coefficient matrix being invertible); see [HoKu-71].

We caution the reader that although the finite difference method can be extended to more complicated PDEs in natural ways, care must be taken with respect to the underlying mathematical theory. Such problems may not have existence and uniqueness for mathematical solutions and in such circumstances we cannot have much hope for any numerical scheme. Mathematical existence and uniqueness theory for PDE is a vast field of contemporary research and much remains to be discovered (especially for nonlinear equations). In the theorem below we give a small sampling of some existence and uniqueness theorems for elliptic boundary value problems. In each we make the underlying assumption that the domain Ω lies in, the plane and that its boundary $\partial\Omega$ is **piecewise smooth**, meaning that $\partial\Omega$ can be broken up into a finite number of pieces, each of which is the graph of a function of either x or y with continuous second derivative. We also say that a function is smooth if its second (partial) derivatives are all continuous.

THEOREM 11.3: *(Existence and Uniqueness for Some Elliptic Boundary Value Problems)* Suppose that Ω is a smooth domain in the plane.
(a) If $g(x,y)$ is a continuous function on $\partial\Omega$ then the Dirichlet problem for the Laplace equation:

$$\begin{cases} \Delta u = 0 \text{ on } \Omega \\ u = g(x,y) \text{ on } \partial\Omega \end{cases}$$

has a unique solution that is continuous on $\Omega \cup \partial\Omega$ and agrees with $g(x,y)$ on $\partial\Omega$.
(b) Suppose that the PDE (11):

$$\begin{aligned} & a(x,y)u_{xx} + b(x,y)u_{xy} + c(x,y)u_{yy} + d(x,y)u_x + e(x,y)u_y + f(x,y)u \\ & = q(x,y), \end{aligned}$$

is uniformly elliptic on $\Omega \left(b^2 - 4ac < -\delta < 0\right.$ throughout Ω, for some positive number δ), that the coefficients have piecewise continuous partial derivatives throughout Ω, and that $a(x,y) \geq 0$ and $f(x,y) \leq 0$ throughout Ω. If $g(x,y)$ has continuous second derivatives in some tube about $\partial\Omega$, then there exists a unique solution $u(x,y)$ of the PDE (11) that satisfies the BC $u = g(x,y)$ on $\partial\Omega$.

Some of the smoothness requirements can be weakened. The result of part (a), for example, can be extended to work for very general domains, including domains with fractal boundaries. The proofs of such theorems are quite involved and are not within the scope of the text; we refer the interested reader to [GiTr83]. In part (b), the requirement that $f(x,y) \leq 0$ is essential. Indeed, it is well known that for any such domain Ω the Laplace operator can have **eigenvalues** $\lambda < 0$ for which the PDE $\Delta u = \lambda u$ has nonzero solutions with zero boundary data. By analogy with matrix theory, these nonzero solutions are called (associated) **eigenfunctions**. As a simple example, in case Ω is the unit square $\{(x,y) : 0 < x,y < 1\}$, the eigenvalues are $\lambda = -\left(n^2 + m^2\right)\pi^2$ for any integers n and m, and associated eignfunctions are $u(x,y) = \sin(n\pi x)\sin(n\pi y)$. It can be readily checked that these functions satisfy the PDE $\Delta u = \lambda u$ and have zero boundary values. Since $u(x,y) = 0$ is also (an obvious) solution of this same BVP, uniqueness is violated. Furthermore, if λ is a negative number not equal to one of these eigenvalues, it can be shown that one cannot arbitrarily assign continuous boundary data for the PDE $\Delta u = \lambda u$ and always have a solution (nonexistence).[10]

EXERCISES 2.3.

[10]There is an interesting problem about these so-called eigenvalues of the Laplacian. If the planar domain Ω is thought of as a drumhead, the (negatives of these) associated eigenvalues can be shown to be natural frequencies of vibration (see Chapter 10 of [Str-92] for details). The set of all of eigenvalues of Ω, the so-called **spectrum**, can be shown to be an infinite set satisfying: $\lambda_1 \geq \lambda_2 \geq \lambda_3 \cdots \to -\infty$. This spectrum can thus be thought of as the totality of the range of tones which can be emmitted from a drumhead of shape Ω. In a famous 1966 paper entitled "*Can you hear the shape of a drum*" [Kac-66], it was asked that if one knows the spectrum of a given domain Ω, is the shape of Ω completely determined (up to congruence): In other words, do domains of different shapes have different spectra? The problem drew a lot of interest but remained open until 1992, when Carolyn Gordon, David Webb, and Scott Wolpert published their paper "*One cannot hear the shape of a drum,*" where they found a counterexample of two noncongruent planar domains with the same spectra. Despite the fact that their domains were simple polygons, their construction actually relied on some sophisticated results from group theory.

1. (a) Use the finite difference method with $N=4$ interior grid values on the x-axis and $M=9$ interior grid values on the y-axis, to solve the following steady-state temperature distribution problem:

$$\begin{cases} \text{(PDE)} & \Delta u = 0, \quad u = u(x,y) \quad 0 < x < 1, 0 < y < 2 \\ \text{(BC)} & u(x,1) = 8 \equiv t(x), u(x,0) = 0 \equiv b(x) \\ & u(.5,y) = y^3 \equiv r(y), u(0,y) = 4y \equiv \ell(y). \end{cases}$$

   Afterwards, graph the resulting approximation to the solution.
   (b) Repeat with $N=9$ and $M=19$.

2. (a) Redo part (a) of Exercise 1 using $N=9=M$ interior grid values on both the x - and y-axis.
   (b) Repeat with $N=19=M$.

3. Consider a steel alloy rectangular plate that is 10 feet long and 6 feet wide. Suppose that the bottom (10 -foot) edge is maintained at 400°F, the top edge is maintained at 250°F and both vertical edges are maintained at 150°F. Assume that the flat faces of the plate are insulated.
   (a) Use the finite difference method to solve for the temperatures within the plate using a spacing of 1 foot (= $h=k$). Plot the approximate location of the 300°F isothermal curve within the plate (i.e., the contour in the plate on which the temperature is constantly 300°F) and also the 200°F contour.
   (b) Repeat part (a) using a 4 inch step size $(=h=k)$.

4. Consider a steel alloy plate that is 10 feet long and 6 feet wide, and is insulated on its flat surfaces. Suppose the left edge is maintained at 1000°F (very hot) and the other three edges are all maintained at 50°F.
   (a) Use the finite difference method to solve for the temperatures within the plate using a spacing of 1 foot (= $h=k$).
   (b) Shade (or color, preferably in red) the part of the plate which will be over 140°F (hot part of the plate).
   (c) Repeat parts (a) and (b) using a grid spacing of 4 inches $(=h=k)$.

5. (a) Use the finite difference method with $N=4$ interior grid values on the x-axis and $M=9$ interior grid values on the y-axis, to solve the following Poisson boundary value problem:

$$\begin{cases} \text{(PDE)} & \Delta u = \sin(\pi x), \quad u = u(x,y) \quad 0 \le x \le 1, 0 \le y \le 2 \\ \text{(BC)} & u(x,2) = 6 \equiv t(x), u(x,0) = 0 \equiv b(x) \\ & u(1,y) = 3y \equiv r(y), u(0,y) = 3\cos(\pi y) \equiv \ell(y). \end{cases}$$

   Afterwards, graph the resulting approximation to the solution.
   (b) Repeat with $N=9$ and $M=19$.

6. (a) Use the finite difference method with $N=9$ interior grid values on the x-axis and $M=9$ interior grid values on the y-axis, to solve the following Poisson boundary value problem:

$$\begin{cases} \text{(PDE)} & \Delta u = x^2 + y^2, \quad u = u(x,y) \quad 0 \le x \le 1, 0 \le y \le 2 \\ \text{(BC)} & u(x,2) = 0 \equiv t(x), u(x,0) = 0 = b(x) \\ & u(1,y) = 100 \equiv r(y), u(0,y) = 100 \equiv \ell(y). \end{cases}$$

   Afterwards, graph the resulting approximation to the solution.
   (b) Repeat with $N=M=19$.

7. (a) Prove that the indexing scheme (18) $k = i + N(M-j)$ always results in the reading order of indices k for the nodes (x_i, y_j) in a rectangle.
   (b) How would the formula (18) change if we wished to index the nodes so they started on the bottom row, went left to right, and then moved up one row at a time?

8. (a) Set up the finite difference method for the steady-state heat problem for the Laplace equation $\Delta u = 0$ on the domain shown in Figure 11.18(a) with specified Dirichlet data and using a common stepsize $h=k=1$.
   (b) Get MATLAB to solve the linear system and give a surface plot of the solution.
   (c) Get MATLAB to give a corresponding isotherm plot.
   (d) Repeat parts (a) through (c) using a step size $h=k=0.5$.

9. Repeat each part of Exercise 8 on the domain of Figure 11.18(a), but change the boundary Dirichlet data as follows: $u \le 0$ on all four sides of the outer boundary, while on the four inner boundary sides, u is specified by: $u(x,3) = u(x,7) = 25(x-5)^2, \quad u(3,y) = u(7,y) = 100$.

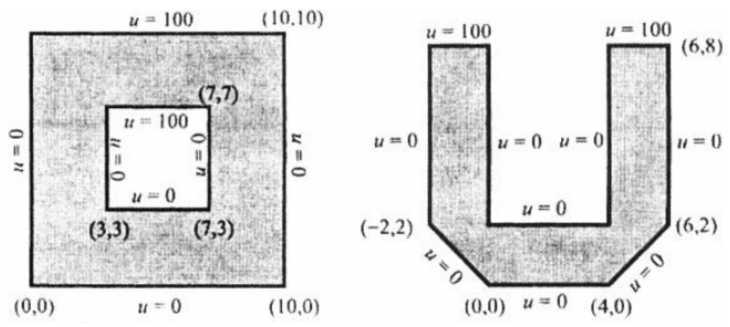


Figure 2.18: (a) (left) and (b): Two planar domains with boundary data for Dirichlet problems for Exercises 8-11.

10. (a) Set up the finite difference method for the steady-state heat problem for the Laplace equation $\Delta u = 0$ on the domain shown in Figure 11.18(b) with specified Dirichlet data and using a common stepsize $h = k = 1$.
(b) Get MATLAB to solve the linear system and give a surface plot of the solution.
(c) Get MATLAB to give a corresponding isotherm plot.
(d) Repeat parts (a) through (c) using a step size $h = k = 0.25$.

