Numerical Simulation of the 2D Lid-Driven Cavity Benchmark Problem Using a Finite Volume Based Method

Jagannath Suresh

Department of Mechanical Engineering, Virginia Tech.

The lid-driven cavity problem is a benchmark flow case in CFD due its simplicity and the unambiguous nature of its boundary conditions. Considerable work has been done since the 1980s in this area, particularly with the increase of mesh refinement and Reynolds numbers. In this report, the 2D lid driven cavity problem is solved using a second-order Adams Bashforth explicit time integration scheme. The pressure poisson equation which results from the manipulation of the continuity equation is solved using a Gauss Seidel solver.
A staggered grid is used in this work to negate the effects of checkerboard pressure oscillations and to alleviate pressure velocity decoupling. The computations were carried out for multiple grid sizes which are: 32 $\times$ 32, 64 $\times$ 64, and 128 $\times$ 128. The results obtained were compared with existing data from literature which eventually showed good coherence.

I. Introduction

The classical lid-driven cavity problem has been investigated by many authors since some pio- neer works giving good results of steady solutions twenty years ago[1].[2] The problem statement is as follows. A 2-D rectangular (or square) cavity as shown in Fig. 1 is filled with an incompressible fluid. The top edge is a lid which slides at a uniform velocity while the other three edges are stationary. Since Kawaguti's original paper, many studies have been published about this problem. Erturk et al.[3] performed numerical simulations on a grid size of 601x601 with Reynolds numbers as high as 21000.Across all the studies, two important flow regimes were noted. One was the presence of primary and secondary vortices (around the center and lower corners of the domain respectively), and the occurrence of singularities at the corners of the domain.

In this study, a second order Adams Bahforth explicit time integration scheme is used to solve the 2D incompressible Navier Stokes equations. The Gauss Seidel solver was also implemented to solve the pressure poisson equation. Multiple grid sizes were used for simulations starting from 32 $\times$ 32 to 128 $\times$ 128 and the results were compared with data from existing literature.

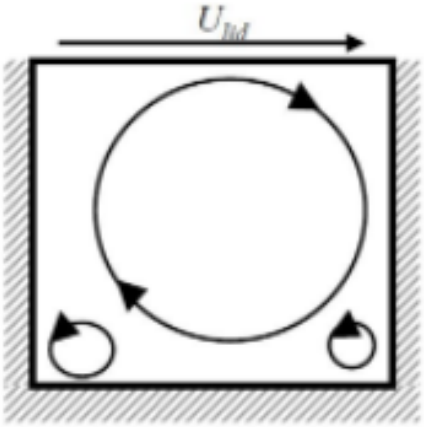


Figure 1: Representative diagram of 2D lid driven cavity

II. Governing Equations

Since the primary objective of the numerical study is to understand the development of the velocity profile in the 2D lid driven cavity, the governing equations for the study are the 2D incompressible Navier Stokes equations with non-dimensionalized primitive variables. The dimensional form of the respective equations is expressed below:

- **Continuity Equation**

$$[\frac{\partial U}{\partial x} + \frac{\partial V}{\partial y}] \tag{1}$$

- **X Momentum Equation**

$$\rho\frac{\partial U}{\partial t} + \rho[\frac{\partial UU}{\partial x} + \frac{\partial UV}{\partial y}] = \mu[\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}] - \frac{\partial P}{\partial x} \tag{2}$$

- **Y Momentum Equation**

$$\rho\frac{\partial V}{\partial t} + \rho[\frac{\partial UV}{\partial x} + \frac{\partial VV}{\partial y}] = \mu[\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2}] - \frac{\partial P}{\partial y} \tag{3}$$

The above equations are non dimensionalized using scaling factors. The velocities in the X and Y directions are scaled using the lid velocity U_{lid}. The length scale used is L and hence the time t is scaled using L/U_{lid}. The pressure is scaled using $\rho \times \mathrm{U}_{lid}^2$ where ρ is the density of the fluid. The non dimensionalized form of the equations are expressed below:

- **Continuity Equation**

$$[\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}] \tag{4}$$

- **X Momentum Equation**

$$\frac{\partial u}{\partial t} + [\frac{\partial uu}{\partial x} + \frac{\partial uv}{\partial y}] = \frac{1}{Re}[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}] - \frac{\partial p}{\partial x} \tag{5}$$

- **Y Momentum Equation**

$$\frac{\partial v}{\partial t} + [\frac{\partial uv}{\partial x} + \frac{\partial vv}{\partial y}] = \frac{1}{Re}[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}] - \frac{\partial p}{\partial y} \tag{6}$$

where u, v, and p are the non dimensionalized variables. And,

$$Re = \frac{\rho L U_{lid}}{\mu} \tag{7}$$

III. Boundary Conditions

A set of equations used along a domain boundary to obtain a particular solution to the problem is called a boundary condition. The equation solved and it's known variables influence the boundary condition for a given physical condition. For example on a physical "Wall", the boundary condition for the momentum equation could be a slip or a no-slip condition, while for the energy equation it could be a specified flux, an imposed temperature, or a convection heat transfer condition. The general categories of boundary conditions are mentioned below:

- a Dirichlet condition, where the unknown variable is defined at the boundary;
- a von Neumann condition, where the flux expressed in the conservation equation is defined at the boundary face;
- a Robin-type condition, where the unknown variable and flux at the boundary are expressed via a constitutive relation

The boundary conditions we use in the present study is:

- **No-Slip Boundary Condition** - This condition assumes that the speed of the liquid in contact with the boundary is equal to the velocity of the boundary which is zero in this study as we consider a stationary wall. The relative movement between the boundary and the fluid layer in contact with the boundary is zero and hence the name "no-slip condition".

No slip boundary conditions are applied on the bottom and side walls. Hence, on these walls:

- u = 0
- v = 0

And on the top wall,

- u = U_{lid}
- v = 0

IV. Numerical Discretization

One of the quantities that is key to Finite Volume formulations for the fluid flow equations is the mass flux across cell faces. And, to compute the mass flux at cell faces, we need velocities at cell faces. Unfortunately, in a normal grid the velocities are calculated at cell centres, not faces and calculation of cell face velocities from cell centre velocities requires some kind of interpolation. Unfortunately, early researchers in CFD found that if standard distance weighted interpolation is used, the resulting solution flip-flops between two or more solutions which is also called checkerboard pressure oscillations. The nature of this oscillation is shown in for structured Cartesian grids and in for unstructured grids. The staggered grid was proposed to counter the issue of checkerboard pressure oscillations which uses different control volumes for momentum and continuity equations and eliminates interpolation of cell centre velocities to find the face velocities.

As shown in Fig. 2, the u and v velocities are calculated on a staggered location compared to the nodes on which scalar quantities are calculated. The u velocities are calculated on the (i,j) marked to the left of the scalar node (i,j) marked with a circle. Similarly, the v velocities are calculated just below the scalar nodes.From the figure, it is also visible that dx_s and dy_s are the distances between the scalar nodes while dx_u and dx_v are the distances between the u and v nodes respectively.

The discretized X-momentum equation is expressed below:

$$\frac{\partial u}{\partial t}dx_s dy_v + uu|_e - uu|_w dy_v + (uv|_n - uv|_s)dx_s = -(p(i,j) - p(i-1,j))dy_v + \frac{1}{Re}(\frac{\partial u}{\partial x}|_e - \frac{\partial u}{\partial x}|_w)dy_v + \frac{1}{Re}(\frac{\partial u}{\partial y}|_n - \frac{\partial u}{\partial y}|_s)dx_s \tag{8}$$

where e,w,n, and s represent the east, west, north, and south faces respectively.
The cell face fluxes can be calculated as follows:

$$u_e = \frac{u(i+1,j) + u(i,j)}{2} \tag{9}$$

$$u_w = \frac{u(i-1,j) + u(i,j)}{2} \tag{10}$$

$$v_n = \frac{v(i,j+1) + v(i-1,j+1)}{2} \tag{11}$$

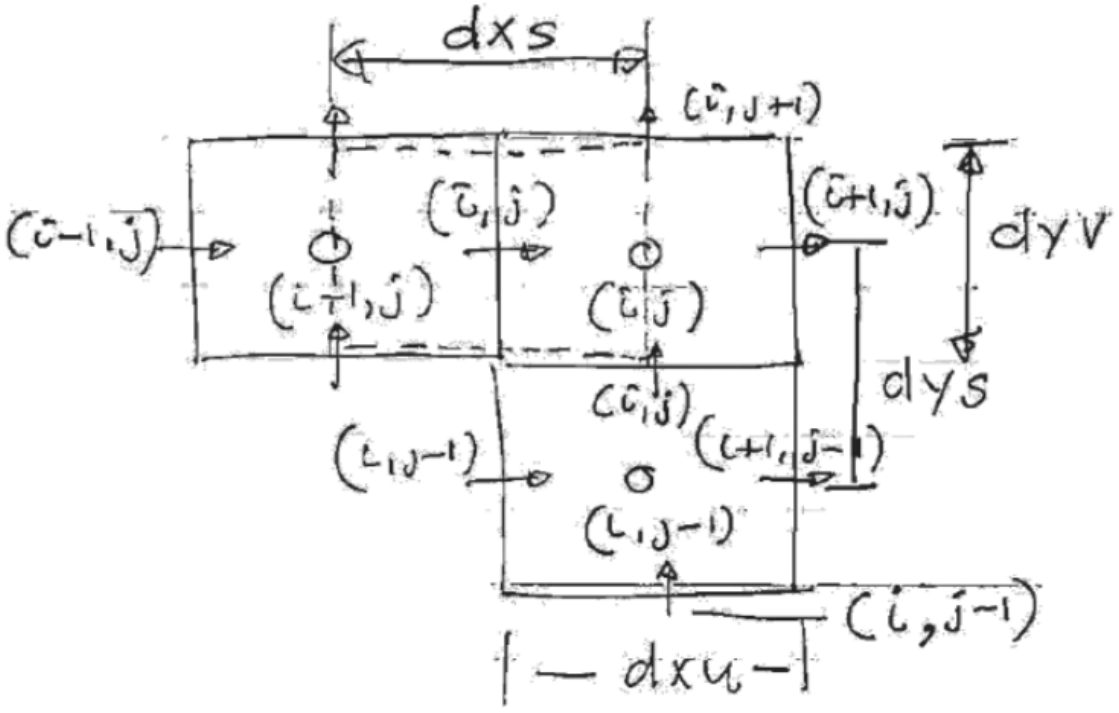


Figure 2: Representative figure of staggered grid

$$v_s = \frac{v(i-1,j) + v(i,j)}{2} \tag{12}$$

Using Central Difference on the values of u at cell faces, we get:

$$u_e = \frac{u(i+1,j) + u(i,j)}{2} \tag{13}$$

$$u_w = \frac{u(i-1,j) + u(i,j)}{2} \tag{14}$$

$$v_n = \frac{v(i,j+1) + v(i-1,j+1)}{2} \tag{15}$$

$$v_s = \frac{v(i-1,j) + v(i,j)}{2} \tag{16}$$

The cell face derivatives are:

$$\frac{\partial u}{\partial x}|_e = \frac{u(i+1,j) - u(i,j)}{dx_u} \tag{17}$$

$$\frac{\partial u}{\partial x}|_w = \frac{u(i,j) - u(i-1,j)}{dx_u} \tag{18}$$

$$\frac{\partial u}{\partial y}|_n = \frac{u(i,j+1) - u(i,j)}{dy_v} \tag{19}$$

$$\frac{\partial u}{\partial y}|_s = \frac{u(i,j) - u(i,j-1)}{dy_v} \tag{20}$$

The above discretized equations can be used in Eq. 8 which would give:

$$\frac{u_{t+dt} - u_t}{dt} = -\frac{p(i,j) - p(i-1,j)}{dx_s} + H_x \tag{21}$$

where,

$$H_x = -\frac{{A_E}^u u_E + {A_W}^u u_W + {A_N}^u u_N + {A_S}^u u_S + {A_P}^u u_P}{dx_s dy_v} \tag{22}$$

where,

$${A_E}^u = (\frac{u_e}{2} - \frac{1}{Re \times dx_u})dy_v \tag{23}$$

$${A_W}^u = (-\frac{u_w}{2} - \frac{1}{Re \times dx_u})dy_v \tag{24}$$

$${A_N}^u = (\frac{v_n}{2} - \frac{1}{Re \times dy_s})dx_s \tag{25}$$

$${A_S}^u = (-\frac{v_s}{2} - \frac{1}{Re \times dy_s})dx_s \tag{26}$$

$${A_P}^u = (\frac{u_e}{2} + \frac{1}{Re \times dx_u})dy_v + (-\frac{u_w}{2} + \frac{1}{Re \times dx_u})dy_v + (\frac{v_n}{2} + \frac{1}{Re \times dy_s})dx_s + (-\frac{v_s}{2} + \frac{1}{Re \times dy_s})dx_s \tag{27}$$

Similarly, for the y momentum equation we get:

$$\frac{v_{t+dt} - v_t}{dt} = -\frac{p(i,j) - p(i,j-1)}{dy_s} + H_y \tag{28}$$

where,

$$H_y = -\frac{{A_E}^v v_E + {A_W}^v v_W + {A_N}^v v_N + {A_S}^v v_S + {A_P}^v v_P}{dx_u dy_s} \tag{29}$$

where,

$$A_E{}^v = (\frac{u_e}{2} - \frac{1}{Re \times dx_s})dy_s \tag{30}$$

$$A_W{}^v = (-\frac{u_w}{2} - \frac{1}{Re \times dx_u})dy_s \tag{31}$$

$$A_N{}^v = (\frac{v_n}{2} - \frac{1}{Re \times dy_v})dx_u \tag{32}$$

$$A_S{}^v = (-\frac{v_s}{2} - \frac{1}{Re \times dy_s})dx_u \tag{33}$$

$$A_P{}^v = (\frac{u_e}{2}+\frac{1}{Re \times dx_s})dy_s+(-\frac{u_w}{2}+\frac{1}{Re \times dx_s})dy_s+(\frac{v_n}{2}+\frac{1}{Re \times dy_v})dx_u+(-\frac{v_s}{2}+\frac{1}{Re \times dy_v})dx_u \tag{34}$$

In the above equations, point P is the point (i,j). E stands for east which is (i+1,j), W stands for west which is (i-1,j), N stands for north which is (i,j+1) and south stands for south (i,j-1). And, because of the use of a uniform grid, $dx_u = dy_v = dx_s = dy_s$.

