Workshop 3: The advection equation

This workshop is about calculating numerical solutions to the 1D advection equation. We will look at the effect of changing the number of grid points (spatial resolution) and the size of the time step. We will also look at how to choose a suitable finite difference expression.

The workshop is designed to be carried out using the Linux PCs in the Lovelace laboratory. It is assumed that you are not on campus and will access these computers remotely. On the module ELE page there are instructions describing how to log in to a remote Linux computer under the "Workshop 2" section.

1 Getting set up

- Start by logging in to a Lovelace lab Linux PC, using the Linux computer availability dashboard (`http://students.emps.ex.ac.uk/dashboard/`) and choose a computer where the system load is low (you may need to use the VPN to view the dashboard).
- The example programs for this workshop are provided as a tar file. Take a copy of the tar file for this workshop by running the command:

  ```
  cp /secamfs/userspace/ug/shared/ecm3446/workshop3.tar .
  ```

  where the dot at the end of the command causes the file to be copied into the current working directory.
- Unpack the tar file by running the following command:

  ```
  tar xvf workshop3.tar
  ```

  This will create a directory called `workshop3` which contains the example programs for this workshop.

2 Solving the advection equation

The transport of a scalar field $u(x,t)$ by a constant velocity v is described by the one-dimensional advection equation

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x} \tag{1}$$

This equation can be solved numerically by using a finite difference expression to approximate the spatial derivative

$$\frac{\partial u}{\partial x} \approx \frac{u_i - u_{i-1}}{\Delta x} \tag{2}$$

where u is represented at discrete points u_i and the separation of the points is Δx. The numerical solution can be advanced by stepping forward from time t to time $t + \Delta t$ using a forward Euler time step

$$u(t + \Delta t) \approx u(t) + \frac{\partial u}{\partial t}\Delta t \tag{3}$$

In this way the solution can be evolved from an initial value at $t = 0$ to the desired end time. In this workshop you will be running a program which solves the one-dimensional advection equation using this method.

3 Running the example program

The example program calculates the advection of a Gaussian by a constant velocity $v = 0.01$. The Gaussian is initially centred at $x = 0.3$ and has a width $\sigma = 0.05$. The values of u are represented with a grid of 10 000 points over the interval $x = 0$ to $x = 1$. The time step is calculated from the CFL condition using the equation

$$\Delta t = C\frac{\Delta x}{v} \tag{4}$$

where the Courant number $C = 0.9$ and 5 000 time steps are calculated. Figure 1 shows the Gaussian at $t = 0$ (purple curve) and after 5000 time steps (green curve).

