A Comparison of Numerical Solutions to the Navier Stokes Equations in the 2D Lid-driven Cavity Problem

David Lin, Horace Chu
Department of Physics, Stanford University, Stanford, California 94305, USA
(Dated: May 31, 2019)

The Navier Stokes equations are a cornerstone of modern computational fluid dynamics. Although the ideal discretization scheme and solution method for solving these equations varies by case, some of the most popular numerical methods include the finite difference method, SIMPLE algorithm with finite volume discretization, and Galerkin finite element method. Thus, we compare the effectiveness of these three methods as applied to the standard benchmark of 2D lid-driven cavity flow. These methods were evaluated against 3 different Reynolds numbers (20, 100, 1000) for robustness, compared against *Ghia et al* for accuracy, and measured across different mesh sizes for runtime. Our results suggest that finite difference methods are fastest and most robust to changes in flow, but are not as accurate as the SIMPLE implementation. The finite element methods performed poorly in all of our tests because they require significantly more case-by-case parameter tuning than the other two methods.

INTRODUCTION

In computational fluid dynamics, the Navier-Stokes equations are the fundamental partial differential equations describing the viscous motion of fluids [1,2,3]. The equations, derived from Isaac Newton's Second Law to fluid motion, assume that the stress term comprises of a diffusing viscous term and a pressure term. Written in momentum form, the Navier Stokes equation becomes

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

where $\mathbf{u}$ represents the 2-d velocity components of the system and $\mathbf{p}$ represents the pressure components of the system [1,2].

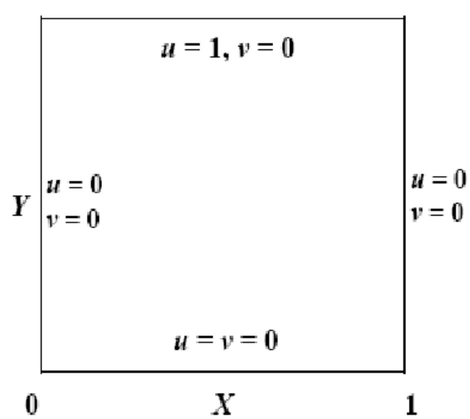

FIG. 1: Lid-driven Cavity Schematic. Figure courtesy of *Rusi et al.* [4]

While there is no general analytical solution to the Navier Stokes equations, there are a number of numerical solutions that leverage grid-based iterative methods [3,5,6,7,8]. In computational fluid dynamics, the lid-driven cavity problem is considered a classical test of novel numerical methods and a popular basis for benchmark comparisons that has been studied for decades [3,4,5]. In particular, we consider the problem of 2-D lid-driven cavity flow with the assumption of incompressible fluids and no-slip conditions at all fixed boundaries. The setup consists of a square cavity with three rigid walls and a fourth "lid" moving tangentially with constant velocity (See Fig. 1). This problem is heavily favored by researchers for its simple geometry that still retains enough complexity to test numerical methods against a rich set of fluid physics [4].

In this paper, we will consider three methods: finite-difference, SIMPLE, and finite-element. Each of these methods is described in more detail in the methods section. We will evaluate the efficacy of each method on the basis of precision, speed, and robustness. We can evaluate the precision of our results by comparing velocity distributions within the cavity against past literature benchmarks [3,4,5]. Further, we will test how the runtime of each algorithm scales with the granularity of our grid.

To evaluate robustness, we must consider the Reynolds number, a dimensionless constant that dictates the types of flow patterns within the cavity. The Reynolds number is defined as

$$Re = \frac{\rho u L}{\mu} = \frac{uL}{v}$$

where ρ is the density of the fluid, u is the velocity of the fluid with respect to the object, L is a characteristic linear dimension, μ is the dynamic viscosity of the fluid and v is the kinematic viscosity of the fluid [1]. At low Reynolds numbers, laminar flow dominates over other types of flow, while at high Reynolds numbers, differences in the fluid's speed and direction cause turbulence, which may even run counter to the original fluid flow. As such, we will be tuning the Reynolds number to gauge how robust our computational methods are against different types of flow.