V. Algorithm and Solution Procedure

The algorithm used is a fractional step method where an intermediate velocity field ($\tilde{u}$ and $\tilde{v}$) is computed using the H_x and H_y from the present field. This intermediate velocity field is used to solve the Pressure Poisson equation from where we get the pressure. Furthermore, this pressure is used to calculate the velocity at the next time step. Adams Bashforth time stepping technique has been used to march forward in time. The following are the Predictor and Corrector steps:

- **Predictor**:

$$\tilde{u} = u^n + \frac{1}{2}\Delta t(3H^n - H^{n-1}) \tag{35}$$

- **Corrector**:

$$u^{n+1} = \tilde{u} - \Delta t \nabla p \tag{36}$$

Since the pressure is unknown at the $(n+1)^{th}$ level, the solution of the corrector step is used in the continuity equation at the $(n+1)^{th}$ level to form the Pressure Poisson equation. On applying the discretization to the continuity equation, we get:

$$(u(i+1,j) - u(i,j))dy_v + (v(i,j+1) - v(i,j))dx_u = 0 \tag{37}$$

The pressure equations are:

$$u(i+1,j)^{n+1} = \tilde{u}(i+1,j) - \Delta t(\frac{p(i+1,j) - p(i,j)}{dx_s}) \tag{38}$$

$$u(i,j)^{n+1} = \tilde{u}(i,j) - \Delta t(\frac{p(i,j) - p(i-1,j)}{dx_s}) \tag{39}$$

$$v(i,j+1)^{n+1} = \tilde{v}(i,j+1) - \Delta t(\frac{p(i,j+1) - p(i,j)}{dy_s}) \tag{40}$$

$$v(i,j)^{n+1} = \tilde{v}(i,j) - \Delta t(\frac{p(i,j) - p(i,j-1)}{dy_s}) \tag{41}$$

Substituting the above values in the continuity equation, we get the Poisson equation of the form:

$$A_P^P P_P + \sum_l A_l^P P_l = S_{i,j} \tag{42}$$

where ${A_E}^P = {A_W}^P = \frac{dy_v}{dx_s}$, ${A_N}^P = {A_S}^P = \frac{dx_u}{dy_s}$, and ${A_P}^P$ = -2 ($\frac{dy_v}{dx_s}$ + $\frac{dx_u}{dy_s}$).

The source term $S_{i,j}$ is:

$$S_{i,j} = \frac{1}{\Delta t}[(\tilde{u}(i+1,j) - \tilde{u}(i,j))dy_v + (\tilde{v}(i,j+1) - \tilde{v}(i,j))dx_u] = 0 \tag{43}$$

The basic steps of the algorithm and solution procedure followed are:

- Solve for the intermediate velocity using Eq. 35 i.e the Predictor step.
- Solve the Pressure Poisson equation using the intermediate velocity field. A Gauss Seidel solver was used with the residual at each iteration converging to a value of 1 $\times 10^{-5}$.
- Correct the intermediate velocity field using the Corrector step (Eq . 36) and obtain the velocity in the subsequent time step.
- Repeat the algorithm until steady state is reached. The system can be said to be steady when the changes in the solution are less than desired tolerance, which in the case studied is 1 $\times 10^{-8}$.

VI. Treatment of Boundaries

The boundaries has to be treated specially due to the staggered nature of the grid. And hence, the boundaries have to be dealt with differently for different equations at the boundaries.

- For the X momentum equation, H_x has a distinctly different value at the top and bottom wall boundaries since the coefficients change their values due to the grid spacing becoming half of the original.

  At the bottom wall:

$$A_S{}^u = (-\frac{v_s}{2} - \frac{2}{Re \times dy_s}) \tag{44}$$

$$A_P{}^u = (\frac{u_e}{2} + \frac{1}{Re \times dx_u})dy_v + (-\frac{u_w}{2} + \frac{1}{Re \times dx_u})dy_v + (\frac{v_n}{2} + \frac{1}{Re \times dy_s})dx_s + (-\frac{v_s}{2} + \frac{1}{Re \times dy_s})dx_s + \frac{dy_v}{Redx_u} \tag{45}$$

  At the top wall:

$$A_N{}^u = (-\frac{v_n}{2} - \frac{2}{Re \times dy_s}) \tag{46}$$

$$A_P{}^u = (\frac{u_e}{2} + \frac{1}{Re \times dx_u})dy_v + (-\frac{u_w}{2} + \frac{1}{Re \times dx_u})dy_v + (\frac{v_n}{2} + \frac{1}{Re \times dy_s})dx_s + (-\frac{v_s}{2} + \frac{1}{Re \times dy_s})dx_s + \frac{dy_v}{Redx_u} \tag{47}$$

- For the Y momentum equation, H_y has a different value at the east and west wall boundaries.