11. Repeat each part of Exercise 10 on the domain of Figure $11.18(b)$, but change the boundary Dirichlet data (only) on the four vertical sides to be linear, increasing from 0 to 100 as y increases from 2 to 8.

12. (a) Formulate a finite difference method using $N = M = 9$ interior grid values on both the x - and y-axes to solve the following elliptic boundary value problem:

$$\begin{cases} \text{(PDE)}\, (e^x u_x)_x + (e^y u_y)_y = 2e^{x+y}(e^x + e^y), 0 < x < 1, 0 < y < 1, u = u(x,y) \\ \text{(BC)} \quad u(x,0) = e^x, u(x,1) = e^{x+1}, u(0,y) = e^y, u(1,y) = e^{y+1} \end{cases}.$$

Use MATLAB to numerically solve it and compare with the exact solution $u(x,y) = e^{x+y}$. What does the existence and uniqueness theorem (Theorem 11.3) say about this problem?
(b) Solve again with the finer grid $N = M = 19$.
(c) Repeat parts (a) and (b) for the BVP obtained by the above by changing all boundary values to zero, i.e., $u(x,y) = 0$ at all boundary points.

NOTE: *(Steady-State Fluid Flow Equation)* An important BVP that arises in applications of steadystate two-dimensional fluid flow is the following:

$$\begin{cases} \text{(PDE)}\, (au_x)_x + (bu_y)_y + cu = f(x,y) \text{ on } D \\ \text{(BC)} u = g(x,y) \text{ on the boundary of } D \end{cases}.$$

Here the function $u(x,y)$ denotes the pressure, a and b (which can be functions of x and y) denote fluid conductivity coefficients in the x - and y - directions. In this setting, the vector $(-au_x, -bu_y)$ turns out to be the fluid flux and $f(x,y)$ denotes the amount of fluid being added at the point (x,y). The PDE can be derived in a similar fashion to how the heat equation was done.

13. Set up a finite difference scheme for the steady-state fluid flow problem above using the parameters: $a = b = 1, c = -1$, and $f(x,y) = x^2$. Use $N = M = 9$ interior grid points on both the x-and y-axis on the square domain $D = \{0 < x < 1, 0 < y < 1\}$. Imposing zero boundary conditions on each side, solve the linear system and plot the resulting surface. Give also a contour plot of the isopressure lines. What does the existence and uniqueness theorem (Theorem 11.3) say about this problem?

14. Repeat all parts of exercise 13 on the problem modified as follows:

$$a(x,y) = 2e^x, b(x,y) = x + y + 1, f(x,y) = 3, \text{ if } x > 0.5; = 0, \text{ if } x = 0.5; -3, \text{ if } x < 0.5$$

NOTE: *(Nine-Point Formula for the Laplacian)* The following so-called **nine-point formula**

$$\Delta u(x,y) \approx \frac{1}{6h^2}\left[\begin{array}{c} 4u(x+h,y) + 4u(x-h,y) + 4u(x,y+h) + 4u(x,y-h) \\ u(x+h,y+h) + u(x-h,y+h) + u(x+h,y-h) \\ +u(x-h,y-h) - 20u(x,y) \end{array}\right],$$

turns out to be extremely accurate, with truncation order $O(h^6)$ when used for the Laplace equation.[11] The next three exercises will examine its use.

[11]Letting $\Delta_h^{(9)} u(x,y)$ denote this approximation, it can be shown using Taylor's theorem that if u has sufficiently many continuous partial derivatives,

15. (a) Use the nine-point formula in a finite difference method to solve the Dirichlet problem

$$\begin{aligned}&\text{(PDE) } \Delta u(x,y)=0, 0<x<1, 1<y<2\\&\text{(BC) } u(x,1)=\ln\left(x^2+1\right), u(x,2)=\ln\left(x^2+2\right), u(0,y)=\ln\left(y^2\right), u(1,y)=\ln\left(y^2+1\right),\end{aligned}$$

using $N=4$ interior grid points on the x-axis and $M=4$ interior grid points on the y-axis. Look at the errors by comparing with the exact solution $u(x,y)=\ln\left(x^2+y^2\right)$. Solve also using the standard five-point finite difference formula and compare the errors.
(b) Repeat part (a), this time using $N=M=9$.
(c) Re-solve Exercise for the Reader 11.4, this time incorporating the nine-point formula. How does the performance compare (use the exact solution provided to get the errors) with that of the standard finite difference method?

16. Repeat each part of Exercise 8, but this time incorporating the nine-point formula.

17. (a) Prove Theorem 11.2 in the general case of step sizes that are not necessarily equal.
(b) Prove an extension to Theorem 11.2 to the case of the following more general boundary value problem:

$$\begin{cases} au_{xx}+bu_{xx}=f(x,y) \text{ on } D \\ u=g(x,y) \quad \text{ on } \partial D \end{cases} \qquad u=u(x,y)$$

where a and b are nonzero real numbers of the same sign. Is the result still valid if a and b are real numbers of opposite signs? How does this tie in with ellipticity?
(c) Does the extension of part (b) continue to be valid in case a and b are allowed to be (continuous) functions $a=a(x,y), b=b(x,y)$ that are of the same sign?

2.4. GENERAL BOUNDARY CONDITIONS FOR ELLIPTIC PROBLEMS AND BLOCK MATRIX FORMULATIONS

Our introduction to finite difference methods in the last section centered on elliptic problems with Dirichlet boundary conditions. Allowing more general boundary conditions will lead to related methods, all of which can be very nicely expressed in the language of block matrices. The notations and concepts of this section coupled with MATLAB's ease of handling matrices will make the task of writing MATLAB codes for finite difference methods a very natural one. Furthermore, this centralized approach will carry over well into the development of finite difference methods for other sorts of PDEs, some more of which will be examined in the next chapter.

We begin with the Dirichlet problem for the Poisson equation in two space dimensions:

$$\begin{cases} \text{(PDE)} \Delta u=f(x,y) \text{ on } \Omega, u=u(x,y) \\ \text{(BC)} \quad u=g(x,y) \text{ on } \partial\Omega \end{cases}, \tag{2.22}$$

The domain Ω will be a rectangle in the plane, which for convenience we assume has its lower-left vertex at the origin: $\Omega=\{(x,y): 0<x<a, 0<y<b\}$. The symbol $\partial\Omega$ denotes the boundary of the domain Ω, which in this case consists of the four sides of the rectangle. Unlike for the parabolic and hyperbolic problems that we will discuss in the next chapter, the relation of horizontal to vertical step sizes is not so important for elliptic problems, so for simplicity we will use equal step sizes $(h=k)$. We assume that a step size $h>0$ has been chosen so that $a=(N+1)h$ and $b=(M+1)h$. The x-grid points and y-grid points are then specified by:

$$x_i=ih(0\leq i\leq N+1), \quad y_j=jh(0\leq j\leq M+1) \tag{2.23}$$

and the corresponding functional values are then denoted by:

$$u_{i,j}=u\left(x_i,y_j\right)=u(ih,jh) \quad (0\leq i\leq N+1, 0\leq j\leq M+1) \tag{2.24}$$

We use analogous notation for the data functions of the problem: $f_{i,j}, g_{i,j}$ (of course, the latter is defined only for indices corresponding to boundary points).

---

then

$$\begin{aligned}\Delta_h^{(9)}u(x,y)-\Delta u(x,y)=&\frac{1}{2}h^2\Delta^2u(x,y)+\frac{2}{6!}h^4\left[\Delta^3u(x,y)+2(\Delta u)_{xxyy}(x,y)\right]\\&+\frac{2}{8!}\frac{h^6}{3}\left[3\Delta^4u(x,y)+16\left(\Delta^2u\right)_{xxyy}(x,y)+20u_{xxxxyyyy}(x,y)\right]+O\left(h^7\right),\end{aligned}$$

where Δ^2u means $\Delta(\Delta u)$, etc.; see [KaKr- 58]. From this it clearly follows that in case u is harmonic (Laplace equation) the approximation is $O\left(h^6\right)$, but in general it is only $O\left(h^2\right)$, and thus cannot be much better than the usual five-point approximation. Note that the approximation is still $O\left(h^6\right)$ when used for a Poisson equation $\Delta u=f(x,y)$ whenever $\Delta f=f_{xxyy}=0$, and so in particular whenever $f(x,y)$ is a polynomial of the form $A+Bx+Cy+Dxy+E\left(x^2-y^2\right)$.

Substituting the central difference approximations (14) and (15) into the PDE (22) results in the following discretization of the Poisson PDE:

$$\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}+\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h^2}=f_{i,j}(1\le i\le N,1\le j\le M). \tag{2.25}$$

Since the central difference formula employed has error $O\left(h^2\right)$, it follows that the local truncation error of the discretization (25) is $O\left(h^2\right)+O\left(h^2\right)=O\left(h^2\right)$. We rewrite (25) in the following simpler form:

$$4u_{i,j}-u_{i+1,j}-u_{i-1,j}-u_{i,j+1}-u_{i,j-1}=-h^2f_{i,j},\quad(1\le i\le N,1\le j\le M). \tag{2.26}$$

This is a linear system with NM equations in the unknown interior grid values of $u(x,y)$ that we index using the scheme (18) into the components of a vector U :

$$P_k\equiv(x_i,y_j),U_k\equiv u(P_k)=u_{i,j},k=i+N(M-j). \tag{2.27}$$

Recall that this indexing scheme results in the "reading order" labeling of the interior grid points (see Figure 11.11).

We wish to look carefully in the matrix form of this linear system:

$$AU=C. \tag{2.28}$$

Observe that we have relabeled only the unknown values of the $u_{i,j}$ that appear in (26); the Dirichlet boundary conditions of (22) give us the following data:

$$\begin{aligned}&u_{i,0}=g_{i,0},u_{i,M+1}=g_{i,M+1}\quad(0\le i\le N+1),\\&u_{0,j}=g_{0,j},u_{N+1,j}=g_{N+1,j}(0\le j\le M+1).\end{aligned} \tag{2.29}$$

If we directly incorporate the indexing scheme (27) into (26), we arrive at the following:

$$4U_k-U_{k+1}-U_{k-1}-U_{k-N}-U_{k+N}=-h^2f_k, \tag{2.30}$$

which, however, needs to be corrected in case any of the u-values in (26) is a known boundary value. If this happens, such values need to be substituted by the g-values in (29) and then moved to the right side. Let us carefully consider each case when such corrections are needed. It is most helpful to view such cases by thinking about the "reading order" indexing of the interior nodes $P_k=(x_i,y_j)$, $(k=i+N(M-j))$ (as in Figure 11.11) as well as the stencil for our finite difference method (Figure 11.10).

