Computational homework 3
Differential equations

Exercises 1 and 2(a)-(e) (marked *) to be handed in and assessed
Deadline: 2200 GMT Sunday 10th January.

Please submit your solutions using the link on the course Moodle page. You should submit a single pdf file, created using the Matlab `publish` command, formatted as per the template provided on the Moodle page.

EXERCISE 1* Consider the initial value problem

$$\begin{cases} y'(t) = 1 - y^2, \quad t \in (0, 20), \\ y(0) = 0, \end{cases}$$

the exact solution of which is $y(t) = \tanh t = (\mathrm{e}^{2t} - 1)/(\mathrm{e}^{2t} + 1)$.

(a) Compute numerical solutions to the initial value problem using the forward Euler method with equally spaced points $t_n = n\frac{20}{N}$, $n = 0, \ldots, N$, for $N = 19, 21, 40, 80$ and 160. Produce a plot comparing the numerical solutions with the exact solution.

For each choice of N compute the error $\max_{n=0,\ldots,N} |u_n - y(t_n)|$ and store these errors in an array $\mathbf{e}_{\mathrm{fe}}$. Also, compute and store the error $|u_N - y(20)|$ at the end of the interval.

(b) Same as a) but using Heun's method, to compute errors $\mathbf{e}_{\mathrm{heun}}$.

(c) Produce a loglog plot of $\mathbf{e}_{\mathrm{fe}}$ and $\mathbf{e}_{\mathrm{heun}}$ against the relevant N values and use your results to read off the approximate convergence orders for the two methods. How do these compare to the theory from lectures? *(Hint: follow the approach we used in computational homework 2)*

(d) The exact solution $y(t)$ has a horizontal asymptote $y(t) \to 1$ as $t \to \infty$. Does every approximation obtained using the methods above reproduce the same asymptotic behaviour? How well is the value of $y(20)$ approximated?

Guided by the theory from lectures, but considering only the maximum value of $|f_y|$ on the solution trajectory (rather than over all $(t, y) \in \mathbb{R}_+ \times \mathbb{R}$), we might expect the critical value of h below which the methods are stable to be $h = 1$. Is this what you observe in your numerical results?

How does the lack of stability for $N = 19$ ($h = 20/19 > 1$) manifest itself for the two different methods (forward Euler and Heun)?

EXERCISE 2 In this exercise we bring together some of the techniques we have studied in the course to solve an initial boundary value problem (IBVP) involving a parabolic partial differential equation, the heat equation.
The problem is: given $T > 0$ find $u : [0,1] \times [0,T] \mapsto \mathbb{R}$ such that

$$\frac{\partial u}{\partial t}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) = 1 \quad \text{in } [0,1] \times [0,T], \tag{1}$$

with $u(0,t) = u(1,t) = 0$ for all $t \in [0,T]$ and $u(x,0) = \sin(\pi x) + \frac{1}{2}x(1-x)$. This problem is a simple model for the evolution of the temperature distribution u in a metal bar undergoing uniform heating, starting from an initial temperature distribution $u(x,0)$, where the bar is held at constant (zero) temperature at its endpoints. For reference, the exact solution to this problem is given by

$$u(x) = e^{-\pi^2 t}\sin(\pi x) + \frac{1}{2}x(1-x).$$

To solve the IBVP numerically we will use the so-called *method of lines*. This involves discretizing the spatial differential operator $\partial^2 u/\partial x^2$ in the same way for all times t, so that the IBVP becomes a finite-dimensional system of ordinary differential equations in t, which can then be solved using a time-stepping scheme such as forward Euler, backward Euler or Crank-Nicolson. For the spatial discretization we shall use a standard finite difference approximation. Given $N > 0$, define equally spaced nodes $0 = x_0 < x_1 < \ldots < x_N = 1$ on the interval $[0,1]$ by $x_n = nh$, $n = 0, \ldots, N$, where $h = 1/N$. For a function $u(x,t)$ we denote the approximation of $u(x_n,t)$ by $u_n(t)$, and we replace the second derivative $\partial^2 u/\partial x^2(x,t)$ by the finite difference formula (cf. theoretical exercise sheet 1)

