Incompressible Navier-Stokes equations

February 29, 2024

Introduction

This project aims to learn how finite element approximation can be used to simulate the fluid equation. Our interest is the following incompressible Navier-Stokes equations:

$$\begin{aligned} \partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \nu \Delta \boldsymbol{u} + \nabla p = 0, \\ \nabla \cdot \boldsymbol{u} = 0, \end{aligned} \tag{1}$$

in $(\boldsymbol{x}, t) \in \Omega \times (0, T]$, $\Omega \subset \mathbb{R}^d$ with appropriate initial and boundary conditions. Here $\boldsymbol{u} = (u_1, \ldots, u_d)^T$ represents the unknown fluid velocity, p is unknown pressure, $\nu > 0$ - kinematic viscosity.

The pair of classical solution $(\boldsymbol{u}, p)$ to the problem (1) must be from the spaces $\mathcal{C}^1(0, T) \times \mathcal{C}^2(\Omega)$ and $\mathcal{C}^0(0, T) \times \mathcal{C}^1(\Omega)$ respectively. Typically, such a solution is difficult to calculate for any given initial data and boundary conditions. Instead, we try to find approximate solutions to (1) in finite element spaces.

Part A

1D simplification

The so-called viscous Burger's equation can be seen as one dimensional simplified model of the Navier-Stokes equations. Let $I = (a, b)$ be the computational domain. We are

looking for $u(x,t)$ from the following initial boundary value problem:

$$
\begin{aligned}
\partial_t u + \partial_x\Big(\frac{u^2}{2}\Big) - \partial_x(\varepsilon \partial_x u) &= 0, && (x,t) \in I \times (0,T], \\
u(a,t) &= g_a(t), && t \in (0,T], \\
u(b,t) &= g_b(t), && t \in (0,T], \\
u(x,0) &= u_0(x), && x \in I,
\end{aligned} \tag{2}
$$

where the values of u_0, a, b and g_a, g_b are yet to be defined. Since the system of Navier-Stokes equations is difficult to solve in general, the viscous Burger's equation is considered to be a good model to understand many interesting complex flow phenomenons.

Assume that the domain is split into N equal sub-intervals. Start by writing a weak and a continuous piecewise linear Galerkin finite element approximation of the Burger's equation. Once the equation is discretized in space, we obtain a system of N Ordinary Differential Equations of the form

$$
\mathbf{M}\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}t} = -\mathbf{A}\beta(\boldsymbol{U}) - \varepsilon \mathbf{S}\boldsymbol{U}, \tag{3}
$$

where $\boldsymbol{U}(t)$ is a vector consisting of the nodal values of the finite element solution $u_h(x,t) = \sum_{i=1}^{N-1} U_j(t)\varphi_j(x)$, where $\varphi_j(x)$ denotes finite element basis function, $\mathbf{M}, \mathbf{A}, \mathbf{S}$ are the mass, advection and diffusion matrices correspondingly:

$$
\mathbf{M}_{ij} := \int_I \varphi_j \varphi_i \,\mathrm{d}x, \quad \mathbf{A}_{ij} := \int_I \partial_x \varphi_j \varphi_i \,\mathrm{d}x, \quad \mathbf{S}_{ij} := \int_I \partial_x \varphi_j \partial_x \varphi_i \,\mathrm{d}x.
$$

Note, that we approximate the nonlinear flux with its finite element interpolant, i.e., $\beta(u) \approx \sum_{i=1}^{N-1} \beta(U_j)\varphi_j$, where $\beta(U_j) = \frac{1}{2}U_j^2$, $j = 1, \ldots, N-1$.

Approximate the resulting system using the standard 4th order explicit Runge-Kutta method in time. Note that at every Runge-Kutta stages you will have to solve a linear system involving the mass matrix.

Problem A.1. Implement a finite element solver using continuous piecewise linear approximations to solve the Burger's equation in (2). Start with initial condition

$$
u_0(x) = c - \tanh\left(\frac{x + \frac{1}{2}}{2\,\varepsilon}\right),
$$

where c is an arbitrary constant. An analytic solution to the Burger's equation using the above initial condition is given by,

$$
u_{exact}(x,t) = c - \tanh\left(\frac{x + \frac{1}{2} - c\,t}{2\,\varepsilon}\right), \tag{4}
$$

Set $a = -1$, $b = 1$, $g_a(t) = u_{exact}(a,t)$, $g_b(t) = u_{exact}(b,t)$, $c = 2$, $\varepsilon = 0.1$, and the final time $T = 0.4$. Investigate a convergence rate of the finite element approximation in a sequence of successively refined intervals: $N = 41, 81, 161, 321, 641$ and plot the error as a function of mesh-size $h = 1/(N-1)$. In one figure, plot your solution at the final time from all different meshes as well as the exact solution.