Case 1: P_k lies on the top row (so $j=M$). The value U_{k+N} is thus known and should be moved to the right side as $g_{i,M+1}$.
Case 2: P_k lies on the bottom row (so $j=1$). The value U_{k-N} is thus known and should be moved to the right side as $g_{i,0}$.
Case 3: P_k lies on the right edge (so $i=M$). The value U_{k+1} is thus known and should be moved to the right side as $g_{N+1,j}$.
Case 4: P_k lies on the left edge (so $i=0$). The value U_{k-1} is thus known and should be moved to the right side as $g_{0,j}$.

Note that Cases 1 and 2 cannot occur simultaneously, nor can Cases 3 and 4, but either of the last two cases can occur in conjunction with either of the first two. In light of the blocklike structure of the above cases, the $NM\times NM$ coefficient matrix A in (28) is readily expressed as an $M\times M$ **block matrix** where each block is an $N\times N$ (ordinary) matrix as indicated below:

$$A=\left[\begin{array}{c|c|c|c|cc|c} T_N & -I_N & 0_N & \cdots & & \cdots & 0_N\\ \hline -I_N & T_N & -I_N & 0_N & & & \vdots\\ \hline 0_N & -I_N & T_N & -I_N & & & \\ \hline \vdots & 0_N & -I_N & T_N & \ddots & & \vdots\\ \hline & & \ddots & \ddots & \ddots & \ddots & 0_N\\ \vdots & & & \ddots & \ddots & \ddots & -I_N\\ \hline 0_N & \cdots & & \cdots & 0_N & -I_N & T_N \end{array}\right] \tag{2.31}$$

where I_N denotes the $N\times N$ identity matrix, 0_N denotes the $N\times N$ zero matrix, and T_N is the following $N\times N$ tridiagonal matrix:

$$T_N=\begin{bmatrix} 4 & -1 & & & & \\ -1 & 4 & -1 & & 0 & \\ & -1 & 4 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & \ddots & \ddots & -1\\ & & & & -1 & 4 \end{bmatrix} \tag{2.32}$$

The block matrix A in (31) is said to have tridiagonal block structure. Using this same decomposition into blocks, the $NM \times 1$ vectors U and C of (28) can be expressed as the following juxtapositions of $MN \times 1$ vectors:

$$U = \begin{bmatrix} U' \\ U^2 \\ \vdots \\ U^M \end{bmatrix}, \quad C = \begin{bmatrix} B_1 - h^2F_1 \\ B_2 - h^2F_2 \\ \vdots \\ B_M - h^2F_M \end{bmatrix} \tag{2.33}$$

where

$$U^j = \begin{bmatrix} U_{1+(M-j)N} \\ U_{2+(M-j)N} \\ U_{3+(M-j)N} \\ \vdots \\ U_{N-1+(M-j)N} \\ U_{(M-j+1)N} \end{bmatrix}, \quad F_j = \begin{bmatrix} f_{1+(M-j)N} \\ f_{2+(M-j)N} \\ f_{3+(M-j)N} \\ \vdots \\ f_{N-1+(M-j)N} \\ f_{(M-j+1)N} \end{bmatrix}, \tag{2.34}$$

and

$$B_i = \begin{bmatrix} g_{1,M+1} + g_{0,M} \\ g_{2,M+1} \\ g_{3,M+1} \\ \vdots \\ g_{N-1,M+1} \\ g_{N,M+1} + g_{N+1,M} \end{bmatrix}, B_j = \begin{bmatrix} g_{0,M+1-j} \\ 0 \\ 0 \\ \vdots \\ 0 \\ g_{N+1,M+1-j} \end{bmatrix} (1 < j < M), B_M = \begin{bmatrix} g_{1,0} + g_{0,1} \\ g_{2,0} \\ g_{3,0} \\ \vdots \\ g_{N-1,0} \\ g_{N,0} + g_{N+1,1} \end{bmatrix} \tag{2.35}$$

Note that the coefficient matrix A is quite sparse; indeed, it is a banded matrix with 5 bands: the main diagonal, the sub- and superdiagonals, and the diagonals that lie M units above and below the main diagonal. By Proposition 7.14,A is invertible (in fact positive definite). In Section 7.7, it was shown how the SOR method can very effectively solve linear systems having A as the coefficient matrix. Such sparse solution techniques will greatly expand the resolution that we will be able to attain in our numerical PDE solution techniques. Indeed, if we wanted to have a resolution of, say, 100 interior grid values on the x - and y-axes (with a square domain), this would mean that the linear system (28) that is needed to be solved would involve a $10,000 \times 10,000$ coefficient matrix A. Even storing such a matrix would tax most home PCs; attempting to solve the system with the general Gaussian elimination method would require on the order of $(10,000)^3 = 10^{12}$ flops. Even at the rate of 1 million flops/second, this would take nearly two weeks to solve! MATLAB's left divide is able to take advantage of the special structure of positive definite matrices, like A, which arise in finite difference methods. This results in MATLAB's ability to solve such linear systems as long as it is possible to store the coefficient matrix. Furthermore, many of the matrices that arise in numerical differential equations are sparse, and so taking advantage of scarcity will permit the solution of even larger systems (see Section 7.7). In order to keep this chapter more accessible, we will be using this left divide solver in lieu of iterative methods; however, readers who have studied Section 7.7 are encouraged to apply some of the methods they have learned on the linear systems that come up in this and subsequent sections.

Our next example will demonstrate how the above notation will allow us to code this finite difference method into a succinct MATLAB program.

EXAMPLE 11.7: (a) Use the finite difference method to solve the following Poisson problem[12]:

$$\begin{cases} \text{(PDE)} & \Delta u = -f(x,y) \text{ on } \Omega = \{0 < x,y < 1\}, u = u(x,y) \\ \text{(BC)} & u(x,0) = u(x,1) = 5x \\ & u(0,y) = 0, u(1,y) = 5 \end{cases},$$

where $f(x,y)$ is defined by:

$$f(x,y) = \begin{cases} 800, \text{ if } \frac{1}{4} < x < \frac{1}{2} \text{ and } \frac{1}{2} < y < \frac{3}{4}. \\ 0, \text{ otherwise} \end{cases}$$

Use a step size $h = 0.02$.
(b) Plot the numerical solution as a surface plot.
(c) Give a two-dimensional contour plot of the solution.

NOTE: Recall that the Poisson equation models steady-state temperature distribution with a time-independent heat source $f(x,y)$. Thus, we can view the solution to the above problem as the steady-state temperature distribution on the unit square Ω with the edges maintained at the temperatures specified by the BC and with a homogeneous heat source concentrated on

[12]The reason that we used a negative coefficient in the Poisson PDE is to highlight the interpretation of the Poisson equation as a model of steady-state heat distributions with internal heat source term $f(x,y)$; cf. (6) of this chapter for the one-dimensional analogue (put $u_t = 0$). For this reason the Poisson equation is often written as $-\Delta u = f$.

the specified smaller square of sidelength 1/4. The contours of the plot in part (c) are then the isotherms of the temperature distribution.

SOLUTION: As we have been doing thus far in our development of numerically solving boundary value problems for PDEs, we will solve this problem in a way that can be easily generalized to the creation of an M-file for solving more general problems.

Notice that with $h = 0.02$, from $N+1 = 1/h$, we see that there will be $N = 49$ interior grid points on both the x - and y-axes. Thus the linear system to be solved, $AU = C$, will have a coefficient matrix of size $N^2 \times N^2$ $\left(N^2 = 2401\right)$. Creating the coefficient matrix A of (32) can be done quite efficiently using MATLAB's `diag` function[13]:

```
>> N=49;
>> A=diag(4*ones(1,N^2))-diag{ones(1,N^2-N), N)-diag{ones(1,N^2-N),âĂŤ
N);
>> %next create vector for sub\super diagonals
>> v1=-ones(1,N-1); v=[v1 0];
>> for i=1:NâĂŤ1
  if i<N-1
   v=[v v1 0];
  else
   v=[v v1]:
  end
end
>> A=A+diag(v,1)+diag(v,-1);
```

In order to create the vector C of (33), we first store the given boundary values at our grid points, using some rather obvious notation:

```
>> leftdata=zeros(1,N+2): rightdata=5*ones(1,N+2);
>> xgrid=0:.02:1;
>> topdata=5*xgrid; bottomdata=topdata:
>> Bprep1=t0pdata(2:N+1); Bpreplast=bottomdata(2:N+1);
>> C=[];
>> Fprep=zeros(1,N);
>> h=0.02;
```

We also store the following M-file for the function on the right side of the PDE:

```
function z = squareheatsource(x,y)
if x>=.25 & x<=.5 & y>=.5 & y<=.75
  z=-800:
else
  z=0;
end

>> for j=1:N
  F=Fprep:
  for i=1:N
    F(i)=-h^2*feval{'squareheatsource',h*i,1âĂŤh*j};
  end
  F(1)=F(1)+leftdata(1+j):
  F(N)=F(N)+rightdata(1+j):
  if j==1
    F=F+Bprep1;
  elseif j==N
    F=F+Bpreplast:
  end
  C=[C; F'];
end
>> $now we can assemble the matrix Z of u-values
>> U=A\C;
>> Z=zeros(N,N):
>> Z(:)=U; Z=Z';
>> z=[topdata(2:N+1); Z: bottomdata(2:N+1)]:
>> Z=[leftdata' Z rightdata']:
```

[13]Note: Some of the matrix creation commands may take a second or two to execute, depending on the speed of your computer.

