# 22C:171/22M:171 Numerical Analysis II
# Programming assignment
# Due Wednesday, May 7th, 2014

In this assignment we will consider the problem of solving a reaction–diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left[D\frac{\partial u}{\partial x}\right] + \phi(t,x,u) \qquad \text{in } \Omega,$$

where $\Omega$is a region in $\mathbb{R}$. The solution $u(t,x)$ can represent the concentration of chemical at time $t$ and position $x$. The coefficient $D$ is the *diffusion coefficient* for the chemical, and $\phi(t,x,u)$ is the rate at which the chemical is being produced.

To make things simple, we will assume that $\Omega = (0,1)$, with boundary conditions $u(t,0) = u(t,1) = 0$. We have initial conditions $u(0,x) = u_0(x)$ with $u_0(x)$ a given function.

## Space discretization

To approximate the functions over $\Omega$, we will use a regular grid with spacing $k = \Delta x$ in the $x$ direction with $k = 1/N$: $x_i = i\,k$.

So for each time $t$ we have a value $u_i(t) \approx u(t,x_i)$. The differential equation satisfied by $u_i(t)$ is given by means of a centered difference formula $\nabla \cdot [D\nabla u]$ for constant $D$:

$$\frac{du_{ij}}{dt} = \frac{D}{k^2}\left[u_{i+1} + u_{i-1} - 2\,u_i\right] + \phi(t,x_i,u_i).$$

This can be written in the vector form

$$\frac{d\mathbf{u}}{dt} = B_k\mathbf{u} + \boldsymbol{\phi}(t,\mathbf{u})$$

where $B_k$ is the matrix (or linear operator) where $\mathbf{w} = B_k\mathbf{u}$ means

$$w_i = \frac{D}{k^2}\left[u_{i+1} + u_{i-1} - 2\,u_i\right],$$

and $\phi_i(t,\mathbf{u}) = \phi(t,x_i,u_i)$.

## Time discretization

We will use an implicit Runge–Kutta method known as the Radau IIA method of order 3. It's Butcher tableau is

$$\begin{array}{c|cc} 1/3 & 5/12 & -1/12 \\ 1 & 3/4 & 1/4 \\ \hline & 3/4 & 1/4 \end{array}$$

Recall that for $dy/dt = f(t, y)$, for the time-step from $t = t_n$to $t = t_{n+1}$we have to solve the equations

$$v_{n,p} = f\left(t_n + c_p h,\, y_n + h \sum_{q=1}^{s} a_{pq} v_{n,q}\right)$$

for $v_{n,p}$, $p = 1, 2, \ldots, s$ and then set

$$y_{n+1} = y_n + h \sum_{q=1}^{s} b_q v_{n,q}.$$

Now $y_n$ is our approximation of $u_i(t_n)$, $i = 0, 1, \ldots, N-1$.

Let $\mathbf{u}_n = \mathbf{u}(t_n)$. For our reaction–diffusion problem, this means solving

$$\mathbf{v}_{n,p} = B_k\left(\mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}\right) + \boldsymbol{\phi}\left(t_n + c_p h, \mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}\right), \qquad p = 1, \ldots, s.$$

Since the derivative part of the right-hand side of the reaction–diffusion equation is dominant, we will set up an iterative scheme

$$\mathbf{v}_{n,p}^{(r+1)} = B_k\left(\mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}^{(r+1)}\right) + \boldsymbol{\phi}\left(t_n + c_p h, \mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}^{(r)}\right), \qquad p = 1, \ldots, s.$$

for $r = 0, 1, 2, \ldots$. The task now before use, is if we know

$$\mathbf{b}_p^{(r)} := \boldsymbol{\phi}\left(t_n + c_p h, \mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}^{(r)}\right),$$

how do we solve

$$\mathbf{v}_{n,p}^{(r+1)} = B_k \left( \mathbf{u}_n + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q}^{(r+1)} \right) + \mathbf{b}_p^{(r)}, \qquad p = 1, \ldots, s \,?$$

Let's drop the superscripts and the subscript $n$ for now, and write it as

$$\mathbf{v}_p = B_k \left( \mathbf{u} + h \sum_{q=1}^{s} a_{pq} \mathbf{v}_{n,q} \right) + \mathbf{b}_p, \qquad p = 1, \ldots, s.$$

Stacking the components $\left(\mathbf{v}_p\right)_i$ like this:

$$\begin{bmatrix} \left(\mathbf{v}_1\right)_1 \\ \left(\mathbf{v}_2\right)_1 \\ \left(\mathbf{v}_1\right)_2 \\ \left(\mathbf{v}_2\right)_2 \\ \left(\mathbf{v}_1\right)_3 \\ \vdots \\ \left(\mathbf{v}_1\right)_{N-1} \\ \left(\mathbf{v}_2\right)_{N-1} \end{bmatrix}$$