RELATED WORK

The most frequently cited work in lid-driven cavity flow is the early paper *Ghia et al.* [3]. Their work leveraged a coupled strongly implicit multigrid (CSI-MG) method to solve the discretized Navier Stokes equation on a 129x129 grid. They optimize their algorithms to ensure stable solutions for high Reynolds numbers up to 10,000. Although the algorithms are not robust to such high Reynolds numbers, we will use their lower Reynolds solutions as a basis for comparison.

The three main bodies of numerical solutions to fluid flow problems and more generally partial differential equations are finite difference, finite volume, and finite element methods. The oldest and most popular of these techniques is finite difference. As the name suggests, these body of methods approximate the derivatives of Taylor expanded differential equations using finite differences among local grid values. Popular works in this domain include that of *Li et al.* and *Weinan and Guo* [6,7]. Accuracy of such finite difference methods is primarily determined by mesh size and boundary approximations.

The second body of numerical PDE solutions is that of finite volume. These methods involve resolving the differential equations at the discretized cell level and balancing the fluxes across these different cells. Popular works in this domain include *Versteeg and Malasekera* and *Sahin and Owen* [2,9]. Similar to the prior reference, we make use of a staggered grid as well as their use of the SIMPLE algorithm to resolve the pressure equation.

Finally, finite element methods constitute the most complex but also most powerful techniques for modeling fluid dynamics. Finite element methods reformulate the PDEs as a linear algebra problem that can be resolved using matrix methods. The variable of interest within each grid element is approximated via an approximate shape function. These methods typically afford the most modelling flexibility, but that additional representation power comes at the expense of interpretability. The Galerkin method of weighted residuals is particularly popular among finite element methods [10,11]. In our project, we use an implementation of this method adapted from *Tsega and Katiyar* [11].

In recent years, there have been many bells and whistles added onto this standard body of algorithms. Most of these papers either seek to speed up existing algorithms or improve accuracy for a particular geometric set up and flow type. However, we do not consider most of these more subtle optimizations since they introduce significant additional complexity and the goal of this study is to compare the efficacy of the most commonly used numerical techniques with minimal parameter tuning.

METHODS

Finite Difference Problem

Perhaps the simplest way of solving the Navier Stokes equations is the finite difference method, where the differential equations in question are approximated with difference equations, and the derivatives are approximated by finite differences. The u-momentum equation becomes

$$\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t}+u_{i,j}^{n}\frac{u_{i,j}^{n}-u_{i-1,j}^{n}}{\Delta x}+v_{i,j}^{n}\frac{u_{i,j}^{n}-u_{i,j-1}^{n}}{\Delta y}= -\frac{1}{\rho}\frac{p_{i+1,j}^{n}-p_{i-1,j}^{n}}{2\Delta x}+ \nu(\frac{u_{i+1,j}^{n}-2u_{i,j}^{n}+u_{i-1,j}^{n}}{\Delta x^2}+\frac{u_{i,j+1}^{n}-2u_{i,j}^{n}+u_{i,j-1}^{n}}{\Delta y^2})$$

And the v-momentum equation becomes

$$\frac{v_{i,j}^{n+1}-v_{i,j}^{n}}{\Delta t}+u_{i,j}^{n}\frac{v_{i,j}^{n}-v_{i-1,j}^{n}}{\Delta x}+v_{i,j}^{n}\frac{v_{i,j}^{n}-v_{i,j-1}^{n}}{\Delta y}= -\frac{1}{\rho}\frac{p_{i+1,j}^{n}-p_{i-1,j}^{n}}{2\Delta y}+ \nu(\frac{v_{i+1,j}^{n}-2v_{i,j}^{n}+v_{i-1,j}^{n}}{\Delta x^2}+\frac{v_{i,j+1}^{n}-2v_{i,j}^{n}+v_{i,j-1}^{n}}{\Delta y^2})$$

