AM5630: INTRODUCTION TO COMPUTATIONAL FLUID DYNAMICS

-XXXX

-TEJASWIN.P, ME12B069

-XXXX

-XXXX

CONTENTS

This presentation is mainly focussed on the Finite Difference Scheme of discretising the Navier-Stokes Equation in a two dimensional plane, using stream function- vorticity approach for a lid driven cavity . It is broadly classified into:

- General Introduction
- Problem Definition and Assumptions
- Difference Schemes used and Discretisation
- Boundary Conditions
- Algorithm for solution
- Solvers for Poisson Equation
- Results and Discussions

GENERAL INTRODUCTION – THEORY

THE INCOMPRESSIBLE NAVIER-STOKES EQUATION

The incompressible Navier Stokes Equation, obtained from the Reynolds Transport Theorem is given by

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

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{\mu}{\rho}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \frac{\mu}{\rho}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)$$

VORTICITY

- Fundamentally, vorticity is a measure of the amount of rotation in a flow.
- The vorticity transport equation talks about the net deformation to the fluid elements.
- This also allows us to get rid of the troublesome pressure in the N-S equations.
- The rotation of the fluid element about the z-axis is defined to be the average of these two angular velocities (positive counterclockwise):

$$\omega_z = \frac{1}{2}\left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)$$

- Similarly, the same is done for x and y directions as well.

VORTICITY

- The rotations can be combined to form a rotation vector:

$$\boldsymbol{\omega} = \omega_x \mathbf{i} + \omega_y \mathbf{j} + \omega_z \mathbf{k}$$

- The vorticity vector is defined as twice the rotation and is also equal to the curl of the velocity.

$$\zeta = 2\omega = \nabla \times \mathbf{V}$$

STREAM FUNCTION-VORTICITY FORMULATION

- Cross differentiating the momentum equations and subtracting

$$\frac{\partial \zeta}{\partial t} + u\frac{\partial \zeta}{\partial x} + v\frac{\partial \zeta}{\partial y} = \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 \zeta}{\partial x^2} + \frac{\partial^2 \zeta}{\partial y^2}\right)$$

- By definition of stream function, to eliminate the continuity equation:

$$u = \frac{\partial \psi}{\partial y}; v = -\frac{\partial \psi}{\partial x}$$

- Hence

$$\zeta = -\left(\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2}\right)$$

$$\frac{\partial \zeta}{\partial t} + \frac{\partial \psi}{\partial y}\frac{\partial \zeta}{\partial x} - \frac{\partial \psi}{\partial x}\frac{\partial \zeta}{\partial y} = \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 \zeta}{\partial x^2} + \frac{\partial^2 \zeta}{\partial y^2}\right)$$

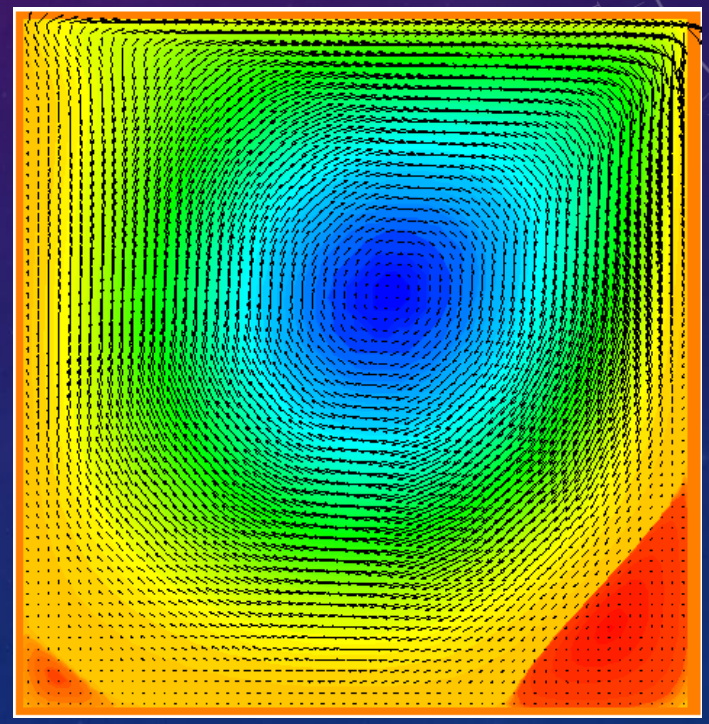
PRESSURE EQUATION

- The pressure can be recovered by taking the divergence of the momentum equation and using the velocities computed from stream function, Y:

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = 2\left[\left(\frac{\partial^2 \psi}{\partial x^2}\right)\left(\frac{\partial^2 \psi}{\partial y^2}\right) - \left(\frac{\partial^2 \psi}{\partial x \partial y}\right)^2\right]$$

LID DRIVEN CAVITY PROBLEM

- The driven cavity problem is a classical problem.
- Incompressible viscous flow driven by the uniform translation of the lid.
- The vorticity-stream function method is used to solve the driven cavity problem.



PROBLEM DEFINITION AND ASSUMPTIONS

The lid driven cavity is a classical fluid dynamics benchmark problem. The driving velocity is taken as 1,without any loss in generality.

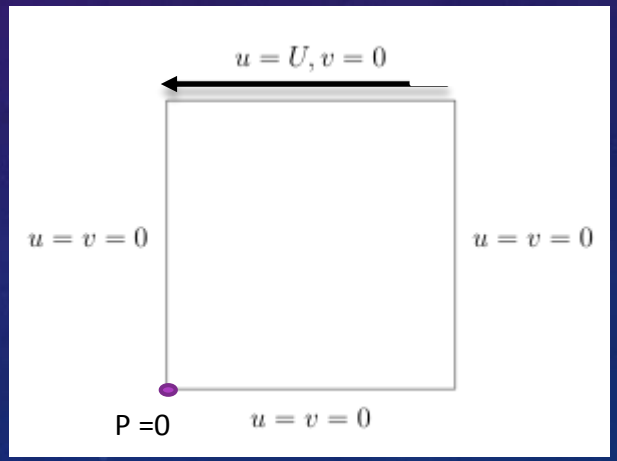
APPLICATION TO LID DRIVEN CAVITY

- On non dimensionalisation of the equations, we get

$$x' = x/L; y' = y/L; t' = tU/L$$

$$u' = u/U; v' = v/U; p' = p/\rho U^2$$

$$\zeta' = \zeta L/U; \psi' = \psi/UL$$



EQUATIONS USED FOR VORTICITY TRANSPORT ALGORITHM

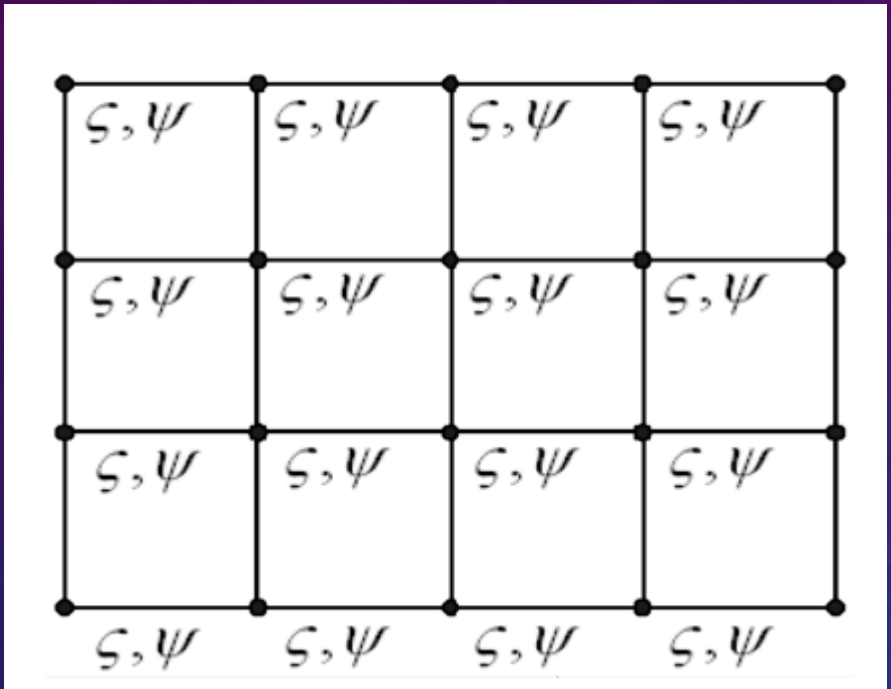
$$\frac{\partial \zeta}{\partial t} + u\frac{\partial \zeta}{\partial x} + v\frac{\partial \zeta}{\partial y} = \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 \zeta}{\partial x^2} + \frac{\partial^2 \zeta}{\partial y^2}\right)$$

