Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG
FAKULTÄT
FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN
Prof. Dr. J. Behrens, C. v. Hallern, M. Avakumović
Summer Term 2024

Numerical Approximation of PDEs - Exercise 1

Exercise 1. (Epsilon)

In numerical analysis epsilon ε is defined as the smallest machine number for which the statement

$$1 + \varepsilon \neq 1$$

holds. One way to determine ε has already been shown in the lecture. The Python script-file `oldeps.py` can be found on our website at STiNE.

a) Save the script in your working directory and run it in your preferred Python environment; record the result.

b) Find another way to determine ϵ by using a `while`-loop and write your own MATLAB or Python script.

c) Compare the results of `oldeps` and your own script.

Exercise 2. (Derivation of Leap-Frog Scheme)

You saw in the lectures the derivation of a *Forward-in-Time-Centered-in-Space (FTCS)* advection scheme. Here we want to derive the *Leap-Frog* scheme, which is centered in time and space.
So, consider the transport equation

$$\frac{\partial \rho}{\partial t} + v \frac{\partial \rho}{\partial x} = 0$$

for a constituent $\rho : \mathbb{R} \times [0, T] \to \mathbb{R}$ with initial condition $\rho(x, 0) = \rho_0(x)$ and a given (constant) velocity v.

a) Use centered finite differences to discretize both partial derivatives in the above PDE.

b) Define grid functions as in the lectures to formulate the discrete equation by means of entries of the form ρ_i^j, where i is a spatial index, and j is a temporal index.

c) By writing all known values to the right hand side, and unknown values to the left, write down an explicit computational formula for solving the transport equation.

d) Discuss this scheme: What is the problem with it? How could one solve it?

Exercise 3.* (Implementation of Leap-Frog Scheme)

Implement the above derived leap-frog scheme. You may use the program for the FTCS scheme provided in the lectures as a template. There you will find an initial condition of the form

$$\rho_0(x) = e^{-\frac{(20x)^2}{2}}.$$

You will also find that the method has periodic boundary conditions implemented to mimic an indefinite spatial domain. Use all these techniques in your own implementation!

a) Implement a function `leapfrogadvect` that takes a grid vector `x`, a time step size `dt`, a final time `T`, and an initial condition `rho_0` on the grid `x`, and returns the final grid function `rho`, similar to the implemented function for the FTCS scheme.

b) Note that your scheme is still unstable, similar to the FTCS scheme. However, it can be stabilized by implementing a so-called *Asselin Filter*. To realize it, the update expression $\rho_\bullet^{j-1} \leftarrow \rho_\bullet^j$ is replaced by:

$$\rho_\bullet^{j-1} \leftarrow \rho_\bullet^j + a\left(\rho_\bullet^{j+1} - 2\rho_\bullet^j + \rho_\bullet^{j-1}\right).$$

Here $0 \leq a < 1$ is a filter parameter. Note that $\rho_\bullet^{j+1}$ should be known at this stage of the algorithm, i.e., this update takes place at the start of each time step, when old values are overwritten by those of the previous iteration. If $a = 0$, we obtain the original (unstable) leap-frog scheme. For $a \approx 1$ heavy damping can be expected.

Implement this simple stabilization for the leap-frog scheme and test it with e.g. $a = 0.05$.