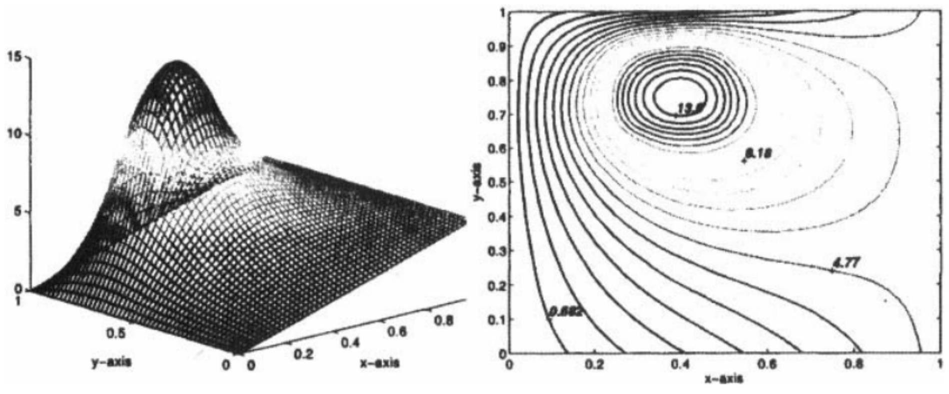


Figure 2.19: (a) (left) Temperature mesh plot for Example 11.7. (b) (right) Corresponding isotherms.

```
>> size(xgrid)
->ans = 1 51
for i=1:51
ygrid(i}=xgrid(52-i};
end %as usual, we reverse the order of y-grid for plots to be
correct.
>> mesh(xgrid,ygrid,Z)
>> hidden off, xlabel('x-axis'), ylabel('y-axis')
>> c=contour(xgrid, ygrid,Z,20):
>> clabel(c, 'manual')
```

EXERCISE FOR THE READER 11.7: (a) Write a MATLAB function M-file that is precisely designed to solve the Dirichlet problem for the Poisson equation $\Delta u = f$ on the rectangle with vertices $(0,0),(\mathrm{a},0),(\mathrm{a},\mathrm{b})$, and (0, b). The syntax should be as follows:

```
[Z, x, y]=rectanglepoissonsolver (h, a, b, varf, leftdata, rightdata, topdata,
                                  bottomdata)
```

where $h=$ the common step size on the x - and y-axes $(h=k)$ (assumed to divide into both a and $b), a$ and b are the dimensions of the rectangular domain, `varf` is either an M-file or an inline function for the function f appearing in the PDE, and the remaining four input variables are vectors for the boundary data. The first two should be column vectors of length $M=b/h+1$, with the boundary values given from top to bottom. The last two should be row vectors of length $N=a/h+$ 1 with boundary values given from left to right. The first output variables is z, an $(N+2)\times(M+2)$ matrix of the values of the solution at the corresponding grid values of the rectangle: The first column of Z should thus be the vector `leftdata`, etc. The last two output variables x and y should simply be the vectors giving the x-and y-grid values; however, y should be given in decreasing order to facilitate plotting.
(b) Test the program by re-solving Example 11.7.

We now proceed to discuss the generalized Neumann problem for the Poisson equation in two space dimensions:

$$\begin{cases}(\text{PDE})\Delta u = f(x,y) \text{ on } \Omega, \quad u=u(x,y)\\ (\text{BC})\partial u/\partial n = g(x,y) \text{ on } \partial\Omega\end{cases} \tag{2.36}$$

Here $\partial u/\partial n$ denotes the derivative of $u(x,y)$ in the direction of the outwardpointing normal vector n for points on the boundary. Recall that when this BVP models steady-state heat distribution (with time-independent heat source), the generalized Neumann boundary conditions specify the rate at which heat is lost $(g>0)$ or gained $(g<0)$ at the boundary point (x,y).[14]

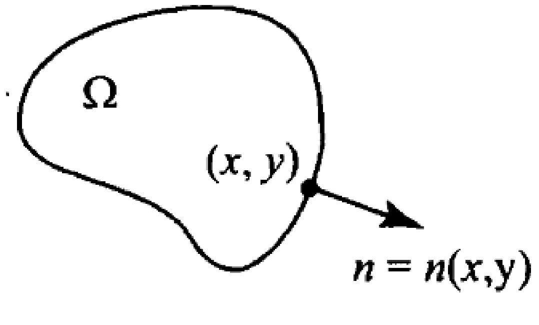


Figure 2.20: Illustration of the unit outward normal vector at a boundary point (x,y) of a planar domain Ω.

[14]Traditionally, the term "Neumann boundary conditions" is reserved for the special case tha $h(x,y)\equiv 0$ (insulated boundary).

Unlike the Dirichet problem for the Poisson equation, the Neumann problem requires an additional consistency hypothesis in order for a solution to exist. Also, it is clear that if $u(x,y)$ is a solution to the Neumann problem (36) and C is any constant, then $u(x,y)+C$ will also be a solution (why?). Apart from this we do have uniqueness. The following theorem makes these statements more precise.

THEOREM 11.4:*(Existence and Uniqueness for the Poisson PDE with Neumann Boundary Conditions)* Suppose that Ω is a smooth bounded planar domain and that in the BVP (36) :

$$\begin{cases} (\text{PDE}) & \Delta u = f(x,y) \quad \text{on } \Omega, \quad u = u(x,y) \\ (\text{BC}) & \partial u/\partial n = g(x,y) \end{cases},$$

$f(x,y)$ has piecewise continuous partial derivatives throughout Ω and $g(x,y)$ is piecewise continuous on $\partial\Omega$. Then the BVP has a solution if and only if the data satisfies the following **compatibility condition**:

$$\iint_\Omega f(x,y)dxdy = \int_{\partial\Omega} g(x,y)ds \tag{2.37}$$

where the latter integral is with respect to arc length along the boundary. In this case, the solution is unique up to an additive constant.

A few comments are in order. If we interpret the BVP (36) as a steady-state heat flow problem, the (BC) can be interpreted as requiring that the heat flux (net heat flow across the boundary) at (x,y) is given by $g(x,y)$. The right-side integral in (37) thus becomes the net heat flux lost through the boundary of the region. The left-side integral is the net heat produced within the region. For equilibrium (steady-state) the amount of internal heat produced must equal the net heat lost through the boundary (conservation of heat). This is why the compatibility condition is required. As for the nonuniqueness, this is plausible since the BVP is only stating the net heat production within the region and the net heat flux. There is no reference to how the temperature is being measured (Fahrenheit or Celsius?) or to how much heat energy is contained in the region, and so it is natural the additive constant must appear as a point of reference.

While for the Dirichlet problem on the square it is quite common for the boundary data to be continuous, for the Neumann problem it is typical that the boundary data is discontinuous at the corners. Indeed, at each corner point, the most typical situation would allow two values of $g(x,y)$, depending on from which side we are looking at normal derivatives. This simply corresponds to the fact that the direction of the normal vector (with respect to which we are measuring the rate of change of $u(x,y)$) takes a sharp (discontinuous) turn at each corner point and the rate of heat flow will change with the direction in which we are measuring it. To make the Neumann problem well-posed (have a unique solution) we could require an additional condition that the temperature be a certain value at a certain point in the domain. Vector calculus can be used to prove Theorem 11.4; some elements of the proof will appear in the exercises.

Turning to finite difference methods for solving the Neumann problem (36), we need to approximate the derivative BC using a finite difference formula. The following so-called forward difference and backward difference formulas appear to be rather plausible to this end:
LEMMA 11.5: *(Forward/Backward Difference Formulas)* (a) Suppose that $f(x)$ is a function having a continuous second derivative in the interval $a \le x \le a+h$; then we have the **forward difference approximation**:

$$f'(a) = \frac{f(a+h)-f(a)}{h} + O(h) \tag{2.38}$$

(b) Suppose that $f(x)$ is a function having a continuous second derivative in the interval $a-h \le x \le a$; then we have the backward difference approximation:

$$f'(a) = \frac{f(a)-f(a-h)}{h} + O(h). \tag{2.39}$$

The lemma is an easy consequence of Taylor's theorem (Exercise 14). A plausible way to set up a finite difference method for the Neumann problem (on a rectangle) would be by incorporating the boundary data with the forward difference scheme on the left and bottom edge nodes, and the backward difference scheme on the right and top edge nodes. While this could indeed be developed into a viable scheme, the drawback is that the $O(h)$ errors introduced by forward/backward difference portions of the schemes would contaminate the much better $O\left(h^2\right)$ local truncation errors that came up from the central difference approximation used in the internal node discretization of the PDE. A better approach is to use the central difference approximations both for the PDE and the boundary conditions. This can be accomplished by introducing additional nodes, so-called ghost nodes, as we will now explain (see Figure 11.21). The forward and backward difference schemes, however, will be of use in finite difference methods for parabolic and hyperbolic BVPs, which are studied in the next chapter.

Our next example will explain how to develop a finite difference scheme for a Neumann problem that will have local disretization error $O\left(h^2\right)$.

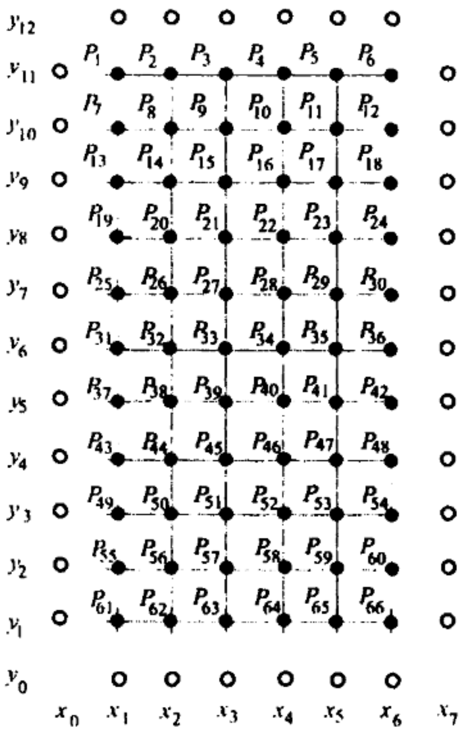
EXAMPLE 11.8: Use a finite difference method with common step size $h = 0.1$ to solve the following steady-state temperature distribution Neumann problem:

$$\begin{cases} \text{(PDE)} & \Delta u = 2, \quad u = u(x,y) \\ \text{(BC)} & u_y(x,1) = -2, u_y(x,0) = 4 \\ & u_x(.5,y) = 4 - 8y, u_x(0,y) = 0 \end{cases}$$

with the additional requirement that $u(0,0) = 0$. Afterwards, graph the resulting approximation to the solution.