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\zeta$$

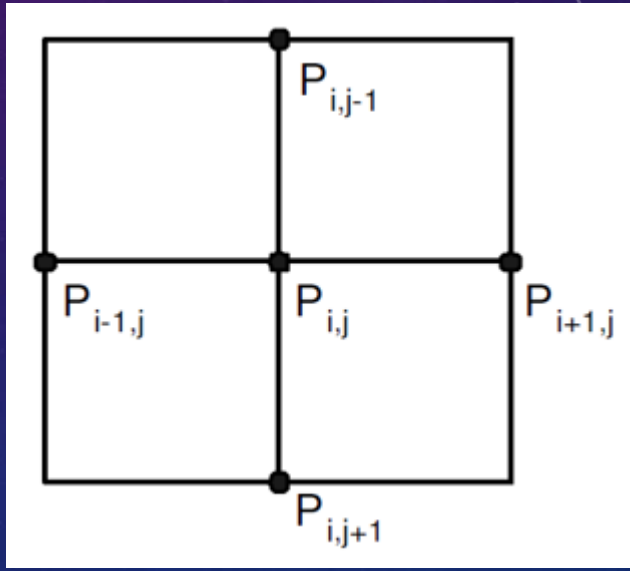
$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = 2\left[\left(\frac{\partial^2 \psi}{\partial x^2}\right)\left(\frac{\partial^2 \psi}{\partial y^2}\right) - \left(\frac{\partial^2 \psi}{\partial x \partial y}\right)^2\right]$$

$$\mathrm{Re} = \frac{UL}{\nu}$$

VORTICITY TRANSPORT ALGORITHM : GRID SYSTEM



For Stream Function – Vorticity Approach



For Pressure Poisson Equation

DEFINITION AND ASSUMPTIONS

For the non dimensionalized benchmark problem, the following are the equations and assumptions :

- Reynolds Number : Variable (100,500,1000)
- Grid size : Variable (10x10,20x20,25x25,30x30,35x35)
- Velocity of lid = 1 m/s
- Dimensions of lid = 1m x 1m
- Properties of fluids set based on the above parameters
- Non dimensionalised pressure at a point on the lower boundary is taken as 1

DIFFERENCE SCHEMES AND DISCRETIZATION

Throughout the entirety of the program , a second order accurate scheme (even the BC) is presented – using central difference schemes throughout, except for the convective term

THE VORTICITY TRANSPORT ALGORITHM – DISCRETIZATION

$$\frac{\zeta_{i,j}^{n+1} - \zeta_{i,j}^{n}}{\Delta t} + \frac{u_{i+1,j}^{n}\zeta_{i+1,j}^{n} - u_{i-1,j}^{n}\zeta_{i-1,j}^{n}}{2\Delta x} + \frac{v_{i,j+1}^{n}\zeta_{i,j+1}^{n} - v_{i,j-1}^{n}\zeta_{i,j-1}^{n}}{2\Delta y}$$

$$= \frac{1}{\mathrm{Re}}\left(\frac{\zeta_{i+1,j}^{n} - 2\zeta_{i,j}^{n} + \zeta_{i-1,j}^{n}}{(\Delta x)^2} + \frac{\zeta_{i,j+1}^{n} - 2\zeta_{i,j}^{n} + \zeta_{i,j-1}^{n}}{(\Delta y)^2} \right)$$

$$\frac{\psi_{i+1,j}^{n} - 2\psi_{i,j}^{n} + \psi_{i-1,j}^{n}}{(\Delta x)^2} + \frac{\psi_{i,j+1}^{n} - 2\psi_{i,j}^{n} + \psi_{i,j-1}^{n}}{(\Delta y)^2} = -\zeta_{i,j}^{n}$$

PRESSURE POISSON EQUATION DISCRETISATION

$$\frac{p^{n}_{i+1,j} - 2\,p^{n}_{i,j} + p^{n}_{i-1,j}}{(\Delta x)^2} + \frac{p^{n}_{i,j+1} - 2\,p^{n}_{i,j} + p^{n}_{i,j-1}}{(\Delta y)^2}$$

$$= 2\left[\left(\frac{\psi^{n}_{i+1,j} - 2\psi^{n}_{i,j} + \psi^{n}_{i-1,j}}{(\Delta x)^2}\right)\left(\frac{\psi^{n}_{i,j+1} - 2\psi^{n}_{i,j} + \psi^{n}_{i,j-1}}{(\Delta y)^2}\right) - \left(\frac{\psi^{n}_{i+1,j+1} - \psi^{n}_{i+1,j-1} - \psi^{n}_{i-1,j+1} + \psi^{n}_{i-1,j-1}}{4\,\Delta x \Delta y}\right)^2\right]$$

BOUNDARY CONDITIONS

Boundary conditions are important for convergence and are an essential part of any solution. The boundary condition for pressure, stream function and vorticity are discussed.

WALL AND LID BOUNDARY CONDITIONS

For stream function

$$\psi_{w+1} = \psi_w + \Delta n \left.\frac{\partial \psi}{\partial n}\right|_w + \frac{1}{2}(\Delta n)^2 \left.\frac{\partial^2 \psi}{\partial n^2}\right|_w + o\left[(\Delta n)^3\right]$$

Hence for the lid and for the walls the equations are respectively, (with u^{lid}=1 m/s)

$$\zeta_{i,NJ} = \frac{2\left(\psi_{i,NJ} - \Delta y - \psi_{i,NJ-1}\right)}{(\Delta y)^2}$$

$$\zeta_w = \frac{2\left(\psi_{w+1} - \psi_w\right)}{(\Delta n)^2}$$

BOUNDARY CONDITIONS –CONTD..

$$\psi = 0$$

This is true for all the walls, based on the NS equations at the wall

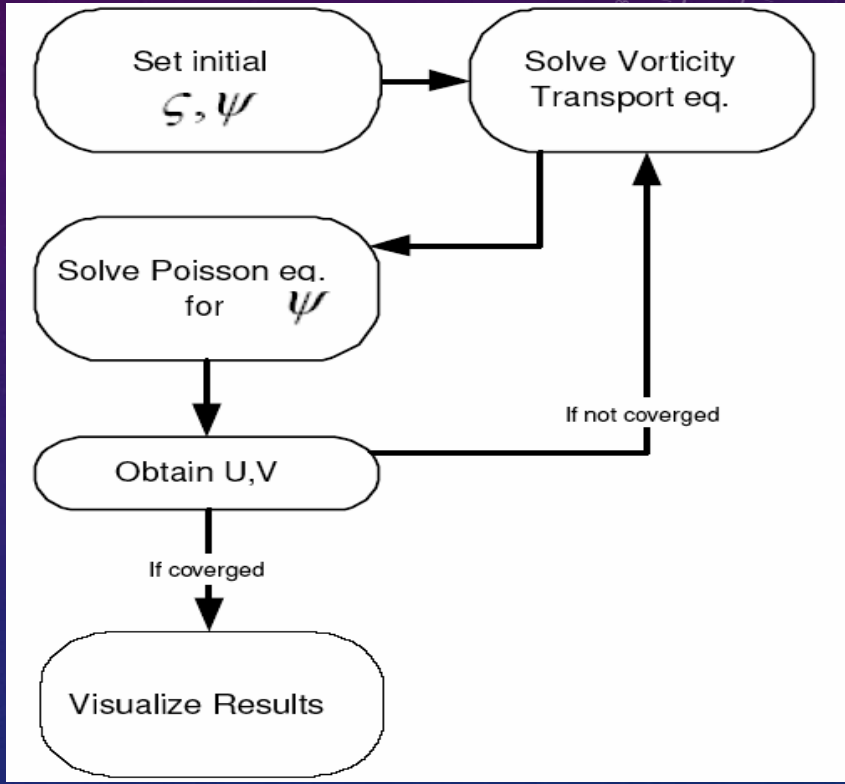
$$\left.\frac{\partial p}{\partial s}\right|_w = -\frac{1}{\mathrm{Re}}\left.\frac{\partial \zeta}{\partial n}\right|_w$$

For the wall pressure, the tangential and normal equations can be used

$$\frac{p_{s+1,w} - p_{s-1,w}}{2\Delta s} = -\frac{1}{\mathrm{Re}_l}\left(\frac{-3\zeta_{s,w} + 4\zeta_{s,w+1} - \zeta_{s,w+2}}{2\Delta n}\right)$$

ALGORITHM

The implementation/pseudo code is discussed here.



WALL AND LID BOUNDARY CONDITIONS

1. Specify the geometry and fluid properties
2. Specify and assume initial conditions (e.g. $u=v=\zeta=\psi=0$).
3. Specify boundary conditions
4. Determine Δt (by stability condition) $\min\left(\frac{\Delta x^2}{2\nu}, \frac{\Delta x}{U}\right)$
5. Solve the vorticity transport equation to get ζ in the next time step
6. Solve stream function equation to get the stream function at the next step
7. Obtain u^{n+1} and v^{n+1}
8. Obtain the ζ BC on walls using the above
9. Continue marching to time of interest, or until the steady state is reached. (Iterate by repeating 5-9)
10. Solve the pressure Poisson equation with appropriate boundary conditions

CONVERGENCE CRITERION

- Run till the code converges or for a fixed amount of iterations.
- Convergence is usually checked by the two equations given below :

$$\frac{1}{NI \times NJ} \sum_{i=1,j=1}^{i=NI,j=NJ} \left| \zeta_{i,j}^{n+1} - \zeta_{i,j}^{n} \right| \leq 1 \times 10^{-8}$$

$$\frac{1}{NI \times NJ} \sum_{i=1,j=1}^{i=NI,j=NJ} \left| \psi_{i,j}^{n+1} - \psi_{i,j}^{n} \right| \leq 1 \times 10^{-8}$$

SOLVERS

Poisson equation solvers implemented ,particularly the Fast Fourier Transform (FFT) is discussed here. We have a five point Poisson Equation for stream function and pressure

FAST FOURIER TRANSFORM IN 1 DIMENSION

The Fast Fourier Transform can be used to solve elliptic partial differential equations in multiple dimensions.

