Final Project

Zeyuan Wang

April 12, 2024

1 Overview

This project aims to solve the Naiver-Stokes equation with Dirichlet boundary condition and incompressible condition in 2D. Two methods : Pressure Poisson Equation and Projection are studied and implemented with Finite Difference Method. The numeric results for both method are discussed.

2 Problem Statement

Consider the following non-forced Naiver-Stokes equation with incompressible condition:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p = \nu \nabla^2 \mathbf{u} \tag{1}$$

with

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

subject to

$$\mathbf{u} = \mathbf{g} \quad \text{on } \partial\Omega \tag{3}$$

$$\int_{\partial\Omega} \mathbf{n} \cdot \mathbf{u} \mathrm{d}A = 0 \tag{4}$$

where $\mathbf{u}$ is the velocity field in n dimension,p is the scalar pressure field, and ν is the kinematic viscosity. In fact, if both $\mathbf{u}$ and p is nondimensionalized, then ν could be replaced with $\frac{1}{Re}$, where Re is the Reynolds number of the flow.

While Eq.(3) is the Dirichlet boundary condition for velocity, Eq.(2) is called continuity equation, conservation of mass, or incompressible conditon, which is necessary to keep the flow field incompressible. Eq.(4) is a consistency condition of Eq.(3), by Divergence Theorem.

Another point that better to be clarified is in Eq.(1), it seems that we need to take the gradient of a vector field $\mathbf{u}$, which is an undefined operation. The answer is we actually need to treat different components of $\mathbf{u}$ separately. In other words, if we expand $\mathbf{u}$ as (u, v) in 2D, we actually get 3 partial differential equations:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} = \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{5}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} = \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{6}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{7}$$

with three unknown $u(x, y)$, $v(x, y)$ and $p(x, y)$.

At the first glance the system consisting of 3 unknowns and 3 equation is well-determined, which is ready to be solved directly. But there are two obstacles preventing us from solving the system naively. The first problem is the three unknowns are highly coupled. The second problem is although we have two evolution equations for u and v, respectively, the explicit evolution equation for pressure is lacking.

3 Pressure Poisson Equation

To solve the second problem, it's tempting to take the divergence of the momentum equation(1), resulting

$$\nabla \cdot [\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p] = \nabla \cdot [\nu \nabla^2 \mathbf{u}] \tag{8}$$

$$\nabla \cdot [\frac{\partial \mathbf{u}}{\partial t}] + \nabla \cdot [(\mathbf{u} \cdot \nabla)\mathbf{u}] + \nabla \cdot [\nabla p] = \nu \nabla \cdot [\nabla^2 \mathbf{u}] \tag{9}$$

Since $\mathbf{u}$ is smooth on time and $\nabla \cdot \mathbf{u}$ must be satisfied on any given time

$$\nabla \cdot [\frac{\partial \mathbf{u}}{\partial t}] = \lim_{\Delta t \to 0} \frac{\nabla \cdot [\mathbf{u}(t + \Delta t)] - \nabla \cdot [\mathbf{u}(t)]}{\Delta t} = 0 \tag{10}$$

. Similarly, if $\mathbf{u}$ is smooth in space

$$\begin{aligned}
\nabla \cdot [\nabla^2 \mathbf{u}] &= \frac{\partial}{\partial x}[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}] + \frac{\partial}{\partial y}[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}] \\
&= \frac{\partial^3 u}{\partial x^3} + \frac{\partial^3 u}{\partial x \partial y^2} + \frac{\partial^3 v}{\partial x^2 \partial y} + \frac{\partial^3 v}{\partial y^3} \\
&= \frac{\partial^2}{\partial x^2}[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}] + \frac{\partial^2}{\partial y^2}[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}] \\
&= 0 + 0 = 0
\end{aligned} \tag{11}$$

Therefore, we have

$$\begin{aligned}
\nabla^2 p &= -\nabla \cdot [(\mathbf{u} \cdot \nabla)\mathbf{u}] \\
&= -\frac{\partial}{\partial x}[u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}] - \frac{\partial}{\partial y}[u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}] \\
&= -(\frac{\partial u}{\partial x})^2 - 2\frac{\partial u}{\partial y}\frac{\partial v}{\partial x} - (\frac{\partial v}{\partial y})^2
\end{aligned} \tag{12}$$

And From $\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$ we know $\frac{\partial u}{\partial x} = -\frac{\partial v}{\partial y}$, $(\frac{\partial u}{\partial x})^2 = (\frac{\partial v}{\partial y})^2 = -\frac{\partial u}{\partial x}\frac{\partial v}{\partial y}$, so

$$\nabla^2 p = 2(\frac{\partial u}{\partial x}\frac{\partial v}{\partial y} - \frac{\partial v}{\partial x}\frac{\partial u}{\partial y}) \tag{13}$$

Eq.(13) is a Poisson equation for pressure, which allow the pressure to be updated as new u and v is obtained with time evolving.

Note the assumption we made here, is that the initial state of velocity field should already be divergence-free. In other words, the role of pressure plays in the system is to "keep the divergence of velocity as what it initially is", instead of "correct a field with divergence into non-divergence state".[1]

To tackle the problem that velocity and pressure coupled, several options can be adopted. For example, one can deliberately lag the pressure for one timestep. So when marching the velocity in time for n to $n+1$, only use the pressure from $n * dt$. When update the pressure, use the latest velocity u^{n+1}.

$$\begin{aligned} \mathbf{u}^{n+1} &= u^n + f(u^n, p^n)\Delta t \\ \nabla^2 p^{n+1} &= g(\mathbf{u}^{n+1}) \end{aligned} \tag{14}$$

However, as the experiment in this project shows that although this semi-implicit can recover pressure in second-order accuracy in time in L_2 norm, the velocity has only first-order accuracy, regardless that second-order scheme is used for convection and diffusion part.

Otherwise, if it's necessary to obtain the pressure in the same time-level as velocity, or one wants to use fully second order implicit scheme like Crank-Nicolson to improve the accuracy of velocity, quick fix is to use substeps, or iteration inside one timestep.

$$\begin{aligned} \mathbf{u}^* &= u^n + f(\mathbf{u}^n, \frac{1}{2}(p^n + p^*))\Delta t \\ \nabla^2 p^* &= g(\mathbf{u}^*) \end{aligned} \tag{15}$$

To be specific, at any given timestep t_n, one solve for u^{n+1} through Eq.(5) with initial v and p, then solve Eq.(6) with latest u and initial p. Finally, solve the pressure Poisson equation(13) with updated u and v. Then, iterate on Eq.(5) again, with the updated v and p. Repeat on this procedure till the solution converged. The numeric result found that even only two iterative substeps can greatly improve the velocity to second-order in time.

3.1 Boundary condition for PPE

Strictly speaking the Dirichlet boundary condition for velocity is all we need to make the system well-posed, both for velocity and pressure. But since we have to solve for Eq.(13) in respect to the pressure, it's still necessary to prescribe appropriate boundary condition on it.

Rewritten Eq.(5) and Eq.(6) in respect to normal($\mathbf{u}_n$) and tangential($\mathbf{u}_\tau$) components to $\mathbf{u}$(which makes no differences than u and v in our 2D grid case), and rearrange terms

$$\frac{\partial p}{\partial n} = -\frac{\partial \mathbf{u}_n}{\partial t} - \mathbf{u}_n \frac{\partial \mathbf{u}_n}{\partial n} - \mathbf{u}_\tau \frac{\partial \mathbf{u}_n}{\partial \tau} + \nu(\mathbf{n} \cdot \nabla^2 \mathbf{u}) \tag{16}$$

$$\frac{\partial p}{\partial \tau} = -\frac{\partial \mathbf{u}_\tau}{\partial t} - \mathbf{u}_n \frac{\partial \mathbf{u}_\tau}{\partial n} - \mathbf{u}_\tau \frac{\partial \mathbf{u}_\tau}{\partial \tau} + \nu(\tau \cdot \nabla^2 \mathbf{u}) \tag{17}$$

To be honest, these are the best thing we can do to constraint pressure on the boundary. While Eq.(14) clearly provides an Neumann BC, Eq.(15) (maybe not so obviously) actually provides an Dirichlet boundary for p since in theory we can line intergal it along the boundary to get p. Also, it's not an easy task to clarify whether we should use Eq.(16), or Eq.(17), or both, as the proper BC for PPE[1][2].

Further simplification is possible, if more conditions about $\mathbf{u}$ is provide on the boundary. Popular options include no-permeate($\mathbf{u}_n = 0$) or no-slip($\mathbf{u} = 0$)(To understand why the tangential component should also be zero in real physics setting is non-trivial, see [3]).

For instance if we prescribe $\mathbf{u} = 0$ at boundary, the Neumann BC is simplified to:

$$\frac{\partial p}{\partial n} = \nu(\frac{\partial^2 \mathbf{u}}{\partial \tau^2}) \tag{18}$$

Another problem not obviously showing in continuum analyse but do cause trouble in discreitization algorithm, is that it require unnecessary smooth up to the boundary: If we agree that the pressure extend to the boundary of velocity, then we don't only need $\mathbf{u}$ to be defined on the boundary, we also need the partial derivative of $\mathbf{u}$ to be defined on the boundary. One way to circumvent this dilemma is to use a so called staggered grid.[4] The staggered grid put velocity and pressure in different position(in a staggered pattern), and the boundary of pressure and velocity has $\frac{1}{2}h$ distance. No standard staggered grid is used in this project, the velocity is extrapolated beyond the boundary when necessary.

4 Projection Method

The idea behind projection method is to allow the velocity field to deviate from divergence-free at an intermediate state $\mathbf{u}^*$. The projection method try to remove the divergence part from $\mathbf{u}^*$ as effectively as possible, by directly project $\mathbf{u}^*$ back to a non-divergent space.

