Problem Set 1
Method of Lines

due date: May 12, 2023 NR_BHs

A Finite difference on a non-uniform grid

Consider a "logarithmic" mesh where the grid spacing increases with x, i.e.

$$x_i = x_{i-1} + h, \quad x_{i+1} = x_i + rh, \tag{1}$$

where $r > 1$ is a fixed ratio and h is the grid spacing between x_{i-1} and x_i. This kind of grid is commonly used in core-collapse simulation to reduce the computation cost by using a coarser grid outside the star.

(A.1)

Derive the finite difference approximation for $u'(x_i)$ using the three-point stencil $(u(x_{i-1}), u(x_i), u(x_{i+1}))$. Show that your approximation is second order accurate.

(A.2)

Derive the finite difference approximation for $u''(x_i)$ using the three-point stencil $(u(x_{i-1}), u(x_i), u(x_{i+1}))$. Show that your approximation is only first order accurate.

B The Advection Equation

Consider the 1D Advection equation with periodic boundary conditions;

$$\frac{\partial u}{\partial t} = c\frac{\partial u}{\partial x}, \quad 0 < x < 1, \quad u(0,t) = u(1,t), \tag{2}$$

where we set $c = 1$.

(B.1)

Show that the solution to [2] for arbitrary (compatible) initial data $u(x,0) = f(x)$ can be written as:

$$u(x,t) = f(x + ct). \tag{3}$$

(B.2)

We will proceed to solve this using a variety of discretization methods. For each of the following, use a uniform grid with N grid-points:

$$x_i = \frac{i}{N}, \quad i = 0, \ldots, N-1. \tag{4}$$

Additionally, let us use the initial data:

$$u(x,0) = e^{-2\cos(2\pi x)}. \tag{5}$$

Write a program that computes $u'(x_i)$ using a 2nd order centered finite difference approximation. Verify your code by performing a convergence test using Eq. (5).

(B.3)

Write a timestepper using the first-order forward Euler method:

$$u_i^{j+1} = u_i^j + \Delta t \left(\frac{\partial u}{\partial t}\right)_i^j, \tag{6}$$

for your choice of Δt (make sure it is at least smaller than the grid spacing! Otherwise you might end up violating the CFL condition, while will lead to unstable simulations.), and use it to evolve the initial data, Eq. (5) from t=0 to t=1. Use the discretization from (B.2) to compute $(\frac{\partial u}{\partial t})_i^j = c\frac{\partial u}{\partial x}(x_i, t_j)$.
You can measure the accuracy of your method by computing the root mean square error:

$$RMSE = \left(\frac{1}{N}\sum_{i=0}^{N-1}\left(u(x_i,1) - u_{\text{analytic}}(x_i,1)\right)^2\right)^{\frac{1}{2}}. \tag{7}$$

To understand the convergence of our method, first fix the grid spacing (N), and plot RMSE vs. step size (Δt). Now repeat, but fixing the step size and varying the grid spacing.

(B.4)

We can now try to improve our simulation via two approaches: improving our computation of the spatial derivative, or improving our time stepping. Pick one, and proceed as follows:
Time stepping: Given a discretization scheme, the advection equation now becomes

$$\frac{\partial \mathbf{u}}{\partial t} = F[t, \mathbf{u}] \tag{8}$$

where, the vector $\mathbf{u}$ contains u evaluated at each grid point. In our case, $F[t, \mathbf{u}]$ is simply $c \cdot \frac{\partial u}{\partial x}(x_i, t)$.
The obvious improvement in our time stepping method would be the Runge-Kutta 4 (RK4) method. Given an equation of the form of Eq. (8), one computes u^{k+1} from u^k via:

$$\begin{aligned}
\mathbf{w}_1 &= F\left(t, \mathbf{u}^k\right) \\
\mathbf{w}_2 &= F\left(t + 0.5\Delta t, \mathbf{u}^k + 0.5\Delta t \mathbf{w}_1\right) \\
\mathbf{w}_3 &= F\left(t + 0.5\Delta t, \mathbf{u}^k + 0.5\Delta t \mathbf{w}_2\right) \\
\mathbf{w}_4 &= F\left(t + \Delta t, \mathbf{u}^k + \Delta t \mathbf{w}_3\right) \\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \frac{\Delta t}{6}\left(\mathbf{w}_1 + 2\mathbf{w}_2 + 2\mathbf{w}_3 + \mathbf{w}_4\right)
\end{aligned} \tag{9}$$

Implement RK4, and remake the plots from (B.3) using RK4 instead of forward euler. Note that because of the rapid convergence of RK4 compared to our finite differencing scheme, it is unlikely you will be able see the convergence of RK4.

Spatial derivative: While one could try to implement a higher order stencil, an alternative to finite difference is to use a pseudo-spectral method. The central idea is to expand the solution in some basis functions. As our problem is periodic, a Fourier series is a natural choice:

$$u(x,t) = Re\sum_{k=0}^{N-1} c_k(t) e^{2\pi i k x}. \tag{10}$$

One can now analytically compute spatial derivatives! If we restrict our focus to even values of N, then the spectral coefficients for the spatial derivative of our function are:

$$c'_k = \begin{cases} 2\pi i c_k k & \text{if } k < N/2 \\ 2\pi i c_k (k-N) & \text{if } N/2 \leq k < N \end{cases} \tag{11}$$

One could then perform an inverse Fourier series, and recover $\frac{\partial u}{\partial x}$! All that's left now is computing the initial c_k. For this we use the technology of the fast Fourier transform:

$$c_k = \frac{1}{N}\sum_{j=0}^{N-1} u(x_j) e^{-2\pi i jk/N} \tag{12}$$

Using the above, replace our finite differencing with a pseudo spectral scheme that 1) computes c_k using [12], 2) computes c'_k using [11], and 3) computes $\frac{\partial u}{\partial x}$ using [10] (which, when discretised, is just the inverse fast Fourier transform!). Make plots depicting the RMSE when applied to [5] for different N. If you are feeling ambitious, try evolving the initial data using this scheme, and produce similar plots as in (B.3).

(B.5)

(optional) Repeat the above exercise, but with the other approach! Play around with different combinations of time stepping and spatial derivatives, and try to see what combinations of step size and grid spacing lead to unstable simulations for particular time stepping and spatial derivatives.

Please send full solutions (code and pdf) to peter.nee@aei.mpg.de

– End of Problem Set 1 –