Solution of the Taylor-Green Vortex Problem Using Artificial Compressibility Method in Generalized Curvilinear Co-ordinates

Sagar Bhatt
Masters Student
Department of Mechanical and Aerospace Engineering,
University at Buffalo

Abstract

The aim of this project was to develop a code for solving the 2D, incompressible Navier-Stokes equations in generalized curvilinear coordinates using artificial compressibility method. A code was developed using MATLAB where three-point, second order finite differencing was used to discretize the convective and viscous fluxes in conjunction with scalar, fourth-difference, third-order artificial dissipation for stability. Dual time-stepping with a second-order backward scheme in real time and four stage Runge-Kutta time stepping was used for pseudo time. The code was applied to an unsteady, incompressible flow field known as Taylor-Green vortex. The solution was marched in time till t=0.1s for a Reynolds number of 100. Periodic boundary conditions were used at the boundaries of the domain ($x, y = (0, 2pi)$). The results were plotted and compared against the analytical solution of Taylor-Green vortex.

I. Introduction

Taylor-Green vortex is an unsteady flow of a decaying vortex which has an exact closed form solution of the incompressible NavierStokes equations. It is named after the British physicist and mathematician Geoffrey Ingram Taylor and his collaborator A. E. Green[11]. They solved for a special class of vortices to illustrate grinding down of large eddies into smaller ones. They defined a special flow field and traced the subsequent motion of the viscous incompressible fluid[9]. Solution to Taylor-Green vortex is one of the few known analytical solutions of Navier-Stokes equations hence they can be used for testing and validating spatial and temporal accuracy of Navier-Stokes solver algorithms. Various researchers, like Don *et al.*[8] and DeBonis [4], have used solution to Taylor-Green vortex to test numerical schemes.

Don *et al.* performed a numerical convergence study of various numerical schemes using Taylor-Green vortices. They used the Fourier collocation method with the sharp-cutoff filter (F-SF), the Fourier collocation method with the exponential filter (F-EF) and fifth-order weighted essentially non-oscillatory (WENO-5) method to examine the consequences of filtering in the numerical simulation of the three-dimensional evolution of nearly-incompressible, inviscid Taylor-Green vortex flow. One of their conclusions was that Taylor-Green vortex is useful to compare the predictions of at least two numerical methods with different algorithmic foundations in order to corroborate the conclusions from the numerical solutions when the analytical solution is not known.

DeBonis developed a computational fluid dynamics code that solves the compressible Navier-Stokes equations and applied it to the Taylor-Green vortex problem to examine the codes ability to accurately simulate the vortex decay and subsequent turbulence. The code, WRLES (Wave Resolving Large-Eddy Simulation), used explicit central-differencing to compute the spatial deriqvatives and explicit Low Dispersion Runge-Kutta methods for the temporal discretization.[4]

Solution to Taylor-Green vortex can therefore be considered a benchmark to examine a code's accuracy in solving incompressible Navier-Stokes equations. This project essentially validates the accuracy of the code developed for 2D incompressible Navier-Stokes equations in generalised curvilinear coordinates using artificial compressibility by applying the code to the flow field of Taylor-Green vortex and comparing the results with the analytical solutions derived by Taylor and Green in their original work[9].

The analytical solution for Taylor-Green vortex is given by $(0 \leq x, y \leq pi))$:

$$u(x, y, t) = -e^{-2t} \cos(x) \sin(y)$$
$$v(x, y, t) = e^{-2t} \sin(x) \cos(y)$$

$P(x,y,t) = -\{[\cos(2x) + \cos(2y)]e^{-4t}]\}/4$

Non-dimensionalized Navier-Stokes equations in Cartesian co-ordinates are given by:

$$\Gamma\frac{\partial Q}{\partial t} + \frac{\partial E^1}{\partial x} + \frac{\partial E^2}{\partial y} - \frac{\partial E_v^1}{\partial x} - \frac{\partial E_v^2}{\partial y} = 0$$

where,

$\Gamma = diag(0,1,1)$,

$$Q = \begin{bmatrix} P \\ u \\ v \end{bmatrix},\ E^1 = \begin{bmatrix} u \\ uu + P \\ uv \end{bmatrix},$$

$$E^2 = \begin{bmatrix} v \\ uv \\ vv + P \end{bmatrix},\ E_v^1 = \frac{1}{Re}\begin{bmatrix} 0 \\ \partial u\ \partial x \\ \partial v\ \partial x \end{bmatrix},$$

$$E_v^2 = \frac{1}{Re}\begin{bmatrix} 0 \\ \partial u\ \partial y \\ \partial v\ \partial y \end{bmatrix}$$

The numerical solution of these equations presents major difficulties, due in part to the special role of the pressure in the equations and in part to the large amount of computer time which such solution usually requires, making it necessary to devise finite-difference schemes which allow efficient computation. In two-dimensional problems the pressure can be eliminated from the equations using the stream function and vorticity, thus avoiding one of the difficulties[2]. Chorin developed a method called "Method of Artificial Compressibilty"[2] and since then has been improved by numerous researchers for solving unsteady Navier-Stokes equations. This method is,typically, combined with dual time stepping where an iteration in dual time is implemented for each time step in real time. This method assures a convergence of solution for incompressible unsteady problem[7]. Chorin himself implemented the method using DuFort-Frankel scheme on a standard grid in the 2D case of thermal convection in a fluid layer heated from below[2].