To be specific,

$$\frac{\mathbf{u}^* - \mathbf{u^n}}{\Delta t} = -[(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+1/2} + \frac{\nu}{2}\nabla^2[\mathbf{u}^* + \mathbf{u}^n] \tag{19}$$

The discretization scheme of convection term $[(\mathbf{u} \cdot \nabla)\mathbf{u}]$ is important but not relevant to the projection. The treatment for convection will be discussed in detailed in following section. For now just note that we totally ignore the pressure constraint for velocity evolution.

$$\mathbf{u}^* = \mathbf{u}^{n+1} - \Delta t \nabla p \tag{20}$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \tag{21}$$

Eq.(19)-(20) will be used to perform the projection step. Substitute Eq.(19) back to Eq.(20) to ensure we get the original NS discretization with pressure term. The fact that projection method divide the original NS equation to two steps to handle with separately gives it names like fractional step method or operator splitting scheme.

To perform the projection step, first, we make usage of Eq.(20) by taking the divergence of Eq.(19), which produces

$$\Delta t \nabla^2 p = \nabla \cdot \mathbf{u}^* \tag{22}$$

Again we get a Poisson equation for pressure to solve with. Then the correction on $\mathbf{u}^*$ is formed by

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \nabla p \tag{23}$$

Note the difference between PPE method and projection method: Once after the projection, $\mathbf{u}^{n+1}$ is guaranteed to be divergence-free immediately. Thus the projection method can be viewed as a more efficient way to solve for velocity field with incompressible constraints. But how can we make sure $\mathbf{u}^{n+1}$ also satisfies the boundary condition? The answer is unfortunately, pretty tricky :$\mathbf{u}^{n+1}$ cannot satisfies the incompressible constraints and boundary condition simultaneously in naive way. This inconsistency brings the projection method so called numerical boundary layer.

Completely leaving aside the pressure in Eq.(18) is the original idea from Chorin's projection[5], which only produces first-order accuracy in time, along with a $O(\Delta t)$ splitting error.[6]

There are numerous ways we can do to increase the accuracy. Just name two, we can integrate pressure gradient back to Eq.(18), in a clever way, or we can prescribe a better boundary condition when solving the intermediate $\mathbf{u}^*$.

For the first idea, note that when using second order midpoint scheme for velocity, we need to evaluate $p^{n+1/2}$,

$$\frac{\mathbf{u}^* - \mathbf{u^n}}{\Delta t} = -[(\mathbf{u} \cdot \nabla)\mathbf{u}]^{n+1/2} + \frac{\nu}{2}\nabla^2[\mathbf{u}^* + \mathbf{u}^n] - \nabla p^{n+\frac{1}{2}} \tag{24}$$

which is impossible to know in advanced. However, by also integrating the old pressure into

Eq.(24):

$$\frac{\mathbf{u}^* - \mathbf{u}^\mathbf{n}}{\Delta t} = -[(\mathbf{u}\cdot\nabla)\mathbf{u}]^{n+1/2} + \frac{\nu}{2}\nabla^2[\mathbf{u}^* + \mathbf{u}^n] - \nabla p^{n+\frac{1}{2}} - \nabla p^{n-\frac{1}{2}} + \nabla p^{n-\frac{1}{2}} \tag{25}$$

Using the spirit of operator splitting, we split Eq.(25) into:

$$\frac{\mathbf{u}^* - \mathbf{u}^\mathbf{n}}{\Delta t} = -[(\mathbf{u}\cdot\nabla)\mathbf{u}]^{n+1/2} + \frac{\nu}{2}\nabla^2[\mathbf{u}^* + \mathbf{u}^n] - \nabla p^{n-\frac{1}{2}} \tag{26}$$

$$\mathbf{u}^* = \mathbf{u}^{n+1} - \Delta t \nabla \phi^{n+1} \tag{27}$$

where $\nabla\phi^{n+1} = \nabla p^{n+\frac{1}{2}} - \nabla p^{n-\frac{1}{2}}$.

As shown in the numeric results, although this scheme improves the velocity into second order accuracy, the accuracy of pressure is still $O(\Delta t)$ at best.

4.1 Boundary condition for projection method

In the above section it should have been implied that the boundary condition is tricky and critical to the accuracy of projection method. Neither $\mathbf{u}^*$ nor pressure has a physical meaningful boundary condition to directly prescribe with. In fact, the choice of any boundary condition has significant effect on the final result.

For $\mathbf{u}^*$, generally we use the same boundary condition of u^{n+1}, assuming that $\mathbf{u}^*$ would not greatly differ from $\mathbf{u}^{n+1}$. However this assumption is only valid when pressure gradient is integrated as in Eq.(24). For the scheme without explicit pressure gradient, non-trivial boundary condition for $\mathbf{u}^*$must be found to ensure the accuracy, such as in [7]. As for the (pseudo)pressure, a homogeneous Neumann boundary condition is often specified

$$\frac{\partial \phi}{\partial n} = 0 \text{ on } \partial\Omega \tag{28}$$

Note that this boundary condition is not realistic, since it also implies that

$$\frac{\partial p^{n+1}}{\partial n} = \frac{\partial p^{n}}{\partial n} = \frac{\partial p^{n-1}}{\partial n} = \cdots = \frac{\partial p^{0}}{\partial n} = 0 \text{ on } \partial\Omega \tag{29}$$

which means the normal component of pressure remains constant on the boundary. It wouldn't be true in general case.

A so-called rotational form pressure correction[8][9] [6]is also proposed, which replace the pressure correction by

$$\nabla p^{n+\frac{1}{2}} = \nabla p^{n+\frac{1}{2}} - \nabla\phi^{n+1} + \frac{\nu\Delta t}{2}\nabla\nabla^2\phi^{n+1} \tag{30}$$

$\Delta t\nabla^2\phi^{n+1}$ can be replaced by $\nabla \cdot \mathbf{u}^*$ from Eq.(27). To see why it works, let's replace the newer pseudo-pressure formula into Eq.(27) and add it to Eq.(26)

$$\frac{\mathbf{u}^{n+1} - \mathbf{u^n}}{\Delta t} = -[(\mathbf{u}\cdot\nabla)\mathbf{u}]^{n+1/2} + \frac{\nu}{2}\nabla^2\mathbf{u}^n + \frac{\nu}{2}[\nabla^2\mathbf{u}^* - \nabla(\nabla\cdot\mathbf{u}^*)] + \nabla p^{n+\frac{1}{2}} \tag{31}$$

Be really careful, $\nabla(\nabla\cdot\mathbf{u})$ is not $\nabla\cdot(\nabla\mathbf{u})$! In fact, it relates to the Laplacian as $\nabla^2\mathbf{u} = \nabla(\nabla\cdot\mathbf{u}) - \nabla\times\nabla\times\mathbf{u}$. Also from Eq.(26) we know $\nabla\times\nabla\times\mathbf{u}^{n+1} = \nabla\times\nabla\times\mathbf{u}^*$. By also putting $\mathbf{n}\cdot u^{n+1} = 0$ on boundary, we finally get

$$(\nabla p^{n+\frac{1}{2}})\cdot\mathbf{n} = -\frac{\nu}{2}\nabla\times\nabla\times\mathbf{u}^{n+1}\cdot\mathbf{n} \text{ on } \partial\Omega \tag{32}$$

which, is a consistent pressure boundary condition. Since $\nabla\times\nabla\times$ plays a critical role, this algorithm is refereed as "rotational form".

As reported in [9] [6]although this correction significantly improve the accuracy for pressure, $O(\Delta t^2)$ accuracy is only achievable for periodic boundary condition. In more general boundary setting, the accuracy of pressure can only be $O(\Delta t^{\frac{3}{2}})$ at best. However, this conclusion is not quite consistence to the numeric result in this project.

5 Other methods

5.1 Saddle Point Form

Combine the velocity and pressure into a state vector$[u, v, p]^T$, a symbolic linear system can be directly formed:

$$\begin{bmatrix} -\nabla^2 - v\frac{\partial}{\partial x} & -\frac{\partial u}{\partial y} & -\frac{\partial}{\partial x} \\ -\frac{\partial v}{\partial y} & -\nabla^2 - u\frac{\partial}{\partial y} & -\frac{\partial}{\partial y} \\ -\frac{\partial}{\partial x} & -\frac{\partial}{\partial y} & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ p \end{bmatrix} = \begin{bmatrix} f_u \\ f_v \\ 0 \end{bmatrix} \tag{33}$$

where f encodes the external force. This form is called the saddle point form since the nullspace of system

$$\begin{bmatrix} \mathbf{A} & k^T \\ k & 0 \end{bmatrix} \tag{34}$$

optimize $\frac{1}{2}x^T\mathbf{A}x + \lambda(k^Tx)^2$. Here we can find that pressure exactly plays the role of λ, which also known as the Lagrangian multiplier. More analysis and numerical solution for saddle point system could be found in [10]. The concept Lagrangian multiplier is especially important since it will be utilized again when dealing with Poisson equation with pure Neumann Boundary condition.