SOLUTION: The reader may check that both of the integrals in (37) have a common value of 1 , so we know this Neumann problem has a solution. In contrast to the corresponding method for the Dirichlet problem (Example 11.5) the new scheme will require us to solve for the values of u at both the interior nodes as well as the boundary nodes. These nodes as well as the ghost nodes we will need are illustrated in Figure 11.21 (compare with Figure 11.11). The picture illustrates how the index scheme for the Dirichlet method should be modified. We briefly highlight the general notations that will be used:

$$x_i = (i-1)h(0 \le i \le N+1), \quad y_j = (j-1)h(-1 \le j \le M+1)$$



(note that now x_0, x_{N+1} correspond to the ghost nodes, so, as before, $x_1, \cdots, x_n$ correspond to unknown function values, and similarly for y 's). The indexing scheme is as before:

$$P_k \equiv (x_i, y_j), \quad U_k \equiv u(P_k) = u_{i,j},$$
$$k = i + N(M-j)$$

(ghost nodes are not indexed with k). The discretization of Poisson's PDE is just as before (26) (with equal step sizes)

$$4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = -h^2 f_{i,j}$$
$$(1 \le i \le N, 1 \le j \le M).$$

Figure 2.21: A grid for the Neumann problem of Example 11.7. The solid-labeled nodes are the ones (function values) that need to be solved for; the hollow nodes are the ghost nodes needed to set up the method.

In our example, of course, $N = 6, M = 11$, and $f(x,y) = -(2y)^2$, but we wish to present the general development. To allow for different Neumann data at corner points, we use the following notations for the Neumann data:

$$g^h_{i,j} = g^h(x_i, y_j) \quad \text{(horizontal side)}$$
$$g^v_{i,j} = g^v(x_i, y_j) \quad \text{(vertical side)}$$

Now we use the central difference formula to eliminate any ghost node values that appear in (26). To see how this works, let us first assume that $i = N$ (so we are dealing with a node on the right side of the rectangular domain). Then $u_{i+1j} = u_{N+1j}$ is a ghost value. The central difference formula gives us that

$$\frac{u_{N+1,j} - u_{N-1,j}}{2h} = g^v_{N,j} \Rightarrow u_{N+1,j} = u_{N-1,j} + 2hg^v_{N,j},$$

which causes (26) to become:

$$4u_{N,j} - 2u_{N-1,j} - u_{N,j+1} - u_{N,j-1} = -h^2 f_{i,j} + 2hg^v_{N,j}(1 < j < M).$$

We left out the two cases $j = 1$ and $j = M$, since these each need another ghost node to be accounted for. Analogously, these give the following:

$$4u_{N,1} - 2u_{N-1,1} - 2u_{N,2} = -h^2 f_{i,j} + 2h\left(g^v_{N,1} + g^h_{N,1}\right), \text{ and}$$
$$4u_{N,M} - 2u_{N-1,M} - 2u_{N,M-1} = -h^2 f_{i,j} + 2h\left(g^v_{N,M} + g^h_{N,M}\right).$$

Similar equations can be thus obtained for nodes on the other three sides. These considerations lead us to the linear system:

$$AU = C \tag{2.40}$$

where the $NM \times NM$ matrix A is given by:

$$A = \left[\begin{array}{c|c|c|c|c|c} W_N & -2I_N & 0_N & \cdots & \cdots & 0_N \\ \hline -I_N & W_N & -I_N & 0_N & & \vdots \\ \hline 0_N & -I_N & W_N & -I_N & & \\ \hline \vdots & 0_N & -I_N & W_N & \ddots & \vdots \\ \hline \vdots & & \ddots & \ddots & \ddots \quad \ddots & 0_N \\ \hline 0_N & \cdots & & \cdots & 0_N \quad -2I_N & W_N \end{array}\right] \tag{2.41}$$

Note that A is an $M \times M$ block matrix made up of the indicated $N \times N$ matrix blocks. Here I_N denotes the $N \times N$ identity matrix, 0_N denotes the $N \times N$ zero matrix, and W_N is the following $N \times N$ tridiagonal matrix:

$$W_N = \begin{bmatrix} 4 & -2 & & & & \\ -1 & 4 & -1 & & 0 & \\ & -1 & 4 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & -1 & \ddots & -1 \\ & & & & -2 & 4 \end{bmatrix} \tag{2.42}$$

The $NM \times 1$ vectors U and C of (28) can be expressed as the following juxtapositions of $MN \times 1$ vectors:

$$U = \begin{bmatrix} U' \\ U^2 \\ \vdots \\ U^M \end{bmatrix}, \quad C = \begin{bmatrix} 2hB_1 - h^2F_1 \\ 2hB_2 - h^2F_2 \\ \vdots \\ 2hB_M - h^2F_M \end{bmatrix} \tag{2.43}$$

where

$$U^j = \begin{bmatrix} U_{1+(j-1)N} \\ U_{2+(j-1)N} \\ U_{3+(j-1)N} \\ \vdots \\ U_{N-1+(j-1)N} \\ U_{jN} \end{bmatrix}, \quad F_j = \begin{bmatrix} f_{1+(j-1)N} \\ f_{2+(j-1)N} \\ f_{3+(j-1)N} \\ \vdots \\ f_{N-1+(j-1)N} \\ f_{jN} \end{bmatrix} \tag{2.44}$$

and

$$B_i = \begin{bmatrix} g^v_{1,M} + g^h_{1,M} \\ g^h_{2,M} \\ g^h_{3,M} \\ \vdots \\ g^h_{N-1,M} \\ g^v_{N,M} + g^h_{N,M} \end{bmatrix}, B_j = \begin{bmatrix} g^v_{1,M-j} \\ 0 \\ 0 \\ \vdots \\ 0 \\ g^v_{N,M-j} \end{bmatrix} (1 < j < M), B_M = \begin{bmatrix} g^v_{1,1} + g^h_{1,1} \\ g^h_{2,1} \\ g^h_{3,1} \\ \vdots \\ g^h_{N-1,1} \\ g^v_{N,1} + g^h_{N,1} \end{bmatrix}. \tag{2.45}$$

This block matrix system shares some resemblance to the one we derived earlier in this section for the Dirichlet problem- but there is one important difference! Whereas the coefficient matrix A for the Dirichlet problem is symmetric, positive definite, and well-conditioned, the above matrix A is, in fact, singular! This can be readily verified since the sum of the entries in each row of A equals zero and therefore the vector $\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$ ' is a solution of the homogeneous system $AU = 0$. This property of our finite difference model corresponds nicely to the property of the BVP mentioned in Theorem 11.3 that the solution of the Neumann problem (if exists) is unique only up to an additive constant. Indeed, the fact that $\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}'$ is a solution of the homogeneous system $AU = 0$ lets us add a constant vector $c\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$ ' to any solution of the linear system $AU = C$ and still have a solution. What is even more interesting is that the compatibility condition (37) $\int_\Omega f(x,y)dxdy = \int_{\partial\Omega} g(x,y)ds$ for the existence of a solution to the Neumann problem turns out to have the following discrete analogue for the solvability of the discrete system $AU = C$:

$$\frac{1}{2}\sum_{P_k \text{ corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{P_k \text{ interior}} c_k = 0 \tag{2.46}$$

We leave the proofs of the facts that (46) is equivalent to $AU = C$ having a solution and that (46) can be viewed as the discrete analogue of the compatibility condition (37) to Exercises 20, and 21. For now, we proceed to verify the compatibility condition and then solve the system, $AU = C$. Since we are dealing with a singular system our usual methods cannot be applied here.

We begin coding the problem into MATLAB; as usual, we do so in a way that is amenable to the creation of more general codes.

```
>> N=6: M=11: h=0.1;
>> %next create vector for +/âĂŤ N-diaqonals
>> vN=-2*ones(1,N);
>> for i=2:MâĂŤ1
  vN((iâĂŤ1)*N+1:i*N)=âĂŤones(1,N);
end
>> for i=1:length(vN}
  vNbott(i)=vN(length(vN)+1-i):
end
>> A=diag(4*ones(1,N*M))+diag(vN, N)+diag(vNbott,âĂŤN);

%next create vector for sub/super diagonals
>> vl=-ones(1,N-1); v1(1)=âĂŤ2; v=[v1 0];
>> v2=-ones(1,N-1); v2(NâĂŤ1)=-2; vbott=[v2 0];
>> for i=1:M-1
if i<M-l
  v=[v v1 0]; vbott = [vbott v2 0]:
else
  v=[v v1]; vbott=[vb0tt v2];
end
end
>> A=A+diag(v,1)+diag(vbott,-1);
```

We next create the relevant boundary data and inhomogeneity (right-hand side) function:

```
>> xgrid=0:.1:.5; ygrid=0:.1:1;
>> leftdata=zeros(size(ygrid))'; rightdata= (4-8*ygrid)';
>> topdata=-2*ones(size(xgrid)); bottomdata=4*ones(size(xgrid)}:
>> f = inline('2', 'x', 'y');
```

from which we may now construct the needed vector C:

```
>> C=zeros(N*M,1):
>> for j=M:âĂŤ1:1
for i-lzN
  C(i+N*(M-j))=-h^2*f(xgrid(i),ygrid(j)):
  if i == 1
    C(i+N*(M-j))=C(i+N*(MâĂŤj))+2*h*leftdata{j):
  elseif 1 == N
    C(i+N*(M-j))=C(i+N*(M-j})+2*h*rightdata{j);
end
end
end
>> C(1:N)=C(1:N)+2*h*topdata';
>> C(M*N-N+1:M*N)= C(M*N-N+1:M*N)+2*h*bottomdata':
```

We now have constructed the known matrices A and C of the singular linear system $AU = C$. For good measure we check the validity of the discrete compatibility condition (46):

$$\frac{1}{2}\sum_{P_l \text{ corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{P_k \text{ interior}} c_k = 0.$$

We need to distinguish the indices k in these three sets. Using some MATLAB notation, these three index sets may be expressed as follows:
Corners: $k = 1, N, MN-1, MN$,
Edges: $k = 2:(N-1), (N+1):N:(MN-1), (2N):N:((M-1)N)$
Interior nodes: all remaining indices k

The following loop will now evaluate the sum of (46):

```
>> sum=0;
for k=1:length(C)
  if ismember(k,[l N M*NâĂŤN+1 M*N])
    sum=sum+C(k)/2:
  elseif ismember(k, {2:(N-1) (N+1):N:(M*N) (2*N):N:((MâĂŤ1)*N) ...
    (M*N-N+1):(M*NâĂŤl)])
     sum=sum+C(k};
  else
     sum=sum+2*C(k);
  end
end
>>sum
```

$\rightarrow$**`sum = 2.2204e-015`**
Taking into account floating point errors, this sum is zero, so the system will have a solution. To solve it numerically, we cannot use left divide. We use MATLAB's `rref` to put the corresponding augmented matrix $[A|C]$ into reduced row echelon form:[15]

```
>> Aug=[A C]; Augred=rref(Aug);
```

We now check that the row reduced augmented matrix has the expected form:

```
>> max(abs(Augred(66,:)))
```