II. Method of Solution

Firstly, the generalized coordinate transformation was achieved for curvilinear coordinate system $(x,y) \rightarrow (\xi,\eta)$ where $\xi = \xi(x,y), \eta = \eta(x,y)$. For this purpose first $x_\xi, x_\eta, y_\xi and y_\eta$ were calculated. This was achieved by using forward differencing of x and y and taking $\Delta\xi, \Delta\eta = 1$. For example:

$$\frac{\partial x}{\partial \xi} = \frac{x_{i+1} - x_{i-1}}{\Delta\xi}$$

Using these values Jacobian was calculated in the following manner:

$$G = det\begin{bmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{bmatrix} = x_\xi y_\eta - x_\eta y_\xi$$

Jacobian: $J = \frac{1}{G}$

Then metrics of transformation were calculated using the following:
$\xi_x = Jy_\eta,\ \xi_y = -Jx_\eta,\ \eta_x = -Jy_\xi,\ \eta_y = Jx_\xi$

Using these, metric tensor was calculated:

$$g^{ij} = \begin{bmatrix} g^{11} & g^{12} \\ g^{12} & g^{22} \end{bmatrix} = \begin{bmatrix} (\xi_x)^2 + (\xi_y)^2 & \xi_x\eta_x + \xi_y\eta_y \\ \xi_x\eta_x + \xi_y\eta_y & (\xi_x)^2 + (\xi_y)^2 \end{bmatrix}$$

Now, covariant velocities were calculated using:
$U = u\xi_x + v\xi_y,\ V = u\eta_x + v\eta_y$

Hence now we can do a partial transformation of non-dimensionalised Navier-Stokes equation in curvilinear co-ordinates:

$$\frac{1}{J}\Gamma\frac{\partial Q}{\partial t} + \frac{\partial E^{*1}}{\partial \xi} + \frac{\partial E^{*2}}{\partial \eta} - \frac{\partial E_v^{*1}}{\partial \xi} - \frac{\partial E_v^{*2}}{\partial \eta} = 0$$

where,

$\Gamma = diag(0,1,1)$,

$$E^{*1} = \frac{1}{J}\begin{bmatrix} U \\ uU + P\xi_x \\ vU + P\xi_y \end{bmatrix},$$

$$E^{*2} = \frac{1}{J}\begin{bmatrix} V \\ uV + P\eta_x \\ vV + P\eta_y \end{bmatrix},$$

$$E_v^{*1} = \frac{1}{Re}\frac{1}{J}\begin{bmatrix} 0 \\ g^{11}\dfrac{\partial u}{\partial \xi} + g^{12}\dfrac{\partial u}{\partial \eta} \\ g^{11}\dfrac{\partial v}{\partial \xi} + g^{12}\dfrac{\partial v}{\partial \eta} \end{bmatrix},$$

$$E_v^{*2} = \frac{1}{Re}\frac{1}{J}\begin{bmatrix} 0 \\ g^{12}\dfrac{\partial u}{\partial \xi} + g^{22}\dfrac{\partial u}{\partial \eta} \\ g^{12}\dfrac{\partial v}{\partial \xi} + g^{22}\dfrac{\partial v}{\partial \eta} \end{bmatrix}$$

The viscous and convective fluxes were then discretized using three-point, second order accurate finite differencing. Central differencing satisfies these requirements. The finite difference equations for this method is given by:

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2\Delta x}$$

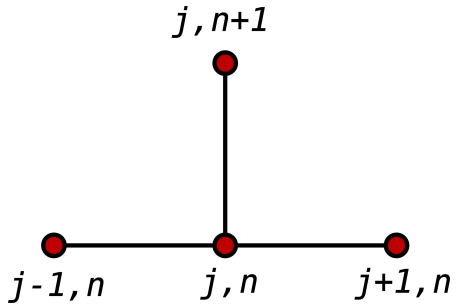

Figure 1: *Stencil of three point central differencing*

Using this discretization, the convective and viscous fluxes were calculated and then right hand side of the Navier-Stokes equations were calculated at t=0. Artificial dissipation was introduced in the system to get corrected values of the calculated RHS.

Boundary Conditions: Periodic boundary conditions were applied on the boundaries. This results in boundary values at N=1 and N=N to be same for both x an y axes. Using this boundary condition, we can now compute values at boundary nodes by employing ghost nodes.