5.2 Streamfunction-Vorticity Form

Another idea is to somehow eliminate the pressure term from the system. The intuition behind is the curl of gradient is 0. So if we apply the curl on both handsides, then ∇p can be removed from Eq.(1)

$$\nabla \times [\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla p] = \nabla \times [\nu \nabla^2 \mathbf{u}] \tag{35}$$

$$\nabla \times [\frac{\partial \mathbf{u}}{\partial t}] + \nabla \times [(\mathbf{u} \cdot \nabla)\mathbf{u}] = \nu \nabla \times [\nabla^2 \mathbf{u}] \tag{36}$$

$$\frac{\partial (\nabla \times \mathbf{u})}{\partial t} + (\mathbf{u} \cdot \nabla)[\nabla \times \mathbf{u}] = \nu \nabla^2 [\nabla \times \mathbf{u}] \tag{37}$$

Let's defined $\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$ as the vorticity, then Eq.(30) can be rewritten as

$$\frac{\partial \omega}{\partial t} + (\mathbf{u} \cdot \nabla)\omega = \nu \nabla^2 \omega \tag{38}$$

Also we define a streamfunction ψ, which satisfies $u = \frac{\partial \psi}{\partial x}$, $v = -\frac{\partial \psi}{\partial y}$, then we have

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} = \frac{\partial}{\partial x}(\frac{\partial \psi}{\partial x}) - \frac{\partial}{\partial y}(-\frac{\partial \psi}{\partial y}) = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \tag{39}$$

$$w = \nabla^2 \psi \tag{40}$$

The formulation above is called streamfunction-vorticity formulation[11], which is one of the non-primitive variables formulations of Naiver-Stokes. This formulation through has two main issues: First, it's only practical in 2D, since in 3D and above the vorticity vector has to be encoded with N components. Second, the boundary condition of ψ and ω is non-trivial to find.

6 Nonlinear Convection

Now let's forget the incompressible constraint for a moment and just focus on the convection-diffuse part of NS equation. To get both second-order accuracy in time and space, one can choose implicit scheme like Crank-Nicolson.

$$\frac{u^{n+1}-u^n}{\Delta t}=\nu\nabla_h^2 u^{n+1/2}-\frac{\partial p^{n+1/2}}{\partial x}-u^{n+1/2}\frac{\partial u^{n+1/2}}{\partial x}-v^{n+1/2}\frac{\partial u^{n+1/2}}{\partial y} \tag{41}$$

$$\frac{v^{n+1}-v^n}{\Delta t}=\nu\nabla_h^2 v^{n+1/2}-\frac{\partial p^{n+1/2}}{\partial y}-u^{n+1/2}\frac{\partial v^{n+1/2}}{\partial x}-v^{n+1/2}\frac{\partial v^{n+1/2}}{\partial y} \tag{42}$$

where $c^{n+1/2}$ is approximated by $\frac{1}{2}(c^n+c^{n+1})$ and ∇_h is second-order finite central difference.

There are several choices, include: 1. Accept that the equation is nonlinear, and use iterative method to solve it. 2. Use an explicit second-order linear multi-steps method for time, like AB2

$$\frac{u^{n+1}-u^n}{\Delta t}=\nu\nabla_h^2 u^{n+1/2}-\frac{\partial p^{n+1/2}}{\partial x}-[\frac{3}{2}(u^n\frac{\partial u^n}{\partial x}+v^n\frac{\partial u^n}{\partial y})-\frac{1}{2}(u^{n-1}\frac{\partial u^{n-1}}{\partial x}+v^{n-1}\frac{\partial u^{n-1}}{\partial y})] \tag{43}$$

$$\frac{v^{n+1}-v^n}{\Delta t}=\nu\nabla_h^2 u^{n+1/2}-\frac{\partial p^{n+1/2}}{\partial y}-[\frac{3}{2}(u^n\frac{\partial v^n}{\partial x}+v^n\frac{\partial v^n}{\partial y})-\frac{1}{2}(u^{n-1}\frac{\partial v^{n-1}}{\partial x}+v^{n-1}\frac{\partial v^{n-1}}{\partial y})] \tag{44}$$

The problem is we need somehow "start-up" the procedure for AB2, means we need both u^0 and u^1 to calculate u^2. In practice u^1 should be obtained by using other explicit first-order method, or multi-step method, but in this project u^1 is grabbed directly from the manufactured solution for the sake of simplicity.

Thanks to the existence of the diffusion term, pure explicit scheme doesn't suffer from the unconditionally instability which occurs in convection only case with forward time-centered space scheme. Interestingly this can be related to the Modified Equation Analysis on other classical conditional stable explicit scheme like upwind or Lax-Friedrichs. For example, the modified equation for Lax-Friedrichs is[12]

$$\frac{\partial u}{\partial t}+a\frac{\partial u}{\partial x}=(\frac{1}{\gamma}-\gamma)\frac{a\Delta x}{2}\frac{\partial^2 u}{\partial x^2}. \tag{45}$$

The presence of second-order space derivative term explains the improvement in the stability. An intuitive interpretation is while the diffusion term scattering the information to its neighbor, it somehow compensate the problem of lacking information from current position caused by central difference.

7 Numerical Implementation

Both Pressure Poisson Method and Projection Method are implemented by using Finite Difference Method.

7.1 Pressure Poisson Equation

Each substep in this method takes the following framework:

1. Update velocity

$$\frac{\mathbf{u}^{n+1,*}-\mathbf{u}^n}{\Delta t} = -[\frac{3}{2}(\mathbf{u}^n\cdot\nabla_h)\mathbf{u}^n - \frac{1}{2}(\mathbf{u}^{n-1}\cdot\nabla_h)\mathbf{u}^{n-1}] + \frac{\nu}{2}\nabla_h^2(\mathbf{u}^{n+1,*}+\mathbf{u}^n) - \frac{1}{2}\nabla_h(p^{n+1,*}+p^n) \tag{46}$$

$$\mathbf{u}^{n+1,*} = \mathbf{g}^{n+1} \text{ on } \partial\Omega(\mathbf{u}) \tag{47}$$

2. Update pressure

$$\nabla_h^2 p^{n+1,*} = \nabla^2 p = 2(\frac{\partial u^{n+1,*}}{\partial x}\frac{\partial v^{n+1,*}}{\partial y} - \frac{\partial v^{n+1,*}}{\partial x}\frac{\partial u^{n+1,*}}{\partial y}) \tag{48}$$

$$\frac{\partial p^{n+1,*}}{\partial n} = \nu(\frac{\partial^2 \mathbf{u}^{n+1,*}}{\partial \tau^2}) \text{ on } \partial\Omega(p) \tag{49}$$

Note that if only one substep is taken, it's equivalent to the idea of pressure lagging. If two more substeps are taken in each timestep, then this scheme somehow behaves in a

Predictor-corrector way.

While solving velocity with Dirichlet boundary condition is straightforward, it worth discussing how to solve the Poisson equation with pure Neumann boundary condition.

The first problem is how to construct a symmetric matrix which represents the partial derivative in second order accuracy. The ghost point method[13] is adopted, which works as follow:

Consider a boundary point $p_{0,j}$, the second order central difference is

$$\frac{-4p_{0,j} + p_{-1,j} + p_{1,j} + p_{0,j-1} + p_{0,j+1}}{h^2} = f_{0,j} \tag{50}$$

Clearly the point $p_{-1,j}$ is out of boundary. But let's assume it exists, as a "ghost point". Impose the Neumman boundary condition we get

$$\frac{p_{1,j} - p_{-1,j}}{2h} = g_{0,j} \tag{51}$$

Thus we get a formula for $p_{-1,j}$ which can be substituted back to Eq.(44) to eliminate the ghost point

$$\frac{-4p_{0,j} + 2p_{1,j} + p_{0,j-1} + 2p_{0,j+1}}{h^2} = f_{0,j} + \frac{2}{h}g_{0,j} \tag{52}$$

Note that we still need to divide it by two to make the matrix symmetric. It also works on corner, where we have

$$\frac{-4p_{0,0} + 2p_{1,0} + 2p_{0,1}}{h^2} = f_{0,0} + \frac{4}{h}g_{0,0} \tag{53}$$

Don't forget to divide it by 4 to keep the symmetry. A more severe problem is that Poisson equation with pure Neumman boundary condition is not well-posed, so the resulting matrix is symmetric but semi-definite. To overcome this problem, many strategies can be used, for example:

1. Adding a diagonal matrix $\epsilon\mathbf{I}$ to the matrix. This equivalent to solve a modified problem $\mathbf{A}u + \epsilon u = f$ with small ϵ.

2. Pose another constraint on u, such as, the zero mean constraint, where $\int_\Omega u = 0$ on the domain. In discrete setting this is saying $e \cdot u = 0$, where $e = [1, 1, \dots, 1]$. So a new system can be formed by using Lagrangian multiplier

$$\begin{bmatrix} \mathbf{A} & e^T \\ e & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix} \tag{54}$$

This project makes the usage of Lagrangian multiplier. The padded matrix is indefinite, but the condition number is moderate, and Conjugate Gradient does converge. The only problem is Incomplete Cholesky(requires positive definite) cannot be used, instead Incomplete LU is chosen as the preconditioner.

7.2 Second-order Projection Method with Pressure gradient

Each timestep in second-order projection method takes the following framework:

1. Update velocity

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -[\frac{3}{2}(\mathbf{u}^n \cdot \nabla_h)\mathbf{u}^n - \frac{1}{2}(\mathbf{u}^{n-1} \cdot \nabla_h)\mathbf{u}^{n-1}] + \frac{\nu}{2}\nabla_h^2(\mathbf{u}^{n+1,*} + \mathbf{u}^n) - \nabla_h p^{n-\frac{1}{2}} \tag{55}$$

$$\mathbf{u}^* = \mathbf{g}^{n+1} \text{ on } \partial\Omega(\mathbf{u}) \tag{56}$$

2. Velocity correction

$$\nabla_h^2 \phi^{n+1} = \frac{1}{\Delta t}\nabla_h \cdot \mathbf{u}^* \tag{57}$$

$$\frac{\partial \phi^{n+1}}{\partial n} = 0 \text{ on } \partial\Omega(p) \tag{58}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t \nabla_h \phi^{n+1} \tag{59}$$

3. Update pressure

$$p^{n+\frac{1}{2}} = p^{n-\frac{1}{2}} + \phi^{n+1} - \frac{\nu \Delta t}{2} \nabla_h^2 \phi^{n+1} \tag{60}$$

Same as above, the Lagrangian multiplier is also used to make the Poisson equation well-posed. Unlike in PPE implementation, non collated grid is used to circumvent the problem when computing the divergence of velocity on the boundary point. Pressure is only evaluated at the center of cell, and the value of both velocity and pressure are interpolated when required. However, linear interpolation is used, which, potentially pollutes the accuracy on space for pressure.