On the other hand, the pressure Poisson equation is

$$\frac{p_{i+1,j}^{n}-2p_{i,j}^{n}+P_{i-1,j}^{n}}{\Delta x^2}+\frac{p_{i,j+1}^{n}-2p_{i,j}^{n}+p_{i,j-1}^{n}}{\Delta y^2}= \rho[\frac{1}{\Delta t}(\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}+\frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}) -\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}- 2\frac{u_{i,j+1}-u_{i,j-1}}{2\Delta y}\frac{v_{i+1,j}-v_{i-1,j}}{2\Delta x}- \frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}]$$

Since the reduction of differential equations into algebraic equations are suited to modern computing, we start our analysis of the problem with this numerical technique. Our implementation of this method follows very closely to the pipeline described in *Barba and Forsyth* [1].

SIMPLE Algorithm

The SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm is another numerical method to solve the Navier Stokes Equations. The main assumption of this algorithm is in the velocity correction equation, which neglects the effect of the pressure correction on the velocity. While this assumption aids runtime complexity, in cases of turbulent flow the algorithm may not necessarily converge.

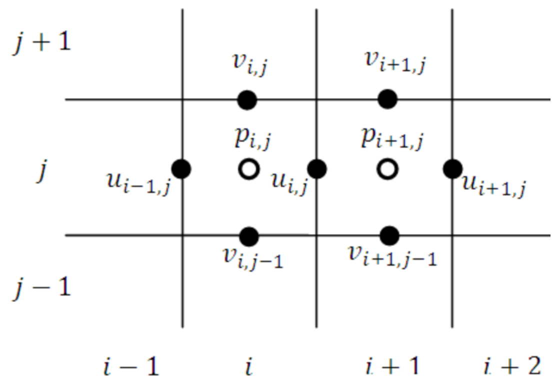

FIG. 2: Staggered Grid Diagram. Figure courtesy of *S.R. Djeddi et al* [13].

In the implementation of this algorithm, we replicate the boundary conditions of the finite difference method. However, in contrast with the previous set up, the SIMPLE algorithm stores velocity and pressure components on a staggered grid (see Fig. 2). Within each square area of the uniform mesh, the pressure is stored at a nodal point in the center while the velocity components alongside the faces. While the staggered grid introduces additional complexity to solving, it improves the accuracy of pressure gradient calculations.

Given the initial pressure, we solve the x and y momentum equations to compute the intermediate u and v velocity estimates. We then use the pressure-correction equation to solve for the pressure correction terms at the center of each staggered grid cell. This term is then used to correct our intermediate estimates for p, u, and v. In each of these correction equations, we include an under-relaxation term between 0 and 1. This constant term determines the strength of the correction, which we need to be high enough to make efficient iterative improvements while low enough to maintain stability. The process above is repeatedly applied until convergence is achieved. Our implementation of this algorithm is adapted from the work of *Sarfi* [12].

Finite Element Method

On the other hand, the finite element method tries to approximate the unknown function (modelled by the differential equation) over the relevant domain. This technique works by subdividing the whole system into smaller sections known as finite elements. The equations describing these finite elements are much simpler, and can easily be assembled into a larger system of equations that model the entire problem. Our implementation of this algorithm is adapted from the work of *Sahin and Owens* [11].

EXPERIMENTS

1. Robustness Test: Reynolds Number

By toggling the Reynolds number, we can simulate the spectrum of laminar to turbulent flow, and then compare the accuracy and precision of our numerical techniques for each of these cases. For the purposes of this project, we used Reynolds numbers 20, 100 and 1000 to respectively model the low, medium and high Re cases. We chose to use a mesh size of 100, since this value is large enough to have good precision in the stream plots, but small enough that the runtime is not excessively large. Thus, across these tests we fix the mesh size to a 100x100 grid and keep all other simulation parameters constant for experimental consistency.