The basic idea is very simple. Consider the one-dimensional equation

$$\frac{\mathrm{d}^2\phi}{\mathrm{d}x^2} = \rho(x) \; .$$

Express f and ρ in terms of their Fourier transforms:

$$f(x) = \frac{1}{\sqrt{2\pi}} \int g(k) e^{ikx} \mathrm{d}k \; , \qquad \rho(x) = \frac{1}{\sqrt{2\pi}} \int \sigma(k) e^{ikx} \mathrm{d}k \; .$$

The equation is diagonalized in k-space

$$-k^2 g(k) = \sigma(k) \qquad \Rightarrow \qquad g(k) = -\frac{\sigma(k)}{k^2} \; .$$

The solution is given by the inverse transform

$$f(x) = -\frac{1}{\sqrt{2\pi}} \int \frac{\sigma(k)}{k^2} e^{ikx} \mathrm{d}k \; .$$

FAST FOURIER TRANSFORM EQUATION IN 2 DIMENSION

Let's consider the discrete form of Poisson's equation

$$\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)\phi(x,y)\simeq\frac{1}{h^2}\left[\phi_{j+1,k}+\phi_{j-1,k}+\phi_{j,k+1}+\phi_{j,k-1}-4\phi_{j,k}\right]=-\rho_{j,k}$$

on an $N\times N$ grid of points in the region $0\leq x,y\leq 1$ with a given charge density, say a single point charge located at the center of the square.

For simplicity, we will impose periodic boundary conditions so that the exponential FFT can be used to obtain the solution.

The discrete Fourier transform is a *linear operation*, so we can apply it separately in the x and y directions, and it does not matter in which order the transforms are done.

The two-dimensional Fourier coefficients are given by

$$\tilde{\phi}_{m,n}=\frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1}W^{mj+nk}\phi_{j,k}\ ,$$

$$\tilde{\rho}_{m,n}=\frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1}W^{mj+nk}\rho_{j,k}\ .$$

The inverse transforms are

$$\phi_{j,k}=\frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}W^{-jm-kn}\tilde{\phi}_{m,n}\ ,$$

$$\rho_{j,k} = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} W^{-jm-kn} \tilde{\rho}_{m,n} \ .$$

Plugging these expressions into the discretized equation and equating coefficients of W^{-mj-nk} gives

$$\frac{1}{h^2} \left[W^m + W^{-m} + W^n + W^{-n} - 4 \right] \tilde{\phi}_{m,n} = -\tilde{\rho}_{m,n} \ ,$$

which is easily solved for

$$\tilde{\phi}_{m,n} = \frac{h^2 \tilde{\rho}_{m,n}}{4 - W^m - W^{-m} - W^n - W^{-n}} \ .$$

The inverse Fourier transform then gives the potential.

STREAM FUNCTION FAST FOURIER TRANSFORM

In the case of the Stream Function Poisson Equation , the Boundary condition for the stream function is a Dirchlet type (=0 on the boundaries). Hence a sinusoidal curve is chosen and solved comfortably, instead of taking a complex function. The equation for the same is

$$u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 \rho_{j,l}$$

$$u_{jl} = \frac{2}{J}\frac{2}{L}\sum_{m=1}^{J-1}\sum_{n=1}^{L-1}\widehat{u}_{mn}\sin\frac{\pi jm}{J}\sin\frac{\pi ln}{L}$$

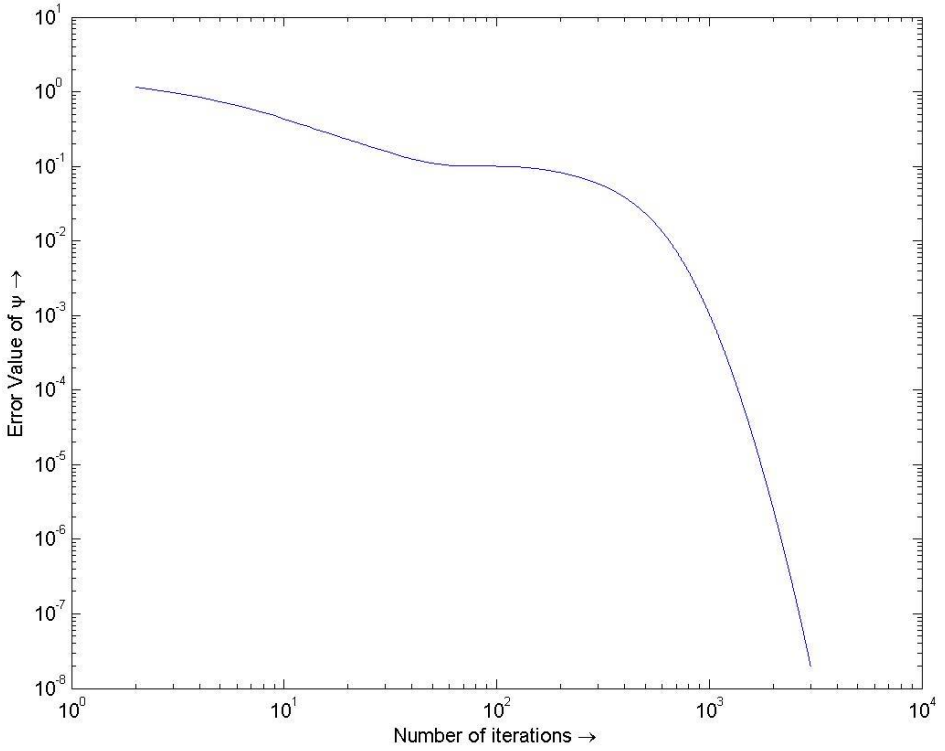
$$\widehat{u}_{mn} = \frac{\Delta^2\widehat{\rho}_{mn}}{2\left(\cos\frac{\pi m}{J} + \cos\frac{\pi n}{L} - 2\right)}$$

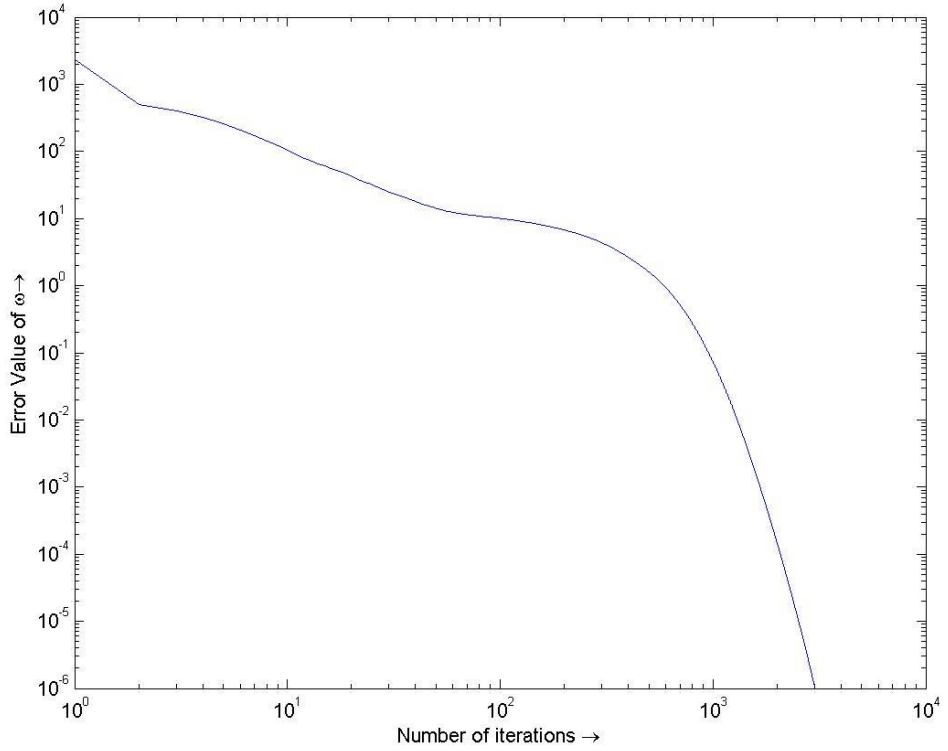
$$\widehat{\rho}_{mn} = \sum_{j=1}^{J-1}\sum_{l=1}^{L-1}\rho_{jl}\sin\frac{\pi jm}{J}\sin\frac{\pi ln}{L}$$