$\rightarrow$**`ans = 0 (Shows the last row is all zeros.)`**
If the compatibility condition (46) failed, then the last entry would not be zero, but all other entries in the last row would be zero so there would be no solution.

```
>> max(max(abs((Augred(1:65,1:65)-eye(65)))))
```

$\rightarrow$ **`ans`** $=0$ **`(Shows the upper`** 65×65 **`submatrix of Augred is the identity matrix.)`** A simple solution of $AU=C$ is now obtained by setting $U_{66}=0$ which, because of the special form observed above of Augred, simply amounts to taking U to be the last column of Augred:

```
>> U=Augred(:,67);
```

Since we would like to have $(u(0,0)=)U_{61}$ to equal zero and since constants can be added to solutions, the solution to our problem will be contained in the vector:

```
>> U=U-U(61);
```

We may now build an appropriate matrix of the u-values and plot as usual:

```
>> Z=zeros(N,M); Z(:)=U; Z=Z';

for i=1:M
y(i)=ygrid(M+1-i);
end %as usual, we reverse the order of y-grid for plots
to be correct.
>> surf(xgrid,y,Z)
>> hidden off, xlabel('x-axis'), ylabel('yâĂŤaxis')
```

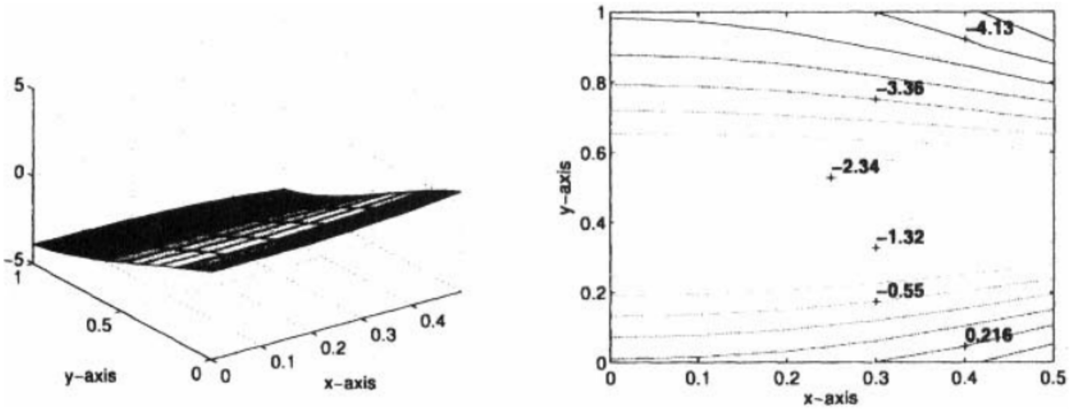


Figure 2.22: (a) (left) Surface plot of the solution to the Neumann problem of Example 11.7. (b) (right) Corresponding plot of isotherms.

The plot is shown in Figure 11.22 along with a corresponding isotherm plot. Note the effect of the various boundary conditions on the heat distribution near the edges. Some comments are in order. Such Neumann problems are not very stable numerically. Indeed, for the linear system to have a solution, an exact condition must hold (the discrete compatibility condition (46)); small roundoff errors can lead to a false conclusion of insolvability. Even worse, the discrete compatiblity condition (46) may not hold even when the integral version holds (37). This would happen, for example, if we changed the PDE in the above example to $\Delta u = 6y^2$ but leave the BCs intact. In this case, the reader could check that although the compatibility condition (37) is still valid, the discrete compatibility condition is no longer valid. See Exercise 22 for

[15]We do not advocate using `rref` to solve singular systems, although we could get more reliable results by working with the Symbolic Toolbox. Apart from this example, we will not be needing to solve any singular linear systems in this book, so we will not delve into a serious discussion of the available numerical methods. Numerical methods for solving singular linear systems are rather sophisticated. The interested reader is encouraged to refer to Chapter 6 of [GoVL-83] or to look at the paper of A. Neumaier [Neu-98] for details.

more details on such pathologies.

Despite the fact that Neumann problems are not very amenable to finite difference schemes (due basically to the exact requirement of the compatibility condition (37)), any other BCs on Poisson's equation (Robin, Dirichlet, or mixed, even with Neumann conditions on some-but not all of the boundary) give rise to a stable problem that can be solved by blending the methods used thus far. We state a relevant theorem and then give an example.

THEOREM 11.6: *(Existence and Uniqueness for the Poisson PDE with Mixed Boundary Conditions)*Suppose that Ω is a smooth bounded planar domain and that in the BVP

$$\begin{cases} (PDE) & \Delta u = f(x,y) \text{ on } \Omega, \quad u = u(x,y) \\ (BC) & p(x,y)\partial u/\partial n + r(x,y)u = g(x,y) \text{ on } \partial\Omega \end{cases}, \tag{2.47}$$

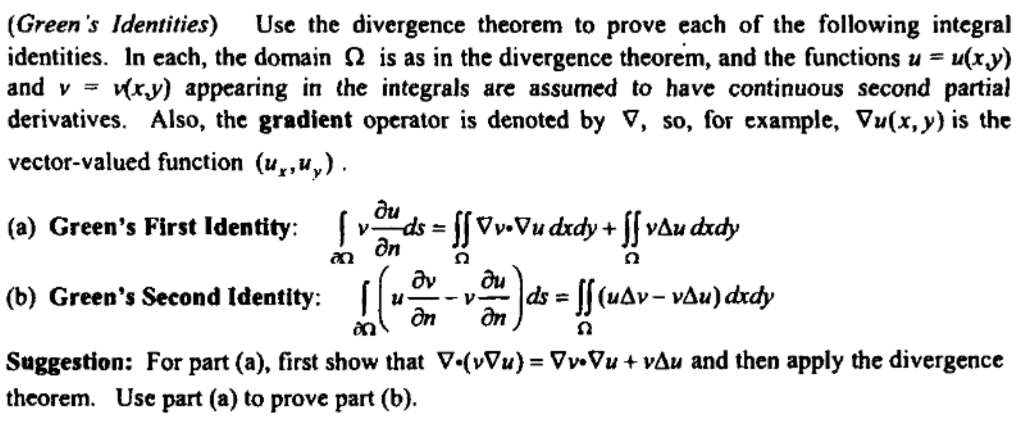
where $f(x,y)$ has piecewise continuous partial derivatives throughout Ω and $g(x,y), p(x,y)$, and $r(x,y)$ are piecewise continuous on $\partial\Omega$, with $p(x,y), r(x,y) \geq 0$, and $p(x,y)+r(x,y) > 0$ throughout $\partial\Omega$. If $r(x,y) > 0$ on some portion of $\partial\Omega$ of positive arclength, then the BVP (47) has a unique solution.

Like Theorem 11.5, this one can be proved using vector calculus (see for example Sections 81 and 82 of classical textbook [BrCh-93]). The last hypothesis appearing in this theorem simply ensures that the BCs are not purely Neumann (without this we we could infer nonuniqueness from Theorem 11.5). Problems with such mixed boundary conditions are usually quite amenable to solution by the finite difference methods of this section. For each boundary node involving Neumann or Robin conditions, we introduce a ghost node, while Dirichlet boundary points ($p = 0, r > 0$) do not require them. The next exercise for the reader requires such a mixing of methods. Even on a simple rectangular domain, for a mixed BVP (having the same type of *BC* on each edge) there are $3^4 = 81$ different types of BC configurations. Thus a general matrix description would not be feasible, but after working through the next exercise for the reader and some of the exercises at the end of this section, the reader should become quite adept at dealing with any sort of BCs.

EXERCISE FOR THE READER 11.8: (a) Numerically solve the following Laplace problem with "mixed type" boundary conditions:

$$\begin{cases} \text{(PDE)} & \Delta u = 0, \quad u = u(x,y) \\ \text{(BC)} & u(x,0) = 0, u_y(x,1) = 20 \qquad 0 \leq x \leq 1, 0 \leq y \leq 1 \\ & u(0,y) = 100, u_x(1,y) = 0 \end{cases}$$

Use a common step size of $h = 0.05$. Obtain a surface graph of the solution along with an isotherm plot and interpret as a steady-state heat distribution. (b) Repeat with $h = 0.02$. Your plots should look like those in Figure 11.23.


(*Green's Identities*) Use the divergence theorem to prove each of the following integral identities. In each, the domain Ω is as in the divergence theorem, and the functions $u = u(x,y)$ and $v = v(x,y)$ appearing in the integrals are assumed to have continuous second partial derivatives. Also, the **gradient** operator is denoted by ∇, so, for example, $\nabla u(x,y)$ is the vector-valued function (u_x, u_y).

(a) Green's First Identity: $\displaystyle\int_{\partial\Omega} v\frac{\partial u}{\partial n}ds = \iint_{\Omega} \nabla v \cdot \nabla u\, dxdy + \iint_{\Omega} v\Delta u\, dxdy$

(b) Green's Second Identity: $\displaystyle\int_{\partial\Omega} \left(u\frac{\partial v}{\partial n} - v\frac{\partial u}{\partial n}\right)ds = \iint_{\Omega} (u\Delta v - v\Delta u)\, dxdy$

Suggestion: For part (a), first show that $\nabla \cdot (v\nabla u) = \nabla v \cdot \nabla u + v\Delta u$ and then apply the divergence theorem. Use part (a) to prove part (b).

Figure 2.23: (a) (left) Mesh plot of the solution of the mixed BVP of Exercise for the Reader 11.8, using a common grid spacing of $h = 0.02$. Note how each of the boundary conditions are well depicted near the edges. (b) (right) Corresponding isotherm contour plot. The plots obtained for $h = 0.05$ appeared quite identical to these.

EXERCISES 2.4.

1. *(Dirichlet Problems for the Laplace Equation)* For each BVP given, do the following: (i) Set up the finite difference method for solving the problem using common x-and y-mesh size $h = 0.1$, and write down the linear system in block matrix notation. (ii) Use MATLAB to solve the resulting linear system, and produce a mesh plot of the solution surface. (iii) Obtain a contour plot of the isotherms. (iv) Repeat parts (i) through (iii) using $h = 0.05$. (v) Repeat parts (i) through (iii) using $h = 0.02$.