2. Runtime Test: Mesh Size

The mesh size determines the desired precision of the system. Of course, this increased precision comes at the expense of computational intensity. It's important to see how the runtimes of different numerical techniques scale in response to different levels of precision in order to gauge how easily it can be adapted to more complicated systems. The runtime of each program execution is measured in terms of the number of seconds it takes for convergence to a tolerance level of 10^{-4}. Most papers don't use mesh sizes finer than 150x150, so we test the runtime scaling up to this level of granularity. Across these tests, we fix the Reynold's number at 100 for experimental consistency.

3. Accuracy Test: Velocity Profile

The first experiment that tests against different Reynold's numbers can be used as a basic eye test for whether the numerical methods are producing reasonable results. However, to further distinguish the extent of accuracy of each method, we will examine the velocity profiles at different Reynolds numbers. Like the first experiment, we fix the mesh size to a 100x100 grid.

RESULTS

1. Robustness Test: Reynolds Number

In this section, we display the velocity stream lines on top of a pressure contour plot for each Reynolds number for all three numerical methods. We experimented with mapping the velocities using stream plots or quiver plots, but decided to display the data using only stream plots because they are more visually intuitive and easier to understand.

For all three Reynolds numbers, our implementation of the finite element method produced incorrect results. As such, we narrow our comparison to our finite difference method vs SIMPLE for the subsequent two experiments.

Low Reynolds Number (20)

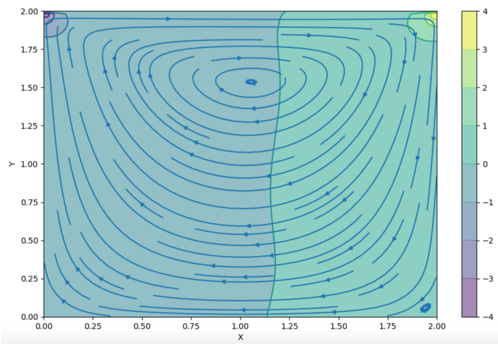

FIG. 3: Finite Difference Method with Re = 20

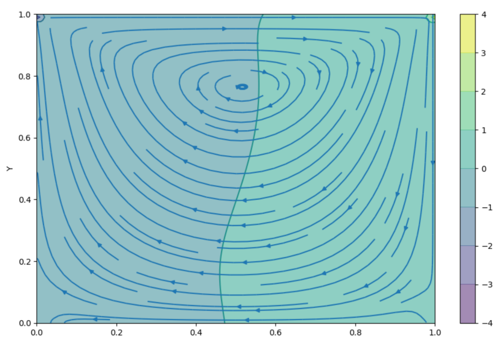

FIG. 4: SIMPLE Method with Re = 20

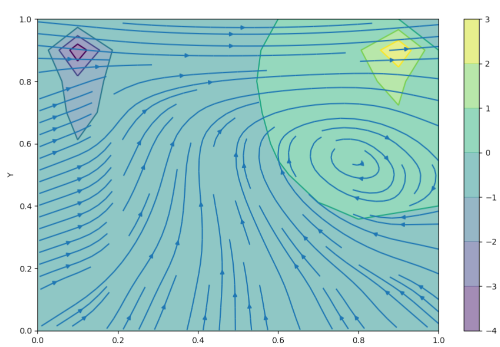

FIG. 5: Finite Element Method with Re = 20

Medium Reynolds Number (100)

We produce an analogous set of graphs for Reynolds number 100. However, we do not include an FEM graph because the velocity field output from the algorithm lost nearly all vorticity and other identifiable qualitative features of cavity flow.

