AEM 4253/5253, Fall 2022, Homework 2
Due on Tuesday, Nov. 1

Notes:

(1) Do not submit code with this HW. Upload a *single* PDF of your HW (either scanned from a legible handwritten version or a typed copy) to Canvas. All your plots must be computer-generated.
(2) Clearly label the axes on all your plots (and indicate the problem # in the caption).
(3) Use colors and/or symbols to differentiate different solutions on the plots.

1.

a) Using Taylor series expansions, derive a third-order upwind formula for the first derivative of f at point i using data from the stencil $\{i-2, i-1, i, i+1\}$. Clearly identify the leading term in the truncation error.

b) Consider the following schemes for the first derivative:

S1. First order backward difference: $f'_i \approx \dfrac{f_i - f_{i-1}}{\Delta x}$

S2. Second order central: $f'_i \approx \dfrac{f_{i+1} - f_{i-1}}{2\Delta x}$

S3. The third order upwind scheme you derived in part (a)

S4. Fourth order central: $f'_i \approx \dfrac{-f_{i+2} + 8f_{i+1} - 8f_{i-1} + f_{i-2}}{12\Delta x}$

For each of these schemes, derive the modified wavenumber k^*, and plot the real *and* imaginary parts of $k^*\Delta x$ vs. $k\Delta x$ in the range $0 \leq k\Delta x \leq \pi$.

c) Consider the case where we are solving the advection equation

$$\frac{\partial f}{\partial t} + a\frac{\partial f}{\partial x} = 0,$$

on an idealized periodic domain with uniform spacing. Recall that the choice of spatial discretization leads to a dispersive error: the speed of a Fourier mode $\exp(Ikx)$ is different from the correct speed a. For a specific problem, we want to ensure that all modes with wavenumbers less than a certain threshold ($k \leq k_m$) to travel with speeds that are within $p\%$ of the exact speed: this can be done if we have a suitably fine grid (i.e. by using small enough values of Δx) . How would you choose Δx to make this true for schemes S2 and S4 given above, and for 10% accuracy? (A different way to frame this is to ask how many points we need per wavelength to achieve the stated accuracy). NOTE: you can do this either graphically or analytically - make sure you clearly state *how* you would go about finding this out.

2. Consider solving the one-dimensional Poisson equation

$$\frac{d^2\phi}{dx^2} = f(x),$$

on a uniform grid with spacing Δx. We are solving for ϕ_i, given f_i, for all points i in the grid. A second order discretization of the left hand side (for the interior points) gives us the linear system

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{(\Delta x)^2} = f_i, \tag{1}$$

which can be efficiently solved using methods for tridiagonal matrices.

a) Write out the modified equation that the discrete form, eqn. (1), exactly solves. Retain terms upto at least $(\Delta x)^4$.

b) Note that the stencil is relatively compact (point i only sees data from $\{i-1, i, i+1\}$). By contrast, a fourth-order discretization of the left hand side leads to the system

$$\frac{-\phi_{i+2} + 16\phi_{i+1} - 30\phi_i + 16\phi_{i-1} - \phi_{i-2}}{12(\Delta x)^2} = f_i.$$

Due to the large stencil, there are issues with the boundary treatment: the compact stencil (eqn. (1)) is preferable because of this. Use the modified equation you derived in part (a) to devise a fourth-order method that retains the compact stencil $\{i-1, i, i+1\}$.

3. Develop a program that solves the 1D advection equation,

$$\frac{\partial f}{\partial t} + a\frac{\partial f}{\partial x} = 0, \qquad a \text{ is a constant,}$$

given an initial condition $f_0(x)$, on a periodic domain $0 \le x < L$ with N uniformly distributed points. Make your program flexible enough to accommodate switching between different spatial discretizations and different explicit time stepping methods (including *at least* explicit Euler and RK3). You should be able to change a, L, N, the time step Δt, and the initial condition easily (do not hardwire these). You may want to read through the note on setting this up (at the end of this document) before starting.

Let $L = 1$, $a = 1$, $N = 50$. The time for the initial condition to "wrap around" and come back to its starting point is $T_p = L/a = 1$. Set up the code to run up to a total time of $10T_p$ using a time step corresponding to a $CFL = 0.5$ ($\Delta t = CFL * \Delta x/a$ for this problem).

a) Using the third order Runge-Kutta method, solve using two different choices of initial condition:
I1. The (smooth) Gaussian function $f_0(x) = \exp(-100(x-0.5)^2)$
I2. The square pulse $f_0(x) = 1$ if $0.25 < x \le 0.75$, $f_0(x) = 0$ otherwise,
and using the four different spatial schemes S1, S2, S3 and S4 from problem 1. For each combination of initial condition and spatial scheme, make one plot showing the solutions at times $T = 0,\ 2T_p,\ 4T_p,\ 6T_p,\ 8T_p$ and $10T_p$.

b) Briefly comment (qualitatively) on the solutions based on what we have learnt in class.

c) **BONUS** HW1 (problem 5) writes the discrete advection equation in matrix form using the second order central scheme (S2 above). Modify this to plot the eigenvalues of the matrices you would get if you used the S1, S3 or S4 schemes. Briefly comment on what you observe.

Notes on setting up code for Problem 3

The solution we want, $f(x)$, is periodic on the domain $0 \le x \le L$. This implies that $f(x+L) = f(x)$ (and, in particular, $f(0) = f(L)$).

- Make the domain such that we do not have a redundant periodic point in the solution: choose $\Delta x = L/N$ and your interior grid point locations are given by $x_i = (i-1)\Delta x,\ i = 1, 2, ..., N$. This will define a domain $0 \le x \le L - \Delta x$.

- To account for the periodic BCs without special boundary treatment, it is helpful to include "ghost" or "dummy" cells in your solution. For this exercise two ghost cells on either end suffice. Essentially, instead of working with an array $f(1:N)$, extend this to $f(-1:N+2)$. If your programming language does not allow for negative indices, you will have to use arrays that range $f(1:N+4)$, where your interior solution is in the range $f(3:N+2)$.
- To simplify things, you will want to make your code modular.
  - Write functions (or subroutines) that evaluate the right hand side: for e.g., with the S2 scheme above, the right hand side is the array $R_i = -a\dfrac{f_{i+1} - f_{i-1}}{2\Delta x},\ i = 1, 2, ..., N$.
  - Write a function that sets the BCs (fills the ghost cells): this will make the code more readable to you.
  - You should be be able to switch between time stepping schemes. You can use repeated calls to the BC and RHS evaluation functions for various multi-stage or multi-step (RK or Adams schemes).