3

Green's Function Analysis

This partial differential equation (PDE) application introduces the following mathematical concepts and computational methods:

1. The PDE has an exact solution that can be used to assess the accuracy of the numerical method of lines (MOL) solution.
2. The use of library routines for the finite-difference (FD) approximation of the spatial (boundary-value) derivative is illustrated.
3. The explicit programming of the FD approximations is included for comparison with the use of the library routines.
4. The failure of stagewise differentiation when applied to this PDE problem with a possible explanation for the failure.
5. The analytical solution, which is compared with the numerical solution to assess the accuracy of the latter, is a Green's function.
6. Computation of an invariant for the Green's function to evaluate the accuracy of the numerical solution.
7. The use of the Green's function for the derivation of other analytical solutions.

The PDE is the *one-dimensional (1D) diffusion equation in Cartesian coordinates*

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}$$

or in subscript notation,

$$u_t = Du_{xx} \tag{3.1}$$

D is the *thermal diffusivity*, a positive constant.

The initial condition (IC) is

$$u(x, t = 0) = \delta(x) \tag{3.2}$$

where $\delta(x)$ is the *Dirac delta function* or *unit impulse function*. $\delta(x)$ has the following mathematical properties:

$$\delta(x) = 0, \quad x \neq 0 \tag{3.3a}$$

$$\int_{-\infty}^{\infty} \delta(x)\,dx = 1 \tag{3.3b}$$

$$\int_{-\infty}^{\infty} f(x)\delta(x)\,dx = f(0) \tag{3.3c}$$

which will be discussed subsequently when applied to the numerical solution.

Since Eq. (3.1) is second order in x, it requires two boundary conditions (BCs). For this problem the spatial domain in x is $-\infty \leq x \leq \infty$. But for a computer analysis, we must choose a *finite domain* (because computers work with finite numbers). Thus, we select finite boundary values for x, which are in effect at $x = \pm\infty$; that is, they are *large enough to accurately represent the infinite spatial domain*. This selection of the boundary values of x is based on a knowledge of the PDE solution, or if this is not possible, they are selected by trial and error (these ideas are illustrated by the subsequent analysis).

Additionally, we choose BCs that are *consistent with the IC* (Eq. 3.2). In this way, we ensure a smooth solution for $t > 0$; that is, we do not introduce discontinuities between the IC and the BCs (which could lead to computational problems in addition to violating the smoothness properties of the PDE solution).

In the case of IC (3.2), we can use the *homogeneous (zero) Dirichlet BCs* $u(x = x_l, t) = u(x = x_u, t) = 0$, where x_l and $x_u (x_l < x_u)$ are the finite boundary values of x we have selected so that the solution does not depart from zero at the boundaries (and therefore homogeneous BCs apply). Alternatively, we can use the *homogeneous Neumann BCs* $u_x(x = x_l, t) = u_x(x = x_u, t) = 0$, again, because the slope of the solution at $x = x_l, x_u$ does not depart from zero for the values of t considered. In the subsequent programming, we use the homogeneous Dirichlet BCs.

The analytical solution to Eq. (3.1) with the IC function $u(x, t = 0) = f(x)$ is [1]

$$u(x, t) = \frac{1}{2\sqrt{\pi Dt}} \int_{-\infty}^{\infty} f(x) e^{-(x-\xi)^2/4Dt}\,d\xi \tag{3.4a}$$

For the special case of the IC function of Eq. (3.2), Eq. (3.4a)reduces to

$$u(x, t) = \frac{1}{2\sqrt{\pi Dt}} \int_{-\infty}^{\infty} \delta(x) e^{-(x-\xi)^2/4Dt}\,d\xi = \frac{1}{2\sqrt{\pi Dt}} e^{-x^2/4Dt} \tag{3.4b}$$

Equation (3.4b) follows from the property of the $\delta(x)$ function (Eq. 3.3c). The verification of Eq. (3.4b) as a solution of Eq. (3.1) is given in an appendix at the end of this chapter.

Equation (3.4a) can be written as

$$u(x, t) = \int_{-\infty}^{\infty} f(x) g(x, \xi, t) d\xi \tag{3.4c}$$

where

$$g(x, \xi, t) = \frac{1}{2\sqrt{\pi Dt}} e^{-(x-\xi)^2/4Dt} \tag{3.4d}$$

$g(x, \xi, t)$ in Eq. (3.4d) is the *Green's function* of Eq. (3.1) for the infinite domain $-\infty \leq x \leq \infty$. Equation (3.4c) indicates that the Green's function can be used to derive analytical solutions to the diffusion equation for IC functions $f(x)$ that damp to zero sufficiently fast as $|x| \to \infty$ ([1], p. 95). Also, Eq. (3.4b) indicates that the Green's function can be considered as the *response of the diffusion equation to a unit impulse at* $x = \xi$ (compare Eqs. (3.4b) and (3.4d)).

Equation (3.4a) can be interpreted as the *superposition of a train of unit impulse solutions of Eq. (3.1)* throughout the spatial domain $-\infty \leq x \leq \infty$ (superposition achieved through integration) to produce the solution to Eq. (3.1) for the IC $u(x, t = 0) = f(x)$. Also, the solution of Eq. (3.4b) has the integral property

$$I(t) = \int_{-\infty}^{\infty} u(x, t)dx = \frac{1}{2\sqrt{\pi Dt}} \int_{-\infty}^{\infty} e^{-x^2/4Dt}\, dx = 1 \tag{3.5}$$

Since the integral $I(t)$ is a function of t, it has the counterintuitive property that it is actually a constant. This *invariant* is used subsequently as a check of the numerical solution of Eq. (3.1).

A main program in Matlab for the MOL solution of Eqs. (3.1) and (3.2) is given in Listing 3.1.

```
%
% Clear previous files
  clear all
  clc
%
% Parameters shared with the ODE routine
  global ncall ndss n xl xu x dx
%
% Spatial grid
  n=101;
  xl=-10.0;
  xu= 10.0;
  dx=(xu-xl)/(n-1);
  for i=1:n
    x(i)=xl+dx*(i-1);
  end
  D=1.0;
%
% Initial condition
  n2=(n-1)/2+1;
  for i=1:n
    if(i==n2)u0(i)=25.0*dx;
    else u0(i)=0.0;
    end
  end
%
```

```
% Independent variable for ODE integration
  t0=0.0;
  tf=2.0;
  tout=(t0:0.5:tf);
  nout=5;
  ncall=0;
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  mf=3;
  if(mf==1) % explicit FDs
    [t,u]=ode15s(@pde_1,tout,u0,options); end
  if(mf==2) ndss=4; % ndss = 2, 4, 6, 8 or 10 required
    [t,u]=ode15s(@pde_2,tout,u0,options); end
  if(mf==3) ndss=44; % ndss = 42, 44, 46, 48 or 50 required
    [t,u]=ode15s(@pde_3,tout,u0,options); end
%
% Store analytical solution and the difference between the
% numerical and analytical solutions
  for it=2:nout
  for i=1:n
    u_anal(it,i)=1.0/(2.0*sqrt(D*pi*t(it)))*...
                 exp(-x(i)^2/(4.0*D*t(it)));
    err(it,i)=u(it,i)-u_anal(it,i);
  end
  end
%
% Display selected output
  fprintf('\n mf = %2d   abstol = %8.1e   reltol = %8.1e\n',...
          mf,abstol,reltol);
%
% t = 0 not included
  for it=2:nout
  fprintf('\n   t   x      u(num)     u(anal)         err\n');
  for i=1:n
    fprintf('%6.2f%6.1f%15.6f%15.6f%15.6f\n',...
            t(it),x(i),u(it,i),u_anal(it,i),err(it,i));
  end
%
%   Calculate and display the integral of the solution
    ui=u(it,:);
    uint=simp(xl,xu,n,ui);
    fprintf('\n Integral of u(x,t=%4.2f) = %7.4f\n',t(it),uint);
  end
  fprintf('\n ncall = %4d\n',ncall);
%
```

```
% Plot numerical solution
  figure(1);
  plot(x,u,'-');
  xlabel('x')
  ylabel('u(x,t)')
  title('Green's function; t = 0, 0.5, 1, 1.5, 2; numerical')
  hold on
%
% Plot numerical and analytical solutions
  figure(2);
  plot(x,u(2:nout,:),'o',x,u_anal,'-');
  xlabel('x')
  ylabel('u(x,t)')
  title('Green's function; t = 0.5, 1, 1.5, 2;
         o - numerical; solid - analytical')

% Plot numerical solution in 3D perspective
  if(mf==3)
  figure(3);
  view([220 10]);
  hold on
  h1=surfl(x,t,u);
  shading interp;
  colormap gray;
  hold off
  axis('tight');
  grid on
  zlim([0 0.5])
  ylabel('t, time')
  xlabel('x, distance')
  zlabel('u')
  s1=sprintf('Green's Function - MOL Solution');
  title(s1, 'fontsize', 12);
  rotate3d on;
  end
% print -deps -r300 pde1.eps; print -dps -r300 pde1.ps;
  print -dpng -r300 pde1.png
```

Listing 3.1. Main program `pde_1_main.m`

We can note the following points about this main program:

1. After declaring some parameters *global*, so that they can be shared with other routines called via this main program, a spatial grid is defined over 101 points, extending over the interval $-10 \leq x \leq 10$.

```
%
% Clear previous files
  clear all
  clc
%
% Parameters shared with the ODE routine
  global ncall ndss n xl xu x dx
%
% Spatial grid
  n=101;
  xl=-10.0;
  xu= 10.0;
  dx=(xu-xl)/(n-1);
  for i=1:n
    x(i)=xl+dx*(i-1);
  end
  D=1.0;
```

The computation of a numerical solution of Eqs. (3.1) and (3.2) indicates that the infinite domain $-\infty \leq x \leq \infty$ can be replaced with the finite domain $-10 \leq x \leq 10$. The justification for this is given when the numerical solution of Eqs. (3.1) and (3.2) is discussed subsequently.