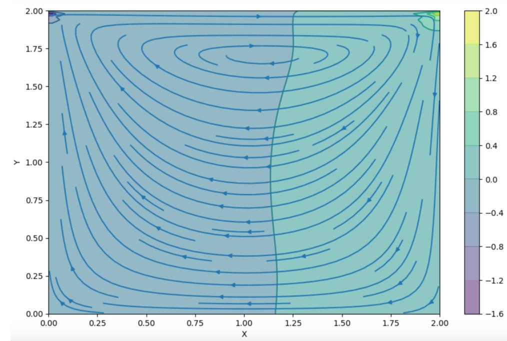

FIG. 6: Finite Difference Method with Re = 100

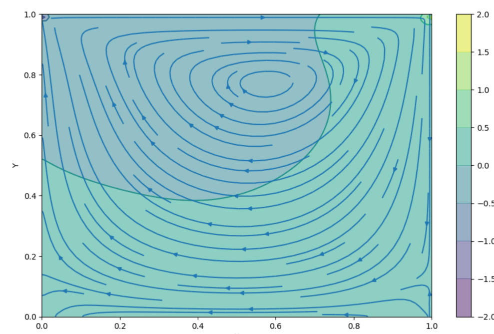

FIG. 7: SIMPLE Method with Re = 100

High Reynolds Number (1000)

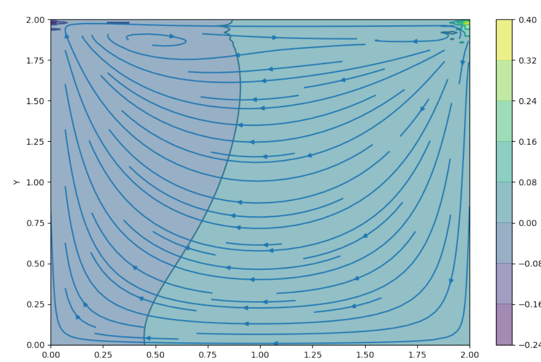

FIG. 8: Finite Difference Method with Re = 1000

For the same reason described previously, we do not include the FEM counterpart in this section. Further, for this high turbulence case, the SIMPLE algorithm does not achieve convergence. Thus, the only pressure and velocity fields displayed here come from the finite difference method.

2. Runtime Test: Mesh Size

The following graphs plot runtime measured in seconds against the dimension of the mesh grid.

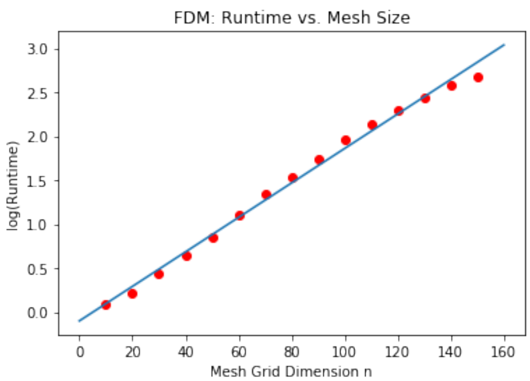

FIG. 9: Finite Difference: Runtime

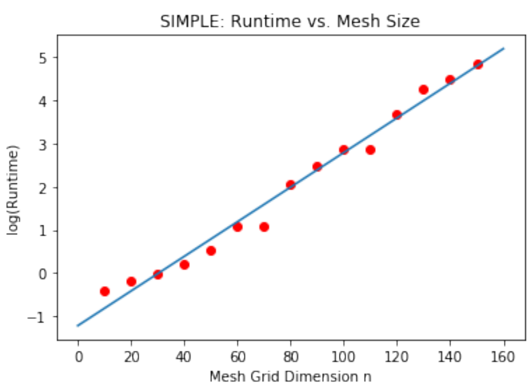

FIG. 10: SIMPLE: Runtime

Accuracy Test: Velocity Profile

We consider the velocity profile as measured across the two central axes. Specifically, we look at how the horizontal velocity u varies along the vertical axis through the cavity's center and how the vertical velocity v varies along the horizontal axis through the cavity's center. These velocity profiles are mapped for several Reynolds numbers.