Artificial Dissipation: To calculate artificial dissipation, first the Jacobian matrix is calculated as follows:

$$A^j = \frac{1}{J}\begin{bmatrix} 0 & \xi_x^j & \xi_y^j \\ \xi_x^j & U^j + u\xi_x^j & u\xi_y^j \\ \xi_y^j & v\xi_x^j & U^j + v\xi_y^j \end{bmatrix},$$

$\xi^1 = \xi,\ \xi^2 = \eta\ , and$

$U^1 = U\ , U^2 = V$

Now, the spectral radius was calculated using:

$\rho(A^j) = \frac{1}{J}(|U^j| + \sqrt{(U^j)^2 + g^{jj}})$

Now, the dissipation was calculated using the following finite difference equations:

$Diss_{(i,j)} = \widetilde{\delta}_\xi D^1_{(i,j)} + \widetilde{\delta}_\eta D^2_{(i,j)} = (D^1_{(i+1/2,j)} -$

$D^1_{(i-1/2,j)}) + (D^2_{(i,j+1/2)} - D^2_{(i,j-1/2)})$

$D^1_{(i+1/2,j)} = \epsilon\rho(A^1)(Q_{(i+2,j)} - 3Q_{(i+1,j)} +$

$3Q_{(i,j)} - Q_{(i-1,j)})$

Where ϵ is a small number which controls the dissipation. RHS with artificial dissipation was then used in time marching inorder to calculate pressure, x-velocity and y-velocity of the the flow field. For time marching, dual time stepping technique was used. The Artificial compressibility equations, dual time-stepping are given by:

$$\frac{\partial Q}{\partial \tau} = -\Gamma\frac{\partial Q}{\partial t} - J\left(\frac{\partial E^{*1}}{\partial \xi} + \frac{\partial E^{*2}}{\partial \eta} - \frac{\partial E_v^{*1}}{\partial \xi} - \frac{\partial E_v^{*2}}{\partial \eta} + Diss\right) = RHS$$

Discretization of real time component of dual time stepping was discretized using second order backward differencing. The finite difference equation for this scheme is given by:

$$\frac{\partial Q}{\partial t} = \frac{3Q^{n+1} - 4Q^n + Q^{n-1}}{2\Delta t}$$

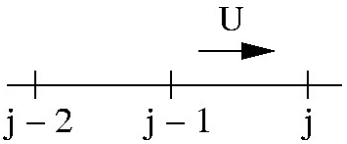

Figure 2: *Stencil of second order backward differencing*

Pseudo time marching was achieved by implementing fourth stage Runge-Kutta scheme, the finite difference equation for which is given by:

$$Q^{k+1,1} = Q^k + \frac{1}{4}\Delta\tau RHS$$

$$Q^{k+1,2} = Q^k + \frac{1}{3}\Delta\tau RHS^1$$

$$Q^{k+1,3} = Q^k + \frac{1}{2}\Delta\tau RHS^2$$

$$Q^{k+1} = Q^k + \Delta\tau RHS^3$$

For each real time step, the solution was marched in pseudo time until it reached solution. The program was executed for different grid sizes, stretching ratios and time steps to verify accuracy of the code. The results are discussed later while code itself could be found in Appendix.
The error between the exact solution and the computed solution was computed using the following norm:

$$Err = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[(\widetilde{u}(x_i,y_j) - u_{(i,j)})^2 + (\widetilde{v}(x_i,y_j) - v_{(i,j)})^2 + (\widetilde{P}(x_i,y_j) - P_{(i,j)})^2\right]$$

III. Grid generation

For the domain size $x, y = (0, 2pi)$, both uniform and stretched grids were generated. For uniform grid, a constant spacing of Δx was chosen. Figure 3 shows a uniform grid for the given domain. The solution was calculated for successively finer grids for a uniform grid and error was plotted in a log-log plot. The results are discussed in the next section.

