Navier-Stokes Discrete Solution

Eilon Zohar

May 2024

1 Introduction

I saw an interesting job opening at Corintis - "Computational Scientist Intern". I currently live in Israel but I am ready to move to Lausanne and looking for my next challenge. I am the guy for this job, even though I didn't know what Navier-Stokes equation is. Of course I heard of it, but I have never properly learned about it. Today I learned what it is and built a python model to solve it numerically for 2 dimensions with simple boundary conditions. I hope the talent acquisition personnel in Corintis will appreciate this effort while considering my application.

In this file I will explain my code to solve Navier-Stokes equation numerically.

2 Navier-Stokes Equation

For an incompressible flow with $2D$ velocity $U = \begin{pmatrix} u \\ v \end{pmatrix}$,

$$\nabla U = \partial_x u + \partial_y v = 0, \tag{1}$$

Newton's 2nd law for a single liquid molecule in an infinite volume element dV is

$$\rho(\partial_t U + (U\nabla)U) = f - \nabla p + \mu\nabla^2 U, \tag{2}$$

where f is the sum of external forces, $-p$ is the pressure other molecules apply, and $\mu\nabla^2 U$ is the viscosity term. The L.H.S of this equation is simply ma/V, let us show for x:

$$\frac{ma_x}{V} = \frac{m}{V}a_x = \rho a = \rho\frac{du}{dt} = \rho(\partial_t u + \partial_x u\frac{dx}{dt} + \partial_y u\frac{dy}{dt}) = \rho(\partial_t u + u\partial_x u + v\partial_y u) = \rho(\partial_t u + (U\nabla)u) \tag{3}$$

and it is equivalent to show that

$$\frac{ma_y}{V} = \rho(\partial_t v + (U\nabla)v) \tag{4}$$

3 Discrete Solution

3.1 Prologue

Assume $U = \begin{pmatrix} u \\ v \end{pmatrix}$ is the $2D$ velocity of an incompressible liquid. Let us split the spacetime into 3 discrete grids with $\Delta x, \Delta y$ and Δt, so that $u^n_{i,j}$ is the x-component of the velocity assigned to time stamp n, x stamp i and y stamp j. In this case,

$$\partial_x A \approx \frac{A^n_{i+1,j} - A^n_{i-1,j}}{2\Delta x}, \partial_y A \approx \frac{A^n_{i,j+1} - A^n_{i,j-1}}{2\Delta y} \tag{5}$$

$$\partial^2_x A \approx \frac{A^n_{i+1,j} + A^n_{i-1,j} - 2A^n_{i,j}}{\Delta x^2}, \partial^2_y A \approx \frac{A^n_{i,j+1} + A^n_{i,j-1} - 2A^n_{i,j}}{\Delta y^2} \tag{6}$$

$$\partial_t A \approx \frac{A^{n+1}_{i,j} - A^n_{i,j}}{\Delta t} \tag{7}$$

for any function A.

3.2 Code

Python notebook