14

Diffusion Equation in Spherical Coordinates[1]

An important goal of current cellular biophysics is the characterization of signal-dependent redistribution of macromolecules in living cells. In this chapter we present a simplified system of equations governing diffusional redistribution in three dimensions (3D) of green fluorescent protein (GFP). However, the biophysical details are not discussed here, and the reader is referred to the original papers for more details [1, 2].

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

1. To demonstrate the origin of a PDE starting with the relevant differential operators in spherical coordinates.
2. These operators are then substituted in the coordinate-free PDE to arrive at the PDE in spherical coordinates.
3. Consideration is given to the variable coefficients in the PDE, specifically how they lead to various singularities.
4. Methods for resolving or regularizing the singularities are reviewed and included in the Matlab routines for the method of lines (MOL) solution of the PDE.
5. Library routines for the calculation of the PDE spatial derivatives are discussed and called from the PDE routines.

Since we are interested in computing a numerical solution to the diffusion equation in spherical coordinates, we start with some differential operators in spherical coordinates (r, θ, ϕ).

[1] This chapter was written in collaboration with Dr. Peter D. Calvert, Department of Ophthalmology, Upstate New York Medical University, Syracuse, NY, and Dr. E. N. Pugh, Jr., Department of Ophthalmology, University of Pennsylvania, Philadelphia, PA.

$\nabla\cdot$ (*gradient of a vector*):

$$\begin{aligned}[\nabla]_r &= \frac{1}{r^2}\frac{\partial}{\partial r}(r^2)\\ [\nabla]_\theta &= \frac{1}{r\,\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta)\\ [\nabla]_\phi &= \frac{1}{r\,\sin\theta}\frac{\partial}{\partial\phi}\end{aligned} \tag{14.1}$$

∇ (*gradient of a scalar*):

$$\begin{aligned}[\nabla]_r &= \frac{\partial}{\partial r}\\ [\nabla]_\theta &= \frac{1}{r}\frac{\partial}{\partial\theta}\\ [\nabla]_\phi &= \frac{1}{r\,\sin\theta}\frac{\partial}{\partial\phi}\end{aligned} \tag{14.2}$$

$\nabla\cdot\nabla$ (*divergence of the gradient of a scalar*):

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right)+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right)+\frac{1}{r^2\sin^2\theta}\frac{\partial^2}{\partial\phi^2} \tag{14.3}$$

$\nabla\cdot\nabla$ ($=\nabla^2$ the *Laplacian*) follows directly from the preceding components of $\nabla\cdot$ (divergence of a vector) and ∇ (gradient of a scalar).

$$\begin{aligned}\nabla\cdot\nabla &= \left(\mathbf{i}_r\frac{1}{r^2}\frac{\partial}{\partial r}(r^2)+\mathbf{j}_\theta\frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta)+\mathbf{k}_\phi\frac{1}{r\sin\theta}\frac{\partial}{\partial\phi}\right)\cdot\left(\mathbf{i}_r\frac{\partial}{\partial r}+\mathbf{j}_\theta\frac{1}{r}\frac{\partial}{\partial\theta}+\mathbf{k}_\phi\frac{1}{r\sin\theta}\frac{\partial}{\partial\phi}\right)\\ &= \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{1}{r}\frac{\partial}{\partial\theta}\right)+\frac{1}{r\sin\theta}\frac{\partial}{\partial\phi}\left(\frac{1}{r\sin\theta}\frac{\partial}{\partial\phi}\right)\\ &= \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right)+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right)+\frac{1}{r^2\sin^2\theta}\frac{\partial^2}{\partial\phi^2}\end{aligned}$$

The *diffusion equation in coordinate-free form* is

$$\frac{\partial u}{\partial t}=D\nabla^2u \tag{14.4}$$

Substitution of ∇^2 from Eq. (14.3) gives the *diffusion equation in spherical coordinates:*

$$\frac{\partial u}{\partial t}=D\left\{\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial u}{\partial r}\right)+\frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial u}{\partial\theta}\right)+\frac{1}{r^2\sin^2\theta}\frac{\partial^2u}{\partial\phi^2}\right\} \tag{14.5}$$

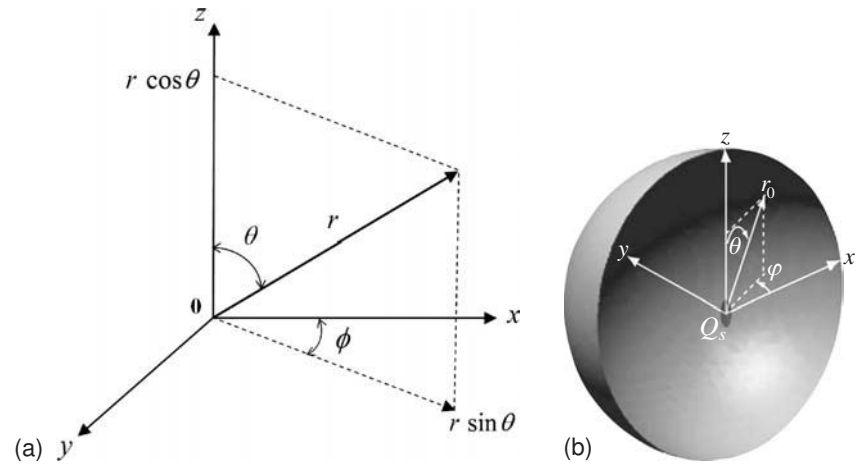

Figure 14.1. (a) Relationship between Cartesian (x, y, z) and spherical (r, θ, ϕ) coordinates. (b) The spherical coordinate system used in the analysis of diffusion in 3D, where a source or sink of protein mass Q_s is created at the cell center. Simplified protein redistribution by diffusion is illustrated in the text by two examples: the first by solving at spherical grid points defined on the radial vector r, assuming angular symmetry, that is, no variations in polar angle θ or azimuthal angle ϕ; and the second by solving at spherical grid points on r and polar angle grid points on θ, but assuming azimuthal angular symmetry, that is, no variations in angle ϕ (image reproduced from [2] with permission of the authors)

To define the angles θ and ϕ, the relationship between *Cartesian coordinates* (x, y, z) and spherical coordinates (r, θ, ϕ) is

$$\begin{aligned} x &= r\ \sin\theta\ \cos\phi \\ y &= r\ \sin\theta\ \sin\phi \\ z &= r\ \cos\theta \\ r &= \sqrt{x^2 + y^2 + z^2} \end{aligned} \tag{14.6}$$

These relationships are illustrated in Figure 14.1a. Thus, ϕ is the angle of r in the x–y plane, and θ is the angle of r relative to the z axis.

Equation (14.5) can be written in expanded form (by differentiating the radial group in r and the angular group in θ as a product) as

$$\frac{\partial u}{\partial t} = D\left\{\frac{\partial^2 u}{\partial r^2} + \frac{2}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\left(\frac{\partial^2 u}{\partial \theta^2} + \frac{\cos\theta}{\sin\theta}\frac{\partial u}{\partial \theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2 u}{\partial \phi^2}\right\} \tag{14.7}$$

Equation (14.7) is first order in t, and second order in r, θ, and ϕ. However, we will select initial conditions (ICs) and boundary conditions (BCs) so that the 3D PDE of Eq. (14.7) has no variation in ϕ and therefore the third RHS term in ϕ is zero. Additionally, we will add an *inhomogeneous term* to Eq. (14.7), $f(r, \theta, t)$. Thus, Eq. (14.7) becomes

$$\frac{\partial u}{\partial t} = D\left\{\frac{\partial^2 u}{\partial r^2} + \frac{2}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\left(\frac{\partial^2 u}{\partial \theta^2} + \frac{\cos\theta}{\sin\theta}\frac{\partial u}{\partial \theta}\right)\right\} + f(r, \theta, t) \tag{14.8}$$

Equation (14.8) is the starting point for our MOL analysis. It is first order in t, and second order in r and θ. We take as the IC (for t)

$$u(r, \theta, t = 0) = 0 \tag{14.9}$$

The BCs in r are

$$\frac{\partial u(r = 0, \theta, t)}{\partial r} = 0 \tag{14.10a}$$

$$\frac{\partial u(r = r_o, \theta, t)}{\partial r} = 0 \tag{14.10b}$$

where r_o is the outer radius of the spherical system.

The BCs in θ are

$$\frac{\partial u(r, \theta = 0, t)}{\partial \theta} = 0 \tag{14.11a}$$

$$\frac{\partial u(r, \theta = \pi/2, t)}{\partial \theta} = 0 \tag{14.11b}$$

As the starting point for the MOL solution of Eqs. (14.8)–(14.11), we start with a simpler problem that does not include the angular variable θ. Thus, Eq. (14.8) reduces to

$$\frac{\partial u}{\partial t} = D\left\{\frac{\partial^2 u}{\partial r^2} + \frac{2}{r}\frac{\partial u}{\partial r}\right\} + f(r, t) \tag{14.12}$$

which has only the radial variation, and the solution is therefore $u(r, t)$. u corresponds to *protein concentration* and f to a *source or sink term representing the volumetric rate of local photoconversion*. Figure 14.1b is a diagrammatic view of the spherical coordinate system used in the diffusion analysis.