(a) $\begin{cases} \text{(PDE) } \Delta u = 0 \text{ on } \Omega = \{0 < x < 2, 0 < y < 1\}, \quad u = u(x,y) \\ \text{(BC)} \quad u(x,0) = 0, u(x,1) = 100, u(0,y) = 0, u(2,y) = 100 \end{cases}$

(b) $\begin{cases} \text{(PDE)} \Delta u = 0 \text{ on } \Omega = \{0 < x,y < 1\}, \quad u = u(x,y) \\ \text{(BC)} u(x,0) = 50x, u(x,1) = -50x, u(0,y) = 0, u(1,y) = y^2 - 101y + 50 \end{cases}$

(c) $\begin{cases} \text{(PDE)} \Delta u = 0 \text{ on } \Omega = \{3 < x < 4, 2 < y < 3\}, u = u(x,y) \\ \text{(BC)} \quad u(x,2) = 0, u(x,3) = 0, u(3,y) = 0, u(4,y) = 100 \end{cases}$

(d) $\begin{cases} \text{(PDE)} \Delta u = 0 \text{ on } \Omega = \{0 < x < 1, 0 < y < 1\}, u = u(x,y) \\ \text{(BC)} u(x,0) = 20\sin(2\pi x), u(x,1) = 50x(1-x), u(0,y) = u(1,y) = 0 \end{cases}$

2. *(Dirichlet Problems for the Poisson Equation)* Go through each of parts (i) through (v) of Exercise | for the following BVPs:

(a) In the BVP of Exercise I(a), change the PDE to $\Delta u = 100\left(1 - (x-1)^2\right)$.

(b) In the BVP of Exercise I(b), change the PDE to $\Delta u = -f(x,y)$, where $f(x,y)$ is the "bump" function of Example 11.7.

(c) In the BVP of Exercise 1(c), change the PDE to $\Delta u = f(x,y)$, where

$$f(x,y) = \begin{cases} 500, & \text{if } x < 3.5 \\ 0, & \text{otherwise} \end{cases}$$

(d) In the BVP of Exercise I(a), change the PDE to $\Delta u = -(2x)^2 - (5y)^2$.

3. *(Mixed Boundary Value Problems for the Laplace Equation)* Go through each of parts (i) through (v) of Exercise 1 by making the following changes in the BVP (a) of Exercise 1:

(a) Replace the corresponding BC with $u_y(x,0) = -20, u_y(x,l) = 40$.

(b) Replace the corresponding BC with $u_y(x,0) = u_y(x,1) = u_x(0,y) = -50$.

(c) Replace the corresponding BC with $u_y(x,0) = u_y(x,1) = u_x(0,y) = 50$.

(d) Replace the corresponding BC with $u_y(x,0) + u(x,0) = u_y(x,1) + u(x,1) = 50$.

4. *(Mixed Boundary Value Problems for the Poisson Equation)* Go through each of parts (i) through (v) of Exercise 1 by making the following changes in the BVP (a) of Exercise 2 :

(a) Replace the corresponding BC with $u_y(x,0) = 40, u_x(0,y) = 40y$.

(b) Replace the corresponding BC with $u_y(x,0) = -40, u_x(0,y) = -40y$.

(c) Replace the corresponding BC with $u_y(x,0) = u_y(x,1) = 10e^x, u_x(0,y) = 0$.

(d) Replace the corresponding BC with $u_y(x,0) = u_y(x,1) = 10e^x, u_x(0,y) + u(0,y) = 0$.

5. *(Mixed Boundary Value Problems for the Poisson Equation)* Go through each of parts (i) through (v) of Exercise 1 by making the following changes in the BVP (c) of Exercise 2 :

(a) Replace the corresponding BC with $u_y(x,3) = u_y(x,2) = u_x(3,y) = 80$.

(b) Replace the corresponding BC with $u_y(x,3) = u_y(x,2) = u_x(3,y) = -80$.

(c) Replace the corresponding BC with $u_x(3,y) + 10u(3,y) = 0, u_x(4,y) + 10u(4,y) = 40$.

(d) Replace the corresponding BC with $10u_x(3,y) + u(3,y) = 0, 10u_x(4,y) + u(4,y) = 40$.

6. (a) Using a finite difference method with grid step sizes $h = k = \pi/5$, solve the following elliptic BVP:

$$\begin{cases} \text{(PDE)} & \Delta u + \left(x^2 + y^2\right)u = 0 \quad u = u(x,y) \quad 0 \le x \le \pi, 0 \le y \le \pi \\ \text{(BC)} & u(x,\pi) = \sin(\pi x) \equiv t(x), u(x,0) = 0 \equiv b(x) \\ & u(\pi,y) = \sin(\pi y) \equiv r(y), u(0,y) = 0 \equiv \ell(y) \end{cases}$$

Plot your numerical solution and then print out the 6×6 matrix whose entries are the absolute values of the differences of the exact solution $\sin(xy)$ with the approximation at each of the grid points. What is the maximum single error occurring at a grid point?

(b) Repeat part (a) using step sizes $h = k = \pi/10$ (the grid for the second question will now be 11×11).

(c) Repeat part (a) once again using step sizes $h = k = \pi/30$.

7. (a) Using a central difference approximation for the first-order partial derivative, use the finite difference method to solve the following elliptic boundary value problem:

$$\begin{cases} \text{(PDE)} & \Delta u - u_x = 2, \quad u = u(x,y) \quad 0 \le x \le 1, 0 \le y \le 1 \\ \text{(BC)} & u(x,1) = 0 \equiv t(x), u(x,0) = 0 \equiv b(x) \\ & u(.5,y) = 0 \equiv r(y), u(0,y) = 0 \equiv \ell(y) \end{cases}$$

with equal grid step sizes $h = k = 0.2$.

(b) Repeat with $h = k = 0.1$ and compare the absolute value of the differences of this solution with that of part (a) on common grid points.

(c) Repeat again with $h = k = 0.05$ and compare the absolute value of the differences of this solution with that of part (b) on common grid points.

NOTE: The next four exercises will take advantage of sparse matrix storage and manipulations in MATLAB; this topic was discussed in Section 7.7.

8. (a) Write an M-file whose syntax and functionality is identical to the `rectanglepoissonsolver` M-file of Exercise for the Reader 11.7 except that internally this new one will create the coefficient matrix as a sparse matrix. Call this new M-file `rectanglepoissonsolversp`.

(b) Test the new program out on the BVP of Example 11.7, and compare performance times with the original one for various step sizes. Allowing a maximum of five minutes on your computer, how many more internal nodes can your new M-file handle compared with the original?

9. *(Dirichet Problems for the Poisson Equation)* For each of the BVPs given in Exercise 2, start off with a common step size $h = 1/4$ and solve it using the finite difference method but by storing the coefficient matrix as a sparse data type. Repeat with $h = 1/8$. Compare the two solutions at common interior grid points. Continue this halving of step sizes and comparing consecutive solutions until the maximum error falls below 1/100 of the maximum observed amplitude of the most recent numerical solution, or the computation takes the computer more than 5 minutes.

10. *(Mixed Boundany Value Problems for the Laplace Equation)* Repeat the instructions of Exercise 9 for each of the BVPs of Exercise 3.

11. *(Mixed Boundary Value Problems for the Poisson Equation)* Repeat the instructions of Exercise 9 for each of the BVPs of Exercise 4 .

12. *(A Block Matrix Finite Difference Method for Unequal Step Sizes)* We apply the finite difference method to the BVP (20): $\begin{cases} \text{(PDE)}\ \Delta u = f(x,y) \text{ on } \Omega, \\ \text{(BC)} \quad u = g(x,y) \text{ on } \partial\Omega \end{cases}$ on a rectangular domain $\Omega = \{(x,y) : 0 < x < a, 0 < y < b\}$; using step size h in the x-direction and k in the y-direction, the scheme is as in (20): $2\left(\frac{h^2}{k^2}+1\right)u_{i,j} - u_{i+1,j} - u_{i-1,j} - \left(\frac{h^2}{k^2}\right)u_{i,j+1} - \left(\frac{h^2}{k^2}\right)u_{i,j-1} = -h^2 q_{i,j}$.

(a) Using the natural ordering of grid points (Figure 11.11), what would the block matrix representation $AU = C$ look like for this scheme?

(b) Adapt the scheme of part (a) to solve the BVP of Exercise 1 (a) using step sizes $h = 0.05$ and $k = 0.025$.

(c) Adapt the scheme of part (a) to solve the BVP of Exercise 1(d) using step sizes $h = 0.02$ and $k = 0.05$.

13. *(A Block Matrix Finite Difference Method for a Mixed BVP with Unequal Step Sizes)* If we apply the finite difference method to the BVP :

$$\begin{cases} \text{(PDE)} & \Delta u = f(x,y), \quad u = u(x,y) \\ \text{(BC)} & u(x,0) = T_b, u_y(x,l) = G_t \quad ,0 \le x \le a, 0 \le y \le b \\ & u(0,y) = T_l, u_x(1,y) = G_r \end{cases}$$

using step size h in the x-direction and k in the y-direction, the scheme is as in (20) :

$$2\left(\frac{h^2}{k^2}+1\right)u_{i,j} - u_{i+1,j} - u_{i-1,j} - \left(\frac{h^2}{k^2}\right)u_{i,j+1} - \left(\frac{h^2}{k^2}\right)u_{i,j-1} = -h^2 q_{i,j}.$$

(a) Using the natural ordering of grid points and using ghost nodes as needed (cf. Figure 11.21), what would the block matrix representation $AU = C$ look like for this scheme?

(b) Adapt the scheme of part (a) to solve the BVP of Exercise for the Reader 11.8 using step sizes $h = 0.05$ and $k = 0.025$.

14. Use Taylor's theorem to prove the forward and backward difference formulas (38) and (39).

NOTE: Many existence and uniqueness theorems for PDE boundary value problems, as well as the theoretical development of the finite element method, depend essentially on some integral identities known collectively as **Green's identfties**. These identities are easily derived from the **divergence theorem** of vector calculus. These theorems are valid in any number of dimensions $(2,3$, or more $)$ but we develop them now in the two-dimensional setting (of this chapter). The next several exercises are thus intended for students who have studied multivariable calculus. Indeed, the divergence theorem is introduced and dealt with extensively in vector calculus courses.