RESULTS

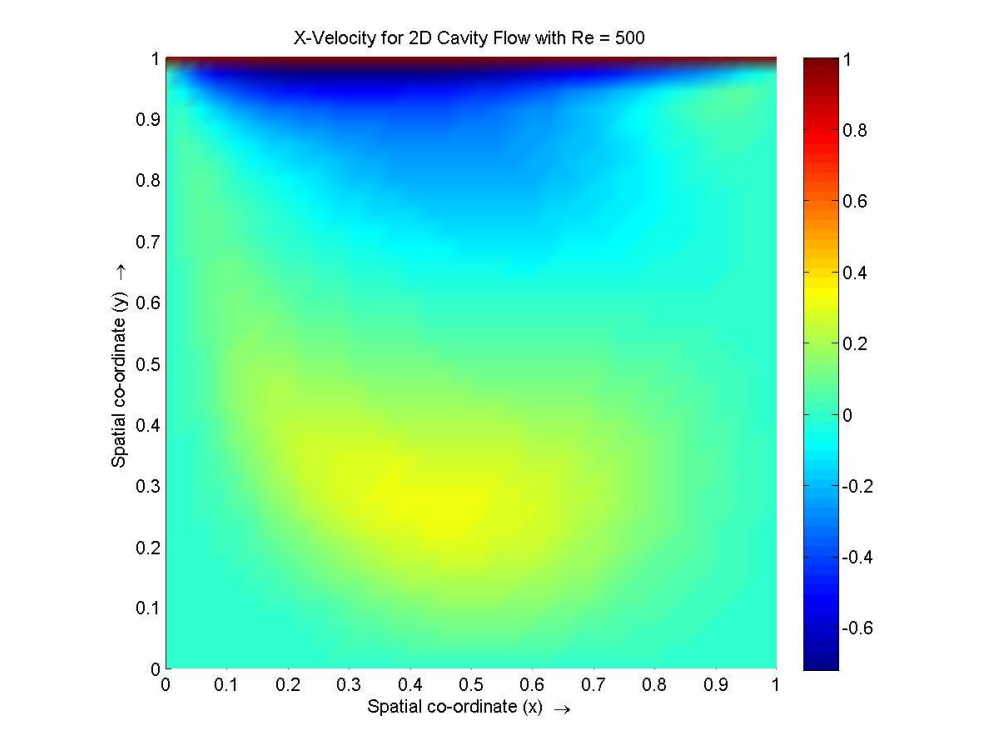
Results for multiple iterations (of grid sizes , Reynolds Number are discussed here)

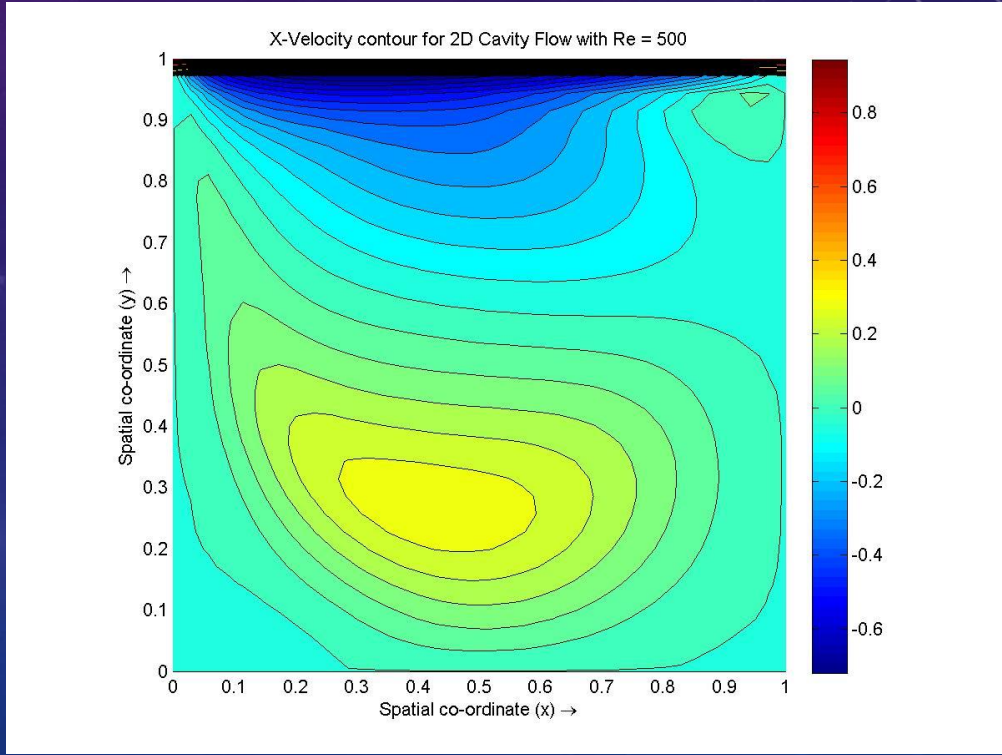
RESULTS FOR A 35X35 GRID ,WITH RE = 500



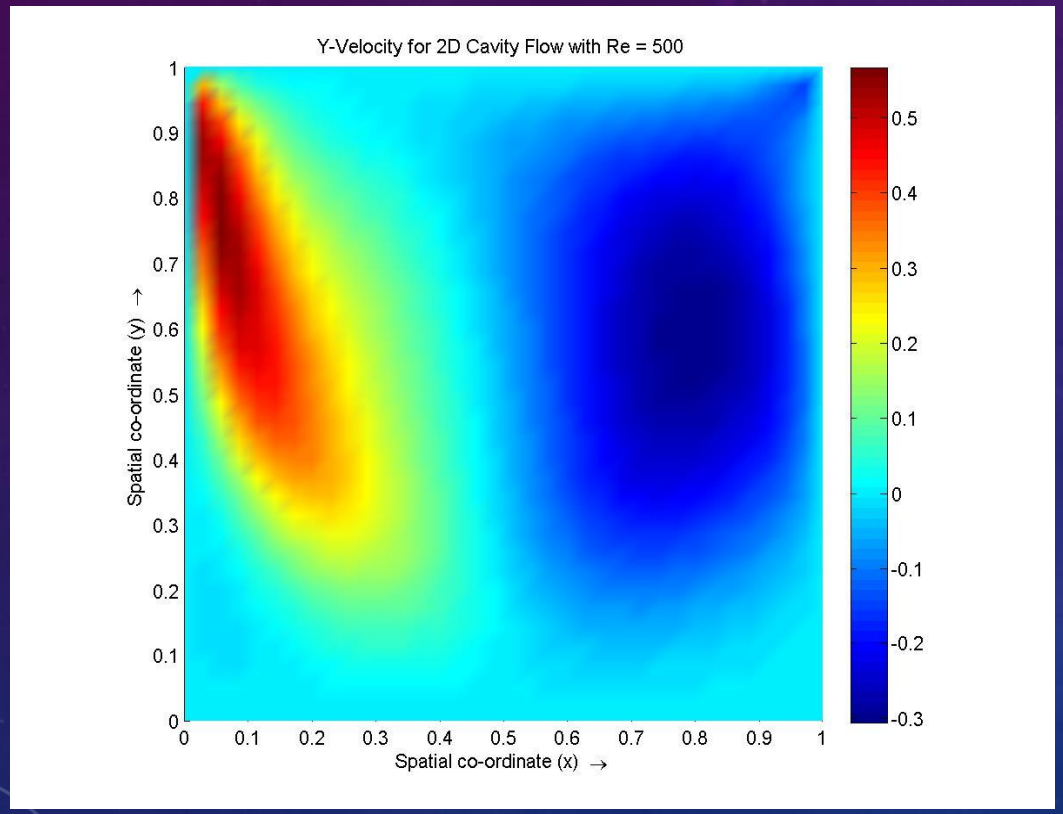


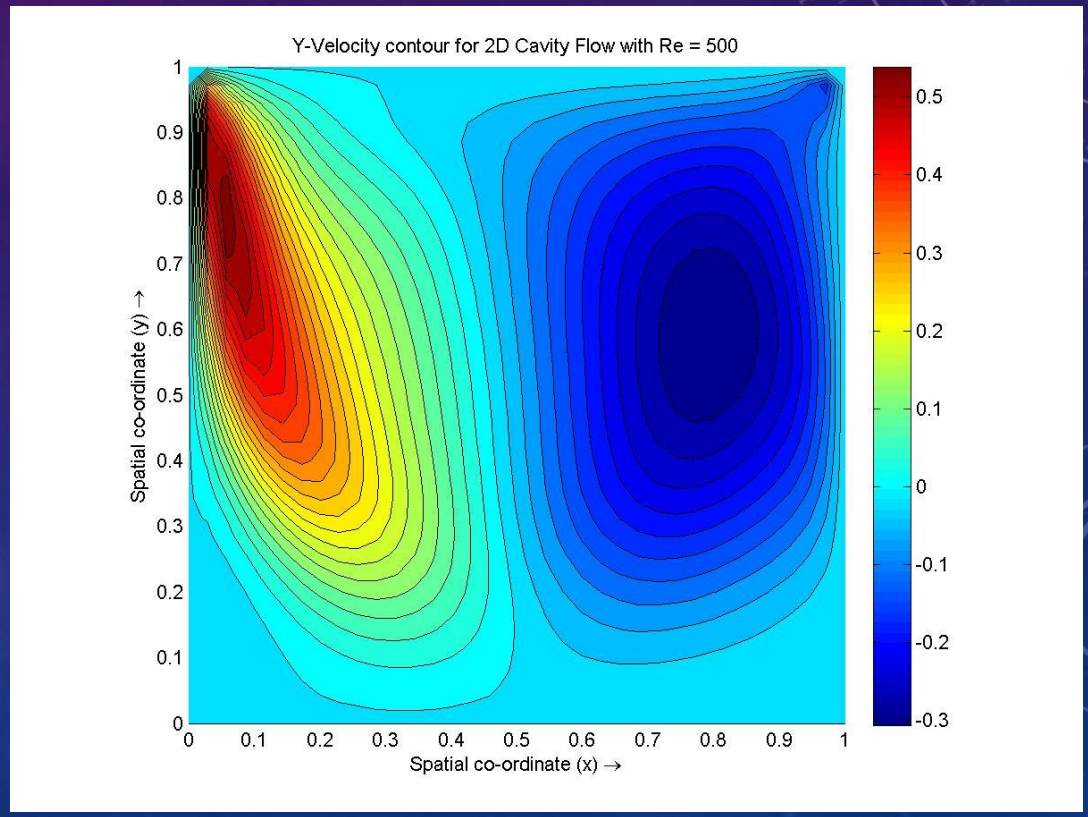
X COMPONENT OF VELOCITY – SMOOTH AND CONTOUR PLOTS