8 Numerical Results

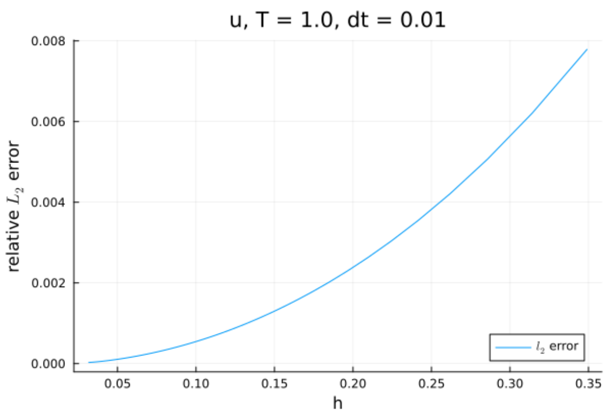
Both methods are tested on a simple square domain $[0, \pi] \times [0, \pi]$. A manufactured solution is grabbed from [7]:

$$\begin{aligned} u(x, y, t) &= -\sin(x)\cos(y)e^{-2t} \\ v(x, y, t) &= \cos(x)\sin(y)e^{-2t} \\ p(x, y, t) &= \frac{1}{4}(\cos(2x) + \cos(2y))e^{-4t} \end{aligned} \tag{61}$$

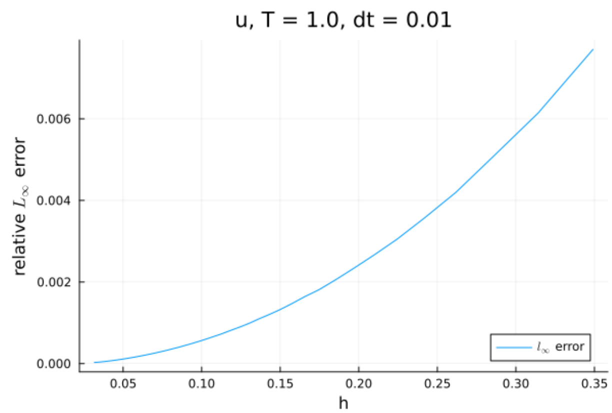
All the errors are measured at $T = 1.0$, when studying the convergence rate for time, h is set to $\frac{\pi}{200}$, when studying the convergence rate for space, t is set to 0.01. The behavior of v is always consistent with u so only the plot of u will be included to keep concise.

8.1 Pressure Poisson Equation

Firstly the convergence rate to space is studied for $h \in [\frac{\pi}{10}, \frac{\pi}{100}]$.

