Homework 5

Projection Method and Staggered Grid for Incompressible Flows

due February 21, 23:59

This homework demonstrates the advantages of a staggered grid compared to a co-located grid for the simulation of incompressible flows.

The techniques used for the solution of the incompressible Navier–Stokes equations are generally different from those used for the compressible equations. One efficient method which ensures that the velocity field is divergence free is the projection method, also known as operator splitting / fractional step method. A detailed description of this method applied to the incompressible Navier–Stokes equations is given in the course; in this homework we use a simple version for a model equation.

A simplified, one-dimensional analogue to the incompressible Navier–Stokes equations, neglecting the viscous and nonlinear convection terms, is given by:

$$\begin{aligned} u_t &= -p_x, && (1) \\ u_x &= 0. && (2) \end{aligned}$$

The first step of the projection method consists in the discretisation of the time derivative, done here using a first-order accurate backward Euler scheme, which results in the following semi-discrete form:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -(p_i^{n+1})_x \quad \Longrightarrow \quad u_i^{n+1} = u_i^n - \Delta t (p_i^{n+1})_x. \tag{3}$$

Taking the x-derivative of equation (3) (this step is the equivalent of taking the divergence of the momentum equation in the incompressible Navier–Stokes equations) yields:

$$(u_i^{n+1})_x = (u_i^n)_x - \Delta t (p_i^{n+1})_{xx}. \tag{4}$$

Requiring $u_x = 0$ (continuity equation) at $t = t^{n+1}$ and rearranging the terms results in a Poisson equation for the pressure:

$$(p_i^{n+1})_{xx} = \frac{1}{\Delta t}(u_i^n)_x. \tag{5}$$

Your task consists in the following:

1. discretise equations (5) and (3) using central differences second order in space for
   (a) a co-located grid,
   (b) a staggered grid (see figure 1);
2. write a program that solves the discretised equations derived above (for both grids).

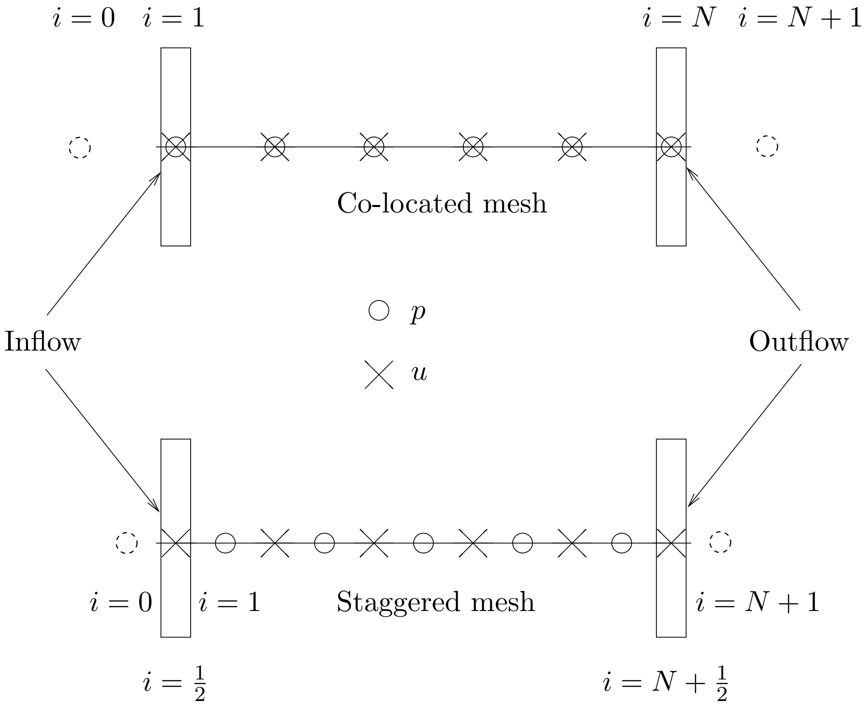

Figure 1: Schematics of a co-located and a staggered grid. Note the numbering of the nodes in both cases and the use of the so-called "ghost points" shown with dashed circles.

Solve the equations for $x \in [-1, 1]$ with the following initial and boundary conditions

$$u(t = 0, x) = 2, \qquad p(t, x = 1) = 5.$$

At each time step, p^{n+1} is computed first from equation (5) and then u^{n+1} is obtained from the equation (3).

The implementation of the boundary conditions should be done very carefully. Note that in these equations the velocity does not require explicit boundary conditions. The pressure, on the other hand, requires a Neumann boundary condition on the inflow border (implement a first order version: $p_1 = p_2$ for the co-located grid and $p_0 = p_1$ for the staggered grid) and a Dirichlet boundary condition on the outflow border. The pressure values in the "ghost points" p_0 and p_{N+1} can be approximated by linear extrapolation when necessary.

In the report (for both the co-located and the staggered grids) you need to:

- write the discretised form of equations (5) and (3);
- explicitly write all equations at the boundaries in the discretised form
- write the obtained discretised system in matrix form, namely

$$\begin{aligned} \underline{\underline{A}}\, \underline{p}^{n+1} &= \underline{\underline{M}}\, \underline{u}^n + \underline{b}, \\ \underline{u}^{n+1} &= \underline{u}^n - \underline{\underline{C}}\, \underline{p}^{n+1} + \underline{d}, \end{aligned}$$

  spelling out all vectors and matrices. Vectors $\underline{b}$ and $\underline{d}$ are needed for correctly including the boundary conditions.

- plot pressure and velocity fields at different time steps. Compare the results for both grids and explain the different behaviour of the solutions.

  The idea is to demonstrate the existence of spurious checker-board solutions. Something to keep in mind is that if the chosen initial solution is exact you may not observe these spurious solutions. Therefore, you should perturb your initial condition with low-amplitude noise ($\sim \mathcal{O}(0.01)$) in the *internal points* of the domain. The MATLAB function `rand` generates random numbers that can be used for that purpose.

- use different $\Delta t/\Delta x$. What effects do you observe? Integrate the equations for long enough such that you can see whether the spurious solutions grow or disappear completely.