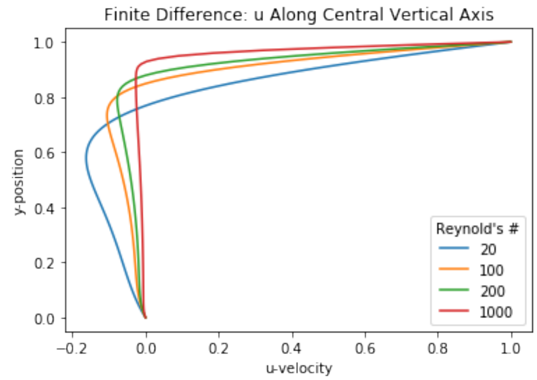

FIG. 11: Finite Difference: u Along Central Vertical Axis

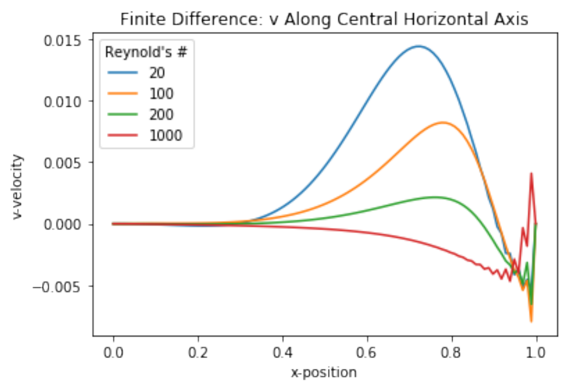

FIG. 12: Finite Difference: v Along Central Horizontal Axis

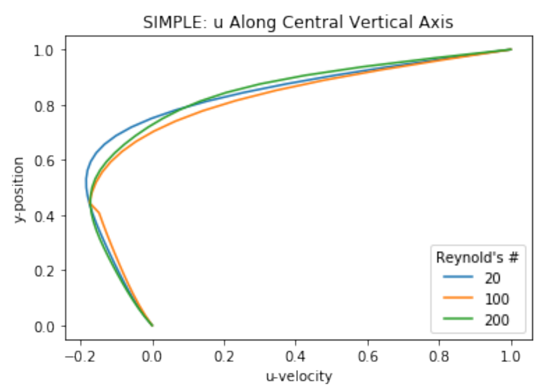

FIG. 13: SIMPLE: u Along Central Vertical Axis

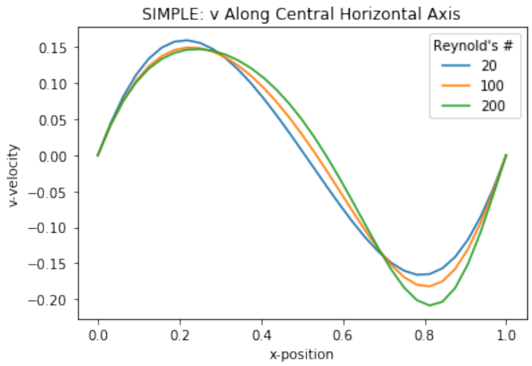

FIG. 14: SIMPLE: v Along Central Horizontal Axis

DISCUSSION AND ANALYSIS

1. Robustness Test: Reynolds number

Finite Difference Method

The pressure and velocity fields from the finite difference method qualitatively match the ones produced in the seminal paper by Ghia et al [3]. In each case, it is easy to discern the two major pressure points or vortices at the bottom corners of the graph and the major vortex at the centre of the cavity. However, the results for Re = 1000 differ significantly from the Ghia paper, which we believe can be attributed to a couple reasons.

First of all, unlike the exact analytic Navier-Stokes equations, the finite difference method introduces discrete spacings in both the x- and y-directions. Hence, the accuracy of the corresponding pressure field is largely dependent on the spacing used. For larger values of Reynolds numbers, the flow within the cavity is more turbulent, so the behaviour is more likely to be less well-defined. In this case, the truncation errors can increase dramatically, explaining the behaviour of the pressure field.