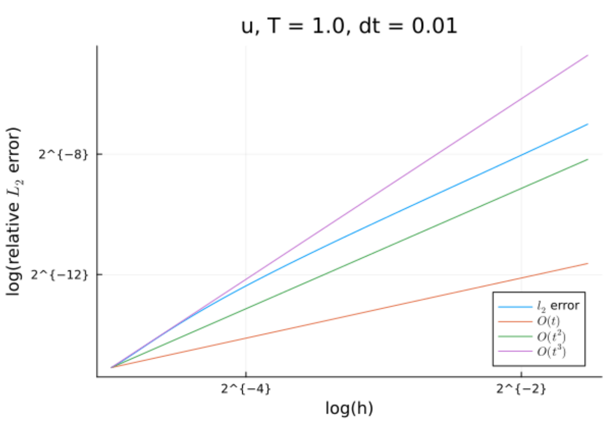


(a) relative L_2 error

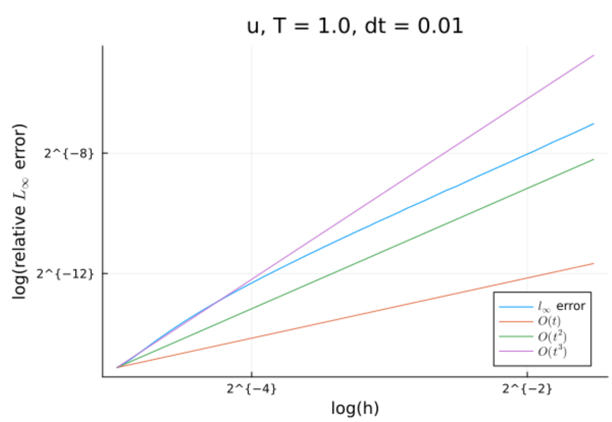


(b) relative L_∞ error

Figure 1: Pressure Poisson Equation method

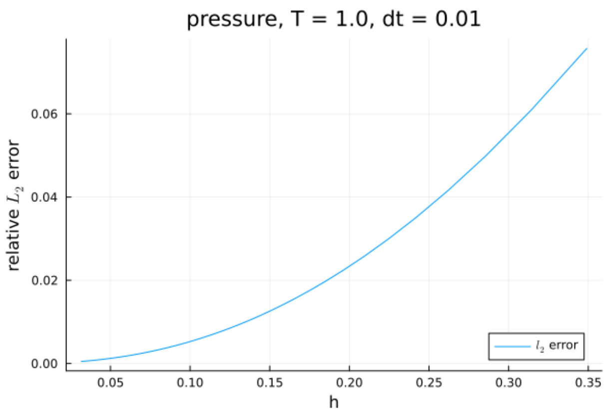


(a) relative L_2 error log-log

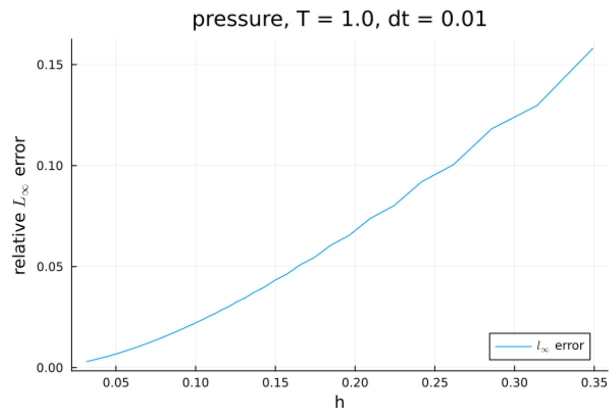


(b) relative L_∞ error log-log

Figure 2: Pressure Poisson Equation method

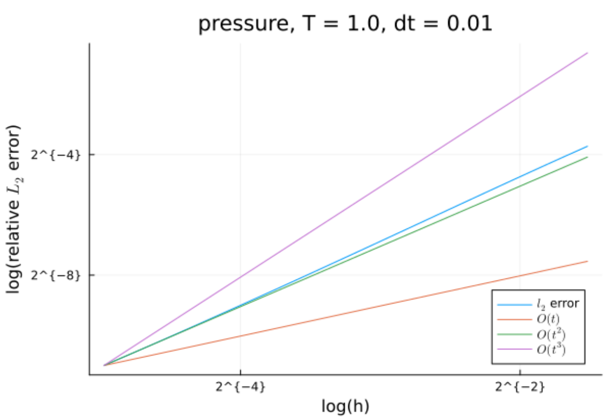


(a) relative L_2 error

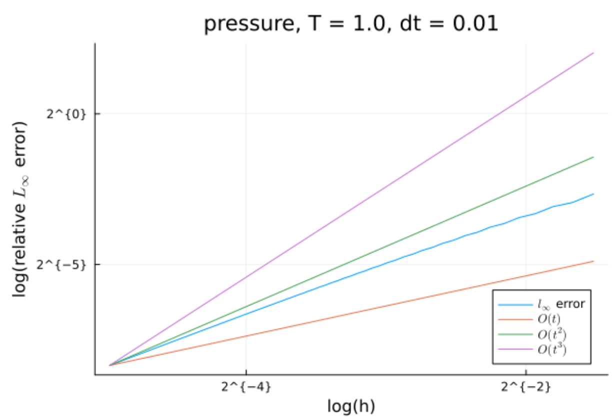


(b) relative L_∞ error

Figure 3: Pressure Poisson Equation method



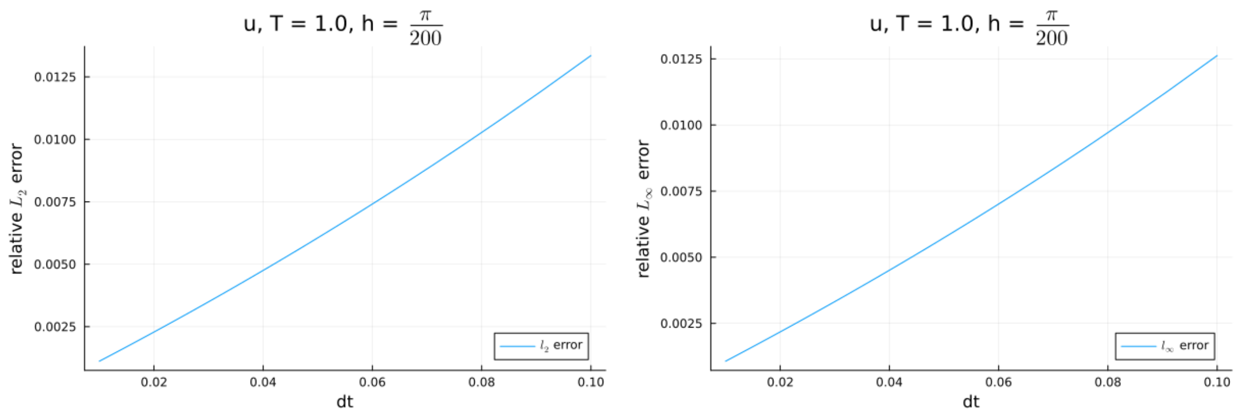
(a) relative L_2 error log-log



(b) relative L_∞ error log-log

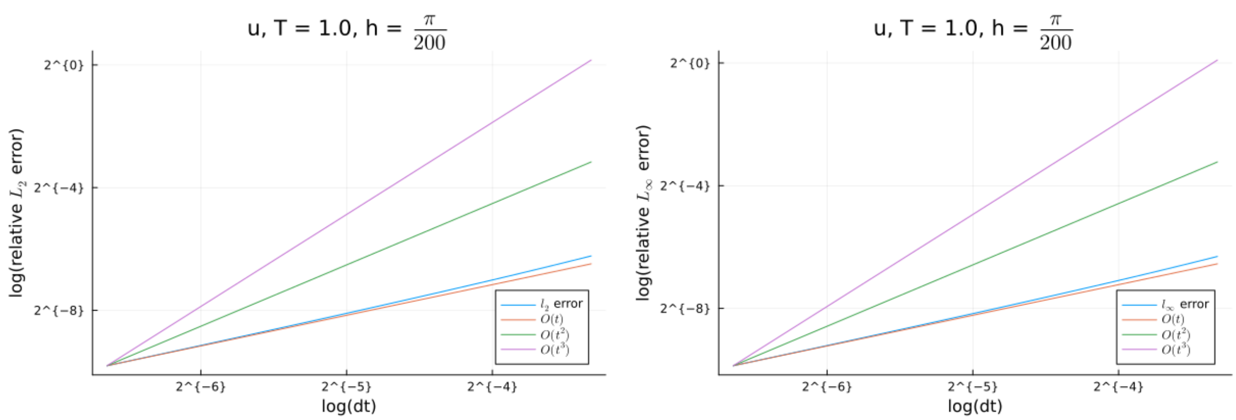
Figure 4: Pressure Poisson Equation method

The $O(h^2)$ convergence rate is shown for every quantity for both L_2 and L_∞, this should not be suprising since every space derivative operator is approximated in second order accuracy. For the convergence rate in time, let's first take a look at when only one substep is used(Only use pressure from p_n).



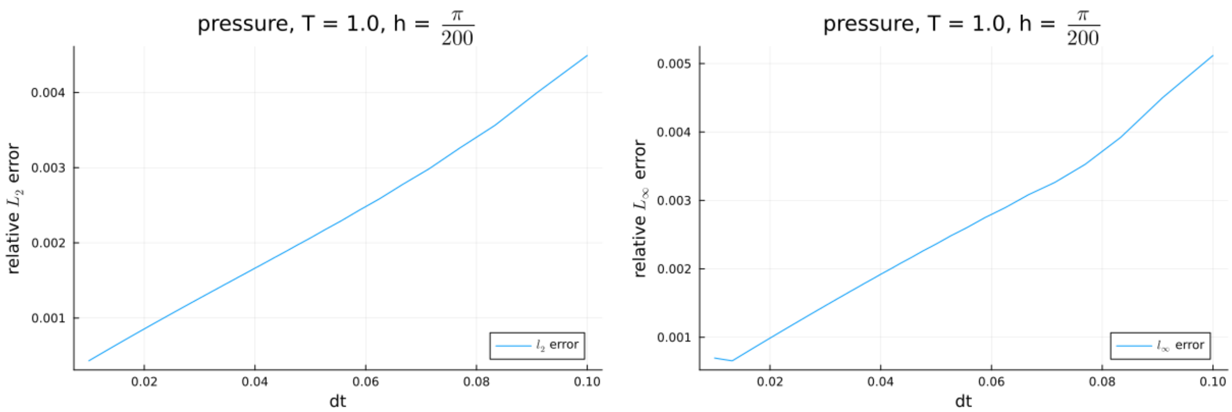
(a) Pressure Poisson Equation method 1-substep (b) relative L_∞ error

Figure 5: Pressure Poisson Equation method 1-substep



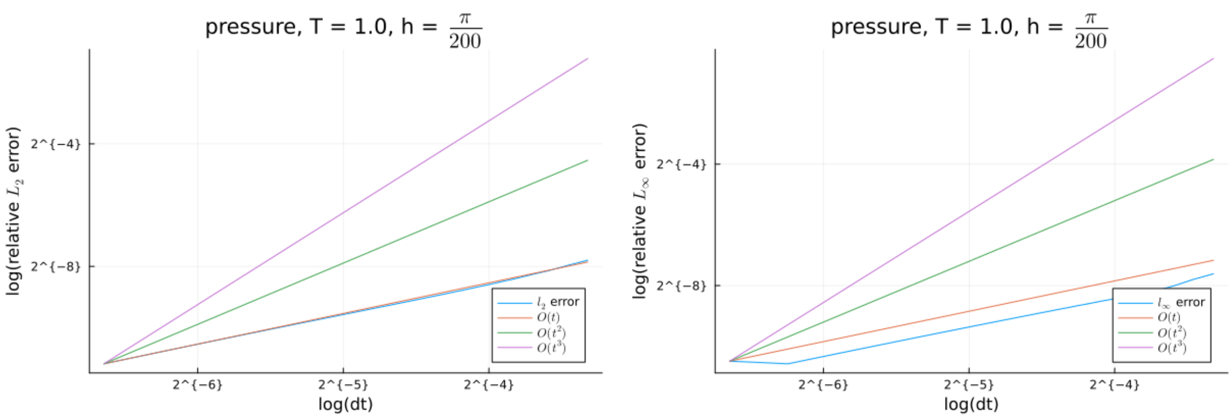
(a) relative L_2 error log-log (b) relative L_∞ error log-log

Figure 6: Pressure Poisson Equation method 1-substep



(a) relative L_2 error (b) relative L_∞ error

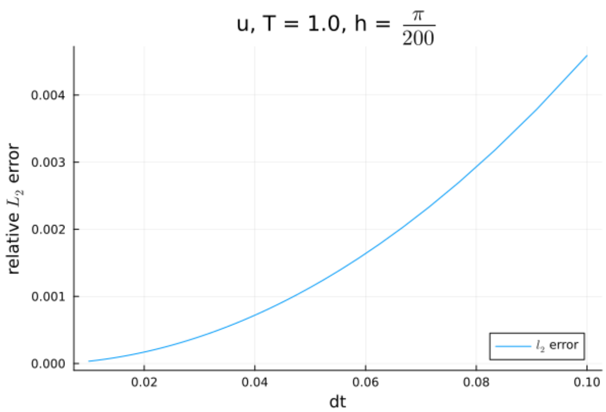
Figure 7: Pressure Poisson Equation method 1-substep



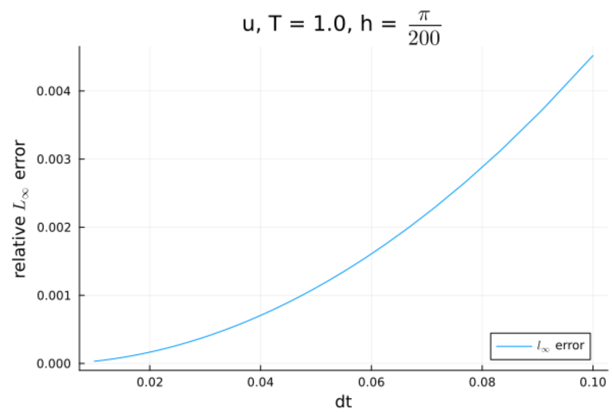
(a) relative L_2 error log-log (b) relative L_∞ error

Figure 8: dt vs error log-log for pressure

The convergence rate is only $O(\Delta t)$. And as stated before, a predictor-corrector scheme will significantly improve the time accuracy.

