Email csyoo@unist.ac.kr your own MATLAB codes together with the results. Zip your source codes and results in the name of your student id and name, i.e. 20160001-홍길동.zip

1. (40 points) Consider a 2-D heat equation,

$$\frac{\partial T}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) + S(x,y), \tag{1}$$

with the boundary and initial conditions

$$\text{B.C.: } T(0,y,t) = T(1,y,t) = 0,\ \partial T/\partial y|_{x,0,t} = 0,\ T(x,1,t) = 0 \quad \text{for } t > 0,$$

$$\text{I.C.: } T(x,y,0) = 0 \quad \text{for} \quad 0 \le x \le 1 \ \text{ and } \ 0 \le y \le 1.$$

$$S(x,y) = 1.0 \ \text{ for } 0 < x < 1 \text{ and } 0 < y < 1 \ \text{ for } t > 0.$$

a) (20 points) We want to solve the PDE using the approximate factorization method (AFM).

① Considering application of the Crank-Nicolson method and the second-order spatial differencing to the PDE, write the PDE in the operator notation. In other words, express the discretized equation using δ_{xx} and δ_{yy} and the corresponding order of errors. Note that

$$\delta_{xx} T = \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\Delta x^2}.$$

② Show that each side of the discretized equation can be factored by two one-dimensional tridiagonal matrices without any loss in the order of accuracy and the final factored from of the discretized equation is written as:

$$\left(\mathbf{I} - \frac{\alpha\Delta t}{2}\delta_{xx}\right)\left(\mathbf{I} - \frac{\alpha\Delta t}{2}\delta_{yy}\right)T^{n+1} = \left(\mathbf{I} + \frac{\alpha\Delta t}{2}\delta_{xx}\right)\left(\mathbf{I} + \frac{\alpha\Delta t}{2}\delta_{yy}\right)T^{n} + \Delta t S, \tag{2}$$

where $\mathbf{I}$ is an identity matrix.

③ The factored algorithm can be implemented in two steps:

$$\left(\mathbf{I} - \frac{\alpha\Delta t}{2}\delta_{xx}\right)T^{*} = \left(\mathbf{I} + \frac{\alpha\Delta t}{2}\delta_{xx}\right)\left(\mathbf{I} + \frac{\alpha\Delta t}{2}\delta_{yy}\right)T^{n} + \Delta t S, \tag{3}$$

$$\left(\mathbf{I} - \frac{\alpha\Delta t}{2}\delta_{yy}\right)T^{n+1} = T^{*}. \tag{4}$$

If $g_{i,j}^n = \left(I + \frac{\alpha\Delta t}{2}\delta_{yy}\right)T_{i,j}^n$, derive the discretized equations for $g_{i,j}^n$ for $j = 1$ and $j = 2 \sim J-1$. At $j = 1$, the second order central difference approximation of $\partial T/\partial y|_{x,0,t} = 0$ should be used.

④ If the right hand side of Eq. (3) is $R_{i,j}^n$, derive the discretized equations for $R_{i,j}^n$ for $i = 2 \sim I-1$.

⑤ Considering the boundary conditions for T^* and using $R_{i,j}^n$, derive the discretized equations for $T^*_{i,j}$ for $j = 1 \sim J-1$.

⑥ Express the above discretized equations in matrix notation

⑦ Derive the discretized equations for $T^{n+1}_{i,j}$ using $T^*_{i,j}$ for $i = 2 \sim I-1$.

⑧ Express the above discretized equations in matrix notation

⑨ Explain how to solve the 2-D heat equation using the approximate factorization method

b) (20 points) (Programming) Based on the above finite difference approximations, solve the equation numerically using the AFM. Make your own code and plot T at $t = 1.0$ using 'surf' function in MATLAB. Use the following parameters; $\alpha = 0.1$, $I = J = 21$ and $\Delta t = 0.05$.

2. (30 points) In this time, we want to solve the PDE of Problem 1 using a time integral method more accurate than the second-order methods.

$$\frac{\partial T}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) + S(x, y), \tag{5}$$

with the boundary and initial conditions

$$\text{B.C.: } T(0, y, t) = T(1, y, t) = 0,\ \partial T/\partial y|_{x,0,t} = 0,\ T(x, 1, t) = 0 \quad \text{for} \quad t > 0,$$

$$\text{I.C.: } T(x, y, 0) = 0 \quad \text{for} \quad 0 \le x \le 1 \quad \text{and} \quad 0 \le y \le 1.$$

$$S(x, y) = 1.0 \text{ for } 0 < x < 1 \text{ and } 0 < y < 1 \text{ for } t > 0.$$

a) (15 points) Solve the parabolic PDE using the Runge-Kutta method.