  At the east wall:

$$A_E{}^v = (-\frac{u_e}{2} - \frac{2}{Re \times dx_s}) \tag{48}$$

$$A_P{}^v = (\frac{u_e}{2} + \frac{1}{Re \times dx_u})dy_v + (-\frac{u_w}{2} + \frac{1}{Re \times dx_u})dy_v + (\frac{v_n}{2} + \frac{1}{Re \times dy_s})dx_s + (-\frac{v_s}{2} + \frac{1}{Re \times dy_s})dx_s + \frac{dx_u}{Redy_v} \tag{49}$$

  At the west wall:

$$A_W{}^v = (-\frac{u_w}{2} - \frac{2}{Re \times dx_s}) \tag{50}$$

$$A_P{}^v = (\frac{u_e}{2} + \frac{1}{Re \times dx_u})dy_v + (-\frac{u_w}{2} + \frac{1}{Re \times dx_u})dy_v + (\frac{v_n}{2} + \frac{1}{Re \times dy_s})dx_s + (-\frac{v_s}{2} + \frac{1}{Re \times dy_s})dx_s + \frac{dx_u}{Redy_v} \tag{51}$$

- For the Pressure Poisson equation:

At the west wall:

$$A_W{}^P = 0 \tag{52}$$

$$A_N{}^P = A_S{}^P = \frac{dx_u}{dy_s} \tag{53}$$

$$A_E{}^P = \frac{dy_v}{dx_s} \tag{54}$$

$$A_P{}^P = (\frac{dy_v}{dx_s} - 2\frac{dx_u}{dy_s}) \tag{55}$$

Similarly, at the east wall:

$$A_E{}^P = 0 \tag{56}$$

$$A_N{}^P = A_S{}^P = \frac{dx_u}{dy_s} \tag{57}$$

$$A_W{}^P = \frac{dy_v}{dx_s} \tag{58}$$

$$A_P{}^P = (\frac{dy_v}{dx_s} - 2\frac{dx_u}{dy_s}) \tag{59}$$

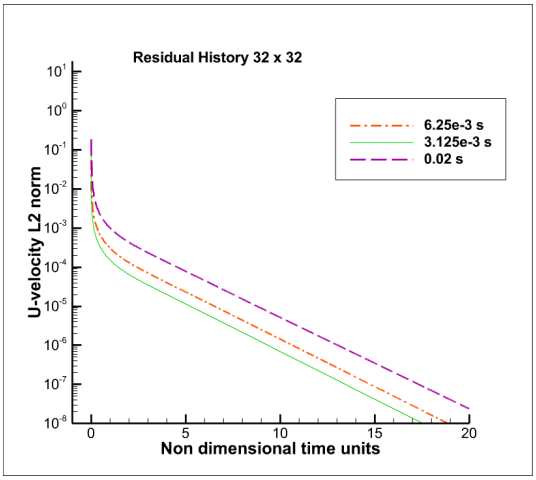
The south and north walls are also treated in a similar fashion.

VII. Results

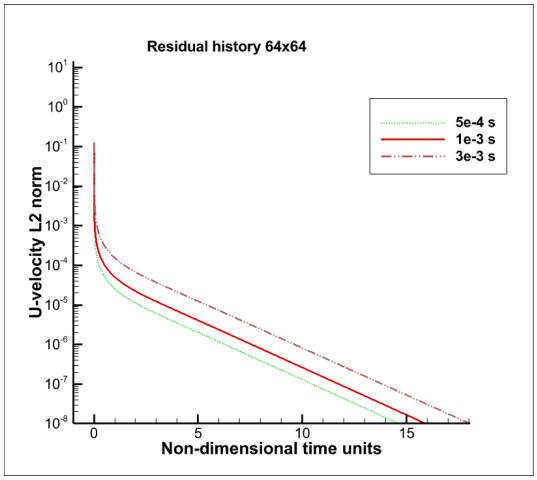
The sections so far gave a detailed overview of the techniques which were opted solve this 2D lid driven cavity problem. MATLAB was used in this study to develop the code to solve the 2D incompressible Navier Stokes equations using the algorithm mentioned in the previous section. Tecplot was used for post-processing the results.

A. Residual History and CPU Time

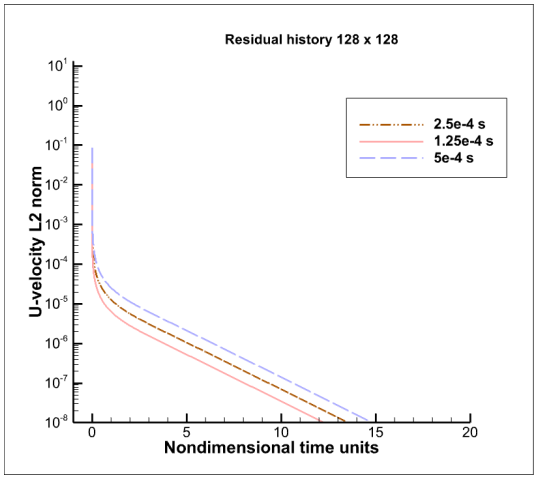
The code was tested on 3 grid levels which are 32×32, 64×64, and 128×128. A Reynolds number of 100 was chosen for the calculation and the residual history for each grid size corresponding to multiple time steps where plotted against the non dimensional time units as depicted in Fig. 3.