We will proceed in two steps: (1) only the singularity $2/r$ in Eq. (14.12) is considered and (2) the singularity in r and θ in Eq. (14.8), $1/r^2((\partial^2 u/\partial\theta^2) + (\cos\theta/\sin\theta)(\partial u/\partial\theta))$, is also considered. In other words, after the solution of Eq. (14.12) is concluded, we will return to Eq. (14.8) that has this considerably more complicated singularity (in r and θ).

A main program for the MOL solution of Eqs. (14.9), (14.10), and (14.12) is given in Listing 14.1.

```
%
% Clear previous files
  clear all
  clc
%
% Global area
  global r r0 nr D f tau ncall
%
% Model parameters
```

```
  D=0.1;
  r0=1.0;
  std=1.0;
  tau=1.0;
%
% Radial grid and inhomogeneous term
  nr=21;
  dr=r0/(nr-1);
  for i=1:nr
    r(i)=(i-1)*dr;
    f(i)=exp(-r(i)^2/std^2);
  end
%
% Independent variable for ODE integration
  tf=2.5;
  tout=[0.0:0.25:tf]';
  nout=11;
  ncall=0;
%
% Initial condition
  u0=zeros(nr,1);
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  [t,u]=ode15s(@pde_1,tout,u0,options);
%
% Display a heading and selected output
  fprintf('\n nr = %2d  r0 = %4.2f  std = %4.2f\n',nr,r0,std);
  for it=1:nout
    fprintf('\n t = %4.2f\n',t(it));
    for i=1:2:nr
      fprintf(' r = %4.1f   u(r,t) = %8.5f\n',r(i),u(it,i));
    end
    for i=1:nr
      u1d(i)=4.0*pi*r(i)^2*u(it,i);
    end
    I1=simp(0.0,r0,nr,u1d)/(4.0/3.0*pi*r0^3);
    fprintf(' \n integral = %7.5f\n',I1);
  end
  fprintf('\n ncall = %5d\n',ncall);
%
% Parametric plot
  figure(1);
  plot(r,u); axis([0 r0 0 1]);
  title('u(r,t), t = 0, 0.25, ..., 2.5');
        xlabel('r'); ylabel('u(r,t)')
```

```
%
% Extend integration for closer approach to steady state
  fprintf('\n Extended solution\n');
  u0=u(nout,:);
  t0=t(nout);
  tf=t0+5.0;
  tout=[t0:2.5:tf]';
  nout=3;
  [t,u]=ode15s(@pde_1,tout,u0,options);end
  for it=1:nout
    fprintf('\n t = %4.1f\n',t(it));
    for i=1:2:nr
      fprintf(' r = %4.1f   u(r,t) = %8.5f\n',r(i),u(it,i));
    end
    for i=1:nr
      u1d(i)=4.0*pi*r(i)^2*u(it,i);
    end
    I1=simp(0.0,r0,nr,u1d)/(4.0/3.0*pi*r0^3);
    fprintf(' \n integral = %7.5f\n',I1);
  end
  fprintf('\n ncall = %5d\n',ncall);
```

Listing 14.1. Main program `pde_1_main.m` for the solution of Eqs. (14.9), (14.10), and (14.12)

We can note the following points about this main program:

1. A *global* area with parameters and variables that can be shared with other routines is defined. Then some model parameters are defined numerically.

```
%
% Clear previous files
  clear all
  clc
%
% Global area
  global r r0 nr D f tau ncall
%
% Model parameters
  D=0.1;
  r0=1.0;
  std=1.0;
  tau=1.0;
```

2. A 21-point grid in r is then defined.

```
%
% Radial grid and inhomogeneous term
  nr=21;
  dr=r0/(nr-1);
  for i=1:nr
    r(i)=(i-1)*dr;
    f(i)=exp(-r(i)^2/std^2);
  end
```

At the same time, the inhomogeneous term $f(r,t)$ in Eq. (14.12) is programmed. This function is

$$f(r,t) = e^{\left[-r^2/\sigma^2\right]}[h(t) - h(t-\tau)] \tag{14.13}$$

where $h(t)$ is the *Heaviside step function*

$$h(t) = \begin{Bmatrix} 0, t < 0 \\ 1, t > 0 \end{Bmatrix} \tag{14.14}$$

The exponential in Eq. (14.13) is a *Gaussian function* with a *standard deviation* σ. τ is a delay in t and therefore $h(t) - h(t-\tau)$ is a pulse with a value of 1 over the interval $0 \le t \le \tau$ and zero otherwise. In other words, $h(t) - h(t-\tau)$ "turns on" the Gaussian function for the interval $0 \le t \le \tau$.

3. A timescale is defined over the interval $0 \le t \le 2.5$ with an output interval of 0.25 so that there are 11 output points in t (including the IC $t = 0$).

```
%
% Independent variable for ODE integration
  tf=2.5;
  tout=[0.0:0.25:tf]';
  nout=11;
  ncall=0;
%
% Initial condition
  u0=zeros(nr,1);
```

Also, homogeneous IC (14.9) is defined over the `nr` points in r.

4. The $nr = 21$ ordinary differential equations (ODEs) are integrated by `ode15s` through the ODE routine `pde_1` (discussed subsequently).

```
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  [t,u]=ode15s(@pde_1,tout,u0,options);
```

5. A heading and selected output are displayed.

```
%
% Display a heading and selected output
  fprintf('\n nr = %2d  r0 = %4.2f  std = %4.2f\n',nr,r0,std);
  for it=1:nout
    fprintf('\n t = %4.2f\n',t(it));
    for i=1:2:nr
      fprintf(' r = %4.1f   u(r,t) = %8.5f\n',r(i),u(it,i));
    end
    for i=1:nr
      u1d(i)=4.0*pi*r(i)^2*u(it,i);
    end
    I1=simp(0.0,r0,nr,u1d)/(4.0/3.0*pi*r0^3);
    fprintf(' \n integral = %7.5f\n',I1);
  end
  fprintf('\n ncall = %5d\n',ncall);
```

Also, the normalized integral

$$I_1 = \int_0^{r_o} 4\pi r^2 u(r,t)dr / \left(\frac{4}{3}\pi r_o^3\right) \tag{14.15}$$

is evaluated numerically by a call to function `simp` (discussed subsequently) that implements *Simpson's rule* for *numerical quadrature* (integration). The integral of Eq. (14.15) is an average value of $u(r, t)$ over $0 \leq r \leq r_o$. As the solution to Eq. (14.12) approaches a final state ($t \to \infty$), it approaches a uniform value given by Eq. (14.15). This property of the solution of Eq. (14.12) is demonstrated in the numerical output discussed subsequently.

The *conservation principle* or *invariant* of Eq. (14.15) is therefore a check on the numerical solution. That is, the numerical solution should approach the constant value of I_1; if it does not, the numerical solution does not *conserve mass* as stated in physical terms. The reason mass is conserved in the present case (Eq. (14.12)) is

(a) For $t > \tau$, mass is no longer added to the system through the inhomogeneous term $f(r, t)$ in Eq. (14.12).

(b) Mass cannot leave the system because of BC (14.10b), that is, this is a *no-flux* BC according to *Fick's first law*, which states that the mass flux q_m at $r = r_o$ is

$$q_m = -D\frac{\partial u(r = r_o, t)}{\partial r} \tag{14.16}$$

Since $\partial u(r = r_o, t)/\partial r = 0$ according to BC (14.10b), $q_m = 0$.

(c) Thus, since mass is not entering (for $t > \tau$) or leaving the system, the mass within the system must remain constant, which is expressed by I_1 in Eq. (14.15) remaining constant.

(d) This is an important test of the preceding code. If I_1 does not remain constant, "mass" must be accumulating in or leaking from the system due to an incorrect approximation of Eqs. (14.9), (14.10), and (14.12), that is, the numerical analysis and/or associated coding has an error.

The number of calls to the ODE routine, `pde_1`, is also displayed to give an indication of the computational effort required to compute the solution. `ncall` is incremented in `pde_1` each time this routine is called.

6. A plot of $u(r, t)$ versus r with t as a parameter ($t = 0, 0.25, \ldots, 2.5$) indicates the evolution of the solution toward the steady state of Eq. (14.15).

```
%
% Parametric plots
  figure(1);
  plot(r,u); axis([0 r0 0 1]);
  title('u(r,t), t = 0, 0.25, ..., 2.5'); xlabel('r');
        ylabel('u(r,t)')
```

The plot is discussed subsequently.