① We can consider the PDE as an initial value problem of a system of ODEs like $d\vec{T}/dt = \vec{F}(t, \vec{T})$. Using the 2nd-order central finite difference approximation for the spatial derivatives of the PDE at $x = x_i$ and $y = y_j$, derive the first-order ODEs for $T_{1,j}$, $T_{I,j}$, $T_{i,1}$, $T_{i,J}$ and $T_{i,j}$. For example, $dT_{i,j}/dt = F_{i,j}(t, T_{i,j}, T_{i+1,j}, T_{i-1,j}, T_{i,j+1}, T_{i,j-1}, S_{i,j})$. Use $\Delta x = \Delta y = h$ and the 2nd-order central difference approximation of $\partial T/\partial y|_{x,0,t} = 0$ for $j = 1$.

② By letting $f(1{:}I) = T(1{:}I, 1)$, $f(I + 1{:}2I) = T(1{:}I, 2)$, $\cdots$, $f\big((J-1)I + 1{:}IJ\big) = T(1{:}I, J)$, convert the above system of 1st order ODEs of $T_{i,j}$ into $f(k)$ corresponding to the PDEs. (Hint: $f\big((j-1)I + i\big) = T(i, j)$)

c) (15 points) (Programming) Using the runKut4 function, integrate the system of ODEs till $t = 1.0$ and plot T at the final time using 'surf'. For the integration, use the following parameters; $\alpha = 0.1$, $I = J = 21$ and $\Delta t = 0.005$.

3. (30 points) Consider 2-D Laplace equation,

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = S(x, y),$$

with the boundary conditions

$$T(0, y) = T(1, y) = 0 \quad \text{for} \quad 0 \le y \le 1.$$

$$\partial T/\partial y|_{x,0} = 0;\ T(x, 1) = 0 \quad \text{for} \quad 0 \le x \le 1,$$

$$S(x,y) = -10 \;\text{ for } 0 < x < 1 \text{ and } 0 < y < 1$$

Assume that $\Delta x = \Delta y = h$.

a) (5 points) Write the finite difference equations for the successive over-relaxation (SOR) iteration

b) (5 points) Write the finite difference equations for the successive line over-relaxation (SLOR) method

c) (5 points) (Programming) Start a guess of $T = 0$ at all points at which T is unknown and use the SOR iteration with $\omega = 1.8$. Error is defined as:

$$\text{err} \equiv \max\left(\text{abs}\left(\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{T_{i,j}^{n}}\right)\right), \quad 2 \le i \le I-1,\ \ 1 \le j \le J-1,$$

and the error tolerance is 1e-6. Use $I = J = 21$. Discuss the convergence rate based on the iteration number when the solution converges. Plot the solution using 'surf' function in MATLAB.

d) (5 points) (Programming) Repeat c) using the SLOR method.

e) (10 points) (Programming) Repeat c) using the Generalized Minimum Residual (GMRES) method (gmres function in MATLAB).

4. (30 points) Let us consider the natural cubic spline. Using Lagrange's two-point interpolation, we can write the expression for $f''_{i,i+1}(x)$ as follows:

$$f''_{i,i+1}(x) = \frac{k_i(x - x_{i+1}) - k_{i+1}(x - x_i)}{x_i - x_{i+1}}, \tag{6}$$

where k_i is the second derivative of the spline at knot i.

a) Derive $f_{i,i+1}(x)$ by integrating Eq. (6) twice with respect to x and imposing $f_{i,i+1}(x_i) = y_i$ and $f_{i,i+1}(x_{i+1}) = y_{i+1}$

b) Derive the equation for the curvatures by applying the slope continuity conditions, $f'_{i-1,i}(x_i) = f'_{i,i+1}(x_i)$, where $i = 2, 3, \cdots, n-1$.

c) Derive the quadrature formula if $y = f(x)$ is approximated by the natural cubic spline with evenly spaced knots at $x_1, x_2, \cdots, x_n$, or $x_{i+1} - x_i = h$ as shown in the figure. (Hint: use $f_{i,i+1}(x)$ derived in (a) and (b)).