we obtain a banded linear system, which can be put into block form

$$\begin{bmatrix} I + 2hDA/k^2 & -hDA/k^2 & & & & \\ -hDA/k^2 & I + 2hDA/k^2 & -hDA/k^2 & & & \\ & -hDA/k^2 & I + 2hDA/k^2 & \ddots & & \\ & & \ddots & \ddots & -hDA/k^2 & \\ & & & -hDA/k^2 & I + 2hDA/k^2 & \end{bmatrix} \begin{bmatrix} \left(\mathbf{v}_*\right)_1 \\ \left(\mathbf{v}_*\right)_2 \\ \left(\mathbf{v}_*\right)_3 \\ \vdots \\ \left(\mathbf{v}_*\right)_{N-1} \end{bmatrix}$$

$$= \begin{bmatrix} \left(B_k\mathbf{u} + \mathbf{b}_*\right)_1 \\ \left(B_k\mathbf{u} + \mathbf{b}_*\right)_2 \\ \left(B_k\mathbf{u} + \mathbf{b}_*\right)_3 \\ \\ \left(B_k\mathbf{u} + \mathbf{b}_*\right)_{N-1} \end{bmatrix}$$

where $(\mathbf{v}_*)_i = [(\mathbf{v}_1)_i, (\mathbf{v}_2)_i, \ldots, (\mathbf{v}_s)_i]^T$ and $(\mathbf{b}_*)_i = [(\mathbf{b}_1)_i, (\mathbf{b}_2)_i, \ldots, (\mathbf{b}_s)_i]^T$. Implement a block tridiagonal linear LU factorization

$$\begin{bmatrix} A_1 & C_1 & & & \\ B_1 & A_2 & C_2 & & \\ & B_2 & A_3 & \ddots & \\ & & \ddots & \ddots & C_{N-2} \\ & & & B_{N-2} & A_{N-1} \end{bmatrix}$$

$$= \begin{bmatrix} L_1 & & & & \\ M_1 & L_2 & & & \\ & M_2 & L_3 & & \\ & & \ddots & \ddots & \\ & & & M_{N-2} & L_{N-1} \end{bmatrix} \begin{bmatrix} U_1 & V_1 & & & \\ & U_2 & V_2 & & \\ & & U_3 & \ddots & \\ & & & \ddots & V_{N-2} \\ & & & & U_{N-2} \end{bmatrix}$$

Note that this gives the equations

$$\begin{aligned} A_1 &= L_1 U_1 \\ C_1 &= L_1 V_1 \\ B_1 &= M_1 U_1 \\ A_2 &= L_2 U_2 + M_1 V_1 \\ C_2 &= L_2 V_2 \\ B_2 &= M_2 U_2 \\ A_3 &= L_3 U_3 + M_2 V_2, \qquad \text{etc.} \end{aligned}$$

which can be solved by means of $LU$ factorizations and solving for matrices.

## Implementation & problem to solve

Implement the block tridiagonal solver as a separate routine, which should also be tested before using it as part of the ODE solver. You should also implement and test the block forward and backward substitution routines separately as well. The block tridiagonal matrices should be stored in a memory efficient way, not (for example) as a standard $sN \times sN$ matrix! Matlab has some data structures (such as cell arrays) that are useful for doing this.

For the test problem, set $D = 10^{-2}$ and $\phi(u) = u(1-u^2)$. For testing, set $(\mathbf{u}_0)_i = \sin(3\pi i/N)$. For the spatial discretization, set $N = 100$ and for the time discretization, set $h = 0.01$. Repeat the process with $N$ doubled, $h$ halved, and both $((N,h) = (200, 0.01), (100, 0.005), (200, 0.005))$ and integrate over a time interval of $[0,T]$ with $T = 20$. Now try $\mathbf{u}_0$ random, but with both positive and negative entries. In Matlab this can be done with either `u0 = rand(N-1,1)-rand(N-1,1)`, or `u0 = randn(N-1,1)`.

Provide evidence to give confidence to the reader that the code is working correctly. (As well as separately testing the linear algebra routines, you could set $\phi(t,x,u) = 0$ and check that the solution is close to the solution of the corresponding heat equation.) Also, for a given exact solution $\widehat{u}(t,x)$, we can solve

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left[D\frac{\partial u}{\partial x}\right] + \phi(t,x,u) - \frac{\partial}{\partial x}\left[D\frac{\partial \widehat{u}}{\partial x}\right] - \phi(t,x,\widehat{u}) + \frac{\partial \widehat{u}}{\partial t}(t,x).$$

## Report writing

Write your report as if you are writing it for a mathematically knowledgeable colleague or engineer who does not know anything in particular about this problem (or this course). Give evidence that your results are to be trusted. (Did you test the routines? What were the results of the tests?) Explain your results, especially anything unexpected, as well as you can. Present your results in a form that is easy to understand. (You may need prose, mathematical equations, tables, and graphs; use them where appropriate.) You will need to describe the situation in written English, and how you proceeded. Clarity of explanation is important.

You should provide the software (provided it is not too awkward), so that others can run and check your code. This means that your code should be documented so that others can read it and understand how it works. You should also provide documentation to explain how to use the software you wrote. (The Matlab `help` system is a good place to start for this.) If supplying the code and documentation on paper is impractical, consider supplying it electronically.