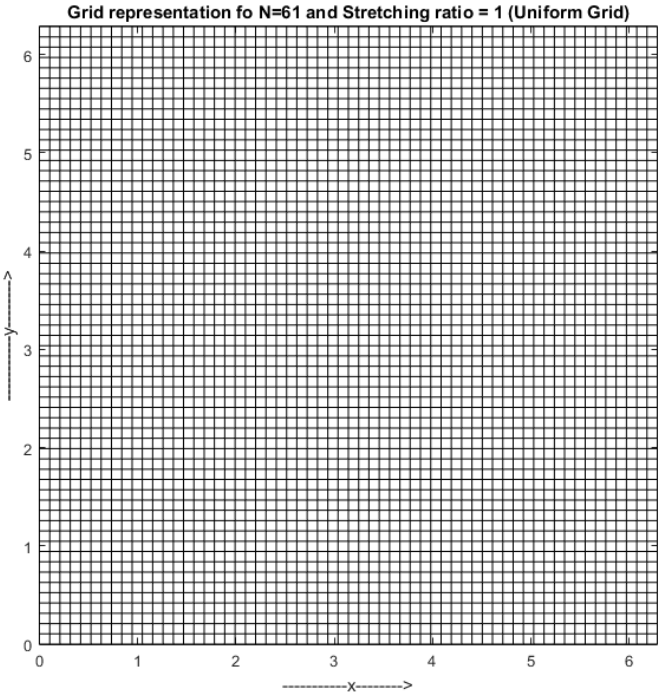

Figure 3: *Uniform Grid for number of grid points, N=61 on each axis*

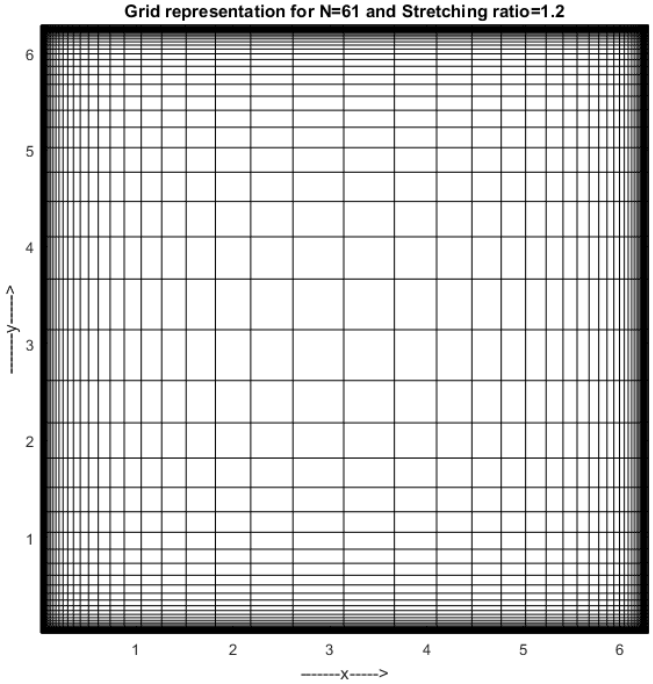

Figure 4: *Stretched Grid for number of grid points, N=61 on each axis and stretching ratio=1.2*

For stretched grids, grid was made finer near the four boundaries. To implement such a grid, a geometric series stretching was performed on one quadrant of the domain i.e. $x, y = (0, pi)$ then the resulting grid was mirrored first in x-direction then in y-direction. Figure 4 shows one such stretched grid with stretching ratio of 1.2 for a grid size NXN where, N=61. The solution was computed on grids with increasing stretching ratio and the error was computed and plotted with respect to increasing stretching ratios. The results are discussed in the next section.

IV. Results

The results were obtained by marching time till t=0.1s by using dual-time stepping method. The solutions obtained thus are plotted here and compared agains analytical solutions. Figure 5 Shows the analytical solution for pressure and Fig. 6 shows the same contour plot for the computed solution after convergence was achieved using dual time stepping.