7. When the solution reaches $t = 2.5$, it has not quite reached the steady-state value of Eq. (14.15). Therefore, it is extended to $t = 7.5$ through a second call to `ode15s`.

```
%
% Extend integration for closer approach to steady state
  fprintf('\n Extended solution\n');
  u0=u(nout,:);
  t0=t(nout);
  tf=t0+5.0;
  tout=[t0:2.5:tf]';
  nout=3;
  [t,u]=ode15s(@pde_1,tout,u0,options);end
  for it=1:nout
    fprintf('\n t = %4.1f\n',t(it));
    for i=1:2:nr
      fprintf(' r = %4.1f   u(r,t) = %8.5f\n',r(i),u(it,i));
```

```
    end
    for i=1:nr
      u1d(i)=4.0*pi*r(i)^2*u(it,i);
    end
    I1=simp(0.0,r0,nr,u1d)/(4.0/3.0*pi*r0^3);
    fprintf(' \n integral = %7.5f\n',I1);
  end
  fprintf('\n ncall = %5d\n',ncall);
```

This extension illustrates an important detail. The numerical solution to an ODE system can start from a previous solution, in this case, at $t = 2.5$. Thus, a new IC is used, which is the solution at the final point of the preceding solution (i.e., $u(r, t = 2.5)$). The solution then continues on to $t = 2.5 + 5.0 = 7.5$; at this final value, the solution is close to the value from Eq. (14.15), as demonstrated in the output discussed subsequently.

During this continuation from $t = 2.5$ to $t = 7.5$, the integral of Eq. (14.15) is computed (at $t = 2.5, 5.0, 7.5$), and the integral along with the continuing `ncall` is displayed.

We briefly review the output before going on to discussions of the routines `pde_1`, `simp`. A portion of the output is given in Table 14.1.

We can note the following details about the output given in Table 14.1:

1. The integral of Eq. (14.15) increases from $t = 0$ (where it is zero from IC (14.9)) to an essentially constant value for $t > 1$ (the inhomogeneous term $f(r, t)$ of Eq. (14.12) is zero for $t > 1$ so that it does not contribute to $\partial u/\partial t$ in Eq. (14.12)).

```
t = 0.00, integral = 0.00000

t = 0.25, integral = 0.14210

t = 0.50, integral = 0.28421

t = 0.75, integral = 0.42631

t = 1.00, integral = 0.56835

t = 1.25, integral = 0.56860

     .                .
     .                .
     .                .


t = 2.50, integral = 0.56860
```

Table 14.1. Selected output from `pde_1_main`

```
nr = 21   r0 = 1.00   std = 1.00

 t = 0.00
 r =  0.0   u(r,t) =  0.00000
 r =  0.1   u(r,t) =  0.00000
 r =  0.2   u(r,t) =  0.00000
 r =  0.3   u(r,t) =  0.00000
 r =  0.4   u(r,t) =  0.00000
 r =  0.5   u(r,t) =  0.00000
 r =  0.6   u(r,t) =  0.00000
 r =  0.7   u(r,t) =  0.00000
 r =  0.8   u(r,t) =  0.00000
 r =  0.9   u(r,t) =  0.00000
 r =  1.0   u(r,t) =  0.00000

 integral = 0.00000

 t = 0.25
 r =  0.0   u(r,t) =  0.23268
 r =  0.1   u(r,t) =  0.23047
 r =  0.2   u(r,t) =  0.22397
 r =  0.3   u(r,t) =  0.21355
 r =  0.4   u(r,t) =  0.19978
 r =  0.5   u(r,t) =  0.18346
 r =  0.6   u(r,t) =  0.16556
 r =  0.7   u(r,t) =  0.14736
 r =  0.8   u(r,t) =  0.13061
 r =  0.9   u(r,t) =  0.11788
 r =  1.0   u(r,t) =  0.11268

 integral = 0.14210

 t = 0.50
 r =  0.0   u(r,t) =  0.43582
 r =  0.1   u(r,t) =  0.43189
 r =  0.2   u(r,t) =  0.42036
 r =  0.3   u(r,t) =  0.40194
 r =  0.4   u(r,t) =  0.37783
 r =  0.5   u(r,t) =  0.34970
 r =  0.6   u(r,t) =  0.31967
 r =  0.7   u(r,t) =  0.29033
 r =  0.8   u(r,t) =  0.26475
 r =  0.9   u(r,t) =  0.24647
 r =  1.0   u(r,t) =  0.23945

 integral = 0.28421
```

```
t = 0.75
r =  0.0   u(r,t) =  0.61680
r =  0.1   u(r,t) =  0.61166
r =  0.2   u(r,t) =  0.59660
r =  0.3   u(r,t) =  0.57270
r =  0.4   u(r,t) =  0.54171
r =  0.5   u(r,t) =  0.50601
r =  0.6   u(r,t) =  0.46851
r =  0.7   u(r,t) =  0.43259
r =  0.8   u(r,t) =  0.40198
r =  0.9   u(r,t) =  0.38064
r =  1.0   u(r,t) =  0.37264

integral = 0.42631

t = 1.00
r =  0.0   u(r,t) =  0.78277
r =  0.1   u(r,t) =  0.77685
r =  0.2   u(r,t) =  0.75955
r =  0.3   u(r,t) =  0.73219
r =  0.4   u(r,t) =  0.69690
r =  0.5   u(r,t) =  0.65653
r =  0.6   u(r,t) =  0.61450
r =  0.7   u(r,t) =  0.57464
r =  0.8   u(r,t) =  0.54107
r =  0.9   u(r,t) =  0.51795
r =  1.0   u(r,t) =  0.50937

integral = 0.56835

t = 1.25
r =  0.0   u(r,t) =  0.70719
r =  0.1   u(r,t) =  0.70299
r =  0.2   u(r,t) =  0.69079
r =  0.3   u(r,t) =  0.67170
r =  0.4   u(r,t) =  0.64751
r =  0.5   u(r,t) =  0.62059
r =  0.6   u(r,t) =  0.59368
r =  0.7   u(r,t) =  0.56963
r =  0.8   u(r,t) =  0.55100
r =  0.9   u(r,t) =  0.53953
r =  1.0   u(r,t) =  0.53581

integral = 0.56860
```

(*continued*)

Table 14.1 (*continued*)

```
      .                .
      .                .
      .                .

 Output for t = 1.5, 1.75, 2, 2.25
            removed
      .                .
      .                .
      .                .

 t = 2.50
 r =  0.0   u(r,t) =  0.58037
 r =  0.1   u(r,t) =  0.57997
 r =  0.2   u(r,t) =  0.57884
 r =  0.3   u(r,t) =  0.57711
 r =  0.4   u(r,t) =  0.57498
 r =  0.5   u(r,t) =  0.57269
 r =  0.6   u(r,t) =  0.57048
 r =  0.7   u(r,t) =  0.56859
 r =  0.8   u(r,t) =  0.56717
 r =  0.9   u(r,t) =  0.56632
 r =  1.0   u(r,t) =  0.56605

 integral = 0.56860

 ncall =    98

 Extended solution

 t =  2.5
 r =  0.0   u(r,t) =  0.58037
 r =  0.1   u(r,t) =  0.57997
 r =  0.2   u(r,t) =  0.57884
 r =  0.3   u(r,t) =  0.57711
 r =  0.4   u(r,t) =  0.57498
 r =  0.5   u(r,t) =  0.57269
 r =  0.6   u(r,t) =  0.57048
 r =  0.7   u(r,t) =  0.56859
 r =  0.8   u(r,t) =  0.56717
 r =  0.9   u(r,t) =  0.56632
 r =  1.0   u(r,t) =  0.56605

 integral = 0.56860
```

```
t =  5.0
r =  0.0   u(r,t) =  0.56868
r =  0.1   u(r,t) =  0.56867
r =  0.2   u(r,t) =  0.56866
r =  0.3   u(r,t) =  0.56865
r =  0.4   u(r,t) =  0.56864
r =  0.5   u(r,t) =  0.56863
r =  0.6   u(r,t) =  0.56861
r =  0.7   u(r,t) =  0.56860
r =  0.8   u(r,t) =  0.56859
r =  0.9   u(r,t) =  0.56859
r =  1.0   u(r,t) =  0.56859

integral = 0.56860

t =  7.5
r =  0.0   u(r,t) =  0.56861
r =  0.1   u(r,t) =  0.56860
r =  0.2   u(r,t) =  0.56860
r =  0.3   u(r,t) =  0.56860
r =  0.4   u(r,t) =  0.56860
r =  0.5   u(r,t) =  0.56860
r =  0.6   u(r,t) =  0.56860
r =  0.7   u(r,t) =  0.56860
r =  0.8   u(r,t) =  0.56860
r =  0.9   u(r,t) =  0.56860
r =  1.0   u(r,t) =  0.56861

integral = 0.56860

ncall =   145
```

2. The extended output indicates a convergence of the solution to the constant value of Eq. (14.15). Thus, for $t = 7.5$,

```
t =  7.5
r =  0.0   u(r,t) =  0.56861
r =  0.1   u(r,t) =  0.56860
r =  0.2   u(r,t) =  0.56860
r =  0.3   u(r,t) =  0.56860
r =  0.4   u(r,t) =  0.56860
r =  0.5   u(r,t) =  0.56860
r =  0.6   u(r,t) =  0.56860
```

```
r =  0.7   u(r,t) =  0.56860
r =  0.8   u(r,t) =  0.56860
r =  0.9   u(r,t) =  0.56860
r =  1.0   u(r,t) =  0.56861

integral = 0.56860

ncall =   145
```

Even with this extension to $t = 7.5$, the number of calls to `pde_1` is quite modest (`ncall = 145`). The plotted output is indicated in Figure 14.2. The effect of BCs (14.10a) and (14.10b) is clear from this plot (zero derivatives in r at the boundaries $r = 0, r_o$). The convergence of the solution to $u(r, t = \infty) = 0.56860$ is clear.