2. Equation (3.2) is defined over the 101-point spatial grid. This IC presents a difficulty in the numerical representation of $\delta(x)$. Note from Eq. (3.3a) that this function is zero everywhere along the spatial grid where $x \neq 0$. This is approximated by the `for` loop where `u0(i)=0.0` except at `i=n2=51` corresponding to $x = 0$. At this point, `u0(n2)=25.0*dx`, where `dx` is the grid spacing. The scaling of 25.0 was selected so that the numerical integral of the numerical solution $u(x, t)$ (`= u(it,i)`) equaled 1 according to Eq. (3.5); this scaling also ensured that the peak values of the analytical and numerical solutions at $x = 0$, $t \neq 0$ are equal (as reflected in the numerical output discussed subsequently).

```
%
% Initial condition
  n2=(n-1)/2+1;
  for i=1:n
    if(i==n2)u0(i)=25.0*dx;
    else u0(i)=0.0;
    end
  end
```

In other words, the `for` loop used to define `u0(i)` is an attempt at a numerical approximation of $\delta(x)$ that satisfies the two basic properties of Eqs. (3.3a) and (3.3b). For the integral conditions of Eqs. (3.3b) and (3.5), two cases can be considered:

(a) For $t = 0$, the approximation of the IC can be considered as two adjacent right triangles, each with a height of `25.0*dx` and a base of `dx`. Thus the total area of the two triangles is (with $dx = (10 - (-10))/(101 - 1) = 0.2$)

$$2[(\text{height})(\text{base})/2] = 2(25\,dx)dx/2 = 2(25)(0.2)(0.2)/2 = 1$$

in agreement with Eq. (3.3b). In other words, the IC is a triangular pulse approximation of $\delta(x)$ spanning the three points $x = -dx, 0, dx$ with the correct "strength" of one, and this is achieved with the scale factor of 25. If the grid spacing is changed through a change of the number of grid points (other than 101), perhaps to achieve better spatial resolution of the numerical solution $u(x, t)$ (by increasing n), the scale factor can be changed accordingly by application of the preceding analysis to maintain the integral property of Eq. (3.3b). In the present case, the agreement between the numerical and analytical solutions is quite acceptable and therefore an increase in the number of grid points would not produce a significantly better numerical solution. This choice of $n = 101$ therefore meets the usual goal of using a grid that produces acceptable accuracy without excessive computation.

(b) For $t > 0$, the numerical integral of the numerical solution equals one (unity) to five figures, in accordance with Eq. (3.5), as demonstrated in the numerical output to follow.

3. t is then defined over the interval $0 \le t \le 2$ with the interval for displaying the numerical solution set to 0.5 so that the solution is displayed five times. The counter for the number of calls to the ordinary differential equation (ODE) routine is also initialized.

```
%
% Independent variable for ODE integration
  t0=0.0;
  tf=2.0;
  tout=(t0:0.5:tf);
  nout=5;
  ncall=0;
```

4. The 101 ODEs are then integrated by a call to the Matlab integrator `ode15s`.

```
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  mf=3;
```

```
if(mf==1) % explicit FDs
  [t,u]=ode15s(@pde_1,tout,u0,options); end
if(mf==2) ndss=4; % ndss = 2, 4, 6, 8 or 10 required
  [t,u]=ode15s(@pde_2,tout,u0,options); end
if(mf==3) ndss=44; % ndss = 42, 44, 46, 48 or 50 required
  [t,u]=ode15s(@pde_3,tout,u0,options); end
```

Three cases are programmed corresponding to `mf=1,2,3`, for which three different ODE routines, `pde_1`, `pde_2`, and `pde_3`, are called (these routines are discussed subsequently). The variable `ndss` refers to a library of differentiation routines for use in the MOL solution of PDEs; the use of `ndss` is illustrated in the subsequent discussion. Note that a stiff integrator, `ode15s`, was selected because the 101 ODEs are sufficiently stiff that a nonstiff integrator results in a large number of calls to the ODE routine.

5. The analytical solution, Eq. (3.4b), is computed over the spatial grid, and the difference between the numerical and analytical solution is also computed.

```
%
% Store analytical solution and the difference between the
% numerical and analytical solutions
  for it=2:nout
  fprintf('\n   t   x      u(num)     u(anal)         err\n');
  for i=1:n
    u_anal(it,i)=1.0/(2.0*sqrt(D*pi*t(it)))*...
                 exp(-x(i)^2/(4.0*D*t(it)));
    err(it,i)=u(it,i)-u_anal(it,i);
  end
  end
```

Note that the analytical solution is not stored at $t = 0$ (corresponding to `it=1`) because of the two terms with a division by t in Eq. (3.4b), that is,

$$\frac{1}{2\sqrt{\pi Dt}} \tag{3.6a}$$

$$e^{-x^2/4Dt} \tag{3.6b}$$

These two terms have some interesting properties:

(a) For $t \to 0$, term (3.6a) becomes arbitrarily large.
(b) For $t \to 0$, term (3.6b) becomes arbitrarily small (for $x \neq 0$).
(c) Since terms (3.6a) and (3.6b) multiply in Eq. (3.4b), the product approaches $\infty \bullet 0$ at $t \to 0$, but the exponential of term (3.6b) dominates, so the product approaches zero (as expected from property (3.3a)).
(d) For $x = 0$, term (3.6b) is unity, so the product of terms (3.6a) and (3.6b) becomes arbitrarily large for $t \to 0$. This arbitrarily large value

demonstrates the difficulty of representing $\delta(x)$ numerically on the spatial grid, and is approximated as `if(i==n2)u0(i)=25.0*dx;` in the programming of IC (3.2). But Eqs. (3.3a) and (3.3b) are the two essential requirements for approximating δx numerically, and this has been done through the programming of Eq. (3.2) described earlier.

6. Selected tabular numerical output is displayed.

```
%
% Display selected output
  fprintf('\n mf = %2d   abstol = %8.1e
           reltol = %8.1e\n', ... mf,abstol,reltol);
%
% t = 0 not included
  for it=2:nout
  fprintf('\n    t    x      u(num)     u(anal)       err\n');
  for i=1:n
    fprintf('%6.2f%6.1f%15.6f%15.6f%15.6f\n',...
            t(it),x(i),u(it,i),u_anal(it,i),err(it,i));
  end
```

Again, the numerical and analytical solutions at $t = 0$ are not displayed.

7. The invariant of Eq. (3.5) is computed by a call to `simp` that implements *Simpson's rule for numerical quadrature (integration)*; `simp` is discussed in an appendix to this chapter.

```
%
%   Calculate and display the integral of the solution
    ui=u(it,:);
    uint=simp(xl,xu,n,ui);
    fprintf('\n Integral of u(x,t=%4.2f) = %7.4f\n',...
            t(it),uint);
  end
  fprintf('\n ncall = %4d\n',ncall);
```

The counter for the calls to the ODE routines is displayed at the end of the numerical solution.

8. `pde_1_main` concludes with plotting (1) the numerical solution including $t = 0$ since it is finite at $x = 0$ (i.e., the coding is `if(i==n2)u0(i)=25.0*dx`, as discussed previously) and (2) the numerical and analytical solutions for $t \neq 0$.

```
%
% Plot numerical solution
  figure(1);
  plot(x,u,'-');
  xlabel('x')
  ylabel('u(x,t)')
  title('Green's function; t = 0, 0.5, 1, 1.5, 2; numerical')
  print -deps -r300 pde.eps; print -dps -r300 pde.ps;...
  print -dpng -r300 pde.png
  hold on
%
% Plot numerical and analytical solutions
  figure(2);
  plot(x,u(2:nout,:),'o',x,u_anal,'-');
  xlabel('x')
  ylabel('u(x,t)')
  title('Green's function; t = 0.5, 1, 1.5, 2;...
        o - numerical; solid - analytical')
```

9. For `mf=3`, a 3D plot of the solution is also produced.

```
% Plot numerical solution in 3D perspective
  if(mf==3)
  figure(3);
  view([220 10]);
  hold on
  h1=surfl(x,t,u);
  shading interp;
  colormap gray;
  hold off
  axis('tight');
  grid on
  zlim([0 0.5])
  ylabel('t, time')
  xlabel('x, distance')
  zlabel('u')
  s1=sprintf('Green's Function - MOL Solution');
  title(s1, 'fontsize', 12);
  rotate3d on;
  end
% print -deps -r300 pde1.eps; print -dps -r300 pde1.ps;
% print -dpng -r300 pde1.png
```

The output from `pde_1_main` is given in Table 3.1.