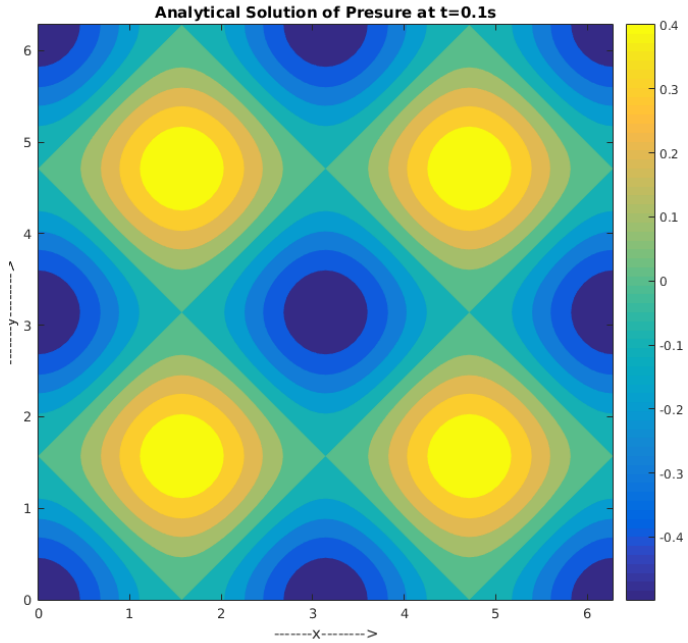

Figure 5: *Analytical Soultion for Pressure at t=0.1s*

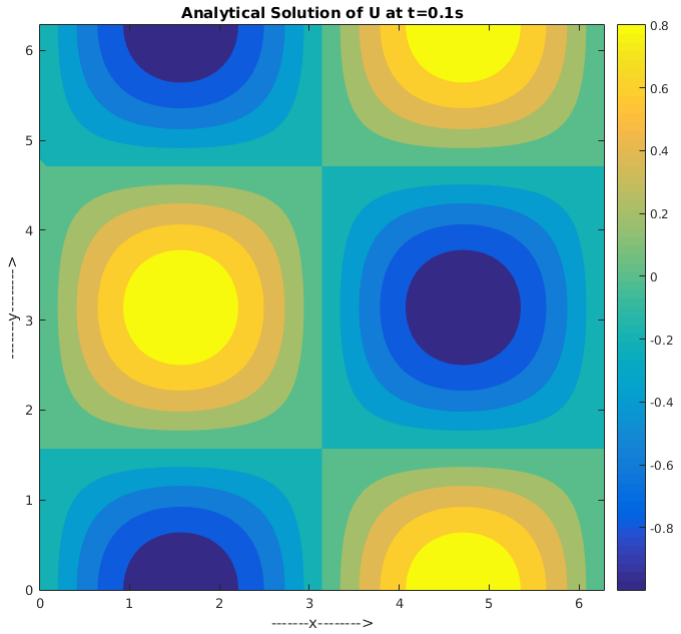

Figure 7: *Analytical Soultion for x-velocity at t=0.1s*

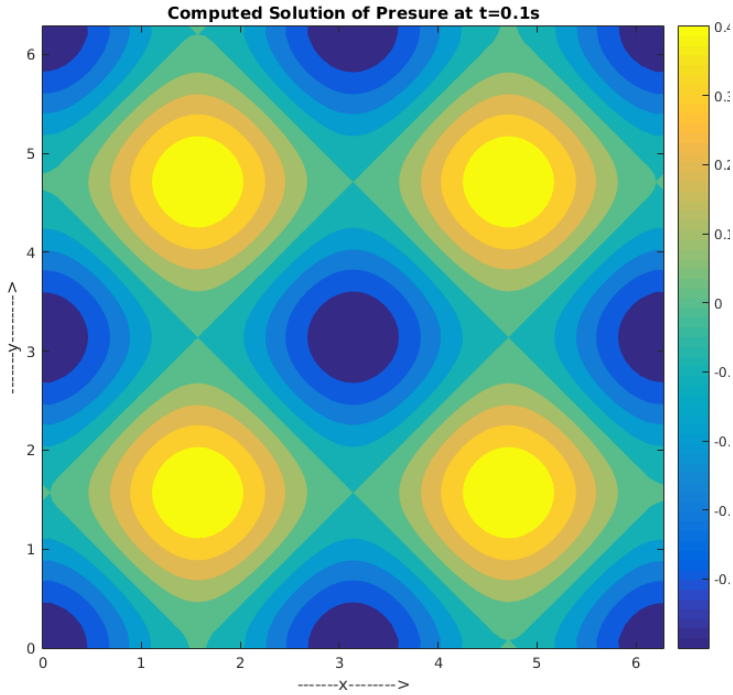

Figure 6: *Computed Soultion for Pressure at t=0.1s*

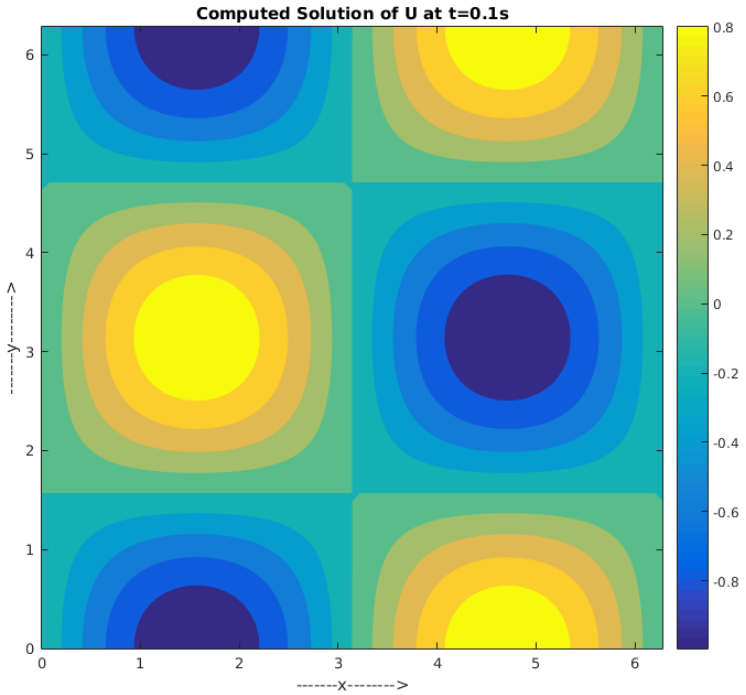

Figure 8: *Computed Soultion for x-velocity at t=0.1s*

Figure 7 shows the analytical solution at t=0.1s for the x-velocity. Figure 8 shows the corresponging contour of the computed solution.

Figure 9 shows the analytical solution at t=0.1s for the x-velocity. Figure 10 shows the corresponging contour of the computed solution.

Figure 11 shows the spatial accuracy of the code. In Fig. 11 we can see that error reduces as the number of grid points reduce. As the number of grid points cross 41, the change in error is of the order 0f 10^{-7} which is a very small quantity. Thereafter, this change in error decays even further until there is hardly any change in it. Considering the order of change in error, we can conclude that the code converges to a grid independence solution for higher values of N than N=61 at t=0.1s. The code, however, was executed till N=81 in order to corroborate that the solution has in fact attained grid independence.The order of spatial accuracy of the code is give by the slope of the log-log plot presented in Fig. 10.:

$$Slope = order \approx \frac{-11.4 - (-12)}{4.4 - 4.1} = 2$$

Hence the code has been determined to be second order accurate in space.