We now consider the remaining routines to produce the solution of Table 14.1 and Figure 14.1. The ODE routine, `pde_1`, is given in Listing 14.2.

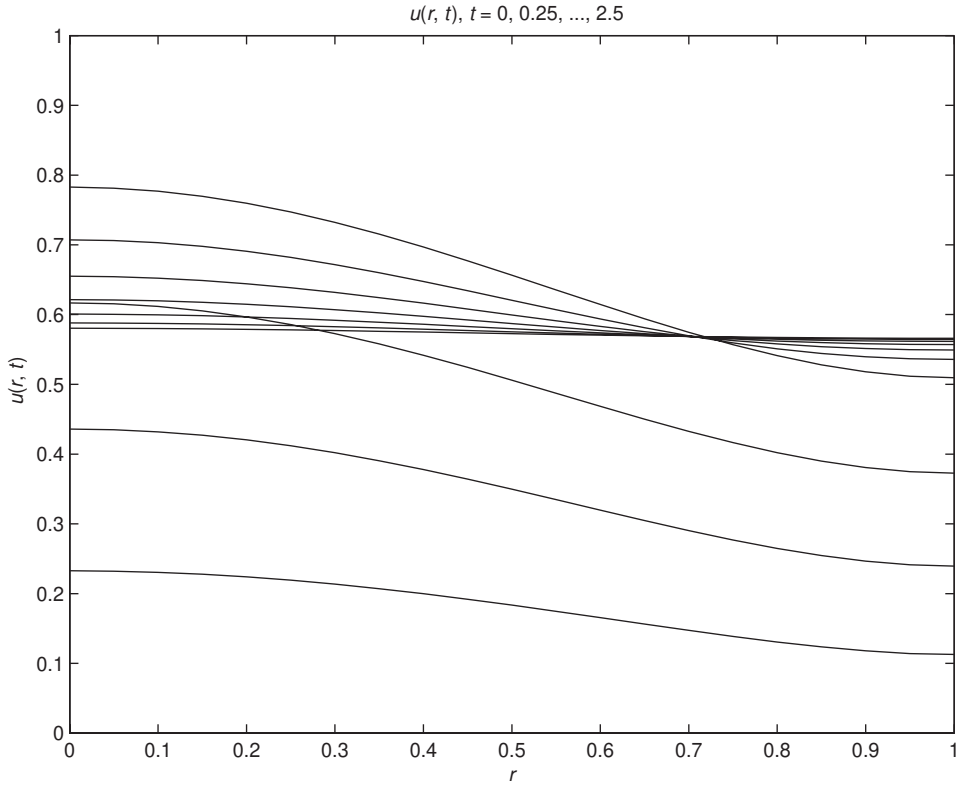

Figure 14.2. Plot of the numerical solution to Eq. (14.12) from pde_1_main and pde_1

```
  function ut=pde_1(t,u)
%
% Global area
  global r r0 nr D f tau ncall
%
% Spatial grid in r
  for i=1:nr
%
%   ur
    ur=dss004(0.0,r0,nr,u);
    ur(nr)=0.0;
    ur( 1)=0.0;
%
%   urr
    urr=dss004(0.0,r0,nr,ur);
  end
%
% PDE
  for i=1:nr
    if(t<=tau)
      fs=f(i);
    else
      fs=0.0;
    end
    if(i==1)
      ut(i)=D*3.0*urr(i)+fs;
    else
      ut(i)=D*(urr(i)+2.0/r(i)*ur(i))+fs;
    end
  end
  ut=ut';
%
% Increment number of calls to pde_1
  ncall=ncall+1;
```

Listing 14.2. pde_1 for Eq. (14.12)

We can note the following points about pde_1:

1. The function is first defined and a global area is then defined to share the problem parameters and variables with main program pde_1_main in Listing 14.1.

```
  function ut=pde_1(t,u)
%
% Global area
  global r r0 nr D f tau ncall
```

Note that the dependent variable `u` and its derivative `ut` are given the names used in Eq. (14.12) since there is only one such variable (i.e., the programming in `pde_1` can be directly in terms of `u`).

2. A `for` loop is then used to step through the `nr` radial grid points (to define the `nr` ODEs in the MOL solution of Eq. (14.12)).

```
%
% Spatial grid in r
  for i=1:nr
%
%   ur
    ur=dss004(0.0,r0,nr,u);
    ur(nr)=0.0;
    ur( 1)=0.0;
```

The first derivative $\partial u/\partial r$ = `ur` is then computed at each of the radial points by a call to `dss004`. After this call, BCs (14.10a) and (14.10b) are imposed to ensure the correct (zero) values of `ur` at $r = 0, r_o$.

3. The second derivative in r in Eq. (14.12), (`urr`), is computed by differentiating the first derivative (using *stagewise differentiation*).

```
%
%   urr
    urr=dss004(0.0,r0,nr,ur);
  end
```

The `for` loop is terminated with the `end` statement.

4. A second `for` loop is used for the programming of the `nr` MOL ODEs. This loop starts with the addition of the inhomogeneous term $f(r, t)$ in Eq. (14.12) (which is available through the global area from the main program and is nonzero for $0 \leq t \leq \tau$).

```
%
% PDE
  for i=1:nr
    if(t<=tau)
```

```
    fs=f(i);
  else
    fs=0.0;
  end
```

5. At $r = 0$ (`i=1`), the first-derivative radial group in Eq. (14.12) is indeterminate; that is,

$$\frac{2}{r}\frac{\partial u}{\partial r} = 0/0$$

as a consequence of *homogeneous Neumann BC* (14.10a). We therefore apply *l'Hospital's rule to resolve (regularize)* this indeterminant form.

$$\lim_{r\to 0}\frac{2}{r}\frac{\partial u}{\partial r} = 2\frac{\partial u^2}{\partial r^2}$$

Thus, the programming of the complete radial group in Eq. (14.12) is

$$\frac{\partial^2 u}{\partial r^2} + 2\frac{\partial^2 u}{\partial r^2} = 3\frac{\partial^2 u}{\partial r^2}$$

and the ODE at $r = 0$ is (with $f(r, t)$)

```
if(i==1)
   ut(i)=D*3.0*urr(i)+fs;
```

Again, BC (14.10a) is included in this result (from when the indeterminate form was evaluated).

6. The programming of the ODEs at the other grid points ($r \neq 0$ or $i \neq 1$) follows directly from Eq. (14.12).

```
    else
      ut(i)=D*(urr(i)+2.0/r(i)*ur(i))+fs;
    end
  end
  ut=ut';
%
% Increment number of calls to pde_1
  ncall=ncall+1;
```

BC (14.10b) is included in this computation of `ut(i)` from `ur(nr)=0` in the preceding `for` loop. The routine then concludes with a transpose of `ut` as

required by `ode15s` and incrementing of `ncall` (to record `pde_1` has been called).

Numerical integration routine `simp` is given in Listing 14.3.

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

Listing 14.3. `simp` for the Evaluation of the Integral I_1 of Eq. (14.15)

`simp` is a straightforward implementation of Simpson's rule applied to Eq. (14.15):

$$I_1 = \int_0^{r_o} 4\pi r^2 u(x,t)dr \approx \frac{h}{3}\left[u(1) + \sum_{i=2}^{n-2} 4u(i) + 2u(i+1) + u(n)\right] \tag{14.17}$$

where the factor $4\pi r^2$ has been included in `u` (in `pde_1_main`).

This concludes the MOL solution of Eqs. (14.9), (14.10), and (14.12). We now go on to the 2D problem of Eqs. (14.8)–(14.11) for the calculation of $u(r, \theta, t)$. The reason this was deferred until the simpler problem was completed is because the second RHS term in θ of Eq. (14.8) requires special attention.

In particular, this term is singular for both $r = 0$ (due to the factor $1/r^2$) and $\theta = 0, \pi/2$ (due to the factor $1/\sin\theta$). The complexity of this r–θ group makes the regularization by a l'Hospital's rule analysis considerably more complex than for the $2/r$ singularity of Eq. (14.12), so we proceed with an alternative approach [3]. Returning to the general form of the PDE, Eq. (14.4), with the inhomogeneous term included

$$\frac{\partial u}{\partial t} = D\nabla^2 u + f(r, \theta, t) \tag{14.18}$$

all that is required to compute a solution of Eq. (14.18) is the accurate evaluation of the RHS. In other words, *it is not necessary to use a particular coordinate system* for the RHS (again, ∇^2 applies to any coordinate system). Thus, at the origin $r = 0$ where a singularity of the θ group in Eq. (14.8) occurs, we can *evaluate the θ group*

in Cartesian coordinates. The reason for this choice is that *this group in Cartesian coordinates does not have a singularity at the origin*.