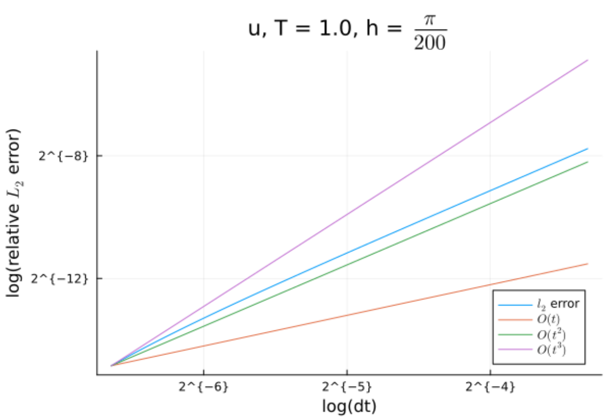


(a) relative L_2 error

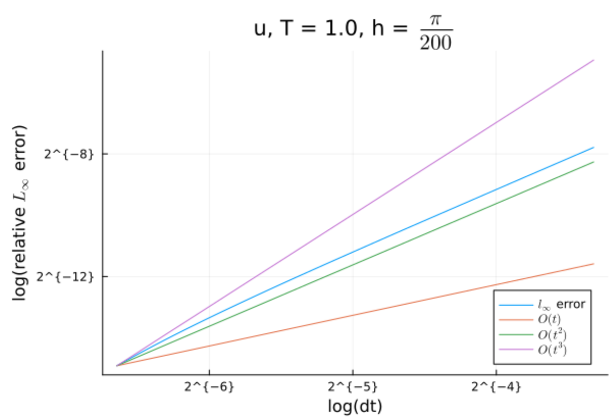


(b) relative L_∞ error

Figure 9: N vs error



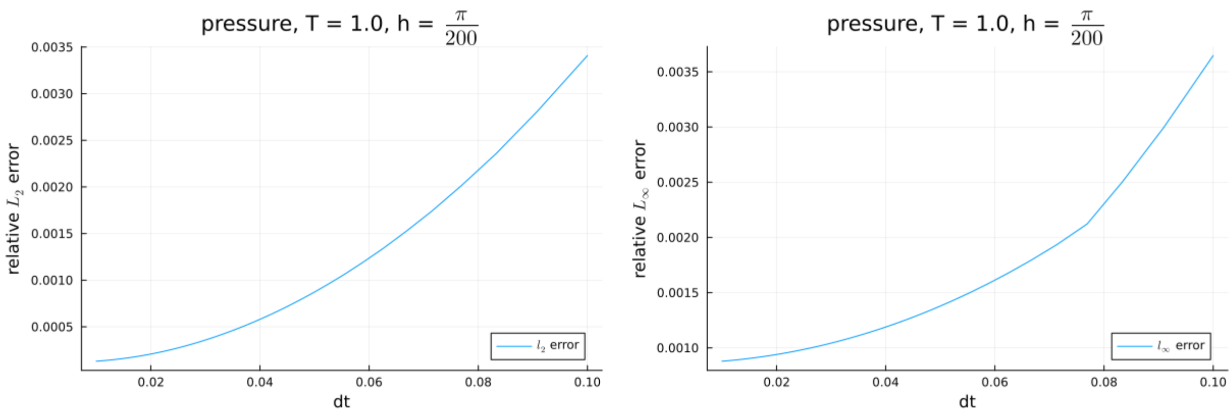
(a) relative L_2 error



(b) relative L_∞ error

Figure 10: N vs error

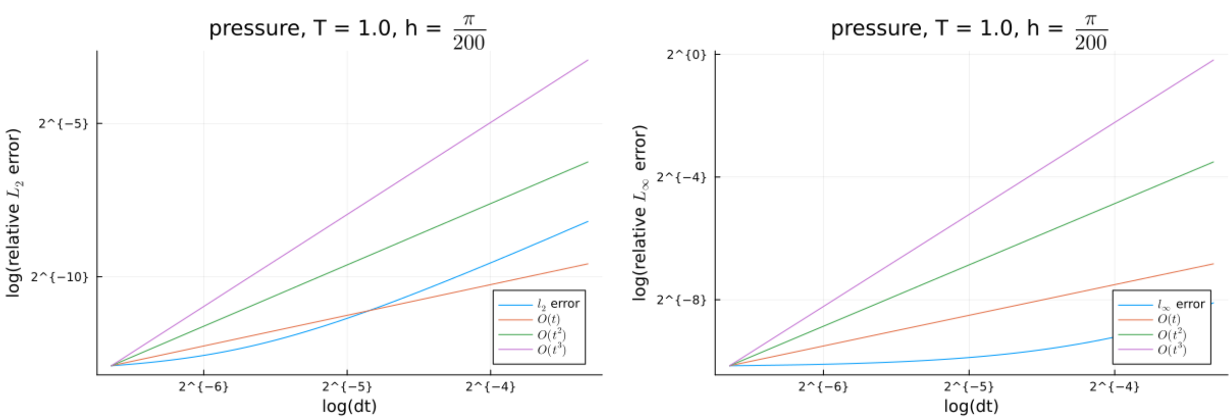
However, the convergence rate of pressure is trickier. It can be shown that the order of accuracy degenerate as smaller timestep is taken, especially for the L_∞ norm.



(a) relative L_2 error

(b) relative L_∞ error

Figure 11: dt vs error for pressure



(a) relative L_2 error log-log

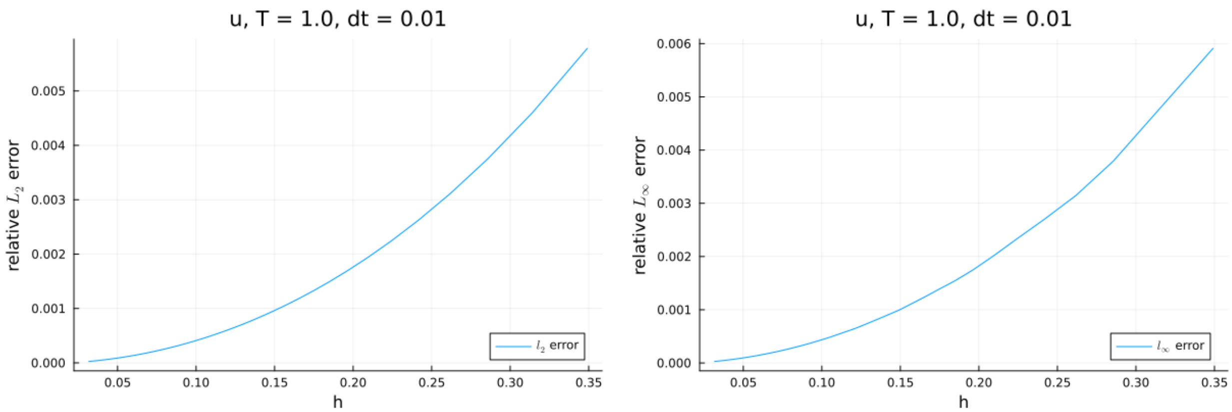
(b) relative L_∞ error

Figure 12: dt vs error log-log for pressure log-log

8.2 Second-order Projection Method

As discussed previously, the second-order accuracy for velocity can be relatively easy to obtain, however, the result of pressure is much more complicate and mysterious.

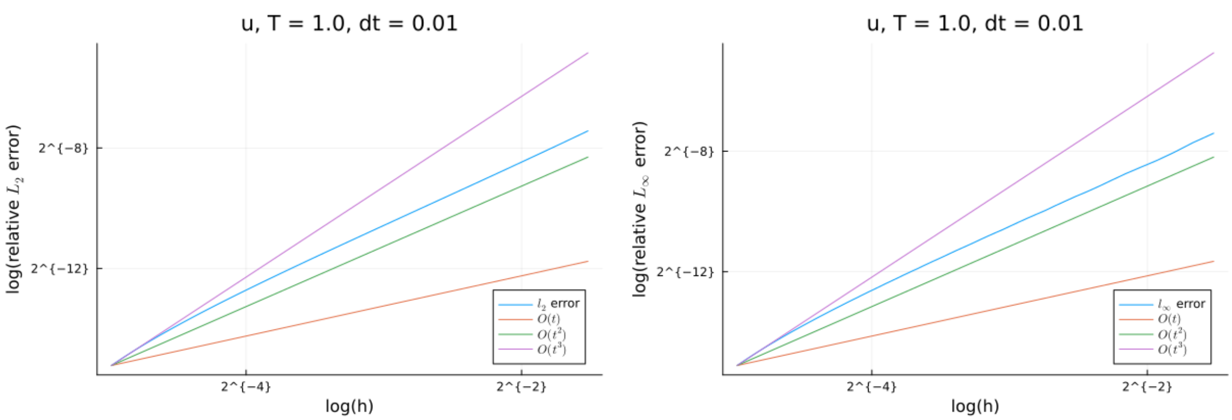
First for convergence rate in space, for velocity, $O(h^2)$ convergence is shown without any surprise.



(a) relative L_2 error

(b) relative L_∞ error

Figure 13: h vs error for u

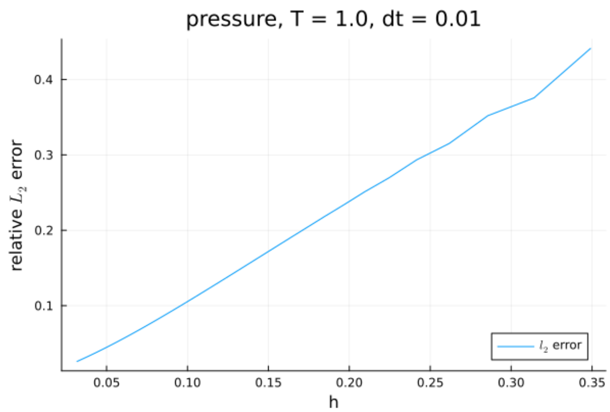


(a) relative L_2 error log-log

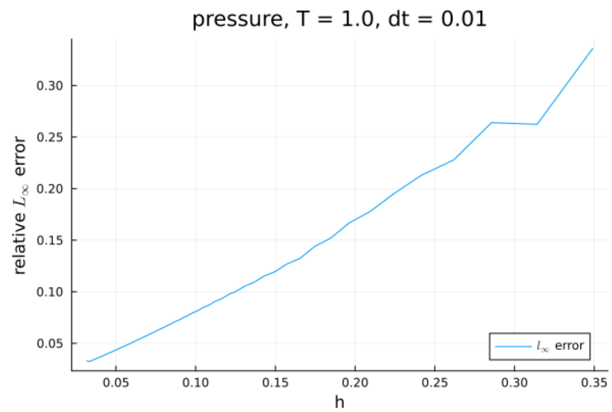
(b) relative L_∞ error log-log

Figure 14: h vs error log-log for u

However, for pressure, only $O(h)$ convergence is observed. The low accuracy linear interpolation might be the culprit, but it cannot explain why the accuracy of velocity is not polluted.