DIVERGENCE THEOREM: If Ω is a bounded domain in the xy-plane that has a smooth boundary $\partial\Omega$, and $\vec{F}(x,y)=(F_1,F_2)$ is any vector-valued function with continuous first partial derivatives on $\Omega\cup\partial\Omega$, then we have $\Omega\cup\partial\Omega$

$$\iint_\Omega \operatorname{div}\vec{F}(x,y)dxdy=\int_\infty \vec{F}(x,y)\bullet n(x,y)ds \tag{2.48}$$

where $\operatorname{div}\vec{F}(x,y)$ denotes the divergence of the vector field $\vec{F}$: $\operatorname{div}\vec{F}(x,y)=\partial F_1/\partial x+\partial F_2/\partial y, n=n(x,y)$ is the outward pointing normal vector at the point (x,y) on the boundary, and ds denotes arclength. The product on the right is the dot product.

15. *(Green's Identities)* Use the divergence theorem to prove each of the following integral identities. In each, the domain Ω is as in the divergence theorem, and the functions $u=u(x,y)$ and $v=v(x,y)$ appearing in the integrals are assumed to have continuous second partial derivatives. Also, the **gradient** operator is denoted by ∇, so, for example, $\nabla u(x,y)$ is the vector-valued function (u_x,u_y).

    (a) **Green's First Identity**: $\iint_{\boldsymbol{\Omega}} v\frac{\partial u}{\partial n}ds=\iint_\Omega \nabla v\cdot\nabla u dxdy+\iint_\Omega v\Delta u dxdy$

    (b) **Green's Second Identity**: $\int_{\partial\Omega}\left(u\frac{\partial v}{\partial n}-v\frac{\partial u}{\partial n}\right)ds=\iint_\Omega(u\Delta v-v\Delta u)dxdy$

    Suggestion: For part (a), first show that $\nabla\cdot(v\nabla u)=\nabla v\nabla u+v\Delta u$ and then apply the divergence theorem. Use part (a) to prove part (b).

16. *(Proof of Uniqueness for the Dirichlet Problem for the Poisson's Equation)* Complete the following outline to prove part of the uniqueness statements of Theorem 11.3: The Dirichlet problem $\Delta u=f(x,y)$ on Ω (smooth) and $u=g(x,y)$ on $\partial\Omega$, can have at most one solution.

    (a) Suppose that we have two solutions u_1 and u_2 of this BVP. Show that $u\equiv u_1-u_2$ is harmonic $(\Delta u=0)$ in Ω and vanishes on $\partial\Omega$.

    (b) Use Green's first identity to get that $\iint_\Omega|\nabla u(x,y)|^2dxdy=0$.

    (c) Show that $|\nabla u(x,y)|^2\equiv 0$ on Ω (see Exercise 14 of Section 10.5).

    (d) Show that u, having vanishing gradient on Ω, must be constant on each component (piece) of Ω. By the zero boundary conditions for u, we must have in fact $u\equiv 0$ and hence $u_1\equiv u_2$ on Ω.

17. *(Proof of Uniqueness for the Neumann Problem for the Poisson's Equation)* Using the outline of the previous exercise as a guide, prove that if u_1 and u_2 both solve the Neumann problem $\Delta u=f(\dot{x},y)$ on Ω (smooth) and $\partial u/\partial n=g(x,y)$ on $\partial\Omega$, then $u_2\equiv u_1+C$ throughout Ω, for some constant C.

18. *(Proof of Uniqueness for the Robin Problem for the Poissons Equation)* Using the outline of the Exercise 16 as a guide, prove that if u_1 and u_2 both solve the Robin problem $\Delta u=f(x,y)$ on Ω (smooth) and $\partial u/\partial n+ru=0$ on $\partial\Omega$, where $r>0$ throughout $\partial\Omega$, then $u_2\equiv u_1$ throughout Ω.

19. *(Dirichlet's Principle)* Consider the Dirichlet problem for the Laplace equation on a smooth domain $\Omega:\Delta u=0$ on Ω and $u=g(x,y)$ on $\partial\Omega$. For any "admissible" function v that has continuous partial derivatives on Ω and satisfies the boundary condition $v=g(x,y)$ on $\partial\Omega$, we define the **energy** of v by:

    $$E(v)=\iint_\Omega|\nabla v(x,y)|^2dxdy.$$

    Dirichlet's Principle states that the solution of the Dirichlet problem has the lowest possible energy (physicists refer to this as the "ground state") among all admissible functions. In other words, if u is the solution of the Dirichlet problem and v is any other admissible function, then $E(v)\geq E(u)$. Follow the outline below to prove Dirichlet's principle:

    (a) Consider the difference function $w=v-u$, which has zero boundary data. Show that we can expand:

    $$E(v)=E(u)+\iint_\Omega\nabla u\cdot\nabla w dxdy+E(w).$$

    (b) Use Green's first identity to show that the middle term (the double integral) in the above expansion is zero. Since each of the three energies is nonnegative, deduce Dirichlet's principle.

20. *(The Discrete Compatibility Condition for the Neumann Problem)*

(a) Consider the smallest possible discretization of a Neumann problem that actually has interior nodes: a 3×3 grid of nodes. The problem then has nine nodes, as shown in Figure 11.24. Show that the discretization $AU = C$ of the Neumann problem (36)

$$\begin{cases} \text{(PDE) } \Delta u = f(x,y) \text{ on } \Omega, u = u(x,y) \\ \text{(BC)} \partial u/\partial n = h(x,y) \text{ on } \partial\Omega \end{cases}$$

has a solution if and only if the condition (46) (specialized to the present setting):

$$\frac{1}{2}c_1 + c_2 + \frac{1}{2}c_3 + c_4 + 2c_5 + c_6 + \frac{1}{2}c_7 + c_8 + \frac{1}{2}c_9 = 0$$

holds.

(b) Generalize your proof in part (a) to show that a general discretization $AU = C$ of the Neumann problem (36) will have a solution if and only if (46)

$$\frac{1}{2}\sum_{P_k \text{corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{P_k \text{ interior}} c_k = 0$$

holds.

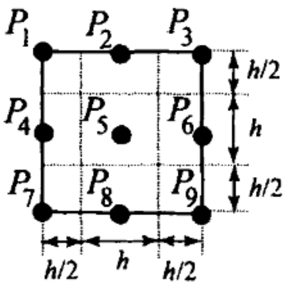


Figure 2.24: A small grid for easy visualization of some properties of finite difference schemes.

Suggestion: For part (a), use (46) to formulate the following sequence of elementary row operations to perform to the last row of the augmented matrix $[A \mid C]$

$$R_9 \to \frac{1}{2}R_9 + R_8 + \frac{1}{2}R_7 + R_6 + 2R_5 + R_4 + \frac{1}{2}R_3 + R_2 + \frac{1}{2}R_1.$$

Verify that this will clear out the last row of A, and leave the expression in (46) in the last entry

21. *(Interpretation of the Discrete Compatibility Condition for the Neumann Problem)*

(a) When specialized to the case of a 3×3 grid on a square as in Figure 11.24, with $a = b = 1$, and $h = 1/2$, interpret the condition (46) (specialized to the present setting) $\frac{1}{2}c_1 + c_2 + \frac{1}{2}c_3 + c_4 + 2c_5 + c_6 + \frac{1}{2}c_7 + c_8 + \frac{1}{2}c_9 = 0$ as a discrete version of the compatibility condition $\int_\Omega f(x,y)dxdy = \int_{\partial\Omega} g(x,y)ds$.

(b) Generalize your proof in part (a) to interpret condition (46) $\frac{1}{2}\sum_{P_k \text{ corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{p_k \text{ interior}} c_k = 0$ as a discrete version of the compatibility condition.

Suggestion: For part (a), write out the vector C as defined by (43), (44), (45):

$$C = \left[2h\left(g^v_{1,3} + g^h_{1,3}\right) - h^2 f_{11}, 2hg^h_{1,3} - h^2 f_{12}, 2h\left(g^v_{2,3} + g^h_{2,3}\right) - h^2 f_{13}, 2hg^v_{2,1} - h^2 f_{21}, \ldots\right]'.$$

Interpret each individual term of (46) as an approximation to a subintegral of one of the integrals in (37); for example, for the first term $\frac{1}{2}c_1$ of (46), we can write:

$$\begin{aligned} \frac{1}{2}c_1 &= h\left(g^v_{1,3} + g^h_{1,3}\right) - \frac{1}{2}h^2 f_{11} \\ &\approx 2\int_{3/4}^{1} g^v(0,y)dy + 2\int_0^{1/4} g^h(x,1)dy - 2\int_0^{1/4}\int_{3/4}^{1} f(x,y)dydx \end{aligned}$$

(simply approximate the functions on the sets of integration by the corresponding constants on the left). Once this is done, it will be apparent that the expression (46) is an approximation to $2\left(\int_{\partial\Omega} g(x,y)ds - \int_\Omega f(x,y)dxdy\right)$ and thus setting this latter expression equal to zero produces (37).

22. In Example 11.8, suppose that the PDE is changed to $\Delta u = 6y^2$ but the boundary conditions are left the same.

    (a) Show that the compatibility condition (37) $\int_\Omega f(x,y)dxdy = \int_{\alpha\Omega} g(x,y)ds$ holds and thus this Neumann problem has a solution.

    (b) Using the same grids as were employed in Example 11.8, show that the discrete compatibility condition (46) $\frac{1}{2}\sum_{P_k \text{ corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{P_i \text{ interior}} c_k = 0$ fails. Thus, for this BVP, although it has a solution, the associated linear system (from the finite difference method) does not have a solution.

    (c) In the language of Exercise 21, explain how these two compatibility conditions are not consistent.

    (d) In the language of Exercise 21, explain why the discrete approximations of (46) correspond exactly to the integrals of (37) for the original BVP of Example 11.8.

    Suggestion: For part (c), the discrete approximations corresponding to the boundary integral $\int_{\partial\Omega} g(x,y)ds$ are exact, but those for the domain integral $\int_\Omega f(x,y)dxdy$ are not.

23. Construct a Neumann problem $\begin{cases} (PDE) & \Delta u = f(x,y) \text{ on } \Omega, u = u(x,y) \\ (BC) & \partial u/\partial n = g(x,y) \text{ on } \partial\Omega \end{cases}$ on a rectangular domain along with a grid of nodes for the finite difference method having the property that the compatibility condition (37) $\int_\Omega f(x,y)dxdy = \int_{\partial\Omega} g(x,y)ds$ fails and thus this Neumann problem does not have a solution, but such that the corresponding discrete compatibility condition (46) $\frac{1}{2}\sum_{P_k\text{corner}} c_k + \sum_{P_k \text{ edge}} c_k + 2\sum_{P_k \text{ interior}} c_k = 0$ does hold. Explain your example in the context of Theorem 11.6 and Exercises 20 and 21 .