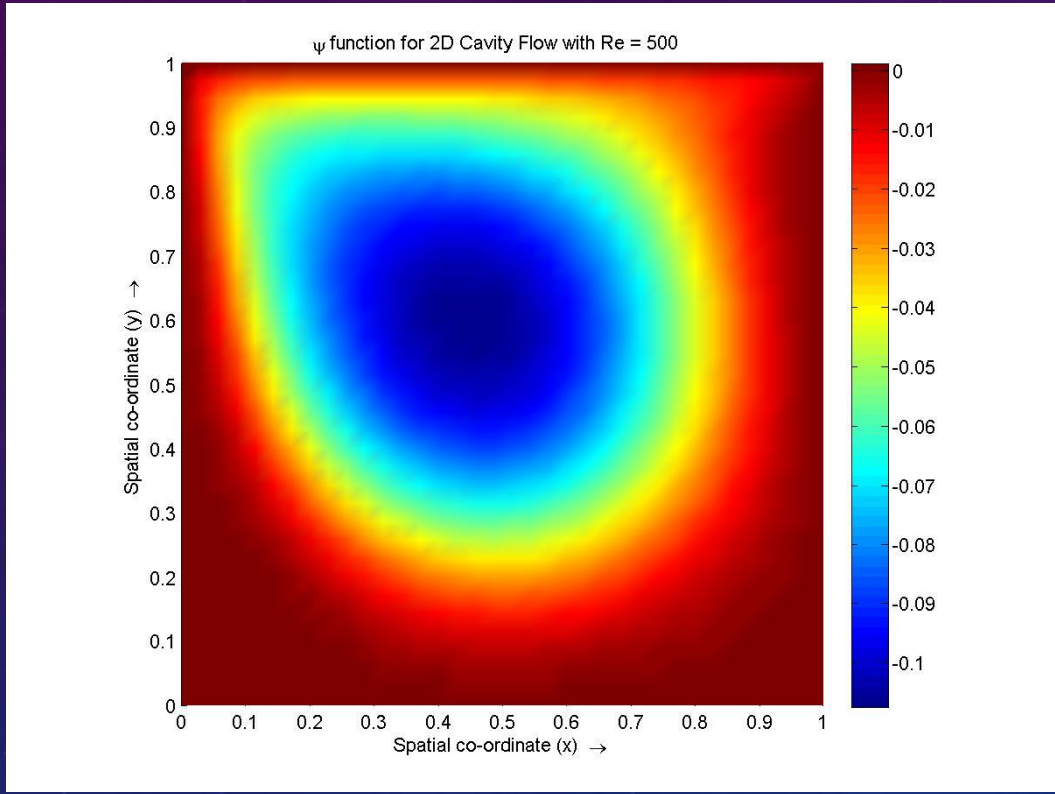


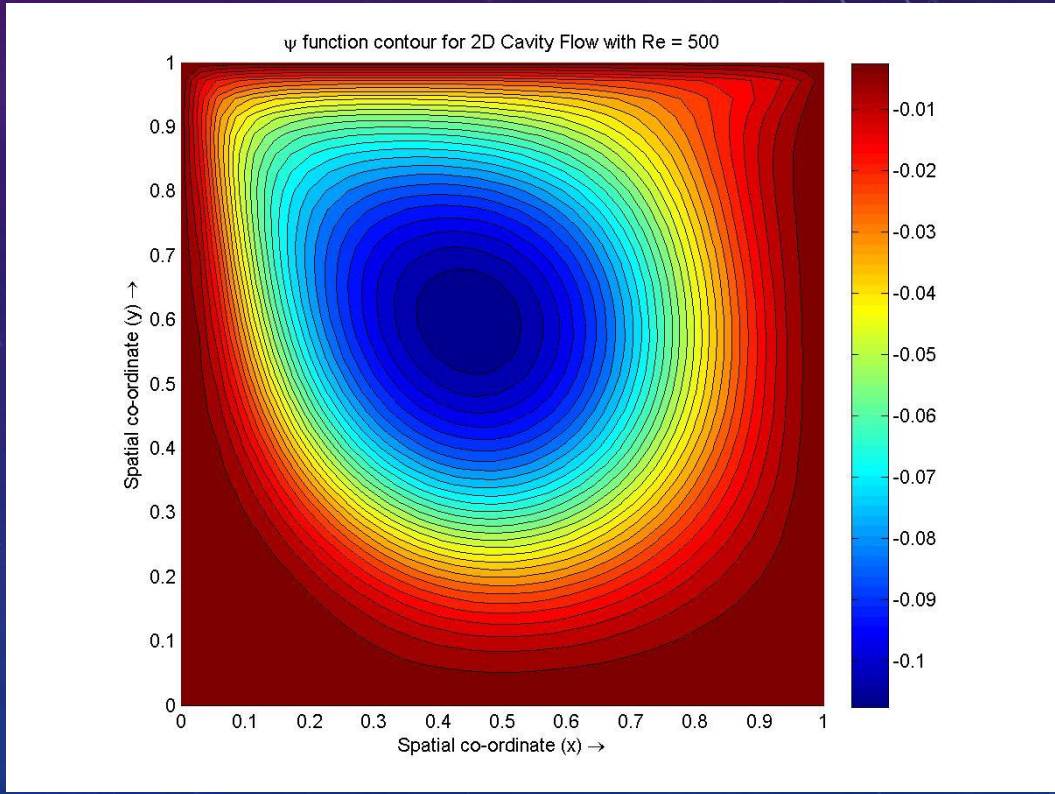
Y COMPONENT OF VELOCITY – SMOOTH AND CONTOUR PLOTS