Table 3.1. Output for `mf=1` from `pde_1_main` and `pde_1`

```
mf =  1   abstol = 1.0e-004   reltol = 1.0e-004

   t      x          u(num)         u(anal)            err
0.50 -10.0        0.000000        0.000000       -0.000000
0.50  -9.8        0.000000        0.000000        0.000000
0.50  -9.6        0.000000        0.000000        0.000000
0.50  -9.4        0.000000        0.000000        0.000000
0.50  -9.2        0.000000        0.000000        0.000000
0.50  -9.0        0.000000        0.000000        0.000000
        .                                          .
        .                                          .
        .                                          .
0.50  -6.0        0.000000        0.000000        0.000000
0.50  -5.8        0.000000        0.000000        0.000000
0.50  -5.6        0.000000        0.000000        0.000000
0.50  -5.4        0.000000        0.000000        0.000000
0.50  -5.2        0.000001        0.000001        0.000001
0.50  -5.0        0.000003        0.000001        0.000001
0.50  -4.8        0.000007        0.000004        0.000003
0.50  -4.6        0.000016        0.000010        0.000006
0.50  -4.4        0.000036        0.000025        0.000011
0.50  -4.2        0.000080        0.000059        0.000021
0.50  -4.0        0.000170        0.000134        0.000036
0.50  -3.8        0.000352        0.000292        0.000060
0.50  -3.6        0.000705        0.000612        0.000093
0.50  -3.4        0.001367        0.001232        0.000135
0.50  -3.2        0.002565        0.002384        0.000181
0.50  -3.0        0.004650        0.004432        0.000219
0.50  -2.8        0.008145        0.007915        0.000230
0.50  -2.6        0.013771        0.013583        0.000188
0.50  -2.4        0.022464        0.022395        0.000070
0.50  -2.2        0.035333        0.035475       -0.000142
0.50  -2.0        0.053554        0.053991       -0.000437
0.50  -1.8        0.078176        0.078950       -0.000774
0.50  -1.6        0.109845        0.110921       -0.001075
0.50  -1.4        0.148486        0.149727       -0.001242
0.50  -1.2        0.193008        0.194186       -0.001178
0.50  -1.0        0.241138        0.241971       -0.000833
0.50  -0.8        0.289465        0.289692       -0.000226
0.50  -0.6        0.333761        0.333225        0.000536
0.50  -0.4        0.369552        0.368270        0.001282
0.50  -0.2        0.392867        0.391043        0.001825
0.50   0.0        0.400966        0.398942        0.002023
0.50   0.2        0.392867        0.391043        0.001825
0.50   0.4        0.369552        0.368270        0.001282
0.50   0.6        0.333761        0.333225        0.000536
```

```
0.50   0.8       0.289465       0.289692      -0.000226
0.50   1.0       0.241138       0.241971      -0.000833
0.50   1.2       0.193008       0.194186      -0.001178
0.50   1.4       0.148486       0.149727      -0.001242
0.50   1.6       0.109845       0.110921      -0.001075
0.50   1.8       0.078176       0.078950      -0.000774
0.50   2.0       0.053554       0.053991      -0.000437
0.50   2.2       0.035333       0.035475      -0.000142
0.50   2.4       0.022464       0.022395       0.000070
0.50   2.6       0.013771       0.013583       0.000188
0.50   2.8       0.008145       0.007915       0.000230
0.50   3.0       0.004650       0.004432       0.000219
0.50   3.2       0.002565       0.002384       0.000181
0.50   3.4       0.001367       0.001232       0.000135
0.50   3.6       0.000705       0.000612       0.000093
0.50   3.8       0.000352       0.000292       0.000060
0.50   4.0       0.000170       0.000134       0.000036
0.50   4.2       0.000080       0.000059       0.000021
0.50   4.4       0.000036       0.000025       0.000011
0.50   4.6       0.000016       0.000010       0.000006
0.50   4.8       0.000007       0.000004       0.000003
0.50   5.0       0.000003       0.000001       0.000001
0.50   5.2       0.000001       0.000001       0.000001
0.50   5.4       0.000000       0.000000       0.000000
0.50   5.6       0.000000       0.000000       0.000000
0.50   5.8       0.000000       0.000000       0.000000
0.50   6.0       0.000000       0.000000       0.000000
        .                                       .
        .                                       .
        .                                       .
0.50   9.0       0.000000       0.000000       0.000000
0.50   9.2       0.000000       0.000000       0.000000
0.50   9.4       0.000000       0.000000       0.000000
0.50   9.6       0.000000       0.000000       0.000000
0.50   9.8       0.000000       0.000000       0.000000
0.50  10.0       0.000000       0.000000      -0.000000

Integral of u(x,t=0.50) =  1.0000

        .                                       .
        .                                       .
        .                                       .
             Output for t = 1, 1.5 removed

        .                                       .
        .                                       .
        .                                       .
```

(*continued*)

Table 3.1 (*continued*)

t	x	u(num)	u(anal)	err
2.00	-10.0	-0.000000	0.000001	-0.000001
2.00	-9.8	0.000001	0.000001	-0.000000
2.00	-9.6	0.000002	0.000002	0.000000
2.00	-9.4	0.000003	0.000003	0.000000
2.00	-9.2	0.000006	0.000005	0.000001
2.00	-9.0	0.000009	0.000008	0.000001
2.00	-8.8	0.000014	0.000012	0.000001
2.00	-8.6	0.000021	0.000019	0.000002
2.00	-8.4	0.000032	0.000029	0.000002
2.00	-8.2	0.000048	0.000045	0.000003
2.00	-8.0	0.000071	0.000067	0.000004
2.00	-7.8	0.000105	0.000099	0.000006
2.00	-7.6	0.000153	0.000146	0.000007
2.00	-7.4	0.000222	0.000212	0.000009
2.00	-7.2	0.000318	0.000306	0.000012
2.00	-7.0	0.000451	0.000436	0.000014
2.00	-6.8	0.000633	0.000616	0.000017
2.00	-6.6	0.000882	0.000861	0.000020
2.00	-6.4	0.001215	0.001192	0.000023
2.00	-6.2	0.001660	0.001633	0.000026
2.00	-6.0	0.002245	0.002216	0.000029
2.00	-5.8	0.003007	0.002976	0.000030
2.00	-5.6	0.003988	0.003958	0.000031
2.00	-5.4	0.005239	0.005210	0.000029
2.00	-5.2	0.006816	0.006791	0.000025
2.00	-5.0	0.008782	0.008764	0.000018
2.00	-4.8	0.011206	0.011197	0.000008
2.00	-4.6	0.014159	0.014164	-0.000005
2.00	-4.4	0.017716	0.017737	-0.000021
2.00	-4.2	0.021952	0.021992	-0.000040
2.00	-4.0	0.026935	0.026995	-0.000061
2.00	-3.8	0.032725	0.032808	-0.000083
2.00	-3.6	0.039371	0.039475	-0.000104
2.00	-3.4	0.046901	0.047025	-0.000124
2.00	-3.2	0.055321	0.055460	-0.000139
2.00	-3.0	0.064609	0.064759	-0.000150
2.00	-2.8	0.074710	0.074864	-0.000154
2.00	-2.6	0.085535	0.085684	-0.000150
2.00	-2.4	0.096955	0.097093	-0.000138
2.00	-2.2	0.108809	0.108926	-0.000117
2.00	-2.0	0.120896	0.120985	-0.000090
2.00	-1.8	0.132987	0.133043	-0.000055
2.00	-1.6	0.144830	0.144846	-0.000016
2.00	-1.4	0.156154	0.156127	0.000027

2.00	-1.2	0.166684	0.166612	0.000071
2.00	-1.0	0.176147	0.176033	0.000115
2.00	-0.8	0.184290	0.184135	0.000155
2.00	-0.6	0.190883	0.190694	0.000189
2.00	-0.4	0.195736	0.195521	0.000215
2.00	-0.2	0.198708	0.198476	0.000231
2.00	0.0	0.199708	0.199471	0.000237
2.00	0.2	0.198708	0.198476	0.000231
2.00	0.4	0.195736	0.195521	0.000215
2.00	0.6	0.190883	0.190694	0.000189
2.00	0.8	0.184290	0.184135	0.000155
2.00	1.0	0.176147	0.176033	0.000115
2.00	1.2	0.166684	0.166612	0.000071
2.00	1.4	0.156154	0.156127	0.000027
2.00	1.6	0.144830	0.144846	-0.000016
2.00	1.8	0.132987	0.133043	-0.000055
2.00	2.0	0.120896	0.120985	-0.000090
2.00	2.2	0.108809	0.108926	-0.000117
2.00	2.4	0.096955	0.097093	-0.000138
2.00	2.6	0.085535	0.085684	-0.000150
2.00	2.8	0.074710	0.074864	-0.000154
2.00	3.0	0.064609	0.064759	-0.000150
2.00	3.2	0.055321	0.055460	-0.000139
2.00	3.4	0.046901	0.047025	-0.000124
2.00	3.6	0.039371	0.039475	-0.000104
2.00	3.8	0.032725	0.032808	-0.000083
2.00	4.0	0.026935	0.026995	-0.000061
2.00	4.2	0.021952	0.021992	-0.000040
2.00	4.4	0.017716	0.017737	-0.000021
2.00	4.6	0.014159	0.014164	-0.000005
2.00	4.8	0.011206	0.011197	0.000008
2.00	5.0	0.008782	0.008764	0.000018
2.00	5.2	0.006816	0.006791	0.000025
2.00	5.4	0.005239	0.005210	0.000029
2.00	5.6	0.003988	0.003958	0.000031
2.00	5.8	0.003007	0.002976	0.000030
2.00	6.0	0.002245	0.002216	0.000029
2.00	6.2	0.001660	0.001633	0.000026
2.00	6.4	0.001215	0.001192	0.000023
2.00	6.6	0.000882	0.000861	0.000020
2.00	6.8	0.000633	0.000616	0.000017
2.00	7.0	0.000451	0.000436	0.000014
2.00	7.2	0.000318	0.000306	0.000012
2.00	7.4	0.000222	0.000212	0.000009
2.00	7.6	0.000153	0.000146	0.000007
2.00	7.8	0.000105	0.000099	0.000006

(*continued*)

Table 3.1 (*continued*)

2.00	8.0	0.000071	0.000067	0.000004
2.00	8.2	0.000048	0.000045	0.000003
2.00	8.4	0.000032	0.000029	0.000002
2.00	8.6	0.000021	0.000019	0.000002
2.00	8.8	0.000014	0.000012	0.000001
2.00	9.0	0.000009	0.000008	0.000001
2.00	9.2	0.000006	0.000005	0.000001
2.00	9.4	0.000003	0.000003	0.000000
2.00	9.6	0.000002	0.000002	0.000000
2.00	9.8	0.000001	0.000001	-0.000000
2.00	10.0	0.000000	0.000001	-0.000001

```
Integral of u(x,t=2.00) =  1.0000

ncall =  195
```

The plotted output from `pde_1_main` is shown in Figures 3.1 and 3.2. We can note the following details of this output:

1. The agreement between the analytical and numerical solutions is quite acceptable (from Table 3.1 and Figure 3.2); note, in particular, this agreement

