Workout — FDM

Non-mandatory exercises

3. In general, a centered, second order finite difference approximation of a second derivative u'', can be written as $(u_{i+1} - 2u_i + u_{i-1})/h^2$, where h is the step size.

   (a) Use this idea to formulate a finite difference approximation of the Laplace equation $u_{xx} + u_{yy} = 0$. Denote the step sizes by Δx and Δy, respectively.

$$u_{i,j} = u(x_i, y_j), i = 1, \ldots, M, j = 1, \ldots, N$$
$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} = 0$$

   (b) Suppose that this approximation is applied to the Laplace equation on the unit square, i.e., $0 \le x \le 1$, $0 \le y \le 1$, with given boundary values. This will result in an algebraic system of equations. Express this system on matrix-vector form.
   Boundary conditions:

$$\begin{aligned} u_{0,j} &= f_j, && j = 0, \ldots, N \\ u_{M,j} &= g_j, && j = 0, \ldots, N \\ u_{i,0} &= h_i, && i = 0, \ldots, M \\ u_{i,N} &= k_i, && i = 0, \ldots, M \end{aligned}$$

   The system of equations is

$$\begin{pmatrix} A_{1,1} & A_{1,2} & 0 & & & \\ A_{2,1} & & & & & \\ 0 & & \ddots & & & \\ & \ddots & \ddots & & & 0 \\ & & & & & A_{N-2,N-1} \\ & & 0 & A_{N-1,N-2} & & A_{N-1,N-1} \end{pmatrix} \begin{pmatrix} U_1 \\ \\ \vdots \\ \\ U_{N-1} \end{pmatrix} = \begin{pmatrix} b_1 \\ \\ \vdots \\ \\ b_{N-1} \end{pmatrix}$$

   where $A_{i,j}$ are the $(M-1) \times (M-1)$ matrices

$$A_{i,i} = \begin{pmatrix} \frac{2}{(\Delta x)^2} - \frac{2}{(\Delta y)^2} & \frac{1}{(\Delta x)^2} & 0 & \cdots & 0 \\ \frac{1}{(\Delta x)^2} & & & & \vdots \\ 0 & & \ddots & & 0 \\ \vdots & & & & \frac{1}{(\Delta x)^2} \\ 0 & \cdots & 0 & \frac{1}{(\Delta x)^2} & -\frac{2}{(\Delta x)^2} - \frac{2}{(\Delta y)^2} \end{pmatrix},$$

$$A_{i,i+1} = A_{i+1,i} = \frac{1}{(\Delta y)^2} I,$$

and U_j and b_j are vectors on the form

$$U_{i\in[1,N-1]} = \begin{pmatrix} u_{1,i} \\ \vdots \\ u_{M-1,i} \end{pmatrix},$$

$$b_1 = \begin{pmatrix} -\frac{f_1}{(\Delta x)^2} - \frac{h_1}{(\Delta y)^2} \\ -\frac{h_2}{\Delta y)^2} \\ \vdots \\ -\frac{h_{M-2}}{(\Delta y)^2} \\ -\frac{g_1}{(\Delta x)^2} - \frac{h_{M-1}}{(\Delta y)^2} \end{pmatrix}, \quad b_{N-1} = \begin{pmatrix} -\frac{f_{N-1}}{(\Delta x)^2} - \frac{k_1}{(\Delta y)^2} \\ -\frac{k_2}{\Delta y)^2} \\ \vdots \\ -\frac{k_{M-2}}{(\Delta y)^2} \\ -\frac{g_{N-1}}{(\Delta x)^2} - \frac{k_{M-1}}{(\Delta y)^2} \end{pmatrix}, b_{i\in[2,N-2]} = \begin{pmatrix} -\frac{f_i}{(\Delta x)^2} \\ 0 \\ \vdots \\ 0 \\ -\frac{g_i}{(\Delta x)^2} \end{pmatrix}.$$

(c) The coeffiecient matrix has a special structure. What will be the complexity for LU factorization of this matrix, if you take advantage of the structure? Your answer can refer to known complexity results from the course Scientific Computing I (*Beräkningsvetenskap I*).

The coefficient matrix is 5-diagonal, but two of the diagonals are located $M-1$ steps away from the main diagonal. This will cause fill-in, and the cost for LU-decomposition is $\sim \mathcal{O}\left(M^2 N\right)$ for large systems.

4. One way to solve time dependent PDEs numerically is to use the "method of lines". This means to first discretize in space but not in time (a so called *semi-discretization*). The result will be a system of ODE. This system can then be solved with a numerical ODE solver.

Now, apply this idea to the heat equation. For the space discretization, use the same approximation of the second order derivative as in Exercise 3 above. Formulate the corresponding ODE system and apply the Trapezoid method to it to get a fully discrete approximation. You do not need to carry out any computations, only to express the approximation as a formula. Do you recognize the resulting method? What is its name according to the course text book, Chapter 9?

The heat equation:

$$u_t = u_{xx}$$

MOL-discretization, using the second order derivative approximation from Exercise 3:

$$u_i'(t) = \frac{u_{i-1}(t) - 2u_i(t) + u_{i+1}(t)}{h^2}$$

The trapezoidal method:

$$\frac{u^{n+1} - u^n}{k} = \frac{1}{2}\left(f\left(u^n\right) + f\left(u^{n+1}\right)\right)$$

Combining the two, we get:

$$\frac{u^{n+1} - u^n}{k} = \frac{1}{2}\left(\frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{h^2} + \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{h^2}\right)$$

This is the Crank-Nicholson method.

5. Suppose that you are going to solve the heat equation $u_t = cu_{xx}$ with a finite difference method. First you discretize in space, using the second order centered finite difference approximation $(u_{j-1} - 2u_j + u_{j+1})/(\Delta x)^2$ of the second derivative. Then you want to choose between Explicit Euler and Implicit Euler for advancing the solution in time. You want to use the time marching method (Explicit or Implicit Euler) that gives the shortest execution time for a given time step Δt. Which method will you choose? Is there any benefit in using the more expensive method?

   Explicit (forward) Euler should be the cheaper one, as we only need one matrix-vector multiplication per iteration. In implicit (backward) Euler, we need to solve a system of equations for each iteration.

   In this case, the stability criterion for Explicit Euler is $\frac{c\Delta t}{(\Delta x)^2} \leq 1$ and for Implicit Euler $\frac{c\Delta t}{\Delta x} \leq 1$. This means that we can use much bigger time steps for Implicit Euler and still have a stable scheme. As a consequence, Implicit Euler is usually faster in practice. If we, for example, use $\Delta x = 0.1$, we need $\Delta t \sim 0.01$ for Explicit Euler to be stable, while $\Delta t \sim 0.1$ is enough for Implicit Euler. Reaching the final solution would then require ten times as many time steps if we were to use Explicit Euler.