Thus, at the origin, Eq. (14.18) becomes (partly in Cartesian coordinates)

$$\frac{\partial u}{\partial t} = D\left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right] + f\,(r = 0, \theta, t) \tag{14.19}$$

The RHS of Eq. (14.19) appears to be a mixture of Cartesian and spherical coordinates. However, the two are related through Eqs. (14.6). In other words, we can express the derivative in x on a grid in r and θ by realizing that the x axis corresponds to $\theta = \pi/2$ (again, we have formulated the problem based on Eq. (14.8) with no variation in ϕ so that $\theta = \pi/2$ also applies to the y axis – see Figure 14.1). Also, the z axis corresponds to $\theta = 0$. With the value of θ specified, the r vector in spherical coordinates corresponds to the x, y, or z axis in Cartesian coordinates.

With these relations in mind, we can easily link the solution (of Eq. (14.19)) in r, θ to the solution in x, y, z at the origin (at $x = y = z = r = 0$). All of this might seem complicated, but we will now illustrate how this can easily be done in the ODE routine `pde_2`. We start first with a main program (from which `pde_2` is called by `ode15s`) (see Listing 14.4).

```
%
% Clear previous files
  clear all
  clc
%
% Global area
  global r r0 nr th th0 nth D std_0 std_pi2 f tau ncall
%
% Model parameters
  D=0.1;
  r0=1.0;
  th0=pi/2.0;
  std_0  =1.0;
  std_pi2=1.0;
  tau=1.0;
%
% Radial grid
  nr=11;
  dr=r0/(nr-1);
  for i=1:nr
    r(i)=(i-1)*dr;
  end
%
% Angular grid
  nth=11;
  dth=th0/(nth-1);
```

```
  for j=1:nth
    th(j)=(j-1)*dth;
  end
%
% Inhomogeneous term
  for i=1:nr
  for j=1:nth
    f(i,j)=exp(-(r(i)*sin(th(j))/std_pi2)^2 ...
               -(r(i)*cos(th(j))/std_0  )^2);
  end
  end
%
% Independent variable for ODE integration
  tf=1.0;
  tout=[0.0:0.25:tf]';
  nout=5;
  ncall=0;
%
% Initial condition
  for i=1:nr
  for j=1:nth
    y0((i-1)*nth+j)=0.0;
  end
  end
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  [t,y]=ode15s(@pde_2,tout,y0,options);
%
% 1D to 2D (3D with t)
  for it=1:nout
  for  i=1:nr
  for  j=1:nth
     u(it,i,j)=y(it,(i-1)*nth+j);
  end
  end
  end
%
% Display a heading and selected output
  fprintf('\n  nr = %2d    r0 = %4.2f', nr, r0);
  fprintf('\n nth = %2d   th0 = %4.2f',nth,th0);
  fprintf('\n   std_0 = %4.2f  ',std_0);
  fprintf('\n std_pi2 = %4.2f\n',std_pi2);
  for it=1:nout
    fprintf('\n t = %4.2f',t(it));
    for i=1:2:nr
```

```
      fprintf('\n');
     for j=1:2:nth
      fprintf(' r = %4.1f  th = %4.2f  u(r,th,t) = %8.5f\n',...
               r(i),th(j),u(it,i,j));
     end
     end
   end
   fprintf('\n ncall = %5d\n',ncall);
%
% Parametric plots
   figure(1);
   subplot(2,2,1)
   uplot(1:nr,1:5:nth)=u(2,1:nr,1:5:nth);
   plot(r,uplot); axis([0 r0 0 1]);
   title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.25');
   xlabel('r'); ylabel('u(r,th,t)');
   subplot(2,2,2)
   uplot(1:nr,1:5:nth)=u(3,1:nr,1:5:nth);
   plot(r,uplot); axis([0 r0 0 1]);
   title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.5');
   xlabel('r'); ylabel('u(r,th,t)');
   subplot(2,2,3)
   uplot(1:nr,1:5:nth)=u(4,1:nr,1:5:nth);
   plot(r,uplot); axis([0 r0 0 1]);
   title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.75');
   xlabel('r'); ylabel('u(r,th,t)');
   subplot(2,2,4)
   uplot(1:nr,1:5:nth)=u(5,1:nr,1:5:nth);
   plot(r,uplot); axis([0 r0 0 1]);
   title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 1');
   xlabel('r'); ylabel('u(r,th,t)');
```

Listing 14.4. Main program `pde_2_main.m` for the Solution of Eqs. (14.8)–(14.11)

`pde_2_main` in Listing 14.4 has several common features with `pde_1_main` in Listing 14.1. The most significant difference is that the inclusion of θ (Eq. (14.12) has only the spatial variable r, while Eq. (14.8) has the spatial variables r, θ). Thus, we have gone from a 1D to a 2D problem.

We can note the following points about this main program:

1. Again, a global area is defined followed by the specification of some problem parameters.

```
%
% Clear previous files
  clear all
  clc
%
% Global area
  global r r0 nr th th0 nth D std_0 std_pi2 f tau ncall
%
% Model parameters
  D=0.1;
  r0=1.0;
  th0=pi/2.0;
  std_0  =1.0;
  std_pi2=1.0;
  tau=1.0;
```

The interval in r is $0 \leq r \leq r_o$ and in θ is $0 \leq \theta \leq \pi/2$. Note that for θ, from symmetry, we can consider just one quadrant (θ could extend to 2π). The two parameters `std_0, std_pi2` are discussed next.

2. Equation (14.13) is now extended to 2D as

$$f(r, \theta, t) = e^{\{-[(r\cos\theta)^2/{\sigma_0}^2 + (r\sin\theta)^2/{\sigma_{\pi/2}}^2]\}}[h(t) - h(t-\tau)] \tag{14.20}$$

where σ_0 = `std_0`, $\sigma_{\pi/2}$ = `std_pi2`. The exponential in Eq. (14.20) is programmed as

```
%
% Inhomogeneous term
  for i=1:nr
  for j=1:nth
    f(i,j)=exp(-(r(i)*sin(th(j))/std_pi2)^2 ...
               -(r(i)*cos(th(j))/std_0  )^2);
  end
  end
```

The indices for r and θ, `i` and `j` respectively, will be used throughout the programming.

3. The radial and angular grids, with `nr=11` and `nth=11` points, are defined (in general, θ will be represented by the characters `th` in the programming).

```
%
% Radial grid
  nr=11;
```

```
  dr=r0/(nr-1);
  for i=1:nr
    r(i)=(i-1)*dr;
  end
%
% Angular grid
  nth=11;
  dth=th0/(nth-1);
  for j=1:nth
    th(j)=(j-1)*dth;
  end
```

4. A timescale is defined over the interval $0 \leq t \leq 1.0$ with an output interval of 0.25 so that there are five output points in t (including the IC $t = 0$).

```
%
% Independent variable for ODE integration
  tf=1.0;
  tout=[0.0:0.25:tf]';
  nout=5;
  ncall=0;
```

5. IC (14.9) follows.

```
%
% Initial condition
  for i=1:nr
  for j=1:nth
    y0((i-1)*nth+j)=0.0;
  end
  end
```

Note in particular the use of the 1D IC array `y0`.

6. The integration of the $nr \times n\text{th} = 11 \times 11 = 121$ ODEs is by `ode15s`. Note in particular the use of function `pde_2`, which is discussed subsequently.

```
%
% ODE integration
  reltol=1.0e-04; abstol=1.0e-04;
  options=odeset('RelTol',reltol,'AbsTol',abstol);
  [t,y]=ode15s(@pde_2,tout,y0,options);
```

```
%
% 1D to 2D (3D with t)
  for it=1:nout
  for  i=1:nr
  for  j=1:nth
     u(it,i,j)=y(it,(i-1)*nth+j);
  end
  end
  end
```

After the solution matrix `y` is returned by `ode15s`, it is put into a 3D array (with t index `it`, r index `i` and θ index `j`); an alternative approach would be to use the Matlab `reshape` utility as illustrated in Chapter 10.