$$D^2 u_n(t) := \frac{u_{n+1}(t) - 2u_n(t) + u_{n-1}(t)}{h^2}.$$

This approximation turns the IBVP into an IVP involving a system of ordinary differential equations satisfied by the functions $u_n(t)$, $n = 1, \ldots, N-1$. Explicitly, we obtain the system

$$\frac{d\mathbf{u}}{dt}(t) = \mathbf{f}(t, \mathbf{u}(t)) = -A\mathbf{u} + \mathbf{b}, \quad t \in [0,T], \tag{2}$$

where $\mathbf{u} = (u_1, \ldots, u_{N-1})^T$, and, for a given $N > 0$, the matrix A and vector $\mathbf{b}$ can be obtained in Matlab using the commands

```
>> h = 1./N;
>> A= (2/h^2)*diag(ones(N-1,1)) - (1/h^2)*diag(ones(N-2,1),1)...
 - (1/h^2)*diag(ones(N-2,1),-1);
>> b = ones((N-1),1);
```

(a)* Consider the temporal discretization of the IVP system (2) using the forward Euler method, the backward Euler method and Crank-Nicolson method, on a time mesh $0 = t_0 < t_1 < \ldots < t_M = T$, where M is the number of subintervals and $t_m = mk$, $m = 0, \ldots, M$, where $k = T/M$ is the (uniform) temporal step size. *(Note that I am using k as the temporal step size and h as the spatial step size - don't confuse the two!)* For each method you have to compute a set of $M+1$ vectors $\mathbf{u}^m = (u_1^m, u_2^m, \ldots, u_{N-1}^m)^T$, $m = 0, \ldots, M$, where u_n^m is the numerical approximation to $u(x_n, t_m)$.

For each of the three methods write down the recurrence relation (in terms of the matrix A) that has to be solved at each timestep to determine $\mathbf{u}^{m+1}$ from $\mathbf{u}^m$.

(b)* Now write a Matlab code which implements the three methods.

Hint: For the two implicit methods you need to solve a linear system at each timestep. For simplicity I suggest you use Matlab's backslash command, which calls a general-purpose direct method. But you should be aware that in large-scale simulations one would use either a special direct solver like the Thomas algorithm, or an iterative solver.

Note: I am suggesting that you implement the recurrence relations directly in your code, rather than using the function files `feuler.m` *etc. that I supplied. If you want to use the latter, be warned that these codes only work for scalar problems, so you would need to modify them appropriately to work for systems.*

Run your code with the parameter choices $N = 40$ (giving a spatial step length $h = 1/40$), $k = h$, and $T = 0.2$. Produce three plots (one for each method) showing the initial solution and the solution at all of the time steps up to and including the final time $T = 0.2$ (on the same axes, using the `hold on` command). Produce a further plot showing snapshots of the exact solution at the same time steps.

What can be said qualitatively about the performance of the three methods, just by studying your plots?

Hint: For forward Euler I suggest using the `ylim` *command to adjust the vertical scale as you see fit. The "vertical zoom" feature of the Matlab plot window is also helpful.*

Warning: remember that your solution vectors $\mathbf{u}^m = (u_1^m, u_2^m, \dots, u_{N-1}^m)^T$, $m = 0, \dots, M$, *represent approximations to the exact solution* $u(x,t)$ *on the INTERIOR spatial grid points* $x_1, \dots, x_{N-1}$. *When plotting them, don't forget to add back on the boundary values* $u_0^m = 0$ *and* $u_N^m = 0$ *to each end of these vectors so you can plot on the whole spatial interval* $[0,1]$.