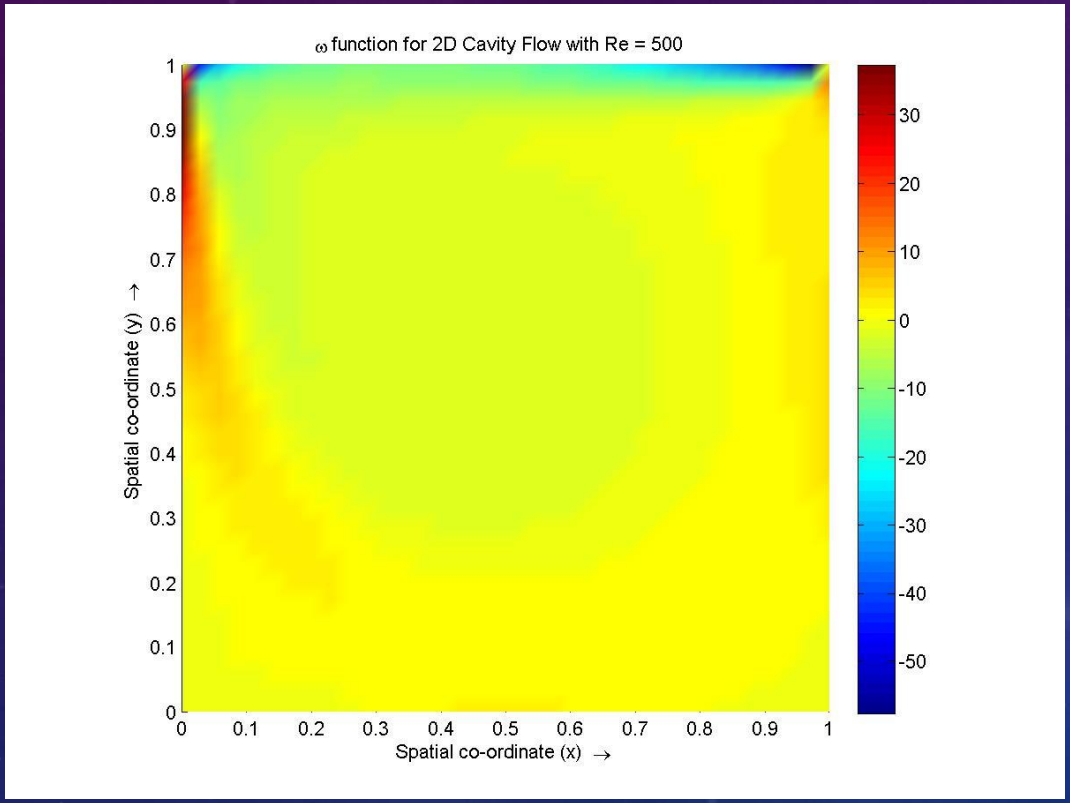


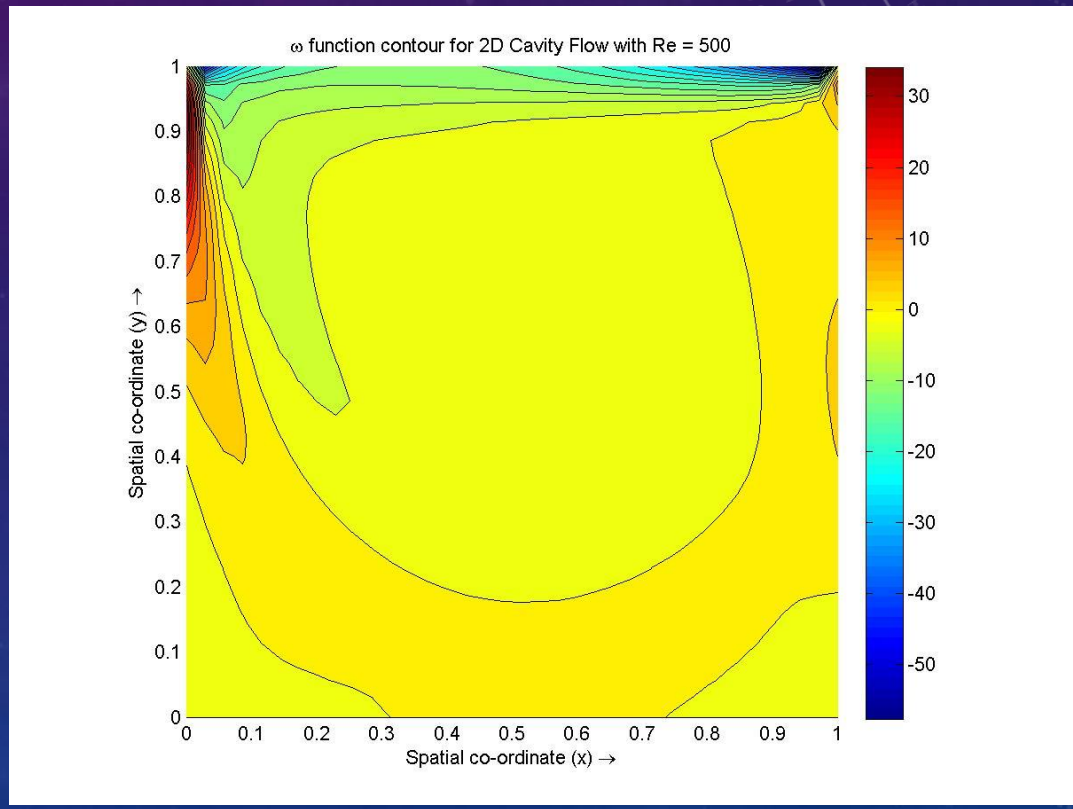
PSI FUNCTION – SMOOTH AND CONTOUR



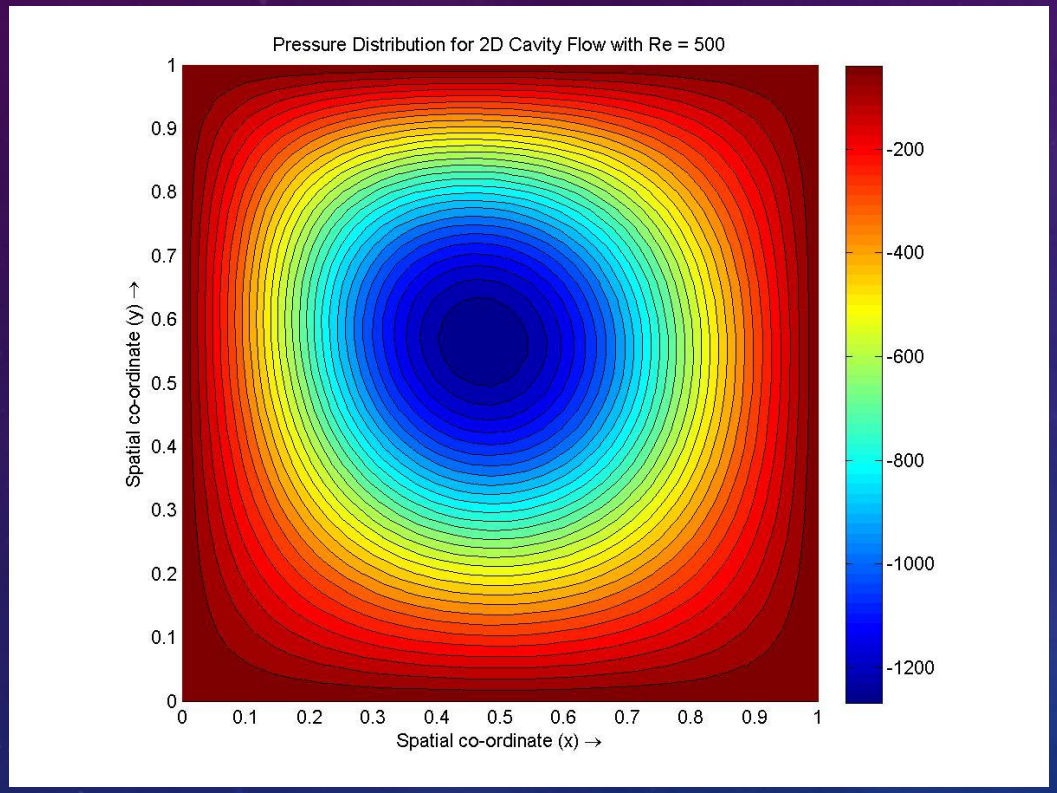


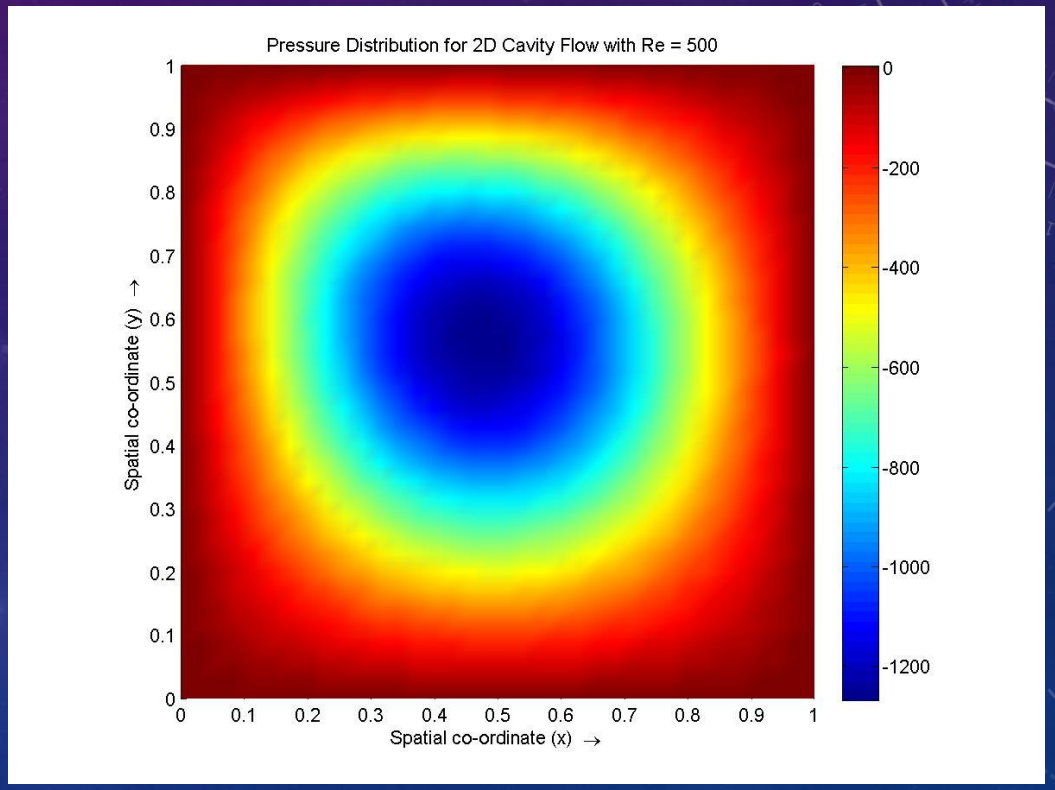
VORTICITY – SMOOTH AND CONTOUR



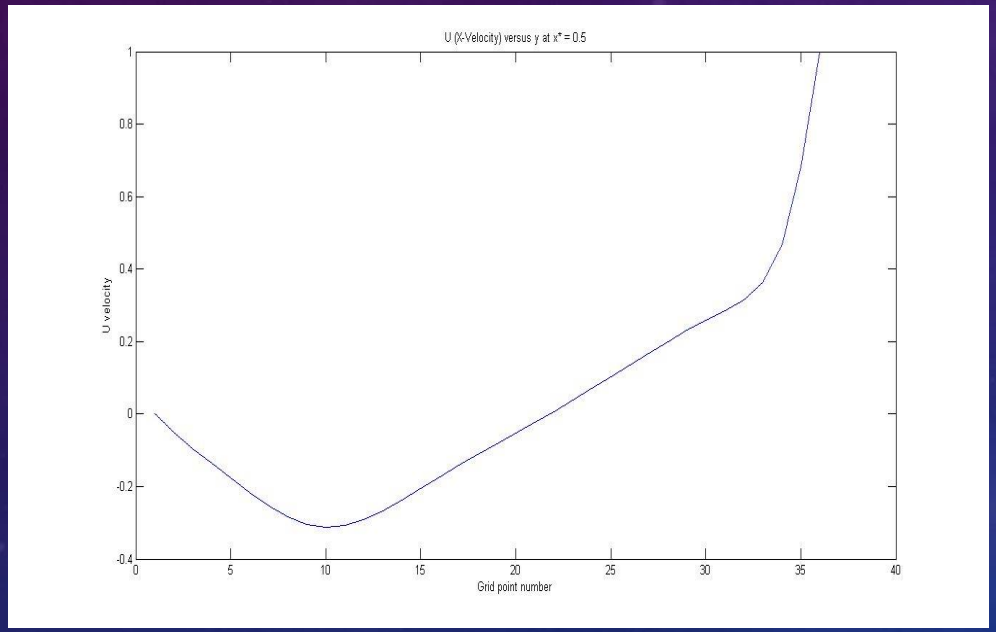