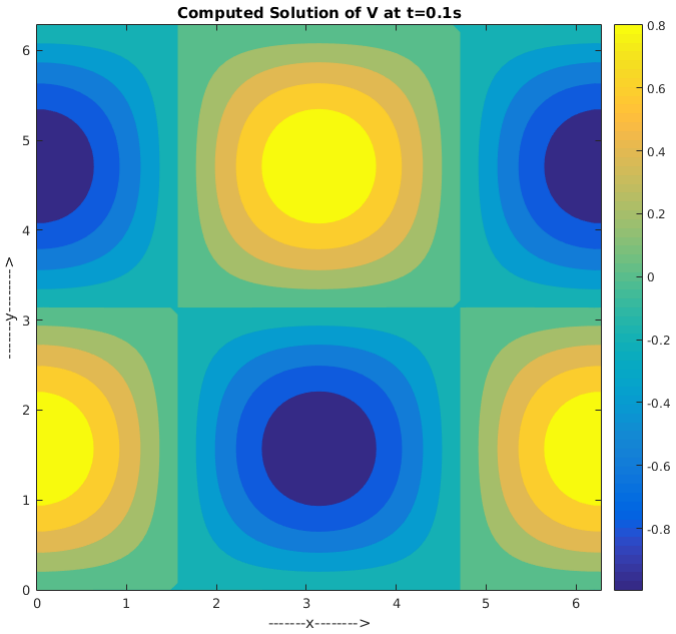

Figure 10: *Computed Soultion for y-velocity at t=0.1s*

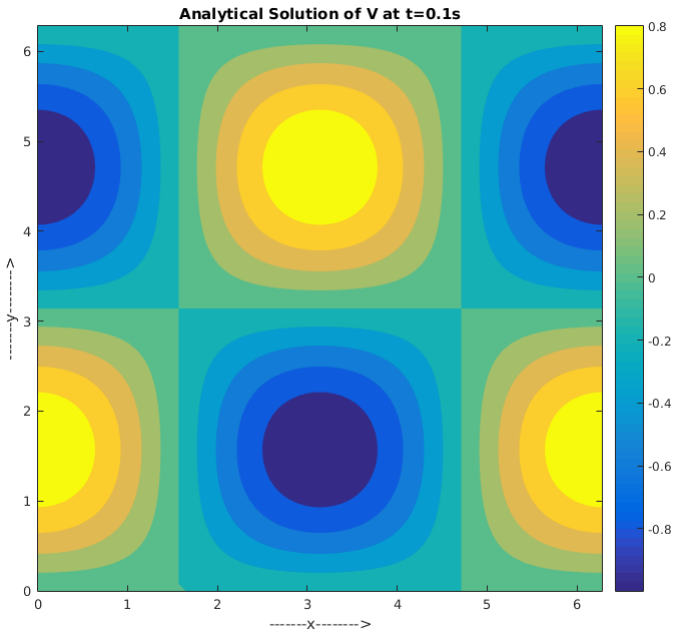

Figure 9: *Analytical Soultion for y-velocity at t=0.1s*

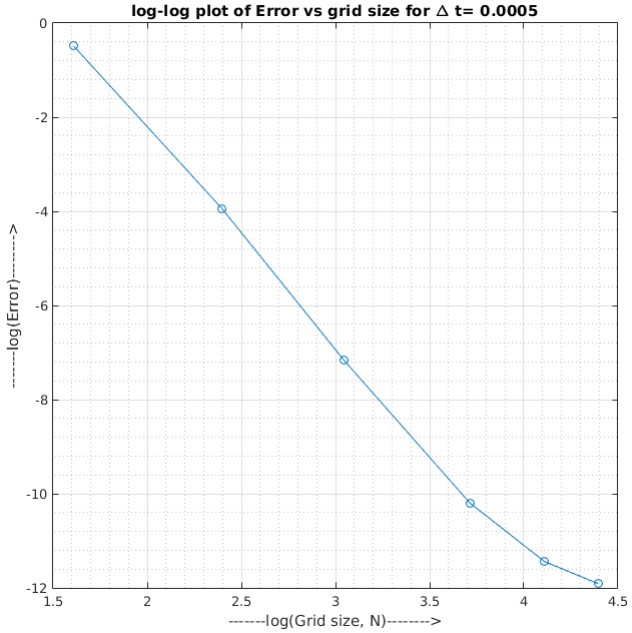

Figure 11: *log-log plot of error vs number of grid points for grid points, N=5 to 81 in each axis*