(a) Residual history of U velocity on 32 x 32 grid



(b) Residual history of U velocity on 64 x 64 grid



(c) Residual history of U velocity on 128 x 128 grid

Figure 3: Residual history comparison on different grid sizes at Re = 100

It appears from Fig. 3 that the non dimensional time units reduce along with the grid sizes. However, it has not be noted that even though the non-dimensional simulation times might seem to reduce with the grid spacing, the actual simulation time is higher for finer grids and smaller time steps as it becomes computationally intensive as we reduce the size of the grid and the time step. This can be very clearly seen in Fig. 4, where the CPU time is plotted against the time step dt for all the cases.

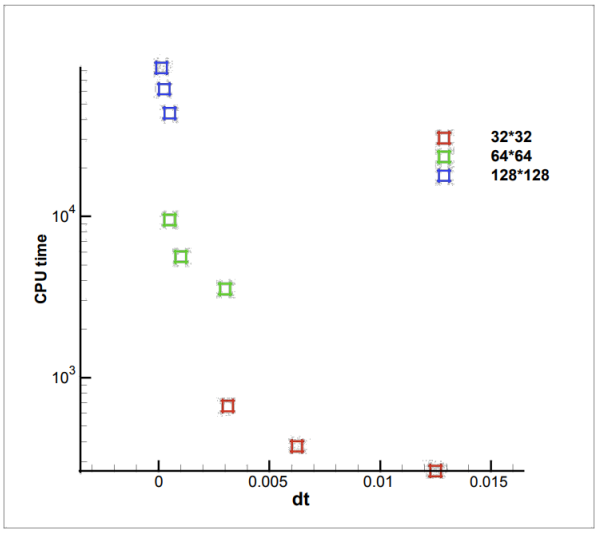


Figure 4: CPU time vs time steps for different grid sizes at Re = 100

As it is visible from Fig. 4, the larger the grid sizes and time step, the larger the CPU time taken for the code to converge.

1. Comparison of time units to converge for Re = 100 and Re = 1000

A maximum time step of 5×10^{-3} is chosen for the 128×128 grid at Re = 1000. This was chosen according to the Von Neumann stability criterion which is directly proportional to the Reynolds number. Hence, as the Reynolds number is increased by a factor of 10, the maximum permissible time step can also be increased by a factor of 10.

On comparison with the equivalent Re = 100 case with a time step of 5×10^{-4}, the Re = 1000 case converged faster.The Re = 100 case took 29551 iterations to converge whereas the Re = 1000 case took only 19930 iterations to reach steady state. The larger permissible time step resulting from the stability criterion allows for faster convergence but in the cost of accuracy.As it is visible from Fig. 5, Re = 1000 took larger number of non dimensional time units to converge, but this is not indicative of the CPU time as the CPU time taken to converge is higher for the Re = 1000 case.