PRESSURE DISTRIBUTION – SMOOTH AND CONTOUR

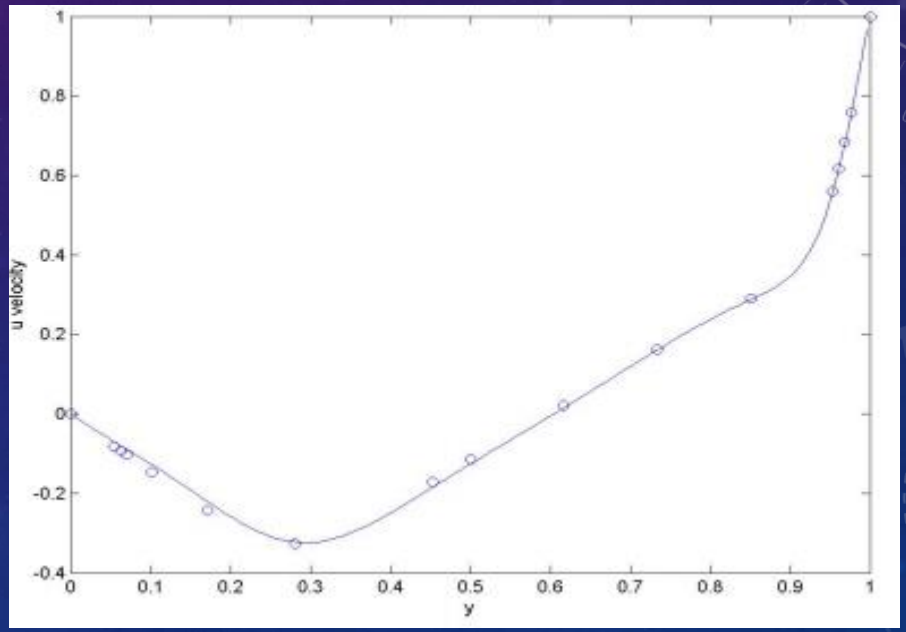




X VELOCITY VS. Y , AT X* = 0.5

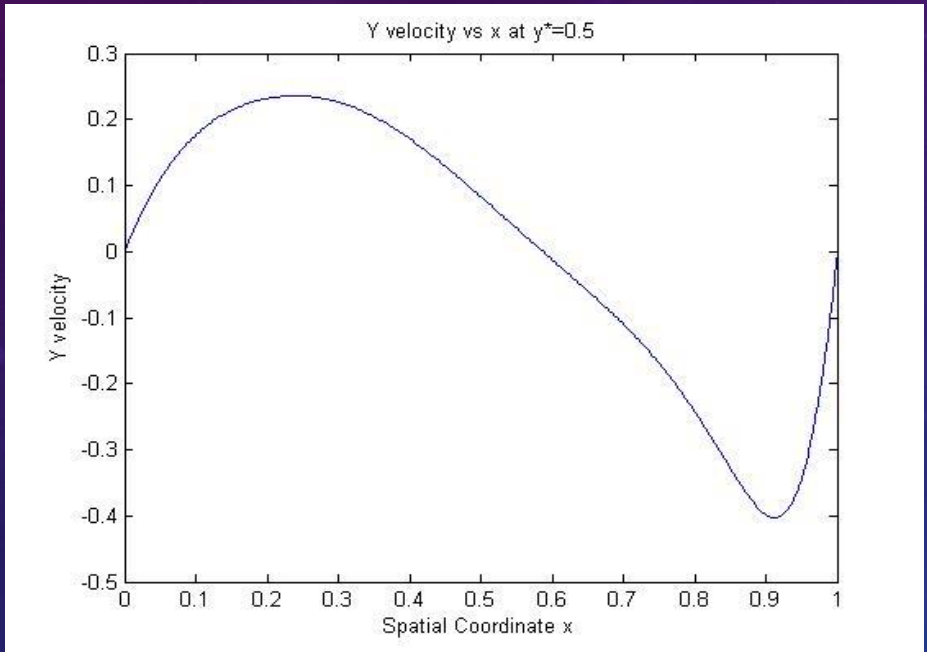


Our Solution

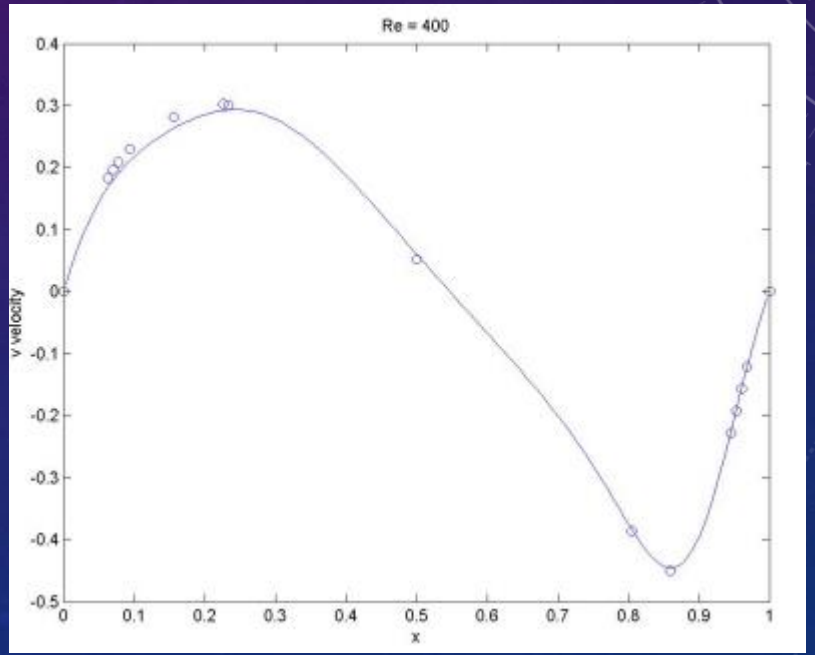


Reference Solution

Y VELOCITY VS. X , AT Y* =0.5