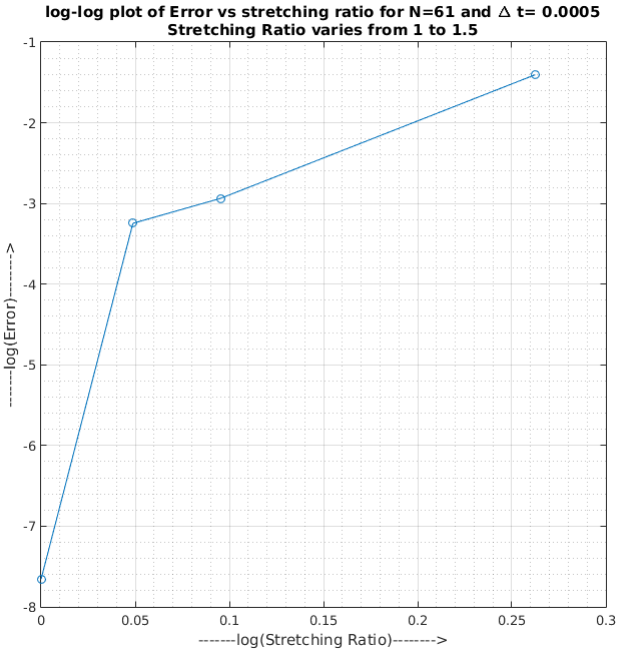

Figure 12: *log-log plot of error vs stretching ratio for N=61 and* $\Delta t = 0.0005s$ *and stretching ratio from 1 to 1.5*

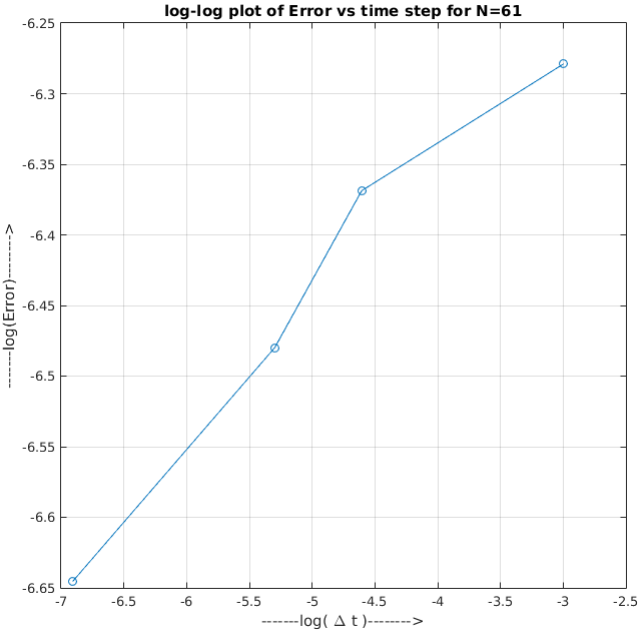

Figure 13: *log-log plot of error vs time-step size for N=61 and* $\Delta t = 0.0005s$ *to* $0.05s$

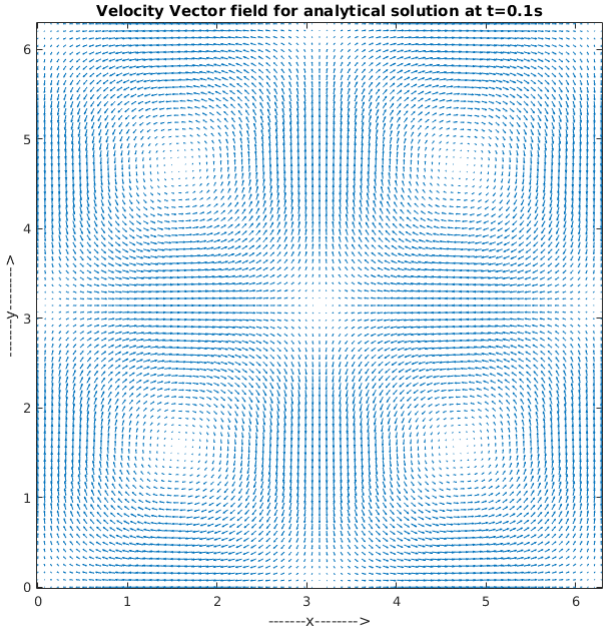

Figure 14: *Analytical Velocity vector field at t=0.1s*

Grid independence shown in Fig. 11 employs a uniform grid with a constant Δx. The accuracy of the code has also been investigated for a stretched grid with finer grid near the boundaries. The stretching ratio is varied from 1 to 1.5 and a log-log plot (Fig. 12) is obtained for Error vs Stretching ratio. As we can observe in the plot, as the stretching ratio increase the error increases. This is due to the fact that as stretching becomes too big, although the grid at boundary is extremely fine, the central nodes are very sparse resulting in higher error. Figure 13 presents temporal accuracy of the code. Δt was varied while keeping everything else constant. The error was computed after complete time marching. In each real time step, convergence takes place for pseudo time. Δt was varied from 0.0005s to 0.05s for a grid size, N=61. As we we can see that as Δt increases, the error increases which was an expected result. The order of temporal accuracy is give by:

$$Slope = order = \frac{-6.275 - (-6.665)}{-3 - (-6)} \approx 0.1$$

The accuracy in time does not change a lot with Δt because of the extremely small time steps to reach convergence for net time marching of 0.1s.