(c)* For each of the three methods, and for each $N \in \{10, 20, 40, 80, 160\}$, compute the root-mean-squared error of the numerical approximation at the final time $T = 0.2$ using the formula

$$E = \sqrt{h \sum_{n=1}^{N-1} (u_n^M - u(x_n, T))^2},$$

where, as above, u_n^m denotes the approximation at time step m and in space node n, and $u(x,t)$ is the exact solution stated at the start of the question. As in part (b), use a time step k equal to h.

Visualize your results on a `loglog` plot and, for the methods that converge, determine the approximate order of convergence p for which $E \approx Ch^p$ for some $C > 0$. *(Hint: follow the approach we used in computational homework 2)*

(Bonus unassessed theoretical question: why do we include the scaling factor h inside the square root in the definition of E?)

(d)* Now remove the assumption that $h = k$ and, by varying h and k appropriately, investigate the convergence order of the two implicit methods (backward Euler and Crank-Nicolson) in the joint limit $h \to 0$ and $k \to 0$. That is, for each method determine constants p and q for which the root-mean-squared error at $T = 0.2$ satisfies, for some $C > 0$, the bound

$$E \le C(h^p + k^q).$$

Hint: to determine p, fix k and vary h, with k very small compared to all your values of h, so that h^p *dominates the bound. For instance, you could try* $M = 6400$ *and* $N = 10, 20, 40$. *To determine q you would do the opposite: fix h very small compared to k (e.g.* $N = 640$ *and* $M = 8, 16, 32$*), so that* k^q *dominates the bound.*

(e)* Now return to the forward Euler method. In lectures we proved that the matrix A is symmetric positive definite, so that all its eigenvalues are real and positive. For $N = [10, 20, 40, 80, 160, 320]$ compute the eigenvalues of A using the `eig` command, and make a precise conjecture about the behaviour of the maximum and minimum eigenvalues $\lambda_{\min}$ and $\lambda_{\max}$ as the spatial step size h tends to zero.

Using your results, determine a condition on the temporal step size k of the form $k \leq Ch^p$ for some $C > 0$ and $p \geq 1$ (which you should specify), which ensures that the forward Euler method is stable. (You will need to consider the spectral radius of the matrix $I - kA$.)

Verify that your stability criterion is correct by redoing your computations in (c) for the forward Euler method, using your new stability criterion to choose k rather than setting $k = h$. What convergence order do you observe as $h \to 0$?

(f) (Bonus unassessed computational question!) Consider now the inital data

$$u_0(x) = \begin{cases} 0 & \text{when } 0 \leq x < \frac{1}{3}, \\ 1 & \text{when } \frac{1}{3} \leq x < \frac{2}{3}, \\ 0 & \text{when } \frac{2}{3} \leq x < 1, \end{cases}$$

which can be approximated in Matlab using (provided $N - 1$ is divisible by 3)

```
>> u0=[zeros((N-1)/3,1);ones((N-1)/3,1);zeros((N-1)/3,1)];
```

Let $N = 22$ and solve for one timestep, taking the timestep k equal to $1/N$, using the forward and backward Euler methods and the Crank-Nicolson method. Repeat using the timesteps $1/(10N)$ and $1/(50N)$. Plot the approximations obtained and comment on the results.

(g) (Bonus unassessed theoretical question!) At the start of the question I gave you an analytical solution of the IVBP. Can you prove that this solution is the only solution? That is, that the solution of (1) is unique? *Hint: Try using the following "energy argument". Suppose that the IBVP has two solutions u_1 and u_2. Show that the difference $v = u_1 - u_2$ satisfies the same IBVP, but with zero right-hand side and zero initial data. Using this fact, along with integration by parts, show that the energy $E(t) = (1/2)\int_0^1 (v(t,x))^2\,dx$ is a non-increasing function of t. Since $E(0) = 0$, deduce that $E(t) = 0$ for all t, and conclude that $v = 0$, i.e. $u_1 = u_2$. You may assume that u_1 and u_2 are both continuous functions of x.*