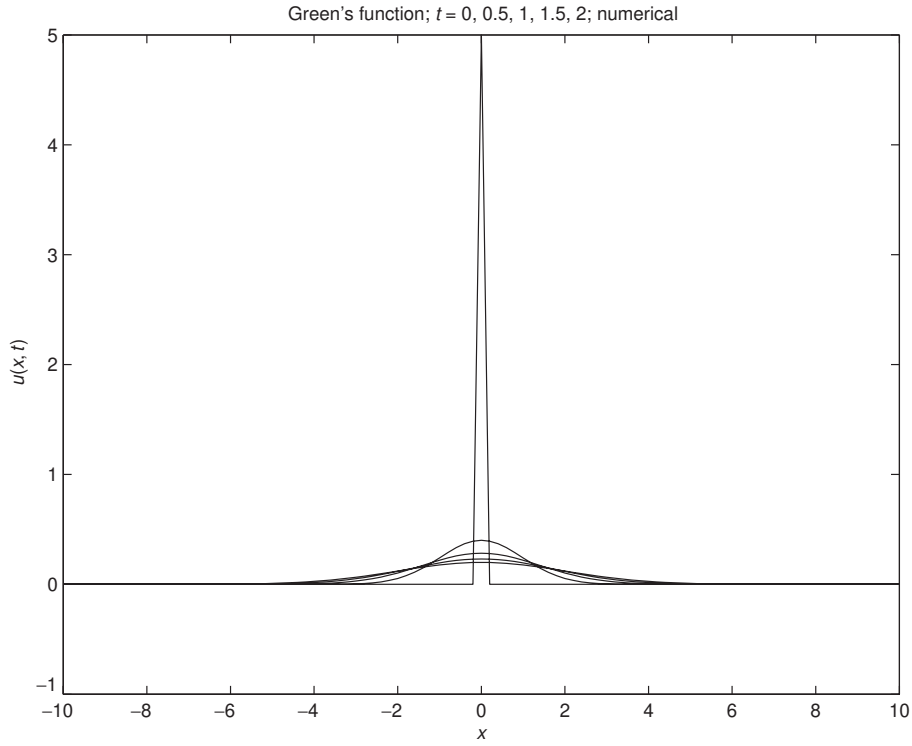

Figure 3.1. Numerical solution from `pde_1_main` including $t = 0$ (`mf=1`)

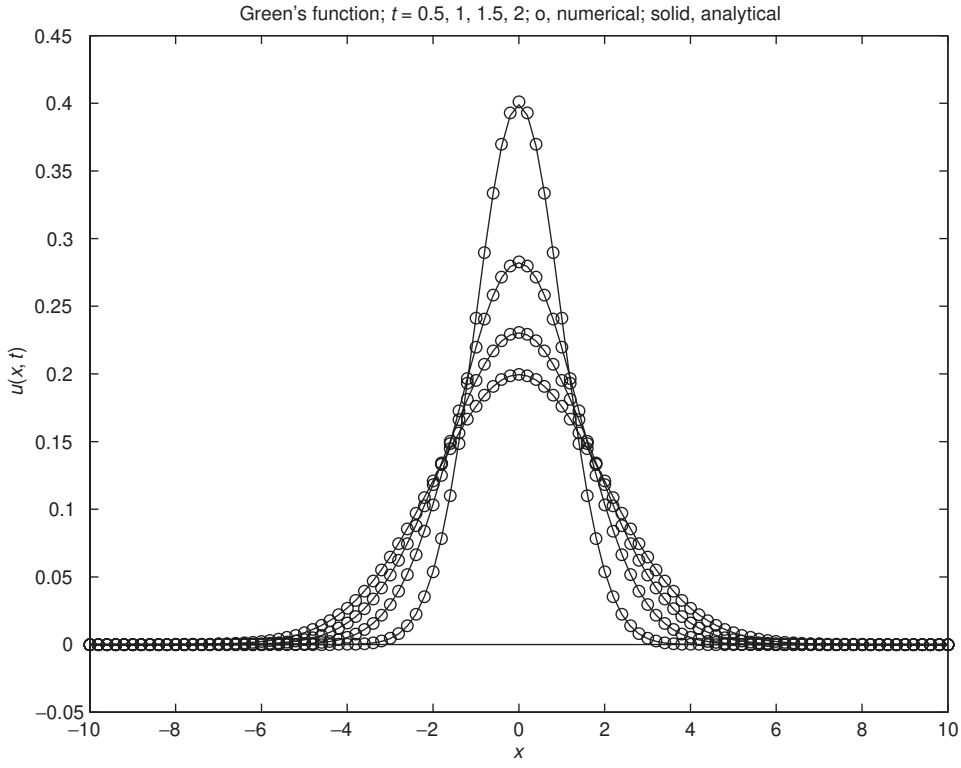

Figure 3.2. Analytical and numerical solutions from `pde_1_main` for $t \neq 0$; (`mf=1`)

in Table 3.1 at $x = 0$ for $t = 0.5$ and $t = 2$ (similar agreement occurred at $t = 1, 1.5$).

2. The integral constraint of Eq. (3.5) is closely approximated (e.g., `Integral of u(x,t=2.00) = 1.0000`), which is perhaps better than expected considering the approximation of $\delta(x)$ of Eq. (3.2) on the spatial grid of 101 points.
3. The height of the numerical approximation of $\delta(x)$ in Figure 3.1, $u(x = 0, t = 0) = 5$, is as expected from the statement `if(i==n2)u0(i)=25.0*dx;` with `dx = 0.2`.
4. The computational effort is quite modest with `ncall = 195`.

The programming of the approximating MOL/ODEs is in one of the three routines called by `ode15s`. We now consider each of these routines. For `mf=1`, `ode15s` calls function `pde_1` (see Listing 3.2).

```
  function ut=pde_1(t,u)
%
% Problem parameters
  global ncall n dx
%
% PDE
  dx2=dx^2;
```

```
  for i=1:n
    if(i==1)     ut(i)=0.0;
    elseif(i==n) ut(i)=0.0;
    else         ut(i)=(u(i+1)-2.0*u(i)+u(i-1))/dx2;
    end
  end
  ut=ut';
%
% Increment calls to pde_1
  ncall=ncall+1;
```

Listing 3.2. Routine `pde_1`

We can note the following points about `pde_1`:

1. After the function call is defined, some problem parameters are declared as *global*.

```
  function ut=pde_1(t,u)
%
% Problem parameters
  global ncall n dx
```

2. The finite-difference (FD) approximation of Eq. (3.1) is then programmed.

```
%
% PDE
  dx2=dx^2;
  for i=1:n
    if(i==1)     ut(i)=0.0;
    elseif(i==n) ut(i)=0.0;
    else         ut(i)=(u(i+1)-2.0*u(i)+u(i-1))/dx2;
    end
  end
  ut=ut';
%
% Increment calls to pde_1
  ncall=ncall+1;
```

The MOL programming of the 101 ODEs is done in the `for` loop:

(a) For the homogeneous Dirichlet BCs $u(x = xl, t) = u(x = xu, t) = 0$, where $xl = -10$ and $xu = 10$, the coding is

```
if(i==1)     ut(i)=0.0;
elseif(i==n) ut(i)=0.0;
```

since the boundary values do not change after being set as part of the IC (Eq. (3.2)) in the main program (and therefore their time derivatives are zero).

(b) The FD approximation of Eq. (3.1) at the interior points is programmed as

```
else ut(i)=(u(i+1)-2.0*u(i)+u(i-1))/dx2;
```

which follows directly from the FD approximation for u_{xx} in Eq. (3.1):

$$u_{xx} \approx \frac{(u(i+1) - 2u(i) + u(i-1))}{\Delta x^2} \tag{3.7}$$

3. Since `ode15s` requires a column vector of derivatives, a final transpose of `ut` is made.

```
  ut=ut';
%
% Increment calls to pde_1
  ncall=ncall+1;
```

Also, the number of calls to `pde_1` is incremented so that at the end of the solution, the value of `ncall` displayed by the main program gives an indication of the computational effort required to produce the entire solution (i.e., `ncall = 195`).

For `mf=2`, function `pde_2` is called by `ode15s`, as given in Listing 3.3.

```
  function ut=pde_2(t,u)
%
% Problem parameters
  global ncall ndss n xl xu
%
% BCs
  u(1)=0.0;
  u(n)=0.0;
%
```

```
% Calculate ux
  if    (ndss== 2) ux=dss002(xl,xu,n,u); % second order
  elseif(ndss== 4) ux=dss004(xl,xu,n,u); % fourth order
  elseif(ndss== 6) ux=dss006(xl,xu,n,u); % sixth order
  elseif(ndss== 8) ux=dss008(xl,xu,n,u); % eighth order
  elseif(ndss==10) ux=dss010(xl,xu,n,u); % tenth order
  end
%
% Calculate uxx
  if    (ndss== 2) uxx=dss002(xl,xu,n,ux); % second order
  elseif(ndss== 4) uxx=dss004(xl,xu,n,ux); % fourth order
  elseif(ndss== 6) uxx=dss006(xl,xu,n,ux); % sixth order
  elseif(ndss== 8) uxx=dss008(xl,xu,n,ux); % eighth order
  elseif(ndss==10) uxx=dss010(xl,xu,n,ux); % tenth order
  end
%
% PDE
  ut=uxx';
  ut(1)=0.0;
  ut(n)=0.0;
%
% Increment calls to pde_2
  ncall=ncall+1;
```

Listing 3.3. Routine `pde_2.m`

We can note the following points about `pde_2`:

1. The initial statements are the same as in `pde_1`. Then the Dirichlet BCs at $x = -10, 10$ are programmed.

```
  function ut=pde_2(t,u)
%
% Problem parameters
  global ncall ndss n xl xu
%
% BCs
  u(1)=0.0;
  u(n)=0.0;
```

Actually, the statements `u(1)=0.0;`, `u(n)=0.0;` have no effect since the dependent variables can only be changed through their derivatives (i.e., `ut(1)`, `ut(n)`) in the ODE derivative routine (a requirement of the ODE integrator `ode15s`). This code was included just to serve as a reminder of the BCs at $x = -10, x = 10$ that are programmed subsequently.