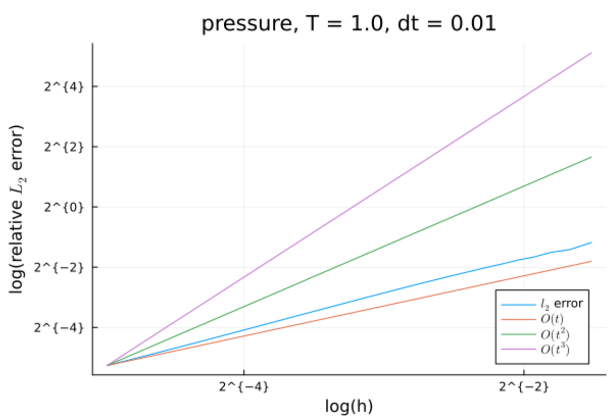


(a) relative L_2 error

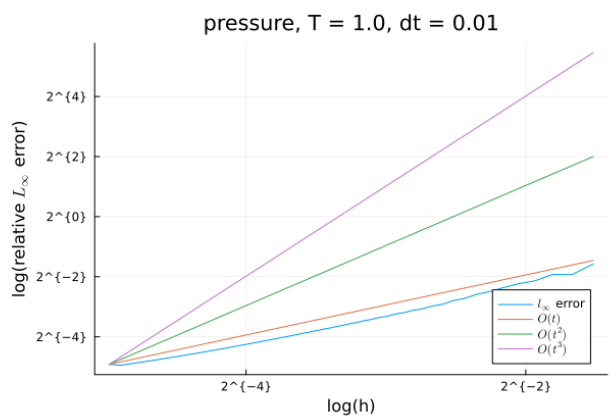


(b) relative L_∞ error

Figure 15: h vs error for u



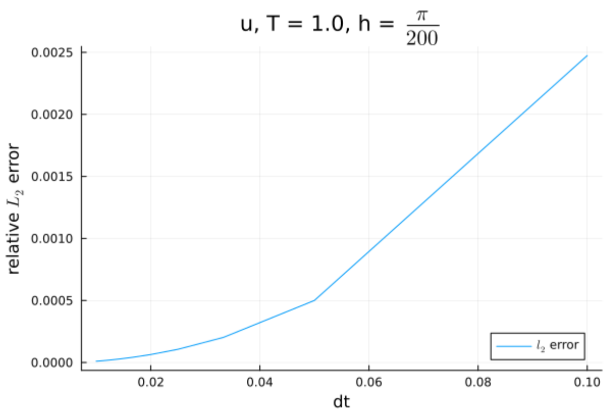
(a) relative L_2 error log-log



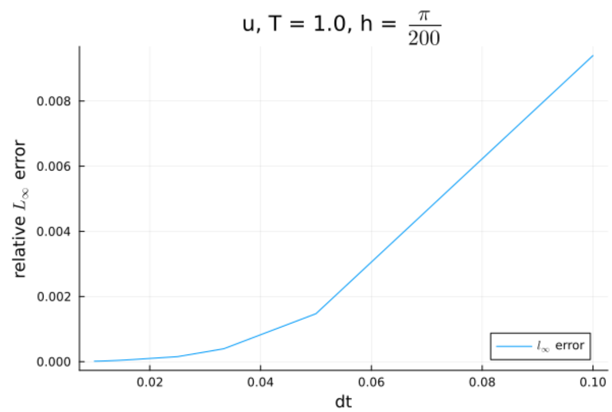
(b) relative L_∞ error log-log

Figure 16: h vs error log-log for u

When it comes to the accuracy in time, even more strange things happen. First, as discussed before, the accuracy of velocity in Chorin projection can be improved to second-order, with the usage of lag pressure value. However, the numerical result demonstrates even better accuracy, especially in L_∞ norm.

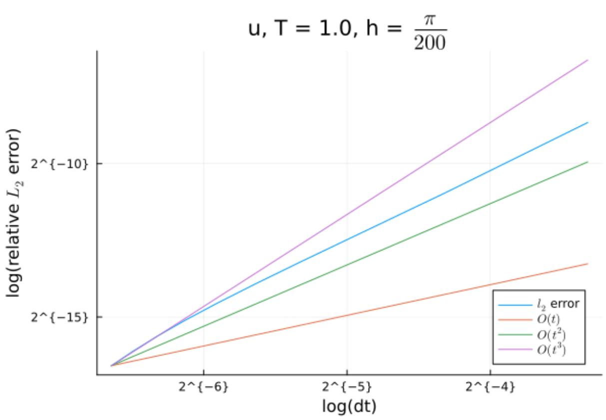


(a) relative L_2 error

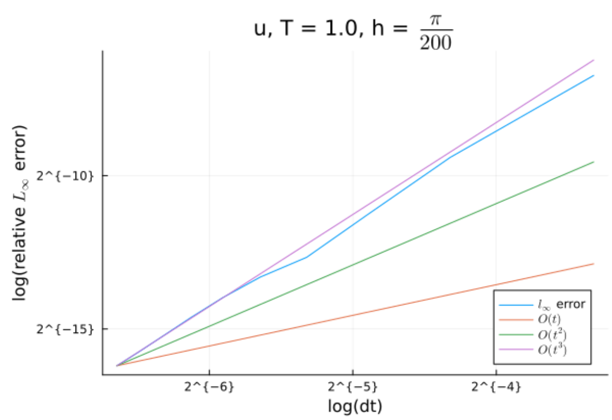


(b) relative L_∞ error

Figure 17: h vs error for u



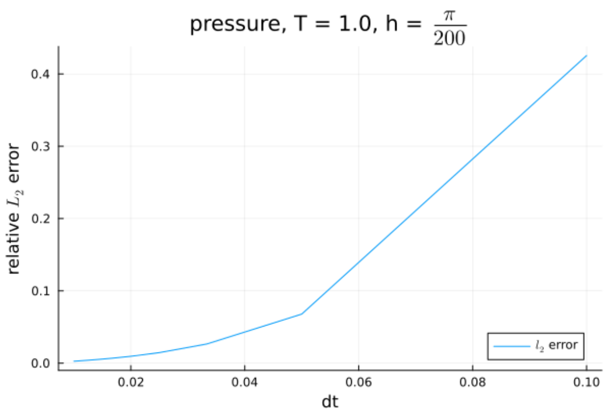
(a) relative L_2 error log-log



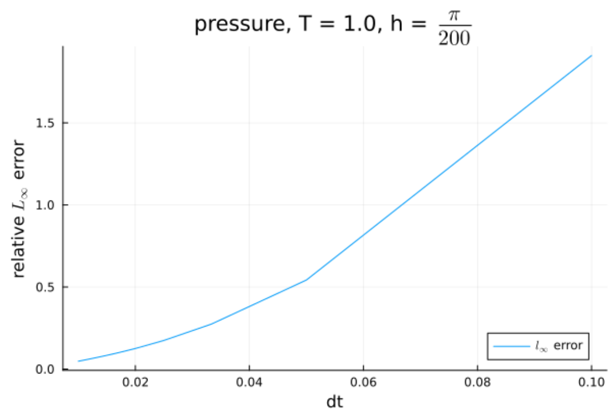
(b) relative L_∞ error log-log

Figure 18: h vs error log-log for u

The convergence rate of pressure is also interesting. First demonstrated is the result without rotational pressure correction.