In addition, round-off error may also become prominent in the finite difference method. A round-off error in one calculation will be replicated many times over, leading to growing errors in the calculation of the velocity and pressure. However, this effect is likely to remain the same for all values of Reynolds number, so it does not adequately explain why the higher Reynolds number pressure fields look more different.

SIMPLE Algorithm

From the pressure field graphs, we see that the Re = 20 case for SIMPLE closely resembles the corresponding case for the finite difference method. The major features and vortices are present in both of these methods, which demonstrate that both numerical techniques work well for laminar, relatively unchaotic flow.

For the Re = 100 case, the u and v velocities for the SIMPLE and finite difference methods are very similar. However, in the SIMPLE case, there seems to be counter flow on the bottom lid. Further, the pressure contour occupies the upper left quadrant as opposed to the entire left half of the cavity as in the FDM cavity. It is likely that the assumption inherent in the SIMPLE algorithm where the pressure correction on the velocity is negligible caused minor differences in the pressure fields. Without considering the velocity profiles (Experiment 3), it is not obvious whether the FDM or SIMPLE output is more accurate based on the contour and streamplots alone.

In the same vein, this same assumption, which is primarily valid for smooth laminar flows, quickly breaks down in the range of turbulent flow. As a result, it was not possible to produce a graph for the SIMPLE algorithm with Re = 1000 as the algorithm does not converge. We can then conclude that finite difference methods accomodate a wider range of Reynolds numbers than SIMPLE in the absence of case-by-case parameter tuning.

Finite Element Method

The finite element method gives markedly different pressure fields for Re = 20. The vortex is heavily skewed right and there are much more pronounced pressure contours on either side of the cavity. Due to the large inaccuracies observed using this method, we decided not to analyze this method in depth.

Since FEM methods have more representation power than the other two methods, one might expect such approaches to sufficiently handle complex geometries and shapes. However, it is important to recognize that the quality of FEM solutions depends heavily on the problem at hand as well as the specific simulation parameters. Our experiments evaluate different metrics while keeping most simulation parameters fixed. Thus, the inefficacy of FEM in this experiment does not reflect on the quality of FEM methods in fluid dynamics at large. Rather, these results suggest that more than the other two methods, any approach with finite elements requires extra attention to parameter tuning to maximize the potential of the method.

2. Runtime Test: Mesh Size

Figures 8 and 9 plot the logarithm of the runtime for various mesh grid sizes for the finite difference and SIMPLE methods respectively. In both cases, the log-lin graphs have a linear relationship, which means that the runtime scales exponentially or at least polynomially with the mesh size.

At every mesh size, we also see that the SIMPLE algorithm takes longer to converge. This result was expected given that our SIMPLE algorithm used a staggered grid with several velocity and pressure correction equations that contributed to added computational complexity. However, we cannot extrapolate too heavily from these relative comparisons since runtimes are heavily dependent on the method of implementation. In the case of SIMPLE, further investigation revealed that the biggest computational bottleneck was the use of the Jacobi method to resolve the matrix equations at every iteration. Other more complex iterative methods like Gauss-Seidel are known for being more efficient linear equation solvers.

3. Accuracy Test: Velocity Profiles

Comparing the u velocity distributions for the finite difference (Fig) and SIMPLE methods (Fig) with the corresponding figures in the Ghia paper, we notice that the general shape is the same, but the SIMPLE method more closely matches the Ghia paper. This is likely because the SIMPLE algorithm is more sophisticated than the finite difference method, and so achieves greater precision.

For the v velocity distributions, the SIMPLE algorithm results (Fig 13, 14) very closely match the figures in the Ghia paper, but the finite difference results (Fig 11, 12) are very different, reinforcing the notion that the SIMPLE algorithm produces more reliable results, particularly when laminar flow can be assumed.