2. The first-order spatial derivative $\partial u/\partial x = u_x$ is then computed.

```
%
% Calculate ux
  if    (ndss== 2) ux=dss002(xl,xu,n,u); % second order
  elseif(ndss== 4) ux=dss004(xl,xu,n,u); % fourth order
  elseif(ndss== 6) ux=dss006(xl,xu,n,u); % sixth order
  elseif(ndss== 8) ux=dss008(xl,xu,n,u); % eighth order
  elseif(ndss==10) ux=dss010(xl,xu,n,u); % tenth order
  end
```

   Five library routines, `dss002` to `dss010`, are programmed that use second-order to tenth-order FD approximations, respectively. Since `ndss=4` is specified in the main program, `dss004` is used in the calculation of `ux`.
3. The second-order spatial derivative is computed from the first-order spatial derivative.

```
%
% Calculate uxx
  if    (ndss== 2) uxx=dss002(xl,xu,n,ux); % second order
  elseif(ndss== 4) uxx=dss004(xl,xu,n,ux); % fourth order
  elseif(ndss== 6) uxx=dss006(xl,xu,n,ux); % sixth order
  elseif(ndss== 8) uxx=dss008(xl,xu,n,ux); % eighth order
  elseif(ndss==10) uxx=dss010(xl,xu,n,ux); % tenth order
  end
```

   Again, `dss004` is called, which is the usual procedure (the order of the FD approximation is generally not changed in computing higher-order derivatives from lower-order derivatives, a process termed *stagewise differentiation*).
4. Finally, Eq. (3.1) is programmed and the Dirichlet BCs at $x = -10, x = 10$ are applied.

```
%
% PDE
  ut=uxx';
  ut(1)=0.0;
  ut(n)=0.0;
%
% Increment calls to pde_2
  ncall=ncall+1;
```

   Note the similarity of the code to the PDE (Eq. (3.1)), and also the transpose required by `ode15s`.

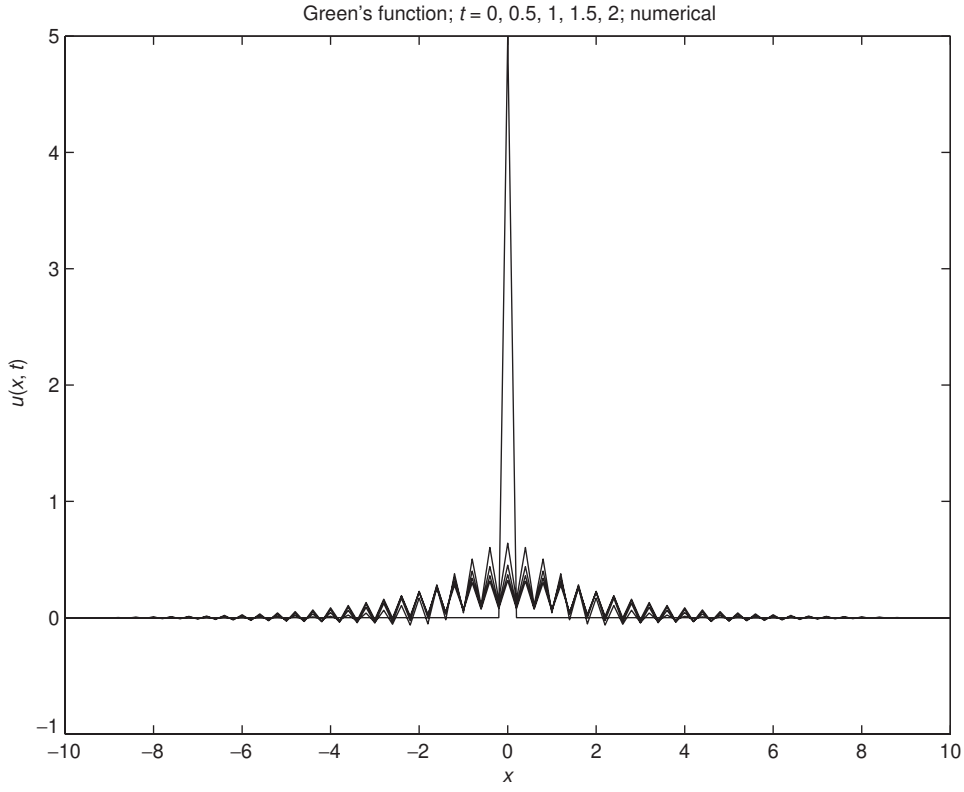

Figure 3.3. Numerical solution from `pde_1_main` including $t = 0$ (`mf=2`)

The numerical output for this case (`mf=2`) would logically be presented and discussed at this point. However, the agreement between the analytical and numerical solutions is not good and is actually difficult to visualize from the numerical output. Therefore, we go directly to the plotted output shown in Figures 3.3 and 3.4 that makes clearer the discrepancies between the analytical and numerical solutions.

We can note the following details of this output:

1. The agreement between the analytical and numerical solutions is unacceptable.
2. The integral constraint of Eq. (3.5) is also substantially in error (from the output of `pde_1_main`).

```
Integral of u(x,t=0.50) =  0.6667

Integral of u(x,t=1.00) =  0.6666

Integral of u(x,t=1.50) =  0.6659

Integral of u(x,t=2.00) =  0.6644
```

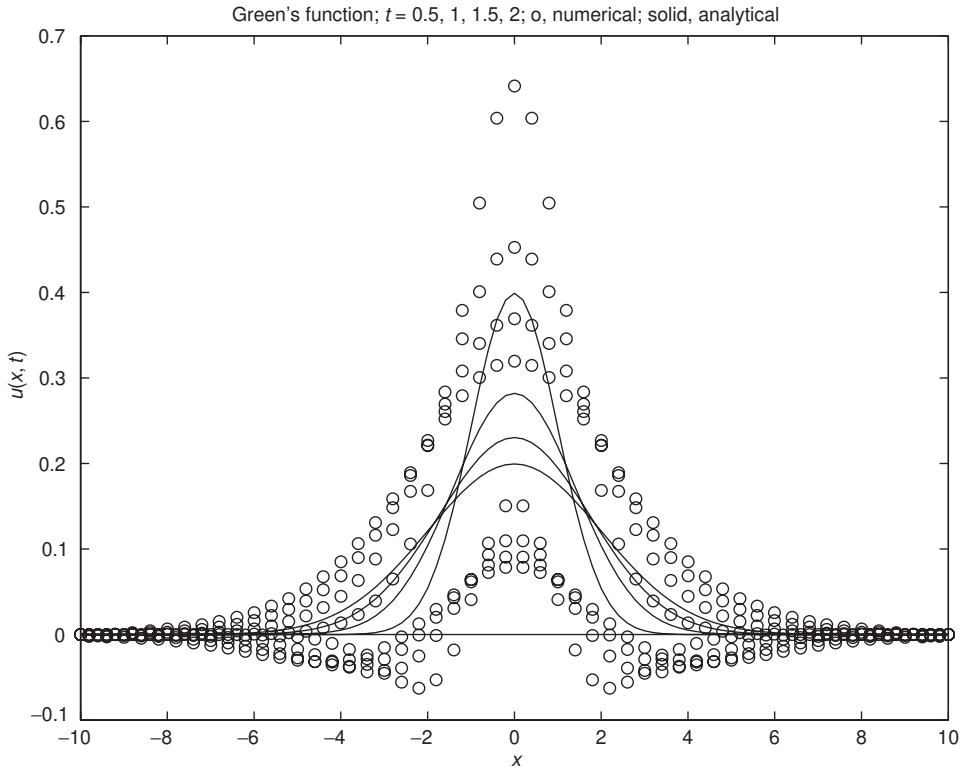

Figure 3.4. Analytical and numerical solutions from `pde_1_main` for $t \neq 0$; (`mf=2`)

Thus, we conclude that the stagewise differentiation of `pde_2` does not work correctly. One possible explanation is that the calculation of the first derivative u_x (e.g., using `dss004`) when applied to the approximation of IC (3.2), as plotted in Figure 3.1, is inaccurate because of the rapid changes in u_x at $x = 0$ for $t \approx 0$. With inaccurate values of u_x, the accurate calculation of u_{xx} (through the stagewise differentiation in `pde_2`) is not possible, and therefore the accurate MOL solution of Eq. (3.1) is not possible.

This explanation is not completely satisfactory since the direct calculation of u_{xx} through Eq. (3.7) produced good agreement between the analytical and numerical solutions (for `mf=1`). With the rapid changes in the solution at $x = 0$ displayed in Figure 3.1, we might also expect that the direct calculation of u_{xx} via Eq. (3.7) would not work well either (in fact, we might expect that u_{xx} calculated numerically would be even more prone to error than u_x). This does not, however, appear to be the case, and thus about all we can conclude from these results is that for this problem (Eqs. (3.1) and (3.2)), direct differentiation worked when stagewise differentiation did not. Clearly, some additional explanation of these results is required, but we will not explore this other than to consider one more approach to the MOL solution of Eqs. (3.1) and (3.2) using `pde_3` (`mf=3`).

This discussion also suggests another point. If we had only the numerical solution for `mf=2` (and not the analytical solution), how would we know that the numerical

solution is substantially in error (since we could not compare it to an analytical solution). This is an important consideration since generally *we will not have an analytical solution to a PDE problem* (the point of computing a numerical solution is usually that we have no recourse other than a numerical solution). In other words, we are generally faced with the requirement of having to *evaluate a numerical solution to try to determine in some fashion whether it is accurate*.

We do not have a general approach to the resolution of this requirement to offer; rather each PDE problem must be considered with respect to the numerical solution, starting with the question of whether the numerical solution makes sense and is reasonable – matters of judgment rather than rigorous analysis. A practical approach to this evaluation is to observe how the numerical solution changes during h- and p-refinement; we would like to have a situation where further reductions in the grid spacing, h, and changes in the order of the approximations, p, do not change the numerical solution beyond a level that is considered acceptable, typically in an application, four significant figures.