7. Selected output (a subset of the 121 ODE dependent variables) is displayed. This output is discussed subsequently.

```
%
% Display a heading and selected output
  fprintf('\n  nr = %2d    r0 = %4.2f', nr, r0);
  fprintf('\n nth = %2d   th0 = %4.2f',nth,th0);
  fprintf('\n   std_0 = %4.2f  ',std_0);
  fprintf('\n std_pi2 = %4.2f\n',std_pi2);
  for it=1:nout
    fprintf('\n t = %4.2f',t(it));
    for i=1:2:nr
      fprintf('\n');
    for j=1:2:nth
      fprintf(' r = %4.1f th = %4.2f u(r,th,t) = %8.5f\n', ...
              r(i),th(j),u(it,i,j));
    end
    end
  end
  fprintf('\n ncall = %5d\n',ncall);
```

8. A series of four plots for $t = 0.25, 0.5, 0.75, 1.0$ is produced with every fifth value of θ plotted parametrically ($\theta = 0, \pi/4, \pi/2$). The plots are of $u(r, \theta, t)$ versus r.

```
%
% Parametric plots
  figure(1);
  subplot(2,2,1)
  uplot(1:nr,1:5:nth)=u(2,1:nr,1:5:nth);
```

```
plot(r,uplot); axis([0 r0 0 1]);
title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.25');
xlabel('r'); ylabel('u(r,th,t)');
subplot(2,2,2)
uplot(1:nr,1:5:nth)=u(3,1:nr,1:5:nth);
plot(r,uplot); axis([0 r0 0 1]);
title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.5');
xlabel('r'); ylabel('u(r,th,t)');
subplot(2,2,3)
uplot(1:nr,1:5:nth)=u(4,1:nr,1:5:nth);
plot(r,uplot); axis([0 r0 0 1]);
title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 0.75');
xlabel('r'); ylabel('u(r,th,t)');
subplot(2,2,4)
uplot(1:nr,1:5:nth)=u(5,1:nr,1:5:nth);
plot(r,uplot); axis([0 r0 0 1]);
title('u(r,th,t), th = 0, \pi/4, \pi/2; t = 1');
xlabel('r'); ylabel('u(r,th,t)');
```

This output was selected to emphasize the variation of the solution with θ, which is the essential difference between Eqs. (14.8) and (14.12). (θ is included in Eq. (14.8).) To explore this idea a bit further, for $\sigma_0 = \sigma_{\pi/2} = \sigma$, Eq. (14.20) reduces to Eq. (14.13) (so that there is no variation in θ in the solution of Eq. (14.8) (which is the case programmed in `pde_1_main` of Listing 14.1 to test the code). Selected output for this case is listed in Table 14.2.

We can note the following details about this output:

1. $u(r, \theta, t)$ becomes larger with increasing t; for the final value $t = 1$, the inhomogeneous term of Eq. (14.20) has just switched to zero ($t = \tau = 1$ in `pde_2`), and for larger t, the radial variation will diminish and approach a constant value as in the case of the output in Table 14.1.
2. There is no θ variation as expected (since $\sigma_0 = \sigma_{\pi/2} = \sigma$) in Eq. (14.20)).

The plotted output is shown in Figure 14.3. We note that there is only one curve in each plot since there is no variation in θ. Actually, there are three superimposed plots for $\theta = 0, \pi/4, \pi/2$, which is a good test since all of the *theta* code is executed but it gives the same result.

We can now change the inhomogeneous term of Eq. (14.20) so that $\sigma_0 \neq \sigma_{\pi/2}$. For example, in `pde_2_main`

```
std_0  =2.0;
std_pi2=1.0;
```

(note we are changing just one number, `std_0`, from 1 to 2). The resulting output is now as given in Table 14.3.

Table 14.2. Selected output from `pde_2_main` for $\sigma_0 = 1, \sigma_{\pi/2} = 1$

```
 nr = 11    r0 = 1.00
nth = 11   th0 = 1.57
  std_0 = 1.00
std_pi2 = 1.00

t = 0.00
r =  0.0   th = 0.00   u(r,th,t) =  0.00000
r =  0.0   th = 0.31   u(r,th,t) =  0.00000
r =  0.0   th = 0.63   u(r,th,t) =  0.00000
r =  0.0   th = 0.94   u(r,th,t) =  0.00000
r =  0.0   th = 1.26   u(r,th,t) =  0.00000
r =  0.0   th = 1.57   u(r,th,t) =  0.00000
      .                              .
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .
r =  1.0   th = 0.00   u(r,th,t) =  0.00000
r =  1.0   th = 0.31   u(r,th,t) =  0.00000
r =  1.0   th = 0.63   u(r,th,t) =  0.00000
r =  1.0   th = 0.94   u(r,th,t) =  0.00000
r =  1.0   th = 1.26   u(r,th,t) =  0.00000
r =  1.0   th = 1.57   u(r,th,t) =  0.00000

t = 0.25
r =  0.0   th = 0.00   u(r,th,t) =  0.23263
r =  0.0   th = 0.31   u(r,th,t) =  0.23263
r =  0.0   th = 0.63   u(r,th,t) =  0.23263
r =  0.0   th = 0.94   u(r,th,t) =  0.23263
r =  0.0   th = 1.26   u(r,th,t) =  0.23263
r =  0.0   th = 1.57   u(r,th,t) =  0.23263
      .                              .
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .
```

```
r =  1.0   th = 0.00   u(r,th,t) =  0.11264
r =  1.0   th = 0.31   u(r,th,t) =  0.11264
r =  1.0   th = 0.63   u(r,th,t) =  0.11264
r =  1.0   th = 0.94   u(r,th,t) =  0.11264
r =  1.0   th = 1.26   u(r,th,t) =  0.11264
r =  1.0   th = 1.57   u(r,th,t) =  0.11264
      .                              .
      .                              .
      .                              .

      Output for t = 0.5, 0.75 removed

      .                              .
      .                              .
      .                              .

t = 1.00
r =  0.0   th = 0.00   u(r,th,t) =  0.78288
r =  0.0   th = 0.31   u(r,th,t) =  0.78288
r =  0.0   th = 0.63   u(r,th,t) =  0.78288
r =  0.0   th = 0.94   u(r,th,t) =  0.78288
r =  0.0   th = 1.26   u(r,th,t) =  0.78288
r =  0.0   th = 1.57   u(r,th,t) =  0.78288
      .                              .
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .
r =  1.0   th = 0.00   u(r,th,t) =  0.50943
r =  1.0   th = 0.31   u(r,th,t) =  0.50943
r =  1.0   th = 0.63   u(r,th,t) =  0.50943
r =  1.0   th = 0.94   u(r,th,t) =  0.50943
r =  1.0   th = 1.26   u(r,th,t) =  0.50943
r =  1.0   th = 1.57   u(r,th,t) =  0.50943

ncall =   156
```

Table 14.3. Selected output from `pde_2_main` for $\sigma_0 = 2, \sigma_{\pi/2} = 1$

```
 nr = 11    r0 = 1.00
nth = 11   th0 = 1.57
  std_0 = 2.00
std_pi2 = 1.00

t = 0.00
r =  0.0   th = 0.00   u(r,th,t) =  0.00000
r =  0.0   th = 0.31   u(r,th,t) =  0.00000
r =  0.0   th = 0.63   u(r,th,t) =  0.00000
r =  0.0   th = 0.94   u(r,th,t) =  0.00000
r =  0.0   th = 1.26   u(r,th,t) =  0.00000
r =  0.0   th = 1.57   u(r,th,t) =  0.00000
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .

r =  1.0   th = 0.00   u(r,th,t) =  0.00000
r =  1.0   th = 0.31   u(r,th,t) =  0.00000
r =  1.0   th = 0.63   u(r,th,t) =  0.00000
r =  1.0   th = 0.94   u(r,th,t) =  0.00000
r =  1.0   th = 1.26   u(r,th,t) =  0.00000
r =  1.0   th = 1.57   u(r,th,t) =  0.00000

t = 0.25
r =  0.0   th = 0.00   u(r,th,t) =  0.24586
r =  0.0   th = 0.31   u(r,th,t) =  0.24586
r =  0.0   th = 0.63   u(r,th,t) =  0.24586
r =  0.0   th = 0.94   u(r,th,t) =  0.24586
r =  0.0   th = 1.26   u(r,th,t) =  0.24586
r =  0.0   th = 1.57   u(r,th,t) =  0.24586
      .                              .
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .
```

```
r =  1.0   th = 0.00   u(r,th,t) =  0.21035
r =  1.0   th = 0.31   u(r,th,t) =  0.19360
r =  1.0   th = 0.63   u(r,th,t) =  0.16146
r =  1.0   th = 0.94   u(r,th,t) =  0.13026
r =  1.0   th = 1.26   u(r,th,t) =  0.11164
r =  1.0   th = 1.57   u(r,th,t) =  0.10523
      .                              .
      .                              .
      .                              .

      Output for t = 0.5, 0.75 removed

      .                              .
      .                              .
      .                              .

t = 1.00
r =  0.0   th = 0.00   u(r,th,t) =  0.94674
r =  0.0   th = 0.31   u(r,th,t) =  0.94674
r =  0.0   th = 0.63   u(r,th,t) =  0.94674
r =  0.0   th = 0.94   u(r,th,t) =  0.94674
r =  0.0   th = 1.26   u(r,th,t) =  0.94674
r =  0.0   th = 1.57   u(r,th,t) =  0.94674
      .                              .
      .                              .
      .                              .

 Output for r = 0.2, 0.4, 0.6, 0.8 removed

      .                              .
      .                              .
      .                              .
r =  1.0   th = 0.00   u(r,th,t) =  0.88327
r =  1.0   th = 0.31   u(r,th,t) =  0.77077
r =  1.0   th = 0.63   u(r,th,t) =  0.65852
r =  1.0   th = 0.94   u(r,th,t) =  0.54209
r =  1.0   th = 1.26   u(r,th,t) =  0.47481
r =  1.0   th = 1.57   u(r,th,t) =  0.43531

ncall =   157
```