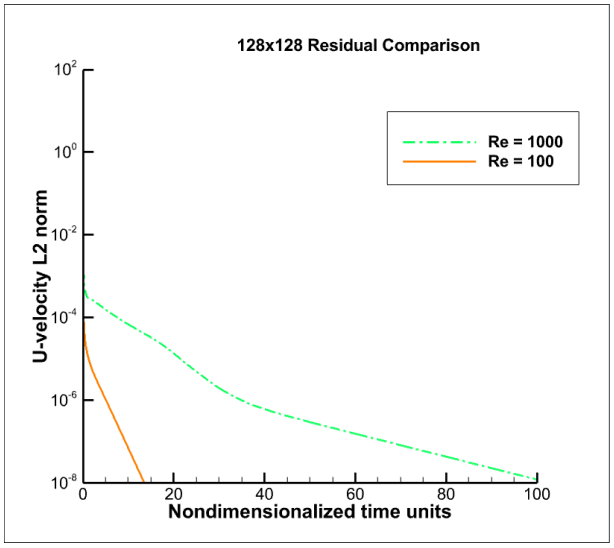


Figure 5: Graph comparing time units to converge for Re = 100 and Re = 1000

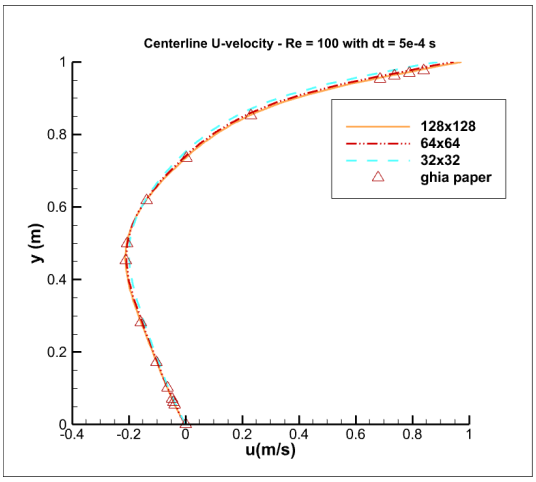
B. Velocity Profiles

Fig. 6 and Fig. 7 represents the centerline velocities in the X and Y directions for all the 3 grid sizes compared to the results from Ghia et al.'s[1] paper for a Reynolds number of 100 and 1000 respectively. As depicted in the figure, the solution is very coherent with the data from the literature. Especially as the grid size becomes finer i.e when the grid is 128 $\times$ 128, the results becomes very close to that obtained in the literature. This is expected because the finer grid captures the intricacies of the flow better than the coarser grids. Another comparison that can be made between Fig. 6 and Fig. 7 is that in Fig. 7 where the Re = 1000, the velocity profiles take sharp reversals along the curve compared to Fig. 6 where the curve is relatively smooth.

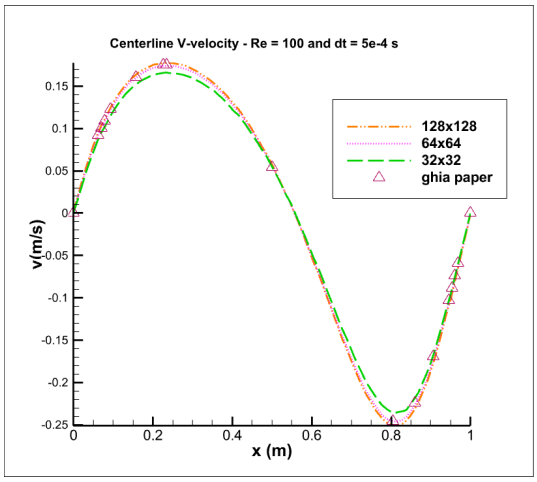
Fig. 8 and Fig. 9 are the contour graphs of u and v velocities in the domain for a Re = 100 and Re = 1000 case respectively. It can be seen that at Re = 100, the velocity of the lid has not diffused into the cavity as much as at Re = 1000. It can also be observed that the primary vortex is closer to the lid at Re = 100 and is away from the lid at Re = 1000. This is due to weaker advection resulting from lower velocities in the Re = 100 case. This results in the formation of a relatively weak primary vortex which is closer to the lid. This effect can also be seen in the sizes of the secondary vortices in the corners of the cavity which is larger when Re = 1000.

C. Possible Improvements

As seen in the results mentioned above, the methods used showed good correspondence with with results from Ghia et al.'s[1] paper. However, a few improvements can be made to expedite