For `mf=3`, function `pde_3` called by `ode15s` is given in Listing 3.4.

```
  function ut=pde_3(t,u)
%
% Problem parameters
  global ncall ndss n xl xu
%
% Calculate uxx
  nl=1; % Dirichlet
  nu=1; % Dirichlet
  ux=zeros(1,n);
  if    (ndss==42) uxx=dss042(xl,xu,n,u,ux,nl,nu); % second order
  elseif(ndss==44) uxx=dss044(xl,xu,n,u,ux,nl,nu); % fourth order
  elseif(ndss==46) uxx=dss046(xl,xu,n,u,ux,nl,nu); % sixth order
  elseif(ndss==48) uxx=dss048(xl,xu,n,u,ux,nl,nu); % eighth order
  elseif(ndss==50) uxx=dss050(xl,xu,n,u,ux,nl,nu); % tenth order
  end
%
% PDE
  ut=uxx';
  ut(1)=0.0;
  ut(n)=0.0;
%
% Increment calls to pde_3
  ncall=ncall+1;
```

Listing 3.4. Routine `pde_3.m`

We can note the following points about `pde_3`:

1. The initial statements are the same as in `pde_1`. Then the Dirichlet BCs at $x = -10, x = 10$ are programmed.

```
  function ut=pde_3(t,u)
%
% Problem parameters
  global ncall ndss n xl xu
%
% Calculate uxx
  nl=1; % Dirichlet
  nu=1; % Dirichlet
```

2. The second-order spatial derivative $\partial^2 u/\partial x^2 = u_{xx}$ is computed.

```
ux=zeros(1,n);
if    (ndss==42) uxx=dss042(xl,xu,n,u,ux,nl,nu); % second order
elseif(ndss==44) uxx=dss044(xl,xu,n,u,ux,nl,nu); % fourth order
elseif(ndss==46) uxx=dss046(xl,xu,n,u,ux,nl,nu); % sixth order
elseif(ndss==48) uxx=dss048(xl,xu,n,u,ux,nl,nu); % eighth order
elseif(ndss==50) uxx=dss050(xl,xu,n,u,ux,nl,nu); % tenth order
end
```

Five library routines, `dss042` to `dss050`, are programmed that use second-order to tenth-order FD approximations, respectively, for a second derivative. Since `ndss=44` is specified in the main program, `dss044` is used in the calculation of `uxx`. Also, these differentiation routines have two parameters that specify the type of BCs: (a) `nl=1` or 2 specifies a Dirichlet or a Neumann BC, respectively, at the lower boundary value of $x = xl(= -10)$, and (b) `nu=1` or 2 specifies a Dirichlet or a Neumann BC, respectively, at the upper boundary value of $x = xu(= 10)$.

For the present analysis of Eq. (3.1), Dirichlet BCs at $x = -10, 10$ are specified as `nl=nu=1`. For these Dirichlet BCs, u_x (= `ux`) is not used by `dss044`. However, it is an input argument to `dss044` and Matlab requires that input arguments to a function have at least one assigned value. This is accomplished with the statement `ux=zeros(1,n);`, but again, this has no effect on the calculation of `uxx`.

3. Finally, Eq. (3.1) is programmed and the Dirichlet BCs at $x = -10, 10$ are applied.

```
%
% PDE
  ut=uxx';
  ut(1)=0.0;
  ut(n)=0.0;
%
```

```
% Increment calls to pde_3
  ncall=ncall+1;
```

Again, the transpose is required by `ode15s`.

The numerical and plotted output for this case (`mf=3`) is given in Table 3.2.

Table 3.2. Output for `mf=3` from `pde_1_main` and `pde_3`

```
mf =  3   abstol = 1.0e-004   reltol = 1.0e-004

   t      x          u(num)         u(anal)            err
0.50 -10.0        0.000000        0.000000       -0.000000
0.50  -9.8        0.000000        0.000000        0.000000
0.50  -9.6        0.000000        0.000000        0.000000
0.50  -9.4        0.000000        0.000000        0.000000
0.50  -9.2        0.000000        0.000000        0.000000
0.50  -9.0        0.000000        0.000000        0.000000
        .                                           .
        .                                           .
        .                                           .
0.50  -6.0        0.000000        0.000000       -0.000000
0.50  -5.8        0.000000        0.000000       -0.000000
0.50  -5.6        0.000000        0.000000       -0.000000
0.50  -5.4        0.000000        0.000000       -0.000000
0.50  -5.2        0.000001        0.000001       -0.000000
0.50  -5.0        0.000001        0.000001       -0.000000
0.50  -4.8        0.000004        0.000004       -0.000000
0.50  -4.6        0.000010        0.000010       -0.000000
0.50  -4.4        0.000024        0.000025       -0.000001
0.50  -4.2        0.000058        0.000059       -0.000001
0.50  -4.0        0.000132        0.000134       -0.000001
0.50  -3.8        0.000290        0.000292       -0.000002
0.50  -3.6        0.000610        0.000612       -0.000002
0.50  -3.4        0.001232        0.001232       -0.000001
0.50  -3.2        0.002386        0.002384        0.000002
0.50  -3.0        0.004438        0.004432        0.000006
0.50  -2.8        0.007926        0.007915        0.000011
0.50  -2.6        0.013597        0.013583        0.000014
0.50  -2.4        0.022408        0.022395        0.000014
0.50  -2.2        0.035483        0.035475        0.000008
0.50  -2.0        0.053987        0.053991       -0.000004
0.50  -1.8        0.078932        0.078950       -0.000018
0.50  -1.6        0.110891        0.110921       -0.000030
```

0.50	-1.4	0.149694	0.149727	-0.000033
0.50	-1.2	0.194161	0.194186	-0.000025
0.50	-1.0	0.241961	0.241971	-0.000009
0.50	-0.8	0.289698	0.289692	0.000007
0.50	-0.6	0.333242	0.333225	0.000017
0.50	-0.4	0.368290	0.368270	0.000020
0.50	-0.2	0.391062	0.391043	0.000020
0.50	0.0	0.398961	0.398942	0.000019
0.50	0.2	0.391062	0.391043	0.000020
0.50	0.4	0.368290	0.368270	0.000020
0.50	0.6	0.333242	0.333225	0.000017
0.50	0.8	0.289698	0.289692	0.000007
0.50	1.0	0.241961	0.241971	-0.000009
0.50	1.2	0.194161	0.194186	-0.000025
0.50	1.4	0.149694	0.149727	-0.000033
0.50	1.6	0.110891	0.110921	-0.000030
0.50	1.8	0.078932	0.078950	-0.000018
0.50	2.0	0.053987	0.053991	-0.000004
0.50	2.2	0.035483	0.035475	0.000008
0.50	2.4	0.022408	0.022395	0.000014
0.50	2.6	0.013597	0.013583	0.000014
0.50	2.8	0.007926	0.007915	0.000011
0.50	3.0	0.004438	0.004432	0.000006
0.50	3.2	0.002386	0.002384	0.000002
0.50	3.4	0.001232	0.001232	-0.000001
0.50	3.6	0.000610	0.000612	-0.000002
0.50	3.8	0.000290	0.000292	-0.000002
0.50	4.0	0.000132	0.000134	-0.000001
0.50	4.2	0.000058	0.000059	-0.000001
0.50	4.4	0.000024	0.000025	-0.000001
0.50	4.6	0.000010	0.000010	-0.000000
0.50	4.8	0.000004	0.000004	-0.000000
0.50	5.0	0.000001	0.000001	-0.000000
0.50	5.2	0.000001	0.000001	-0.000000
0.50	5.4	0.000000	0.000000	-0.000000
0.50	5.6	0.000000	0.000000	-0.000000
0.50	5.8	0.000000	0.000000	-0.000000
0.50	6.0	0.000000	0.000000	-0.000000
	.			.
	.			.
	.			.
0.50	9.0	0.000000	0.000000	0.000000
0.50	9.2	0.000000	0.000000	0.000000
0.50	9.4	0.000000	0.000000	0.000000

(*continued*)

Table 3.2 (*continued*)

```
0.50   9.6       0.000000       0.000000       0.000000
0.50   9.8       0.000000       0.000000       0.000000
0.50  10.0       0.000000       0.000000      -0.000000

Integral of u(x,t=0.50) =  1.0000

         .                                        .
         .                                        .
         .                                        .

              Output for t = 1, 1.5 removed

         .                                        .
         .                                        .
         .                                        .

   t     x         u(num)        u(anal)            err
2.00 -10.0      -0.000000       0.000001      -0.000001
2.00  -9.8       0.000001       0.000001      -0.000000
2.00  -9.6       0.000002       0.000002      -0.000000
2.00  -9.4       0.000003       0.000003      -0.000000
2.00  -9.2       0.000005       0.000005      -0.000000
2.00  -9.0       0.000008       0.000008      -0.000000
2.00  -8.8       0.000012       0.000012      -0.000000
2.00  -8.6       0.000019       0.000019      -0.000000
2.00  -8.4       0.000029       0.000029      -0.000000
2.00  -8.2       0.000044       0.000045      -0.000000
2.00  -8.0       0.000067       0.000067      -0.000000
2.00  -7.8       0.000099       0.000099      -0.000000
2.00  -7.6       0.000146       0.000146      -0.000000
2.00  -7.4       0.000212       0.000212      -0.000000
2.00  -7.2       0.000306       0.000306      -0.000000
2.00  -7.0       0.000436       0.000436      -0.000000
2.00  -6.8       0.000616       0.000616       0.000000
2.00  -6.6       0.000862       0.000861       0.000000
2.00  -6.4       0.001193       0.001192       0.000001
2.00  -6.2       0.001634       0.001633       0.000001
2.00  -6.0       0.002217       0.002216       0.000001
2.00  -5.8       0.002978       0.002976       0.000002
2.00  -5.6       0.003959       0.003958       0.000002
2.00  -5.4       0.005212       0.005210       0.000002
2.00  -5.2       0.006793       0.006791       0.000002
2.00  -5.0       0.008765       0.008764       0.000001
2.00  -4.8       0.011198       0.011197       0.000001
2.00  -4.6       0.014163       0.014164      -0.000000
```