CONCLUSIONS AND FUTURE WORK

In this paper, we present three solution methods: finite difference, SIMPLE algorithm with finite volume discretization, and Galerkin method with finite element discretization. When applied to the 2D lid-driven cavity problem, we found that the most simple method, finite difference, had the fastest runtimes and most flexibility across different Reynolds numbers. By contrast, the SIMPLE algorithm did not converge for turbulent flow but otherwise produced velocities more in line with previous literature than FDM. Both FDM and SIMPLE produced pressure and velocity fields that qualitatively exhibited the same features as *Ghia et al* [3]. Meanwhile, FEM lagged behind in all three of our experimental metrics. We attribute this result to the sensitivity of finite elements to simulation parameters.

Future work would center around improvements to SIMPLE and FEM. For SIMPLE, we could consider faster iterative matrix solution methods like Gauss-Seidel or TDMA to improve runtime. We could also look into ways to correct the behavior of SIMPLE to ensure convergence in cases of turbulent flow. While the focus of our experiments was the effectiveness of these methods with minimal tuning, it might also be worth comparing how a highly tuned FEM method would compared against other methods when optimized specifically for the case at hand.

CONTRIBUTIONS AND ACKNOWLEDGEMENTS

All authors dedicated equal amounts of time and effort into the completion of this project. The repository for this project can be found at *https://github.com/davidlin001/flow_sims*. We would like to thank Professor Blas Cabrera and Avikar Periwal for their support and advice throughout the execution of this project.

REFERENCES

[1]. L.A. Barba, and G.F. Forsyth, CFD Python: the 12 steps to Navier-Stokes equations. Journal of Open Source Education, 1(9), 21, https://doi.org/10.21105/jose.00021 (2018).

[2]. H. K. Versteeg and W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, Longman Scientific and Technical, (1995).

[3]. U. Ghia, K.N. Ghia, and C.T. Shin. High-Re solutions for incompressible flow using the NavierStokes equations and a multigrid method Journal of Computational Physics, 48 (1982), pp. 387-411

[4]. N. Rusli, E.H. Kasiman, A. Kueh, B. Hong, A. Yasreen, Y. Mohd, A.Y. Yassin, and A. Norsarahaida. Numerical Computation of a Two-Dimensional Navier-Stokes Equation Using an Improved Finite Difference Method, (2011).

[5]. O. Botella and R. Peyret. Benchmark spectral results on the lid-driven cavity flow Comput Fluids, 27 (4) (1998), pp. 421-433

[6]. M. Li, T. Tang, and B. Fornberg. A Compact Forth-Order Finite Difference Scheme for the Steady Incompressible Navier-Stokes Equations. Int. J. Numer. Methods Fluids, (1995) 20:11371151

[7]. E. Weinan and L. Jian-Guo. Vorticity Boundary Condition and Related Issues for Finite Difference Schemes. J. Comp. Physics, (1996), 124:368382

[8]. T. Sorensen and F. Hastrup-Nielson. A Lid-Driven Cavity Flow, Technical University of Denmark, https://github.com/franzhastrup/MATLAB-Lid-Driven-Cavity-Flow/, (2015).

[9]. M.Sahin and R.G.Owens. A novel fully-implicit finite volume method applied to the lid-driven cavity prob-

lem. Parts I and II Int J Numer Methods Fluids, 42 (1) (2003).

[10]. W. Jiajan. Solution to incompressible Navier-Stokes equations by using finite element method, MSc thesis, University of Texas at Arlington (2010).

[11]. E.G. Tsega and V.K. Katiyar. Finite Element Solution of the Two-dimensional Incompressible Navier-Stokes Equations Using MATLAB, Applications and Applied Mathematics 13 (1) (2018).

[12]. M.J. Sarfi, 2D Lid Driven Cavity Flow using SIMPLE algorithm, MathWorks, (2018).

[13]. S.R. Djeddi et al. Numerical Simulation of Flow around Diamond-Shaped Obstacles at Low to Moderate Reynolds Numbers. American Journal of Applied Mathematics and Statistics, Vol. 1, No. 1, 11-20. doi:10.12691/ajams-1-1-3, (2013).