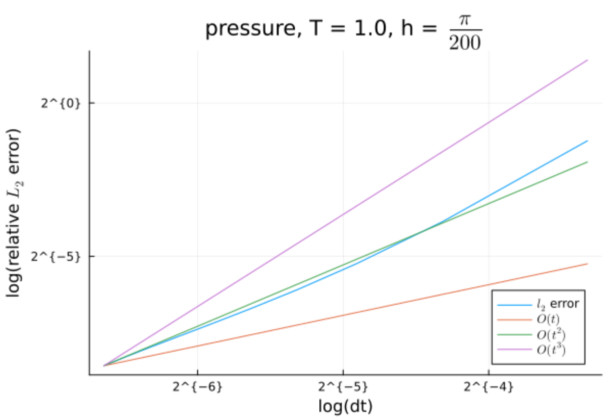


(a) relative L_2 error

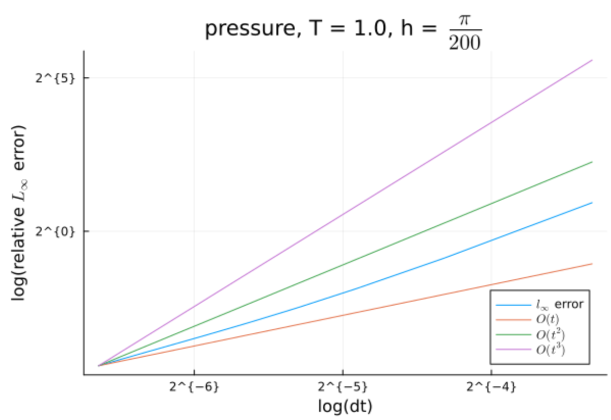


(b) relative L_∞ error

Figure 19: h vs error for u



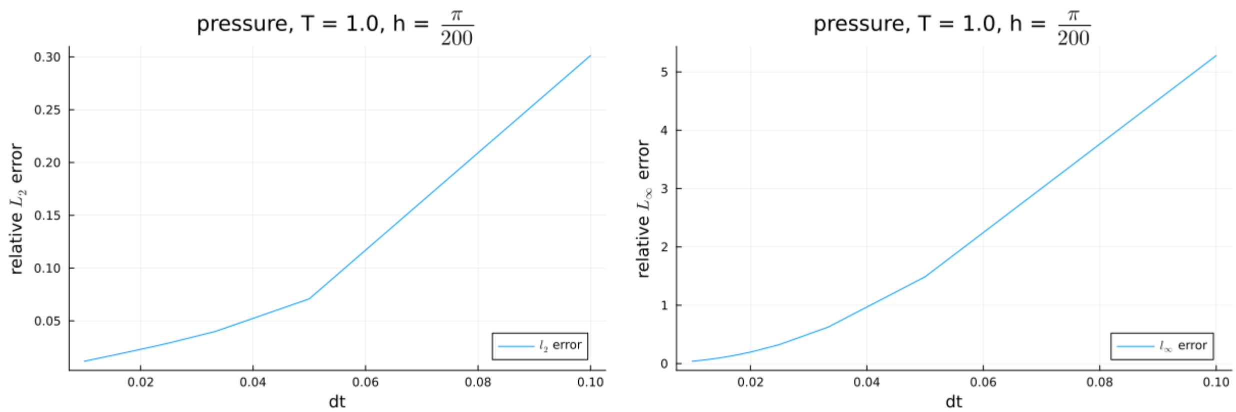
(a) relative L_2 error log-log



(b) relative L_∞ error log-log

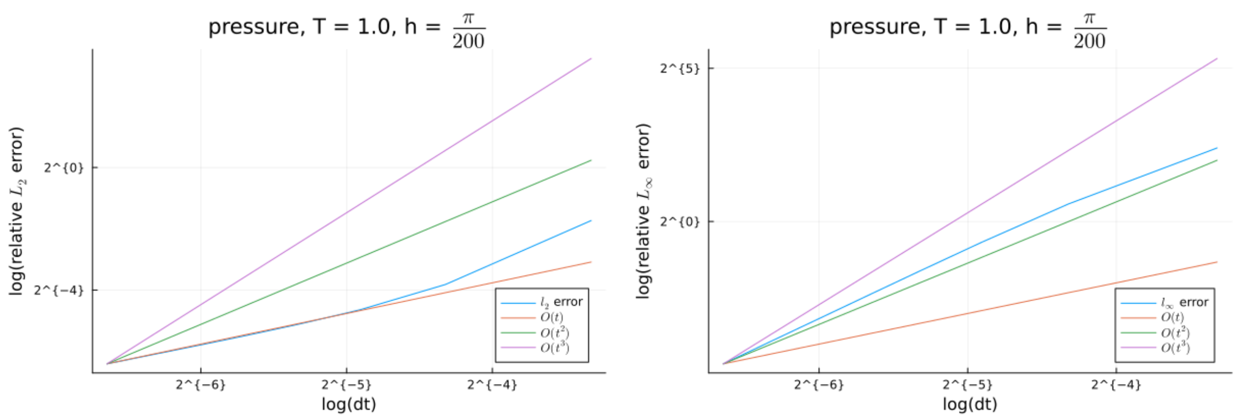
Figure 20: h vs error log-log for u

then with rotational pressure correction



(a) relative L_2 error (b) relative L_∞ error

Figure 21: h vs error for u

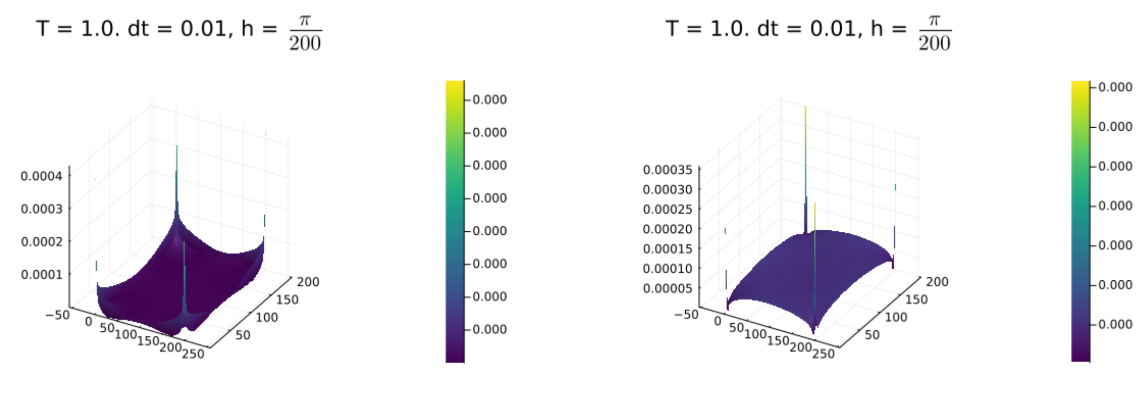


(a) relative L_2 error log-log (b) relative L_∞ error log-log

Figure 22: h vs error log-log for u

Two things are out of expectations: 1. the pressure is recovered in $O(\Delta t^2)$ in L_2 norm, and $O(\Delta t^{\frac{3}{2}})$ in L_∞ norm. 2. the rotational correction improve the accuracy in L_∞ norm to fully second-order, with the price that L_2 becomes only first-order accurate.

To make sure that the rotational correction do improve the accuracy for pressure, the point-wise error for both scheme under same configuration are provided:



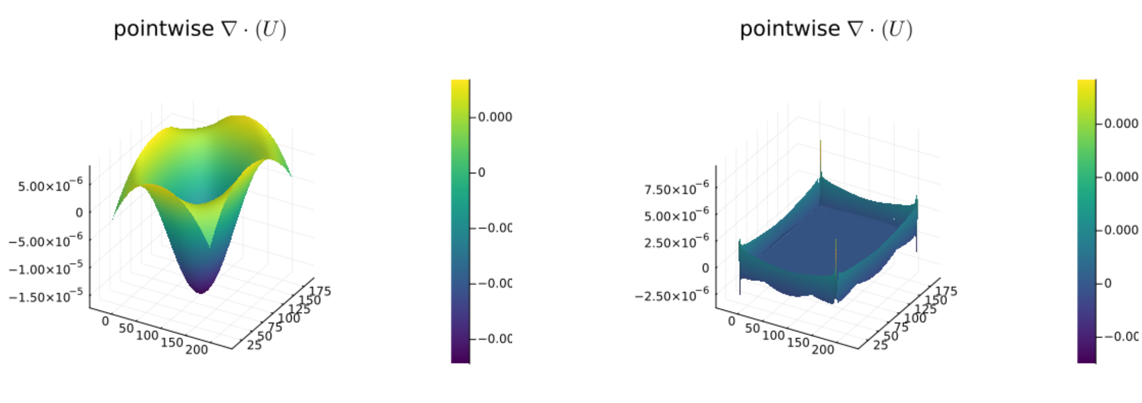
(a) relative L_2 error (b) relative L_∞ error

Figure 23: h vs error for u

It can be shown that although the error is lower at the boundary, the average error in the domain is increased slightly.

8.3 Comparison of Divergence

One might also be interested in how different method express the divergence free constraint on velocity. For this purpose, the point-wise divergence measurements are also provided.



(a) Pressure Poisson Equation (b) Second-order Accuracy Projection

Figure 24: point-wise divergence for **u**

It can be shown that projection method does a significantly better job on keeping the divergence free constraint, while it still suffer from the artificial boundary layer problem.

9 Conclusion

In this project, two popular methods to handle the incompressible NS equation are studied, implemented and tested. While the accuracy of velocity can be recovered with at least second-order in both L_2 and L_∞ norm, the behavior of pressure remains mysterious and unexpected. It could be caused by unexplored bugs in the numerical code. Further exploration on the theory is still required in future work.

References

[1] Dietmar Rempfer. "On boundary conditions for incompressible Navier-Stokes problems". In: (2006).

[2] Philip M Gresho and Robert L Sani. "On pressure boundary conditions for the incompressible Navier-Stokes equations". In: *International journal for numerical methods in fluids* 7.10 (1987), pp. 1111–1145.

[3] Sandeep Prabhakara and MD Deshpande. "The no-slip boundary condition in fluid mechanics: Solution of the sticky problem". In: *Resonance* 9 (2004), pp. 61–71.

[4] Francis H Harlow, J Eddie Welch, et al. "Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface". In: *Physics of fluids* 8.12 (1965), p. 2182.

[5] Alexandre Joel Chorin. "Numerical solution of the Navier-Stokes equations". In: *Mathematics of computation* 22.104 (1968), pp. 745–762.

[6] Jean-Luc Guermond, Peter Minev, and Jie Shen. "An overview of projection methods for incompressible flows". In: *Computer methods in applied mechanics and engineering* 195.44-47 (2006), pp. 6011–6045.

[7] John Kim and Parviz Moin. “Application of a fractional-step method to incompressible Navier-Stokes equations”. In: *Journal of computational physics* 59.2 (1985), pp. 308–323.

[8] JJIM Van Kan. “A second-order accurate pressure-correction scheme for viscous incompressible flow”. In: *SIAM journal on scientific and statistical computing* 7.3 (1986), pp. 870–891.

[9] J Guermond and Jie Shen. “On the error estimates for the rotational pressure-correction projection methods”. In: *Mathematics of Computation* 73.248 (2004), pp. 1719–1737.

[10] Michele Benzi, Gene H Golub, and Jörg Liesen. “Numerical solution of saddle point problems”. In: *Acta numerica* 14 (2005), pp. 1–137.

[11] A Salih. “Streamfunction-vorticity formulation”. In: *Department of Aerospace Engineering Indian Institute of Space Science and Technology, Thiruvananthapuram-Mach* (2013), p. 10.

[12] Tomáš Bodnár, Philippe Fraunié, and Karel Kozel. “Modified equation for a class of explicit and implicit schemes solving one-dimensional advection problem”. In: *Acta Polytechnica* 61.SI (2021), pp. 49–58.

[13] Long Chen. *Finite difference methods for poisson equation*. 2022.