2.00	-4.4	0.017736	0.017737	-0.000001
2.00	-4.2	0.021989	0.021992	-0.000002
2.00	-4.0	0.026992	0.026995	-0.000003
2.00	-3.8	0.032804	0.032808	-0.000004
2.00	-3.6	0.039470	0.039475	-0.000005
2.00	-3.4	0.047020	0.047025	-0.000005
2.00	-3.2	0.055456	0.055460	-0.000005
2.00	-3.0	0.064755	0.064759	-0.000004
2.00	-2.8	0.074861	0.074864	-0.000002
2.00	-2.6	0.085684	0.085684	-0.000000
2.00	-2.4	0.097095	0.097093	0.000002
2.00	-2.2	0.108931	0.108926	0.000005
2.00	-2.0	0.120994	0.120985	0.000008
2.00	-1.8	0.133053	0.133043	0.000010
2.00	-1.6	0.144857	0.144846	0.000012
2.00	-1.4	0.156138	0.156127	0.000011
2.00	-1.2	0.166621	0.166612	0.000009
2.00	-1.0	0.176037	0.176033	0.000004
2.00	-0.8	0.184134	0.184135	-0.000001
2.00	-0.6	0.190688	0.190694	-0.000006
2.00	-0.4	0.195510	0.195521	-0.000011
2.00	-0.2	0.198462	0.198476	-0.000014
2.00	0.0	0.199456	0.199471	-0.000015
2.00	0.2	0.198462	0.198476	-0.000014
2.00	0.4	0.195510	0.195521	-0.000011
2.00	0.6	0.190688	0.190694	-0.000006
2.00	0.8	0.184134	0.184135	-0.000001
2.00	1.0	0.176037	0.176033	0.000004
2.00	1.2	0.166621	0.166612	0.000009
2.00	1.4	0.156138	0.156127	0.000011
2.00	1.6	0.144857	0.144846	0.000012
2.00	1.8	0.133053	0.133043	0.000010
2.00	2.0	0.120994	0.120985	0.000008
2.00	2.2	0.108931	0.108926	0.000005
2.00	2.4	0.097095	0.097093	0.000002
2.00	2.6	0.085684	0.085684	-0.000000
2.00	2.8	0.074861	0.074864	-0.000002
2.00	3.0	0.064755	0.064759	-0.000004
2.00	3.2	0.055456	0.055460	-0.000005
2.00	3.4	0.047020	0.047025	-0.000005
2.00	3.6	0.039470	0.039475	-0.000005
2.00	3.8	0.032804	0.032808	-0.000004
2.00	4.0	0.026992	0.026995	-0.000003
2.00	4.2	0.021989	0.021992	-0.000002
2.00	4.4	0.017736	0.017737	-0.000001

(*continued*)

Table 3.2 (*continued*)

2.00	4.6	0.014163	0.014164	−0.000000
2.00	4.8	0.011198	0.011197	0.000001
2.00	5.0	0.008765	0.008764	0.000001
2.00	5.2	0.006793	0.006791	0.000002
2.00	5.4	0.005212	0.005210	0.000002
2.00	5.6	0.003959	0.003958	0.000002
2.00	5.8	0.002978	0.002976	0.000002
2.00	6.0	0.002217	0.002216	0.000001
2.00	6.2	0.001634	0.001633	0.000001
2.00	6.4	0.001193	0.001192	0.000001
2.00	6.6	0.000862	0.000861	0.000000
2.00	6.8	0.000616	0.000616	0.000000
2.00	7.0	0.000436	0.000436	−0.000000
2.00	7.2	0.000306	0.000306	−0.000000
2.00	7.4	0.000212	0.000212	−0.000000
2.00	7.6	0.000146	0.000146	−0.000000
2.00	7.8	0.000099	0.000099	−0.000000
2.00	8.0	0.000067	0.000067	−0.000000
2.00	8.2	0.000044	0.000045	−0.000000
2.00	8.4	0.000029	0.000029	−0.000000
2.00	8.6	0.000019	0.000019	−0.000000
2.00	8.8	0.000012	0.000012	−0.000000
2.00	9.0	0.000008	0.000008	−0.000000
2.00	9.2	0.000005	0.000005	−0.000000
2.00	9.4	0.000003	0.000003	−0.000000
2.00	9.6	0.000002	0.000002	−0.000000
2.00	9.8	0.000001	0.000001	−0.000000
2.00	10.0	0.000000	0.000001	−0.000001

```
Integral of u(x,t=2.00) =  1.0000

ncall =  199
```

The plotted output from `pde_1_main` is shown in Figures 3.5 and 3.6. The 3D plot from `pde_1_main` is shown in Figure 3.7. Note in particular the numerical approximation of the IC, Eq. (3.2), for small t and how this sharp change in $u(x, t)$ diffuses away for later t.

We can note the following details of this output:

1. The agreement between the analytical and numerical solutions is quite acceptable (from Table 3.2 and Figure 3.6) and is, in fact, better than that from `pde_1` (from Table 3.1 and Figure 3.2).
2. The integral constraint of Eq. (3.5) is closely approximated (e.g., `Integral of u(x,t=2.00) = 1.0000`).
3. The computational effort is quite modest with `ncall = 199`.

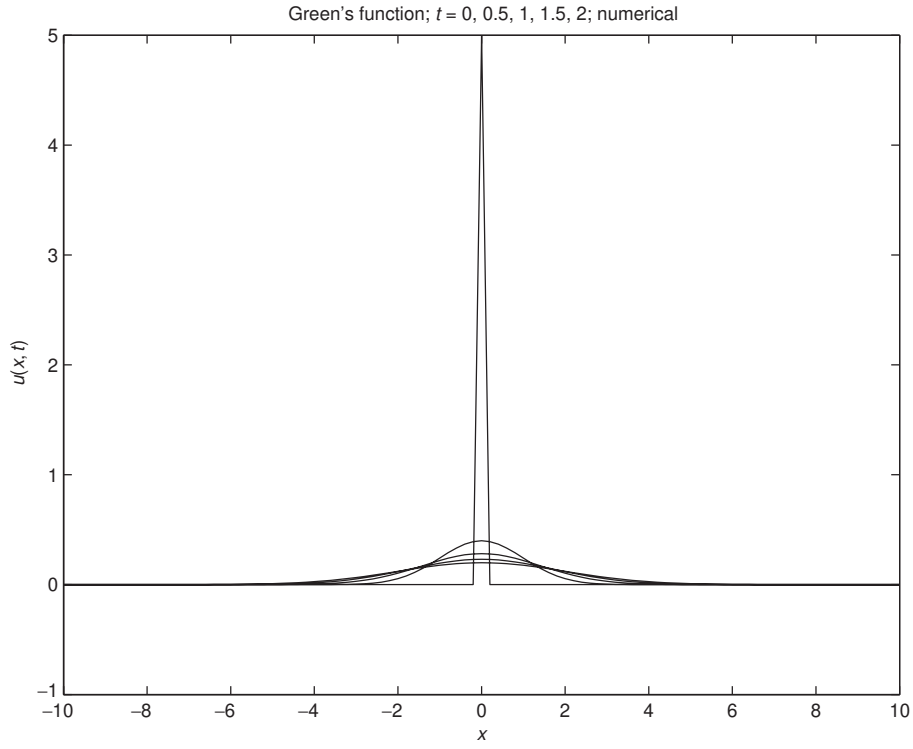

Figure 3.5. Numerical solution from `pde_1_main` including $t = 0$ (`mf`=3)

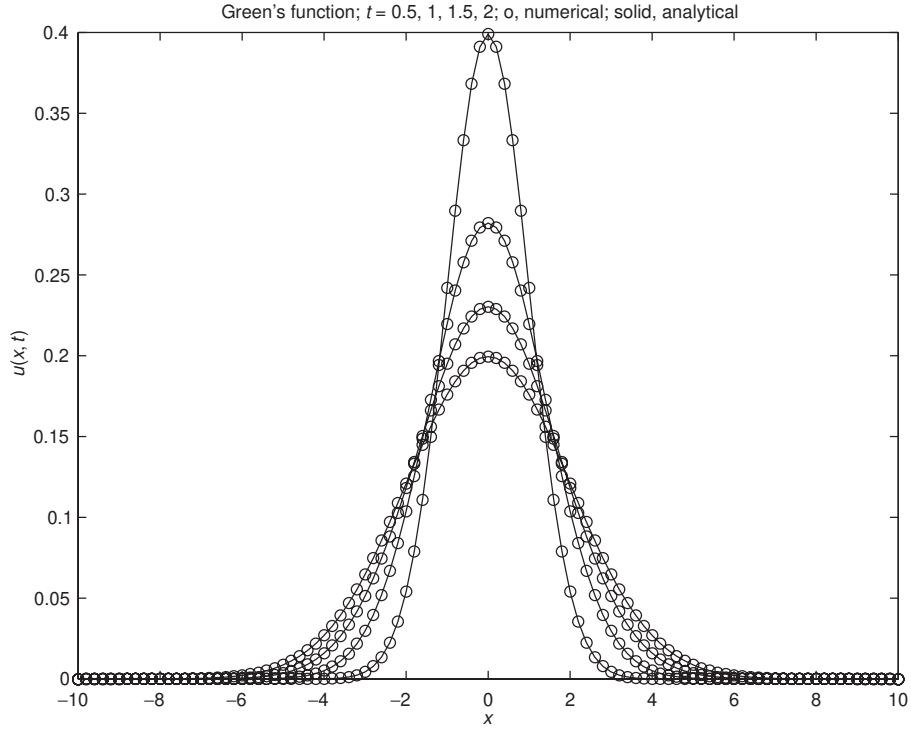

Figure 3.6. Analytical and numerical solutions from `pde_1_main` for $t \neq 0$; (`mf`=3)