Problem A.2. Now, solve the Burger's equation using the following initial condition

$$u_0(x) = \sin(x),$$

over the interval $I = (0, 2\pi)$, (i.e., set $a = 0$, $b = 2\pi$). The boundary condition is $g_a(t) = g_b(t) = 0$. Run the problem until the final time $T = 2$ over the mesh consisting 201 points with $\varepsilon = 1, 0.1, 0.001, 0$. Plot your solution on one figure and discuss your result.

Problem A.3. Consider the same problem setting as in Part C.2. Now, set $\varepsilon = \frac{1}{2}h$ and run your program with a sequence of meshes $N = 41, 81, 161, 321, 641$. Plot your solution at the final time $T = 2$. What do you observe now?

Part B

2D Finite element approximation

Let $0 = t_0 < t_1 < ... < t_N = T$ be a sequence of discrete time steps with associated time intervals $I_n = (t_{n-1}, t_n]$ of length $k_n = t_n - t_{n-1}$, $n = 1, 2, \ldots, N$. Let

$$\begin{aligned} \boldsymbol{V}_h :=& \{\boldsymbol{v} : \boldsymbol{v} \in [H^1(\Omega)]^2, \boldsymbol{v}(\boldsymbol{x}) - \text{cont. pw. linear in } \Omega\}, \\ X_h :=& \{\boldsymbol{v} : \boldsymbol{v} \in H^1(\Omega), \boldsymbol{v}(\boldsymbol{x}) - \text{cont. pw. linear in } \Omega\}, \end{aligned}$$

be finite element spaces consisting of continuous piecewise linear polynomials on a mesh $\mathcal{T}_h = \{K\}$ of mesh size $h(\boldsymbol{x})$. Next, let us denote by $\boldsymbol{U}_0 = \hat{\pi}_h \boldsymbol{u}_0$ the interpolation of the initial data into the finite element space $\boldsymbol{V}_h$.

Problem B.1. Write down a weak formulation of (1) with appropriate spaces for velocity and pressure. Discretize the resulting ODE system using the Crank-Nicholson scheme in time. Note, that the nonlinear term can be treated explicitly, i.e., by taking the solution from the previous time step.

Once you derive a block system for the unknown $(\boldsymbol{u}, p)$, you will realize that the system is indefinite, i.e., has a zero block matrix on the diagonal. This is one of the fundamental problems of scientific computing. Many techniques resolve this issue, however, we will consider one of the simplest approaches.

Problem B.2. Let $\boldsymbol{U}_n$ is an approximate solution from time t_n. Assume that $\nabla \cdot \boldsymbol{U}_n \equiv 0$. Derive a Pressure-Poisson equation by taking a divergence of the momentum equation.

Part C

Implementation in 2D

The book has a section with all the necessary functions in Matlab that are needed for implementation in 2D. However, to solve the problems in this part, you can also use any open-source software, such as Dealii or the FEniCS project. You can start with existing Navier-Stokes solvers in these libraries and modify them to implement a Pressure-Poisson solver.

Flow in the L-shape domain

Consider the L-shaped domain, which is the subset of the unit square obtained by removing the upper right quadrant, see Figure 1.

The flow with vertical velocity $\boldsymbol{u} = (0, -3\sin(2x_1\pi))^T$ enters to the domain from the top boundary $\Gamma_{\text{inflow}} := \{x_2 = 1\}$, then exits from the right boundary $\Gamma_{\text{outflow}} := \{x_1 = 1\}$. The flow has zero velocity at the remaining boundaries $\Gamma_{\text{noslip}} = \partial\Omega \setminus (\Gamma_{\text{inflow}} \cup \Gamma_{\text{outflow}})$. The outflow boundary condition is enforced by setting the pressure to zero.

Problem C.1. Run the program until $T = 1$ with the following parameters: (a) $\nu = 1$; (b) $\nu = 0.1$; (c) $\nu = 0.01$; (d) $\nu = 0.001$. Plot the velocity for each case and discuss your solution. What happens when ν dicreases? You can use different mesh-resolutions to investigate the behaviour of the solution for different viscosity coefficients.

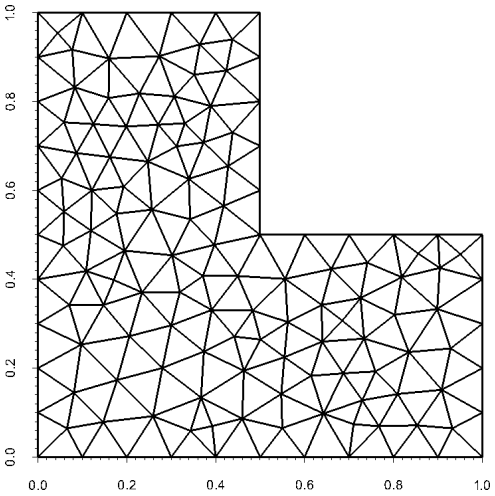

Figure 1: The L-shape domain.

