NM2b
Hydrodynamics, python/MPI

Tomasz Korzec

June 13, 2022

Abstract

Project for a term paper in "Numerical Methods in Classical Field Theory and Quantum Mechanics". In this project the Diffusion equation is solved with a forward-time-center-space (FTCS) method.

1 Diffusion equation

The incompressible Navier Stokes equation is a diffusion-convection equation [1]. Approximating the time-derivative with a forward-difference leads to an "explicit method", where all fields at time $t+\tau$ can be computed from the fields at time t.

The goal of this project is to implement such an explicit method using python and MPI. To simplify things, we focus on the pure diffusion equation, instead of the full Navier Stokes equation, and we also consider the case with only one space and one time dimension. The partial differential equation we want to solve is

$$\frac{\partial v}{\partial t} = \frac{\partial}{\partial x}\left[D(x)\frac{\partial v}{\partial x}\right] + S(x,t)\,. \tag{1}$$

Here $D(x)$ is a diffusion coefficient, that can vary in space, but is constant in time, and $S(x,t)$ is an external source.

Using symmetric finite differences for the space derivative (discretization length $h/2$), and a forward finite difference for the time derivative (discretization step τ) leads to equations

$$\begin{aligned} v(x,t+\tau) &= v(x,t) \\ &\quad + \frac{\tau}{h^2} D(x+h/2)\left[v(x+h,t)-v(x,t)\right] \\ &\quad + \frac{\tau}{h^2} D(x-h/2)\left[v(x-h,t)-v(x,t)\right] \\ &\quad + \tau S(x,t)\,. \end{aligned} \tag{2}$$

2 Geometry

We work with a regular grid with spacing h in the space direction, so we have $v(0), v(h), v(2h), \ldots, v(L)$ as our field values. As boundary condition we use Neumann boundaries, that can be realized by setting $D(x) = 0$ on the boundaries, i.e. $D(-h/2) = D(L + h/2) = 0$. Apart from this, D is supplied by the user, as is the external source S and the initial condition $v(x, t = 0)$.

To parallelize the problem using MPI, distribute the field v among N_{proc} processes. Since the evolution of a field at x by one time-step requires the knowledge of the nearest neighbors $x \pm h$, and since sometimes these neighbors may live on a different process, communication is necessary before every time-evolution step.

3 Task

Write a program that solves the diffusion equation using the FTCS method.

- Use python and MPI. Start with a first version without parallelization (see LAB1[2], the corresponding assignment is attached).
- Parallelization with MPI: partition the fields onto N_{proc} tasks. Before computing $v(t + \tau)$, a communication of fields to neighbor processes will be necessary.
- To test your code, switch off the source $S(x, t) = 0$, set the diffusion parameter to a constant $D(x) = D$ and compare your numerical solution to the analytical one, that can be worked out in this case [2].
- In your term paper of about 10 pages
  - Give a short theoretical introduction to the problem
  - Demonstrate that your implementation is correct. Show that the solution indeed solves the Diffusion equation and plot $v(x, t)$.
  - Demonstrate that your parallelization is efficient. Show a strong scaling plot. Show how timings increase when you reduce h and τ.

References

[1] `https://moodle.uni-wuppertal.de/course/view.php?id=29658`

[2] `https://moodle.uni-wuppertal.de/course/view.php?id=27184`