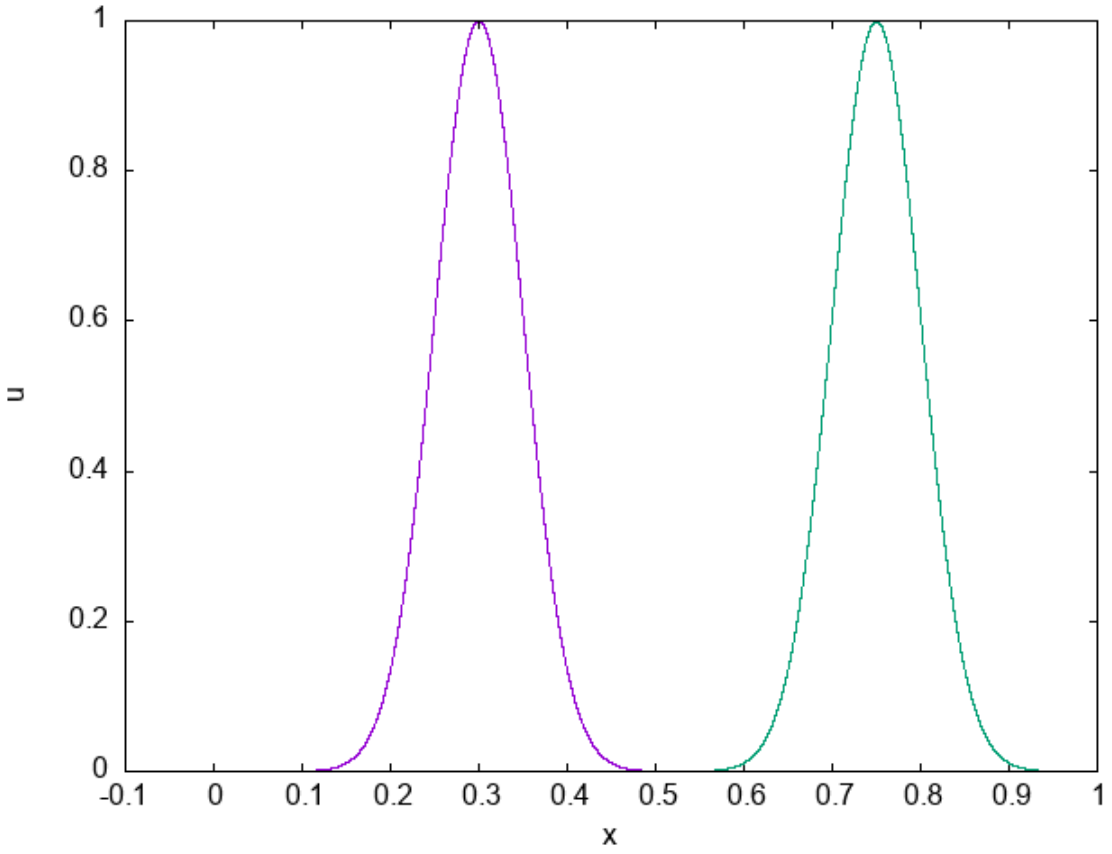

Figure 1: Advection of a Gaussian as calculated by the workshop 3 program. The initial conditions are plotted in purple and the final results are shown in green.

- Start by compiling and running the example program:

  ```
  cd advection_1d
  gcc -o advection1D -std=c99 advection1D.c -lm
  ./advection1D
  ```

  The program will print some information about the calculation to the terminal.

  Question 1: What is the end time of the calculation and how far has the Gaussian advected?

- The program will write two files called `initial.dat` and `final.dat` which contain the initial conditions and the values at the end of the calculation respectively. You can plot the values in these two files using the script provided:

```
gnuplot plot_advection
```

This will generate a plot in a file called `advection.png` which should match the plot in Figure 1. You can use the `scp` command (or `pscp` if using PuTTY) to copy the plot to your local computer for viewing (this is described in the Workshop 2 instructions).

4 Spatial resolution

In this section you will investigate the effect of changing the spatial resolution.

- Reduce the number of grid points by a factor of 100 from 10 000 to 100. Then recompile and re-run the program.

  Question 2: How does this affect the end time of the calculation and why?

- Change the number of time steps so that the end time is the same as your answer to question 1. Then recompile and re-run the program, and plot the new results.

  Question 3: How has the reducing the resolution affected the accuracy of the solution?

  Question 4: How has reducing the resolution changed the computational cost of calculating the solution? To answer this question think about how many calculations need to be performed.

5 The CFL condition

In this section we will look at what happens if the CFL condition is not respected.

- Start by changing the number of grid points back to 10 000 and the number of time steps back to 5 000, so that the program is in its original configuration.
- Try increasing the CFL number to 1.001 and see what effect this has on the solution.
- Then try a CFL number of 1.01 and see what effect this has on the solution.

Question 5: What happens if the CFL condition is not respected?

6 Choosing the finite difference stencil

In this last section we will look at how to solve the advection equation if the Gaussian moves in the opposite direction (from right to left)

- Change the CFL number back to 0.9 so that the program is in its original configuration
- Change the location of the centre of the Gaussian (`x0`) to 0.7
- Change the velocity (`velx`) to -0.01
- Now compile and run the program. This will not work correctly!

  Question 6: What are the values of u in the file `final.dat`?

- Try changing the stencil so that the finite difference is taken on the opposite side

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1} - u_i}{\Delta x} \tag{5}$$

- Check that this now produces the correct result for right to left advection

  Question 7: How would you modify the program to allow for advection in either direction (positive or negative velocities)?