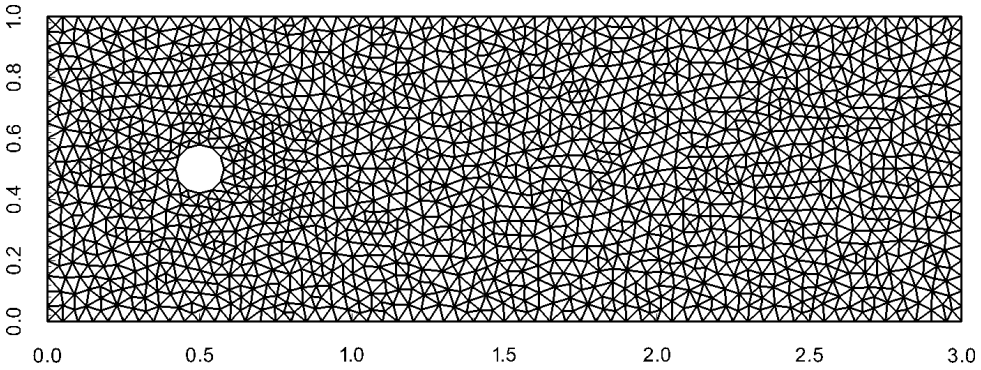

Figure 2: The wind-tunnel domain with a cylinder.

Flow around a circular cylinder in a wind tunnel

Consider incompressible flow in a wind tunnel with a circular cylinder. The size of the tunnel is 3×1 and the cylinder with radius 0.1 is located at $\boldsymbol{x} = (0.5, 0.5)^T$, see Figure 2. The flow with horizontal velocity $\boldsymbol{u} = (4\sin(x_2\pi), 0)^T$ enters to the domain from the left boundary $\Gamma_{\text{inflow}} := \{x_1 = 0\}$, then exits from the right boundary $\Gamma_{\text{outflow}} := \{x_1 = 3\}$. The flow has zero velocity at the remaining boundaries $\Gamma_{\text{noslip}} = \partial\Omega \setminus (\Gamma_{\text{inflow}} \cup \Gamma_{\text{outflow}})$. As for the L-shaped domain the outflow boundary condition is enforced by setting the pressure to zero.

Problem C.3. Modify your code according to the new geometry and initial data.

Run your program until time $T = 10$ on coarser meshes, with number of samples 60 and $\nu = 0.01$. Plot the velocity arrows in paraview, together with the velocity colormap.

In paraview, the velocity arrows can be found in `Filters/Alphabetical/Glyph`. Then press `Apply` in the left panel. At the same panel choose: `Properties/Scale Mode = vector`, `Properties/Scale Factor = 0.05` or any other numbers you may prefer.

Problem C.4. Now, run the program until $T = 10$ on finer meshes, with two different viscosity: (a) $\nu = 0.1$, (b) $\nu = 0.01$, and (b) $\nu = 0.001$. Plot the velocity, pressure and artificial viscosity for both cases. What happens with the solution?

Problem C.5. You can see the stability issues as $\nu \to 0$. (Why?). To fix this problem, as in the Burger equation, we replace ν with $\nu_K = \frac{1}{2}h_K$ for each $K \in \mathcal{T}_h$, where h_K is the diameter of the cell K . Run the program until $T = 10$ on the finer meshes with this new viscosity and discuss your solution.

Problem C.6. (*Optional*) One of the main interests of solving Navier-Stokes equation in a wind tunnel around a cylinder is to compute the so called *lift* and *drag* forces of the cylinder immersed in the fluid. The lift and drag are given by the y and x components of the force generated on the cylinder:

$$F_{\text{lift}} = \int_{\Gamma_{\text{cylinder}}} P\boldsymbol{n} \cdot \boldsymbol{e}_y \, \mathrm{d}s,$$

and

$$F_{\text{drag}} = \int_{\Gamma_{\text{cylinder}}} P\boldsymbol{n} \cdot \boldsymbol{e}_x \, \mathrm{d}s,$$

where $\boldsymbol{n}$ is the outward unit normal vector of Γ_{cylinder}, $\boldsymbol{e}_y$ and $\boldsymbol{e}_x$ are unit vectors in the y and x directions respectively. Compute the lift and drag forces of the cylinder and plot the with respect to time.

(*Hint:* Adapt the demo code on drag and lift forces in FEniCS into your problem.)

Good luck!
Murtazo
Uppsala, February 2024