Introduction to Computational Fluid Dynamics: Coursework 1

Due: See Blackboard

Before the submission deadline, you should submit a pdf file containing all your answers to all the questions in order, together with Matlab `.m` files (or source code files) that you used to answer the questions. Marks will be added for legibility of the code and of the answers in the report.

Make your submission on Blackboard. Do **not** format your pdf file as a report, just answer the questions in order. Your pdf should be a self-contained document and it should not be necessary to look at the Matlab/source files to mark the coursework except for code legibiilty (the code is provided to give partial credit in case of errors). It is not necessary to provide an abstract/introduction/conclusion *etc.* The file "`matlab_hints`" in the Course Content/Coursework 1 folder on Blackboard may prove useful.

In this coursework we will use numerical methods to solve the advection equation

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0, \qquad a > 0, \tag{1}$$

in the domain $0 < x < 1$, with periodic boundary conditions $u(1) = u(0)$.

This coursework is about a class of numerical methods called *semi-Lagrangian* methods. They are often used for advection problems in order to take large timesteps. A semi-Lagrangian method computes the value of u at each grid point x_i at time level $n+1$ (*i.e.* $u_i^{n+1} = u(x_i, t^{n+1})$) by finding the *departure point*, that is the point x_i^* so that (x_i^*, t^n) is on the same characteristic line as (x_i, t^{n+1}).

1. Show that
$$x_i^* = x_i - a\Delta t, \qquad i = 1, \ldots, n. \tag{2}$$

2. Once we have identified our departure points (notice that for equation (2) these are the same for each time level since a is independent of time), we want to use the property that the solution of the equation is constant along characteristics so that
$$u_i^{n+1} = u(x_i^*, t^n).$$
However, we do not know the exact value of u at x_i^* because in general the departure points lie between the gridpoints. Hence, we must *interpolate* u to x_i^*. In this coursework we will only consider *linear interpolation*; more sophisticated methods are usually used in practise. To interpolate u to x_i^* using linear interpolation, we approximate u by finding a straight line between two values defined at gridpoints, and finding the position of the line at the departure point. The simplest method is to choose the stencil (x_{i-1}, x_i).

Show that the linear interpolated value of u using (x_{i-1}, x_i) is

$$u(x_i^*) \approx u_{i-1} + \frac{x_i^* - x_{i-1}}{x_i - x_{i-1}}(u_i - u_{i-1}).$$

3. Hence show that the numerical scheme is equivalent to the *upwind scheme*.

4. For what values of the departure point is the CFL condition satisfied?

5. When the CFL condition is satisfied, explain why the interpolated value of u at the departure point is always between u_i^n and u_{i-1}^n. Use this to show that

$$|u_i^{n+1}| \leq \max_j |u_j^n|$$

and hence that the upwind scheme is stable when the CFL condition is satisfied.

6. If the CFL condition is not satisfied, explain how it is possible to have a situation where

$$|u_i^{n+1}| > \max_j |u_j^n|,$$

and hence the method is not stable.

7. Explain why the upwind method has zero diffusion and dispersion error when the Courant number takes the value

$$\lambda = \frac{a\Delta t}{\Delta x} = 1.$$

(Hint: think about where the departure point is in this case.)

8. If we want a method which is stable for any value of the Courant number

$$\lambda = \frac{a\Delta t}{\Delta x},$$

we select the interpolation stencil so that the numerical domain of dependence contains the departure point. If $x_i^* \in [x_{k-1}, x_k]$ then we choose the stencil (x_{k-1}, x_k) to interpolate the interval. Show that this results in the *first-order semi-Lagrangian* method:

$$u_i^{n+1} = -(i-k)u_{k-1} + (i-k+1)u_k - \lambda(u_k - u_{k-1}), \tag{3}$$

where k is chosen so that

$$x_{k-1} \leq x_i - a\Delta t \leq x_k,$$

and where

$$x_i = i\Delta x.$$

9. Using the answers to the previous questions, explain why this method is always stable.

10. Discretise and implement (1) with the upwind first order semi Lagrangian scheme (3) outlining your implementation in the report part of the submission. In the next question we will modify the code so that the advecting velocity is a function of time, so make sure that the departure point is computed separately at each timestep. Take care with the periodic boundary conditions: when the departure point is greater than 1 or less than 0 you need to "wrap" it around so that it is mapped back into the domain.

    Using the initial condition

$$u = \begin{cases} 0 & x \leq 0.5 \\ 1 & x > 0.5 \end{cases}$$

    and $a = 0.5$, obtain numerical solutions at time $t = 1$ with 101 grid points (100 intervals), and $\lambda = 0.5, 1.0, 1.5, 5.0$. How do the solutions differ? How can you account for the differences? Which values of λ have the greatest diffusion error?

11. We will now modify equation (1) to have time-varying velocity:

$$u_t + a(t)u_x = 0, \quad a(t) = 10\sin(t).$$

    Solve the equation numerically at $T = 1$ using the first-order semi-Lagrangian method for $\Delta t = 0.1$ using the same initial condition, with 11, 21, 51, 101, 201, 251 and 501 gridpoints. Describe and explain the behaviour of the numerical solution.