Project 2

MAE 560 – Computational Fluid Mechanics and Heat Transfer, Fall 2022

Moatasim Farooque[a)]

(Dated: 7 December 2022)

Numerical solutions to 2D parabolic and elliptic PDEs

I. PROBLEM STATEMENT:

Numerically simulate the lid-driven cavity flows by solving the incompressible Navier–Stokes equations using the fractional-step method with 2nd-order Adam–Bashforth (AB2) scheme for the advection term and Crank–Nicolson (CN) for the viscous term . Follow the problem settings reported in Ghia, Ghia, and Shin and validate your results against those reported in the paper. Validation needs to be performed for at least one Reynolds number. Explore interesting cases with your powerful incompressible Navier–Stokes solver that you code up from scratch!

I.A. INS in the form of pressure-Poisson Equation

Using 2nd-order central finite differencing and the conjugate gradient method, numerically solve the Poisson equation (2) for the steady state solution $\tilde{u}(x, y)$ The Poisson Equation is given by:

I.A.1. Linear Operators:

To solve this we need to come up with a few operators whose effect is the same as applying a matrix transformation on our solution vector. These operators can be created by discretizing the domain by finite differences. Matrix representation for $nx = ny = 5$ is shown with each operator

I.A.2. Gradient Operator:

This is a first order backward difference on pressure nodes
This applies on P-type vector and gives out a u-type vector. This means we'll have two types of gradients one along the x-axis and the other along the y-axis.
The x-direction gradient is given by:

X - Direction Gradient

$$(\nabla p_{i,j})_x = \frac{p_{i-1,j} + p_{i,j}}{dx}$$

The y-direction gradient is given by:
Y - Direction Gradient

$$(\nabla p_{i,j})_y = \frac{p_{i,j-1} + p_{i,j}}{dy}$$

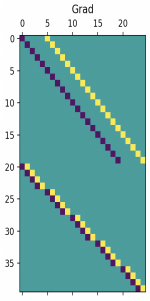

FIG. 1 Gradient Operator

I.A.3. Divergence Operator:

This applies on u-type vector and gives out a p-type vector.

$$(\nabla . \boldsymbol{U})_{i,j} = \left(\frac{-u_{i,j} + u_{i+1,j}}{dx}\right) + \left(\frac{-v_{i,j} + v_{i,j+1}}{dy}\right)$$

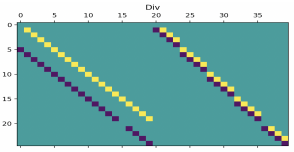

FIG. 2 Divergence Operator

I.A.4. Laplace Operator:

This is a 2nd order central difference on velocity nodes. This applies on u-type vector and gives out a u-type vector.
The x-direction laplacian is given by:
X-Component

$$(\nabla^2 \boldsymbol{u})_{i,j} = \left(\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{dx^2}\right) + \left(\frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{dy^2}\right)$$

The y-direction laplacian is given by:
Y-Component

$$(\nabla^2 \boldsymbol{v})_{i,j} = \left(\frac{v_{i-1,j} - 2v_{i,j} + v_{i+1,j}}{dx^2}\right) + \left(\frac{v_{i,j-1} - 2v_{i,j} + v_{i,j+1}}{dy^2}\right)$$

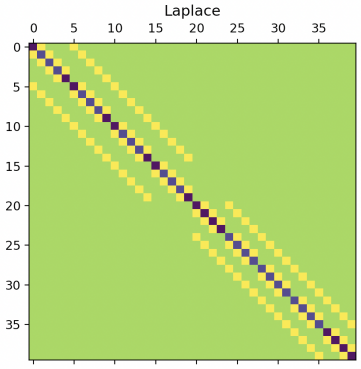

FIG. 3 Laplace Operator

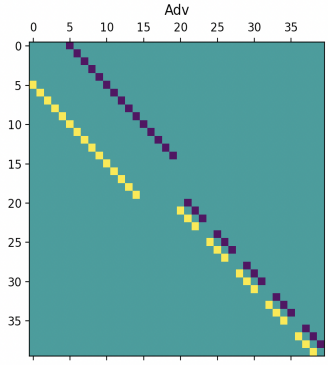

FIG. 4 Advection Operator

I.A.5. Advection Operator:

U-Component

$$\nabla.(\boldsymbol{U}.u)_{i,j} = \frac{1}{dx}\left[\left(\frac{u_{i-1,j}+u_{i,j}}{2}\right)\left(\frac{u_{i-1,j}+u_{i,j}}{2}\right)\right] - \frac{1}{dx}\left[\left(\frac{u_{i,j}+u_{i+1,j}}{2}\right)\left(\frac{u_{i,j}+u_{i+1,j}}{2}\right)\right] \frac{1}{dy}\left[\left(\frac{u_{i,j-1}+u_{i,j}}{2}\right)\left(\frac{v_{i-1,j}+v_{i,j}}{2}\right)\right] - \frac{1}{dy}\left[\left(\frac{u_{i,j}+u_{i,j+1}}{2}\right)\left(\frac{v_{i,j+1}+v_{i-1,j+1}}{2}\right)\right]$$

V-Component

$$\nabla.(\boldsymbol{U}.v)_{i,j} = \frac{1}{dx}\left[\left(\frac{u_{i-1,j}+u_{i,j}}{2}\right)\left(\frac{v_{i-1,j}+v_{i,j}}{2}\right)\right] - \frac{1}{dx}\left[\left(\frac{u_{i+1,j-1}+u_{i+1,j}}{2}\right)\left(\frac{v_{i+1,j}+v_{i,j}}{2}\right)\right] \frac{1}{dy}\left[\left(\frac{v_{i,j-1}+v_{i,j}}{2}\right)\left(\frac{v_{i,j-1}+v_{i,j}}{2}\right)\right] - \frac{1}{dy}\left[\left(\frac{v_{i,j}+v_{i,j+1}}{2}\right)\left(\frac{v_{i,j}+v_{i,j+1}}{2}\right)\right]$$

I.A.6. Fractional Step Method:

Step-1 involves calculating an intermediary Velocity u^F

$$\boldsymbol{R}\boldsymbol{u}^F = \boldsymbol{S}\boldsymbol{u}^n + \Delta t\left(3\boldsymbol{a}^n - \mathbf{a}^{n-1}\right)/2 + \Delta t\nu\left(\boldsymbol{bc}_{\boldsymbol{L}}^{n+1} + \boldsymbol{bc}_{\mathbf{L}}^{n}\right)/2$$

The second step is used to calculate the pressure which serves as a Lagrangian multiplier to guide the velocity towards the optimal direction.

$$\boldsymbol{D}\boldsymbol{R}^{-1}\boldsymbol{G}P^{n+1} = \boldsymbol{D}\boldsymbol{u}^F/\Delta t + \boldsymbol{bc}_{\boldsymbol{D}}^{n+1}/\Delta t$$

The final step is marching towards the correct velocity using the pressure multiplier.

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^F - \Delta t\boldsymbol{R}^{-1}\boldsymbol{G}P^{n+1}$$

$$\boldsymbol{R} \equiv \left(\boldsymbol{I} - \frac{\Delta t}{2}\nu\boldsymbol{L}\right)$$

$$\boldsymbol{S} \equiv \left(\boldsymbol{I} + \frac{\Delta t}{2}\nu\boldsymbol{L}\right)$$

and $\boldsymbol{R}^{-1} \approx \left(\mathbf{I} + \frac{\Delta t}{2}\nu\boldsymbol{L}\right)$

I.A.7. Conjugate Gradient Algorithm:

CG Algorithm will be used in steps 1 and 2 to calculate u^F and P^{n+1} The pseudocode for implementation is as follows:

$$d_{(0)} = r_{(0)} = b - Ax_{(0)}$$

$$\alpha_{(i)} = \frac{r_{(i)}^T T_{(i)}}{d_{(i)}^T A d_{(i)}}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$$

$$\beta_{(i+1)} = \frac{r_{(i+1)}^T T_{(i+1)}}{T_{(i)}^T T_{(i)}}$$

$$d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)}$$

First we will do some benchmarking and check the convergence of this algorithm for this specific problem 5

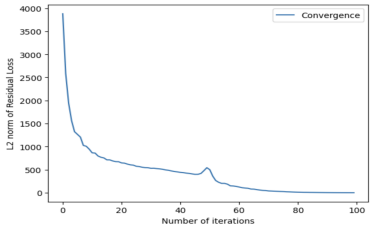

FIG. 5 Convergence Check

I.B. Results for Base Case Reynolds Number: 100

For the first part of the simulation, same problem setup was used as employed by Ghia et al in their paper so we can validate the results. Grid of 128X128 was used with drichlet boundary conditions on all the walls. The top lid is moving with velocity U=1. The rest of the walls are stationary. We get the following contours for Reynolds Number 100. 5 simulation was run till the solution stopped changing within 1e-5 tolerance.