Figure 14 and 15 represent the velocity vector field for analytical and computed solutions.

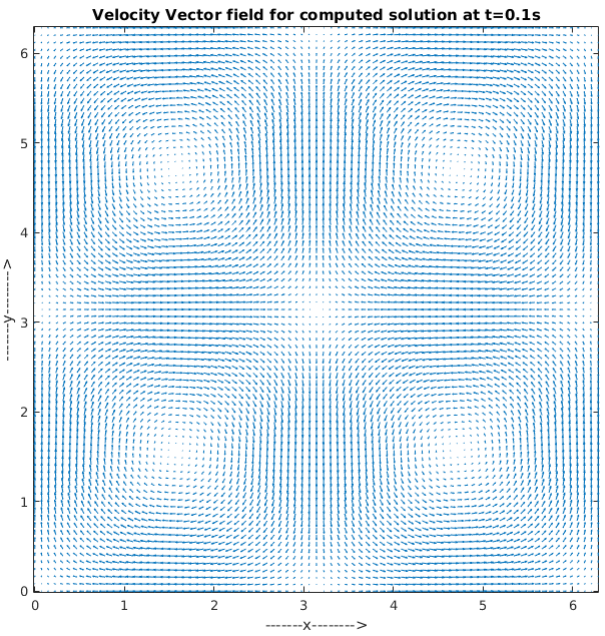

Figure 15: *Computed Velocity vector field at t=0.1s*

V. Summary

A numerical study of Taylor-Green vortex problem was undertaken using Artificial compressibility method where dual time stepping is employed to for time marching. A code was developed for solving the 2D, incoompressible Navier-Stokes equations in the generalized curvilinear coordinate systems. A three-point , second order finite differencing was used to discretize the convective and viscous fluxes in conjunction with scalar, fourth-difference, third-order artificial dissipation for stability. Dual time-stepping with a second-order backward scheme in real time and four stage Runge-Kutta time stepping was used for pseudo time. The solution was marched in time till t=0.1s for a Reynolds number of 100. Periodic boundary conditions were used at the boundaries of the domain $(x, y = (0, 2pi))$. The results were plotted and compared against the analytical solution of Taylor-Green vortex. The spatial accuracy was determined for the solution using both uniform and stretched grids. The temporal accuracy was determined for a constant spatial resolution.

VI. Conclusions

The study investigated the evolution of 2D incoompressible Navier-Stokes equations in Taylor-Green vortex flow field and compared the results with solutions to unsteady, incompressible Taylor-Green vortex. As the grid resolution increased, the results showed remarkable improvement. The solution was almost completely converged till N=61. The solution was determined to be second order accurate spatially.
Temporal accuracy was also explored for the solution and was found to be ≈ 0.1. Such low temporal accuracy was attributed to the fact that the error does not change a lot because of the extremely small time steps that were used to march till t=0.1s. The equations were also solved for a non-uniform grid by stretching the grid near the four boundary conditions. On investigating the error of the non-uniform grid, it was found that the error increases as the stretching ratio increases. This phenomenon can be attributed to the fact that although near the boundary the grid is very fine but this arrangement also results the central nodes being sparse which resulted in a very coarse grid. This lead to an increase in net error.

References

[1] John David Anderson and J Wendt. *Computational fluid dynamics*, volume 206. Springer, 1995.

[2] Alexandre Joel Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of computational physics*, 2(1):12–26, 1967.

[3] Alexandre Joel Chorin. Numerical solution of the navier-stokes equations. *Mathematics of computation*, 22(104):745–762, 1968.

[4] James DeBonis. Solutions of the taylor-green vortex problem using high-resolution explicit finite difference methods. *AIAA paper*, 382:2013, 2013.

[5] Boyce E Griffith. An accurate and efficient method for the incompressible navier–stokes equations using the projection method as a preconditioner. *Journal of Computational Physics*, 228(20):7565–7595, 2009.

[6] Klaus A Hoffmann and Steve T Chiang. {*Computational fluid dynamics, Vol. 1*}. 2000.

[7] Hans Petter Langtangen, Kent-Andre Mardal, and Ragnar Winther. Numerical methods for incompressible viscous flow. *Advances in Water Resources*, 25(8):1125–1146, 2002.

[8] Chi-Wang Shu, Wai-Sun Don, David Gottlieb, Oleg Schilling, and Leland Jameson. Numerical convergence study of nearly incompressible, inviscid taylor–green vortex flow. *Journal of Scientific Computing*, 24(1):1–27, 2005.

[9] GI Taylor and AE Green. Mechanism of production of small eddies from large ones. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 158(895):499–521, 1937.

[10] Socrates Vrahliotis, Theodora Pappou, and Sokrates Tsangaris. Artificial compressibility 3-d navier-stokes solver for unsteady incompressible flows with hybrid grids. *Engineering Applications of Computational Fluid Mechanics*, 6(2):248–270, 2012.

[11] Wikipedia. Taylorgreen vortex — wikipedia, the free encyclopedia, 2016. [Online; accessed 30-April-2016].