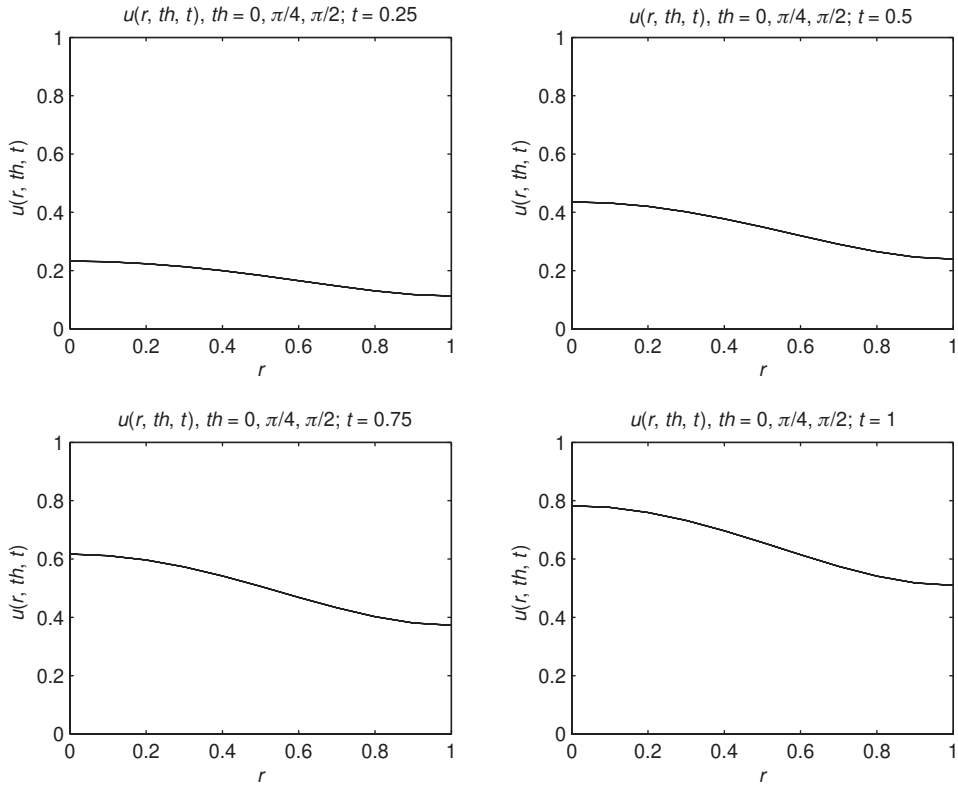

Figure 14.3. Plot of the numerical solution to Eq. (14.8) from pde_2_main and pde_2 for $\sigma_0 = 1, \sigma_{\pi/2} = 1$

We can note the following details about this output:

1. The θ variation is evident (except for $r = 0$, which is just a point value, so the θ variation is not meaningful).
2. The computational effort is quite modest (`ncall = 157`).

The plotted output is shown in Figure 14.4. The three curves in each plot correspond to $\theta = 0, \pi/4, \pi/2$. The overall level of $u(r, \theta, t)$ increases with t as would be expected since the inhomogeneous term in Eq. (14.20) continues to add mass until $t = 1$. Also, the solution is elongated in the $\theta = 0$ (or z) direction (the top curve of the set of three, and also, as demonstrated in Table 14.4) as expected with $\sigma_0 = 2, \sigma_{\pi/2} = 1$ in the Gaussian ellipsoid of Eq. (14.20).

To assist in the interpretation of the numerical solution of Figure 14.4, we have also included a discussion of an animation in Appendix 6.

To conclude this discussion, we now consider `pde_2`. The challenge here is to approximate the θ group in Eq. (14.8). The r group can be approximated as in the case of Eq. (14.12). The ODE routine for Eq. (14.8) is given in Listing 14.5.

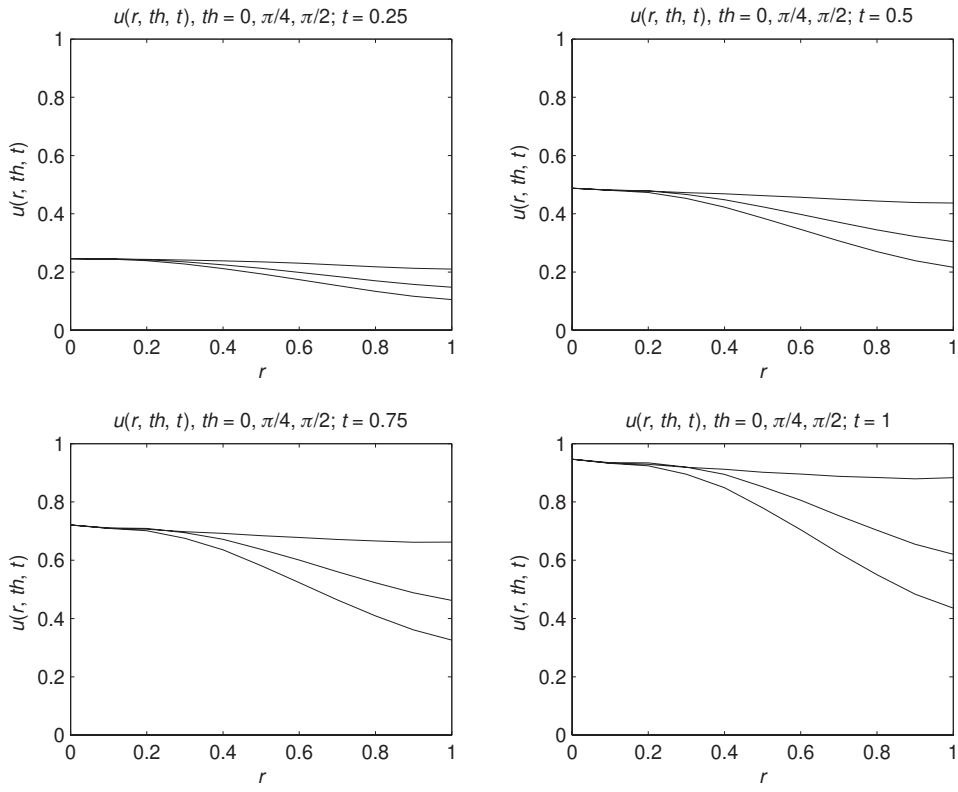

Figure 14.4. Plot of the numerical solution to Eq. (14.8) from pde_2_main and pde_2 for $\sigma_0 = 2, \sigma_{\pi/2} = 1$

```
  function yt=pde_2(t,y)
%
% Global area
  global r r0 nr th th0 nth D std_0 std_pi2 f tau ncall
%
% 1D to 2D
  for i=1:nr
  for j=1:nth
    u(i,j)=y((i-1)*nth+j);
  end
  end
%
% Spatial grids in r and theta
  for i=1:nr
  for j=1:nth
%
%   ur
```

```
    u1d=u(:,j);
    ur1d=dss004(0.0,r0,nr,u1d);
    ur1d(nr)=0.0;
    ur1d( 1)=0.0;
    ur(:,j)=ur1d;
%
%   urr
    urr1d=dss004(0.0,r0,nr,ur1d);
    urr(:,j)=urr1d;
%
%   uth
    u1d=u(i,:);
    uth1d=dss004(0.0,th0,nth,u1d);
    uth1d(nth)=0.0;
    uth1d(  1)=0.0;
    uth(i,:)=uth1d;
%
%   uthth
    uthth1d=dss004(0.0,th0,nth,uth1d);
    uthth(i,:)=uthth1d;
  end
  end
%
% PDE
  for i=1:nr
  for j=1:nth
    if(t<=tau)
      fs=f(i,j);
    else
      fs=0.0;
    end
%
%   r=0
    if(i==1)
%
%     th = 0
      if(j==1)
        u1d(:)=u(:,1);
        ur1d=dss004(0.0,r0,nr,u1d);
        ur1d( 1)=0.0;
        ur1d(nr)=0.0;
        urr1d=dss004(0.0,r0,nr,ur1d);
        urr(:,1)=urr1d;
%
%     th = pi/2
      elseif(j==nth)
        u1d(:)=u(:,nth);
```

```
        ur1d=dss004(0.0,r0,nr,u1d);
        ur1d( 1)=0.0;
        ur1d(nr)=0.0;
        urr1d=dss004(0.0,r0,nr,ur1d);
        urr(:,nth)=urr1d;
      end
%
%     PDE for r = 0
      ut(i,j)=D*(2.0*urr(i,nth)+urr(i,1))+fs;
%
%   r ~= 0
    elseif(i~=1)
%
%     th = 0, pi/2
      if(j==1|j==nth)
        ut(i,j)=D*(urr(i)+2.0/r(i)*ur(i)...
                   +(2.0/r(i)^2)*uthth(i,j))+fs;
      else
%
%     th ~= 0, pi/2
      ut(i,j)=D*(urr(i)+2.0/r(i)*ur(i)+(1.0/r(i)^2)...
               *(uthth(i,j)+cos(j)/sin(j)*uth(i,j)))+fs;
      end
    end
  end
  end
%
% 2D to 1D
  for i=1:nr
  for j=1:nth
    yt((i-1)*nth+j)=ut(i,j);
  end
  end
  yt=yt';
%
% Increment number of calls to pde_2
  ncall=ncall+1;
```