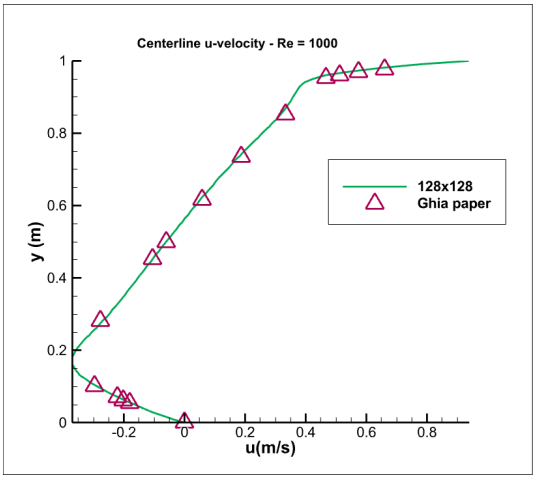


(a) Centerline u velocity

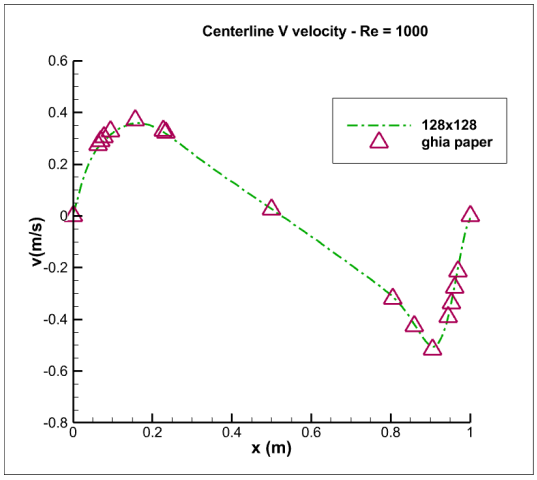


(b) Centerline v velocity

Figure 6: U and V velocities along the centerline at Re = 100



(a) Centerline u velocity

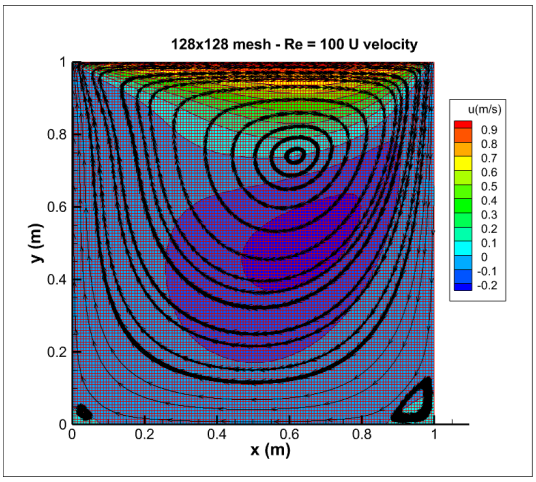


(b) Centerline v velocity

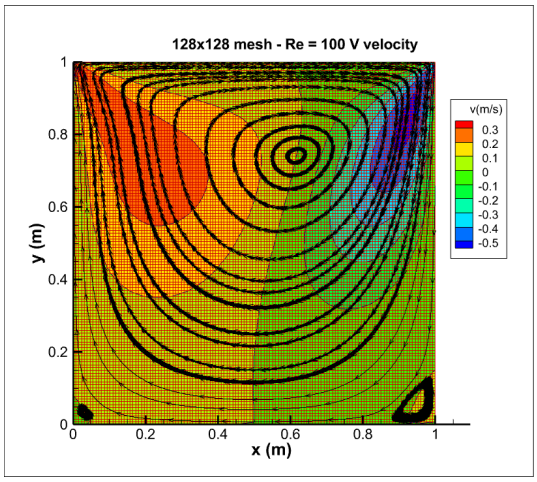
Figure 7: U and V velocities along the centerline at Re = 1000

the convergence of the code like the following:

- The current code uses a Gauss Seidel solver whereas methods like ADI, Conjugate Gradient and multigrid methods would definitely lead to faster convergence.
- Parallelization of the code would make the convergence faster. The domain can split into pieces and computations can be carried out in a piecewise manner which would save a lot of computational time.

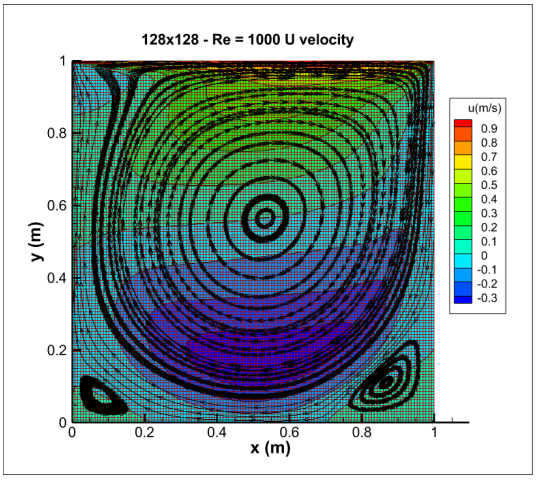


(a) U velocity contour at Re = 100

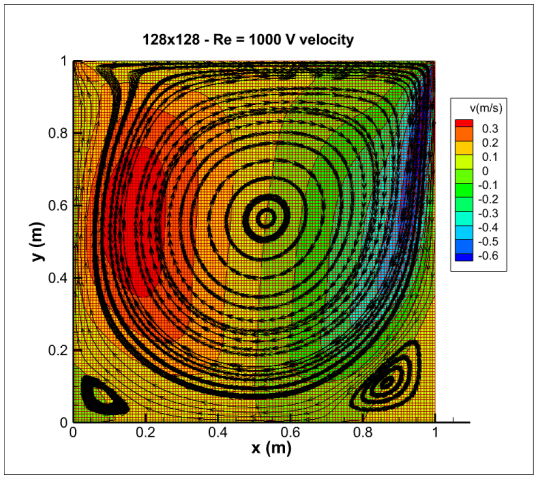


(b) V velocity contour at Re = 100

Figure 8: U and V velocity contours at Re = 100 and grid size 128 × 128 with streamlines



(a) U velocity contour at Re = 1000



(b) V velocity contour at Re = 1000

Figure 9: U and V velocity contours at Re = 1000 and grid size 128 × 128 with streamlines

VIII. Conclusion

The 2D incompressible Navier Stokes equations were solved using time integration method with a second order Adams Bashforth discretization scheme. Gauss Seidel method was implemented to solve for the pressure and the simulation was carried out for 3 different grid sizes along with the experimentation of various time steps and were compared with values from existing literature. The results obtained were in excellent agreement with the results from Ghia et al.'s work.

References

[1] U Ghia, K.N Ghia, and C.T Shin. High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method. *Journal of Computational Physics*, 48(3):387–411, 1982.

[2] Roel Verstappen, Jan Wissink, and Arthur Veldman. Direct numerical simulation of driven cavity flows. *Applied Scientific Research*, 51:377–381, 06 1993.

[3] Ercan Erturk, Thomas Corke, and C. Gokcol. Numerical solutions of 2-d steady incompressible driven cavity flow at high reynolds numbers. *International Journal for Numerical Methods in Fluids*, 48:747 – 774, 07 2005.