This can be compared with the U-velocity results for Reynolds number of 100 given by Ghia. The mid-line of

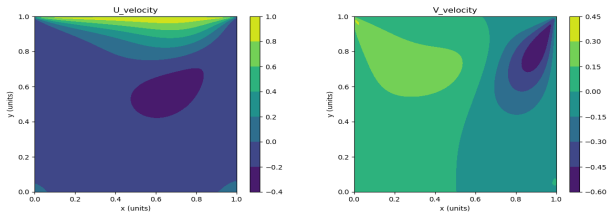

FIG. 6 Steady State U and V velocity for Re=100

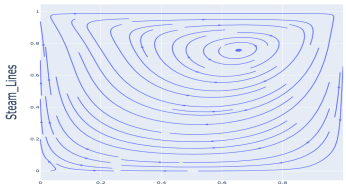

FIG. 7 StreamPlot for Re=100

domain in the x-direction is plotted so our x=64 (from a total of 128 points). The y-direction points are varying in the x-axis. The results show discrepancy at the end points. This could be because of minor boundary condition issue at that point. 8

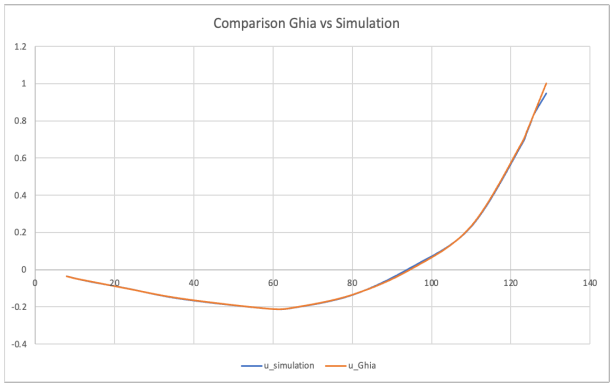

FIG. 8 Comparison

I.B.1. Results for Bottom Plate moving Case Reynolds Number: 100

To explore an interesting case, the bottom plate was moved in the opposite direction to the top plate 9

I.B.2. Vectorization and work on speed

The initial convergence time was way too high to be able to experiment multiple cases so techniques were explored to speed up the process. The first obvious way was vectorization of the operators. This significantly brought down time for one time step iteration from 80 sec/iterations to 1.15 iterations/second. There is a caveat associated with vectorization, that is the number of grid

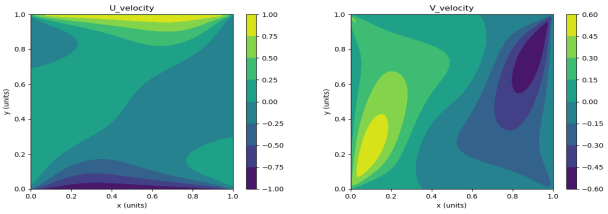

FIG. 9 Steady State U and V velocity for Re=100 and bottom plate moving

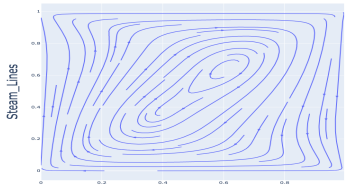

FIG. 10 StreamPlot for Re=100 and bottom plate moving

points should be able to fit in the RAM so calculations can be performed. Another technique applied was the use of parallel conjugate gradient algorithm. This can be employed using the multiprocessing capabilities of the system. For each iteration no. of guesses is provided in the form of a vector and is equal to the number of processes. The guess which minimzes the residual is proceeded to the next iteration. This further brought down the time for each iteration. Resulting in around 1.5 iterations per second.

[a)] Also at Mechanical Engineering Department, NCSU .