Computational PDEs M345N10, 2019-2020, Project 1
The project mark, will be weighted to comprise 20% of the overall Module.
Blackboard upload deadline : 11.59pm, 21 February
Please name your files in following way:
Proj1_part_1of_X_yourCID.m (all your Matlab scripts, with X the total number of scripts etc.); Your accompanying technical report as: Proj1_yourCID.pdf. A zipped folder will also be acceptable, in this case call your zipped folder: Proj1_yourCID.zip (etc.)

Project 1: Diffusion Problem

The idealised heat diffusion equation is given by

$$u_t = u_{xx}, \qquad 0 < x < 1, \tag{1}$$

where t represents time and x a spatial coordinate. The equation satisfies the initial condition $u = 1$, when $t = 0$, $0 < x < 1$, and the boundary condition $u = 0$ at $x = 0$ and $x = 1$ for $t \geq 0$. The PDE has separable solutions of the form $u(x, t) = X(x)T(t)$.

Part A: Marks (25%)

1. Show that the analytic solution satisfying the initial and boundary conditions is given by

$$u(x,\, t) = \frac{4}{\pi} \sum_{n=0}^{\infty} \frac{1}{2n+1} \exp\left[-(2n+1)^2 \pi^2 t\right] \sin\left[(2n+1)\pi x\right]. \tag{2}$$

2. Define $h = \Delta x = 1/(N-1)$ the x-discretisation step size with N the number of intervals in the domain $0 \leq x \leq 1$. Define $\Delta t = k$ (say) an appropriate time interval. Discretise Eqn. 1 by a finite difference based explicit method with $O(h^2)$ and $O(k)$ accuracies; for your code it would be useful to work in the $r = k/h^2$ variable (from lectures).

3. March your numerical solution to $t = 0.075$, and evaluate the difference between your numerical and analytic (Eqn. 2) results at the two coordinates $x = 0.02$ and $x = 0.5$, through the following:

$$error = \log_{10} \mid u_{numerical} - u_{analytic} \mid . \tag{3}$$

   Investigate in a systematic way, how the error changes, on varying r, k and h – since $r = k/h^2$ you will have to fix k or h as you vary r. How do

your errors behave with increasing N and varying time step intervals k? Show in a reasonably concise and appropriate manner by way of plots a summary of your findings. Explain any difference in errors behaviour at $x = 0.5, x = 0.02$. Investigate and demonstrate by way of plots the numerical stability restriction on the maximum value of r.

Assume that your numerical results have converged and are grid independent if your error measure given by Eqn. 3 is below 0.000001*, or closest you can attain to this value with your code.*

Part B: Marks (45%)

1. Next discretise Eqn. 1, with fully implicit ($\Theta = 1$) and semi-explicit ($\Theta = 1/2$, Crank-Nicholson) discretisations. Use the tridiag.m matlab script provided as a template to develop your code – familiarise yourself with its usage through the tridiag_proj1_test.m script (setup to solve a N=5 tri-diagonal matrix problem).

2. Repeat your investigations on accuracy and numerical stability on the parameters r, k, h, $\Theta = (1, 1/2)$ (using the $x = 0.5, x = 0.02$ points as measures of accuracy). Hence contrast your findings between the fully explicit, semi-explicit and fully implicit treatments by appropriate plots in your report.

3. Compare the computation time taken for the explicit and semi-explicit methods to obtain the same overall accuracy at $x = 0.5$ (error measure ≤ 0.000001 (*you may need to use very large N values for some meaningful measure of computer performance!*).

4. Next, use the 3-point second-order accurate time-discretisation formula:
$$u_t \equiv \frac{3U_n^{j+1} - 4U_n^j + U_n^{j-1}}{2k} + O(k^2). \tag{4}$$
to modify your fully implicit code above (Part B-1) and investigate. (*the superscript j refers to the time level, and you may assume that solutions at previous time intervals j and $j-1$ have become available, and solutions at time interval $j+1$ are to be computed.*).

Part C: Marks (30%)

1. Using Lagrangian interpolation or otherwise, develop higher order finite-difference formulae (4^{th}-order accurate for the u_{xx} derivative on a uniform x-grid of step-size h, in the form:

Central difference at x_n:

$$u_{xx} \equiv a(U_{n+2} + U_{n-2}) + b(U_{n+1} + U_{n-1}) + cU_n + O(h^4). \qquad (5)$$

Forward sided difference at x_n:

$$u_{xx} \equiv dU_{n-1} + eU_n + fU_{n+1} + gU_{n+2} + jU_{n+3} + O(h^4). \qquad (6)$$

Backward sided difference at x_n:

$$u_{xx} \equiv lU_{n-3} + mU_{n-2} + nU_{n-1} + pU_n + qU_{n+1} + O(h^4). \qquad (7)$$

In your report show how the coefficients (i.e. $a, b, c, d, \ldots$ etc.) above are deduced.

2. Modify the explicit time-marching code developed in Part A for the same boundary conditions, with the above more accurate expressions, and thus investigate numerical stability and accuracy. Present your results concisely which highlight differences, benefits and or detriments of using the above derived expressions. (use the $x = 0.5, x = 0.02$ points as performance measures).

3. On a (x, u)-plot and a $(x, error)$-plot, compare the numerical solution for u at $t = 0.005$ for $0 \leq x \leq 1$ computed with your code in Part A with the more accurate $O(h^4)$ 5-point explicit scheme for a fixed $N = 21$, and compare your results with the analytic solution given in Eqn 2. Why do differences arise if any?

4. Modify your fully implicit code developed in Part B for the same boundary conditions, thus set up the matrix problem $A\hat{u} = \hat{b}$. Use Matlab functions (LU factorisation or Gaussian Elimination) to determine the solution vector $\hat{u}$. Hence investigate numerical efficiency, errors and computer timings, contrasting issues you have explored in earlier parts.

Notes:

1. Marking will consider both the correctness of your code as well as the soundness of your analysis *and* clarity and legibility of the technical report.

2. All figures created by your code should be well-made and properly labelled.

3. In order to assign partial credit, comment your matlab (or Python) scripts to indicate steps being undertaken or what is being attempted (SHORT COMMENTS!).

4. You are allowed to discuss general aspects of Matlab/Python with each other, however you are trusted not to discuss your code or analysis with other students.

Dr M. S. Mughal
5 February, 2020