Listing 14.5. pde_2 for eq. (14.8)

We can note the following points about pde_2:

1. The function is defined and a global area is specified for the parameters and variables shared with pde_2_main of Listing 14.4.

```
  function yt=pde_2(t,y)
%
% Global area
  global r r0 nr th th0 nth D std_0 std_pi2 f tau ncall
%
% 1D to 2D
  for i=1:nr
  for j=1:nth
    u(i,j)=y((i-1)*nth+j);
  end
  end
```

The ODE dependent-variable vector `y` is then converted to a 2D array, `u(i,j)`, where the indices for r and θ are `i` and `j`, respectively. Thus, we can program in terms of the problem-oriented variable u of Eq. (14.8).

2. Two nested `for` loops step through the r–θ plane (for a total of $nr \times n\text{th} = 121$ ODEs). The derivative $\partial u/\partial r$ is calculated by first storing u in a 1D array, `u1d`, which is then differentiated by a call to `dss004` to give the 1D derivative array `ur1d`. BCs (14.10a) and (14.10b) are imposed and the 1D derivative array is stored in the 2D derivative array `ur` for subsequent use in the programming of Eq. (14.8).

```
%
% Spatial grids in r and theta
  for i=1:nr
  for j=1:nth
%
%   ur
    u1d=u(:,j);
    ur1d=dss004(0.0,r0,nr,u1d);
    ur1d(nr)=0.0;
    ur1d( 1)=0.0;
    ur(:,j)=ur1d;
```

3. The second derivative $\partial^2 u/\partial r^2$ is calculated by a second call to `dss004` applied to the first derivative (using stagewise differentiation). The final result is a 2D array, `urr`, with the second derivative in r.

```
%
%   urr
    urr1d=dss004(0.0,r0,nr,ur1d);
    urr(:,j)=urr1d;
```

4. For the derivatives in θ, we consider the two cases $r = 0$ and $r \neq 0$. For $r = 0$ (`i=1`), $\theta = 0$ (`j=1`), the θ group in Eq. (14.8) is approximated with

```
%
%   r=0
    if(i==1)
%
%     th = 0
      if(j==1)
        u1d(:)=u(:,1);
        ur1d=dss004(0.0,r0,nr,u1d);
        ur1d( 1)=0.0;
        ur1d(nr)=0.0;
        urr1d=dss004(0.0,r0,nr,ur1d);
        urr(:,1)=urr1d;
```

We have here used the Laplacian in Cartesian coordinates as explained with Eq. (14.19). The derivative $\partial^2 u/\partial z^2$ is the derivative $\partial^2 u(r, \theta = 0, t)/\partial^2 r$ (refer to Figure 14.1 to see that the radial vector along $\theta = 0$ is the z axis). The array `urr(:,1)` therefore contains the derivative $\partial^2 u/\partial z^2$. Note also that the preceding code includes BCs (14.11a) and (14.11b).

5. A similar argument equates the $\partial^2 u/\partial x^2$ to $\partial^2 u(r, \theta = \pi/2, t)/\partial^2 r$ (refer to Fig. 14.1 to see that the radial vector along $\theta = \pi/2$ is the x axis; it is also the y axis since we are not considering variations in ϕ). In other words to obtain $\partial^2 u/\partial x^2$ *and* $\partial^2 u/\partial y^2$, we use

```
%
%     th = pi/2
      elseif(j==nth)
        u1d(:)=u(:,nth);
        ur1d=dss004(0.0,r0,nr,u1d);
        ur1d( 1)=0.0;
        ur1d(nr)=0.0;
        urr1d=dss004(0.0,r0,nr,ur1d);
        urr(:,nth)=urr1d;
      end
```

Note the subscript `j = nth` corresponding to $\theta = \pi/2$. Also, this code includes BCs (14.11a) and (14.11b).

6. Having the three derivatives in Cartesian coordinates (the derivatives in x, y, and z of Eq. (14.19)) we can program Eq. (14.19) (which is a replacement for Eq. (14.8) at $r = 0$ or `i=1`) as

```
%
%     PDE for r = 0
      ut(i,j)=D*(2.0*urr(i,nth)+urr(i,1))+fs;
```

The factor 2 in the first RHS term reflects the fact that $\partial^2 u/\partial x^2$ and $\partial^2 u/\partial y^2$ in Eq. (14.19) are the same; again, note index `nth` in the first RHS term (for x and y), and `1` in the second RHS term (for z). Thus, we have handled the r and θ groups of Eq. (14.8) at $r = 0$ by using Eq. (14.19) instead.

7. For $r \neq 0$, the situation is somewhat simpler, except that we still need to consider the singularity for $\theta = 0, \pi/2$ due to the $1/\sin\theta$ term. However, this can be handled by l'Hospital's rule.

$$\lim_{\theta \to 0} \frac{\cos\theta}{\sin\theta} \frac{\partial u}{\partial \theta} = \frac{\cos\theta}{\cos\theta} \frac{\partial^2 u}{\partial \theta^2}$$

Thus, the θ group becomes

$$\frac{1}{r^2} \left(\frac{\partial^2 u}{\partial \theta^2} + \frac{\cos\theta}{\sin\theta} \frac{\partial u}{\partial \theta} \right) = \frac{2}{r^2} \frac{\partial^2 u}{\partial \theta^2} \tag{14.21}$$

Equation (14.8) with Eq. (14.21) (for both $\theta = 0, \pi/2$) is programmed as

```
%
%   r ~= 0
    elseif(i~=1)
%
%     th = 0, pi/2
      if(j==1|j==nth)
        ut(i,j)=D*(urr(i)+2.0/r(i)*ur(i)...
                +(2.0/r(i)^2)*uthth(i,j))+fs;
```

8. For $r > 0$ and $\theta \neq 0$ and $\theta \neq \pi/2$, Eq. (14.8) can be programmed directly as

```
      else
%
%     th ~= 0, pi/2
      ut(i,j)=D*(urr(i)+2.0/r(i)*ur(i)+(1.0/r(i)^2)...
                *(uthth(i,j)+cos(j)/sin(j)*uth(i,j)))+fs;
      end
    end
  end
  end
```

Note the similarity of this programming to Eq. (14.8). The double `end` completes the programming of the two nested `for` loops at the beginning (for stepping through r and θ).

9. This completes the programming of the RHS of Eq. (14.8) (or (14.19) where required). The 2D array for $\partial u/\partial t$ (in array `ut(i,j)`) is put into a 1D array `yt` for use by `ode15s`. The usual transpose and incrementing of `ncall` are included.

```
%
% 2D to 1D
  for i=1:nr
  for j=1:nth
    yt((i-1)*nth+j)=ut(i,j);
  end
  end
  yt=yt';
%
% Increment number of calls to pde_2
  ncall=ncall+1;
```

In summary, our intention in presenting this chapter is to illustrate how:

1. A PDE in spherical coordinates can be integrated numerically by the MOL.
2. Singularities in the PDE can be evaluated.
3. To include a no-flux BC (in this case, BC (14.10b)).
4. To use a conservation principle (in this case, Eq. (14.15)) to check the numerical solution.

The complexity of this procedure, particularly the resolution of singularities resulting from variable coefficients in the PDE, is justified if the physical system is spherical, which facilitates the analysis, particularly the use of BCs such as Eqs. (14.10) and (14.11).

REFERENCES

[1] Calvert, P. D., K. J. Strissel, W. E. Schiesser, E. N. Pugh, Jr., and V. Y. Arshavsky (November 2006), Light-Driven Translocation of Signaling Proteins in Vertebrate Photoreceptors, *Trends in Cell Biology*, **16**(11): 560–568

[2] Calvert, P. D., J. A. Peet, A. Bragin, W. E. Schiesser, and E. N. Pugh, Jr. (January 2007), Fluorescence Relaxation in 3D from Diffraction-Limited Sources of PAGFP or Sinks of EGFP Created by Multiphoton Photoconversion, *Journal of Microscopy*, **225**(1): 49–71

[3] Smith, G. D. (1965), *Numerical Solution of Partial Differential Equations*, Oxford University Press, London, pp. 44–45