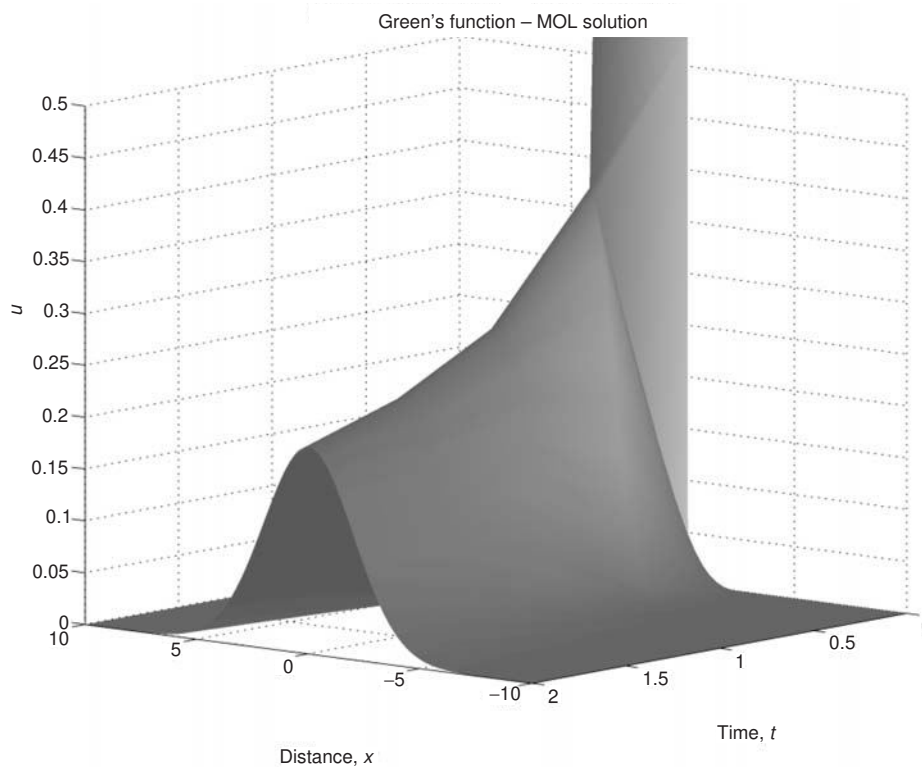

Figure 3.7. Numerical solution from `pde_1_main` including $t = 0$ (`mf=3`) in 3D

Thus, we can again conclude that direct calculation of u_{xx} (via Eq. (3.7)) works well at least for this problem (Eqs. (3.1) and (3.2)).

We conclude this chapter with some additional brief discussion:

1. The Green's function, Eq. (3.4d), is an analytical solution to the diffusion equation (3.1) for the delta function $\delta(x)$ IC of Eq. (3.2). In particular, this Green's function is for an infinite spatial domain. Solutions of Eq. (3.1) for other IC functions represented as $f(x)$ are available through the superposition of an integral (Eq. 3.4a).
2. These properties of the Green's function are applicable to a broad spectrum of (linear) PDE problems. Here are some considerations when attempting a Green's function solution of a PDE:
   (a) Spatial domain, which can be infinite (e.g., $-\infty \leq x \leq \infty$), semi-infinite (e.g., $0 \leq x \leq \infty$), or finite (e.g., $x_l \leq x \leq x_u$). The extent of the spatial domain has a major effect on the form of the Green's function. For example, for an infinite domain, the Green's function tends to zero for large $|x|$ (as with the exponential of Eq. (3.4d)). For a finite domain, the Green's function is generally an infinite series of eigenfunctions.
   (b) Also, for the semi-infinite and finite domains, the form of the Green's function depends on the type of boundary conditions, that is, Dirichlet, Neumann, or a third-type (Robin) BC.

(c) The form of the Green's function is also determined by the coordinate system (Cartesian coordinates in the case of Eq. (3.1)). For example, the Green's function in cylindrical and spherical coordinates is composed of functions specific to these coordinate systems.
(d) If the PDE is 1D, 2D, or 3D, the corresponding Green's functions is made up of parts for each coordinate dimension. For example, for a 3D PDE specified on a finite domain in each coordinate, the Green's function would be a triple infinite series.
(e) As with most analytical methods of PDE solution, the Green's function is quite limited in the class of problems to which it can be applied. For example, it generally applies only to linear PDEs, and practically to a single linear PDE.
(f) But since the Green's function provides analytical PDE solutions, it can be used to test numerical solutions as in the preceding application.
(g) Extensive collections of Green's functions are available for the variety of PDE characteristics mentioned earlier (spatial domain, coordinate system, boundary conditions, number of dimensions), notably those by Polyanin [2].
(h) Finally, the preceding example demonstrates that the Green's function is a convenient analytical method for the solution of PDEs, primarily because (a) a delta function IC facilitates the evaluation of integrals (such as Eq. (3.4b)) and (b) solutions for other ICs (than a delta function) can be readily constructed (using the idea of superposition as in Eq. (3.4d)). However, the delta function is not easy to handle numerically and requires some form of numerical approximation (such as was used in the `for` loop for IC (3.2) of `pde_1_main`).

3. The success of direct numerical differentiation (via Eq. (3.7)) and the failure of stagewise differentiation (in `pde_2`) indicates that numerical procedures that seem reasonable (such as stagewise differentiation) do not always work. Thus, PDE/MOL analysis is to some extent a trial-and-error process to arrive at an accurate numerical solution (with reasonable computing effort).
4. As a related matter, you could ask why stagewise differentiation would ever be used. The answer, generally, is that it facilitates PDE/MOL solutions when direct differentiation cannot be applied as easily. For example, the nonlinear differential group

$$\frac{\partial\left(u\dfrac{\partial u}{\partial x}\right)}{\partial x} = (uu_x)_x \tag{3.8}$$

could occur when analyzing the nonlinear diffusion equation

$$u_t = (D(u)u_x)_x \tag{3.9}$$

with $D(u) = u$.

Stagewise differentiation could be applied conveniently to term (3.8) by first calculating u_x, then multiplying this derivative by u to produce uu_x, and then differentiating this product to produce $(uu_x)_x$. Direct calculation of term (3.8) would probably first require expansion as $uu_{xx} + (u_x)^2$. Both approaches

would be worth trying (and hopefully at least one would work, and ideally both would give essentially the same numerical solution). A third approach would be to program an FD approximation specifically for the term of interest; for example,

$$(uu_x)_x \approx \frac{u_{i+1}\left(\frac{u_{i+2}-u_i}{2\Delta x}\right) - u_{i-1}\left(\frac{u_i - u_{i-2}}{2\Delta x}\right)}{2\Delta x} \tag{3.10}$$

Care is required in this approach for developing an approximation to a term in a PDE to be sure, for example, that it converges to the term of interest as the grid spacing becomes arbitrarily small. One way to check such an approximation is to assume a functional form for $u(x, t)$, for example, a polynomial, which can be substituted into the approximation and then evaluated numerically. Also, the exact value of the term can be computed (using the assumed functional form for $u(x, t)$) and compared with the numerical value.

APPENDIX A

A.1. Verification of Eq. (3.4b) as the Solution to Eq. (3.1)

Equation (3.4b) can be verified as a solution to Eq. (3.1) by direct substitution. First, the derivative u_t is

$$\begin{aligned} u_t(x,t) &= \frac{1}{2\sqrt{\pi D t}} e^{-x^2/4Dt}(-x^2/4D)(-1/t^2) + \frac{1}{2\sqrt{\pi D}}(-1/2)t^{-3/2}\, e^{-x^2/4Dt} \\ &= \left[\frac{1}{2\sqrt{\pi D t}}(-x^2/4D)(-1/t^2) + \frac{1}{2\sqrt{\pi D}}(-1/2)t^{-3/2}\right] e^{-x^2/4Dt} \\ &= \frac{1}{2\sqrt{\pi D}t^{3/2}}\left[\frac{1}{t}(x^2/4D) - (1/2)\right] e^{-x^2/4Dt} \end{aligned} \tag{3.11a}$$

u_x is

$$u_x(x,t) = \frac{1}{2\sqrt{\pi D t}} e^{-x^2/4Dt}(-2x/4Dt)$$

Differentiation of this result with respect to x gives u_{xx}

$$\begin{aligned} Du_{xx}(x,t) &= \frac{D}{2\sqrt{\pi D t}}\left[e^{-x^2/4Dt}(-2/4Dt) + e^{-x^2/4Dt}(-2x/4Dt)^2\right] \\ &= \frac{1}{2\sqrt{\pi D}t^{3/2}}\left[(-1/2) + (x^2/4Dt)\right] e^{-x^2/4Dt} \end{aligned} \tag{3.11b}$$

We see then that Eqs. (3.11a) and (3.11b) satisfy Eq. (3.1).

A.2. The Function `simp`

The integral of Eq. (3.5) is evaluated numerically by application of Simpson's rule

$$u_1(t) = \int_{x_l}^{x_u} u(x,t)dx$$
$$\approx \frac{h}{3}\left[u(1) + \sum_{i=2}^{n-2} 4u(i) + 2u(i+1) + u(n)\right] \tag{3.12}$$

where x_u and x_l are the upper and lower limits of the integral. The integration interval is therefore $h = (x_u - x_l)/(n-1)$, with n the number of quadrature (spatial grid) points in x (must be odd). For Eq. (3.5), $x_l = -\infty$, $x_u = \infty$, $n = 101$.

Function `simp` is a straightforward implementation of Eq. 3.12 (see Listing 3.5).

```
  function uint=simp(xl,xu,n,u)
%
% Function simp computes the integral of the numerical
% solution by Simpson's rule
%
    h=(xu-xl)/(n-1);
    uint=u(1)-u(n);
      for i=3:2:n
        uint=uint+4.0*u(i-1)+2.0*u(i);
      end
      uint=h/3.0*uint;
    end
```

Listing 3.5. Numerical quadrature routine `simp` applied to Eq. (3.5)

REFERENCES

[1] Farlow, S. J. (1993), The D'Alembert Solution of the Wave Equation, *Partial Differential Equations for Scientists and Engineers*, Dover Publications, New York, Chapter 17, pp. 129–136

[2] Polyanin, A. D. (2002), *Handbook of Linear Partial Differential Equations for Scientists and Engineers*, Chapman and Hall/CRC, Boca Raton, FL