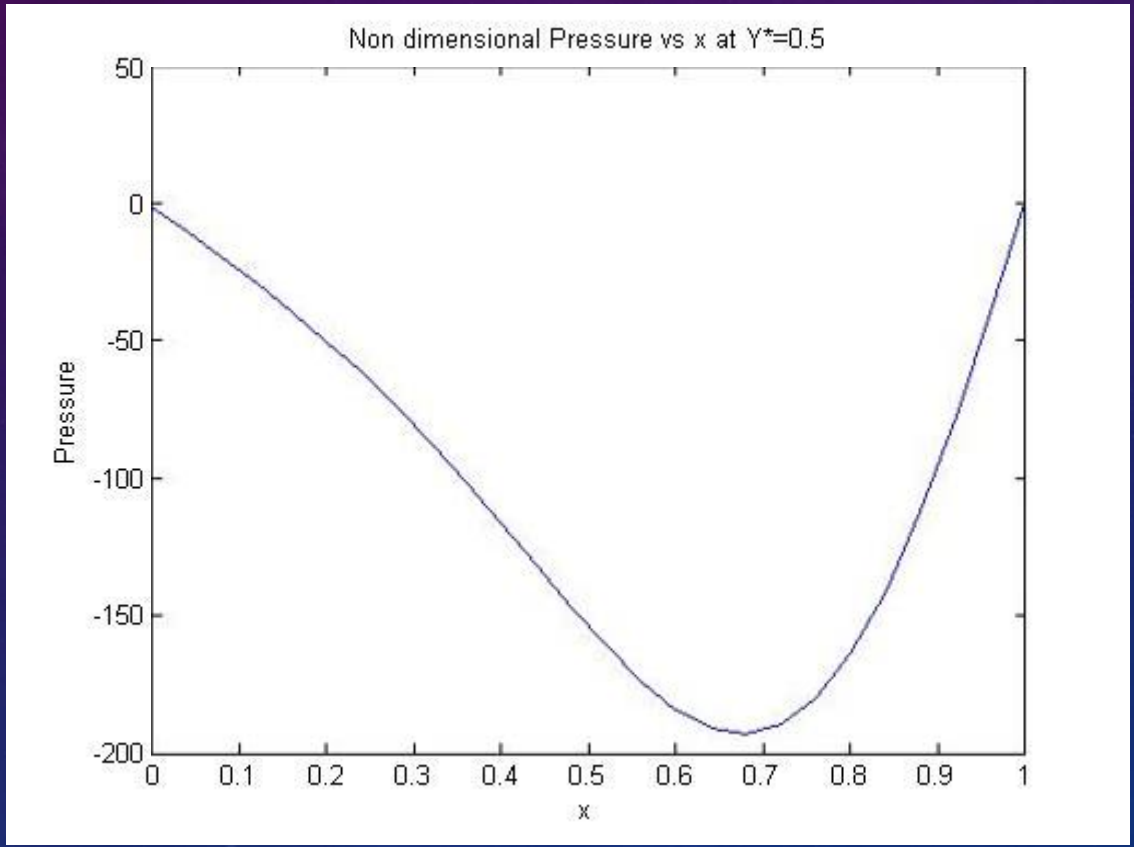


Our Solution

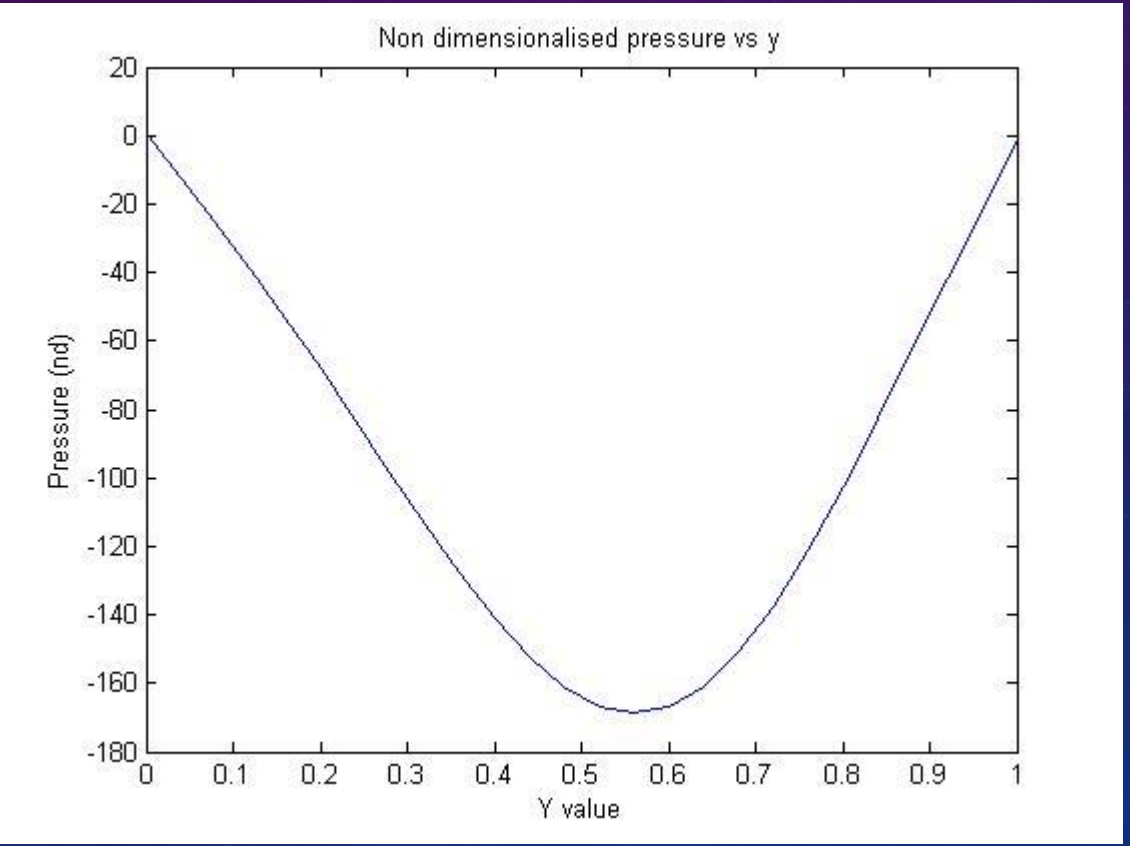


Reference Solution

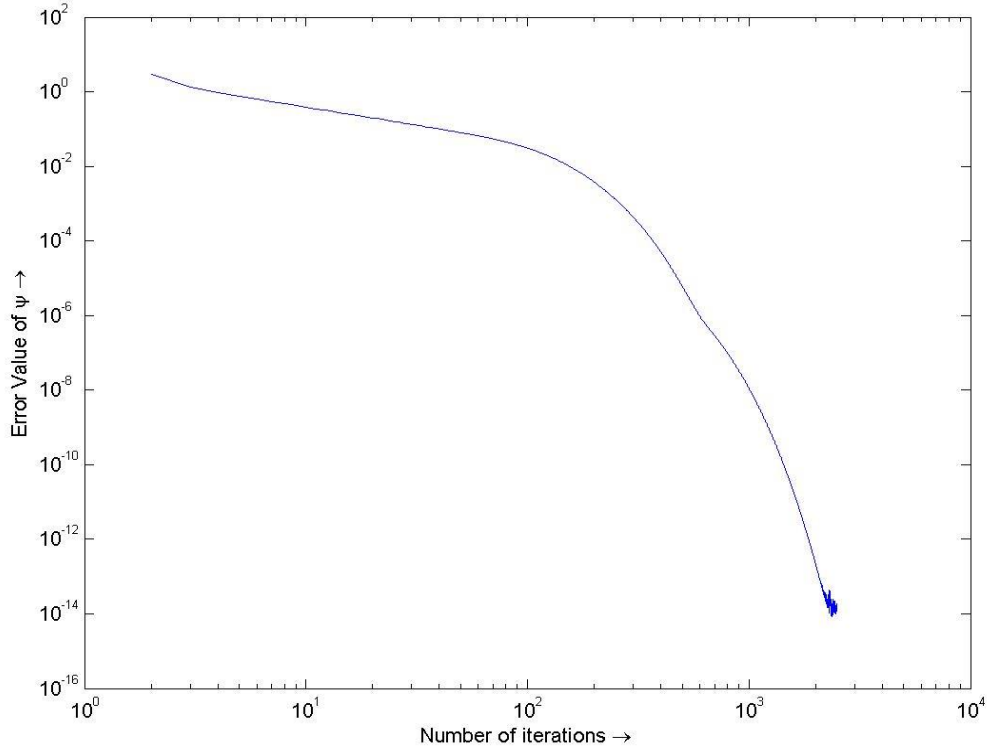
PRESSURE VS. Y , AT X* = 0.5

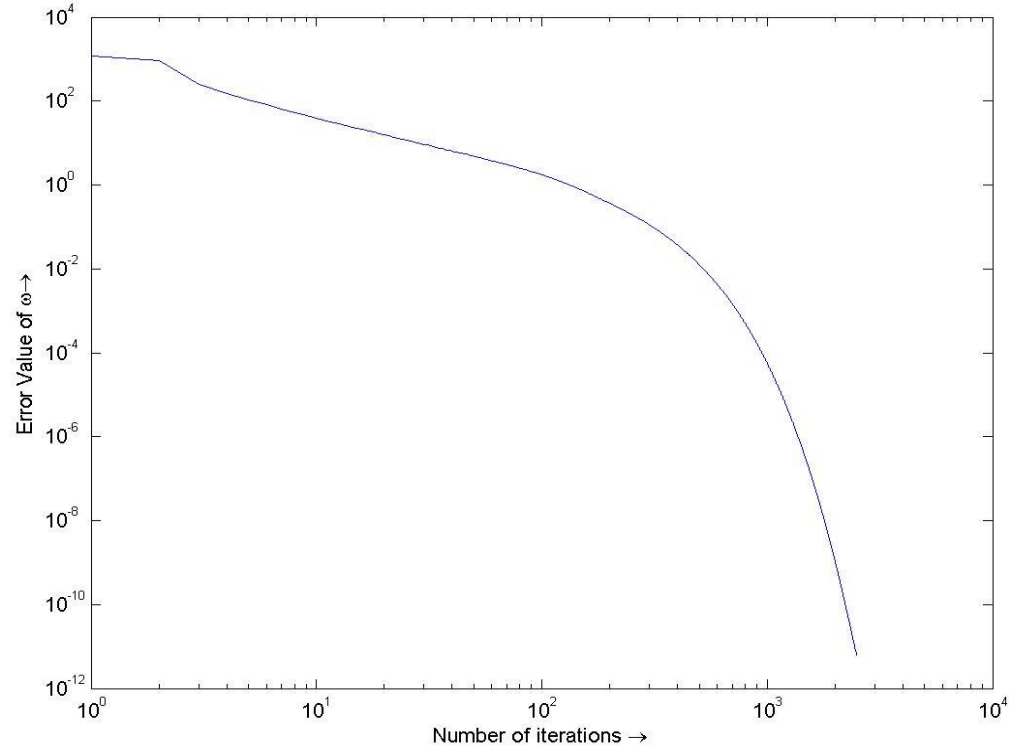


PRESSURE VS. X , AT Y* =0.5

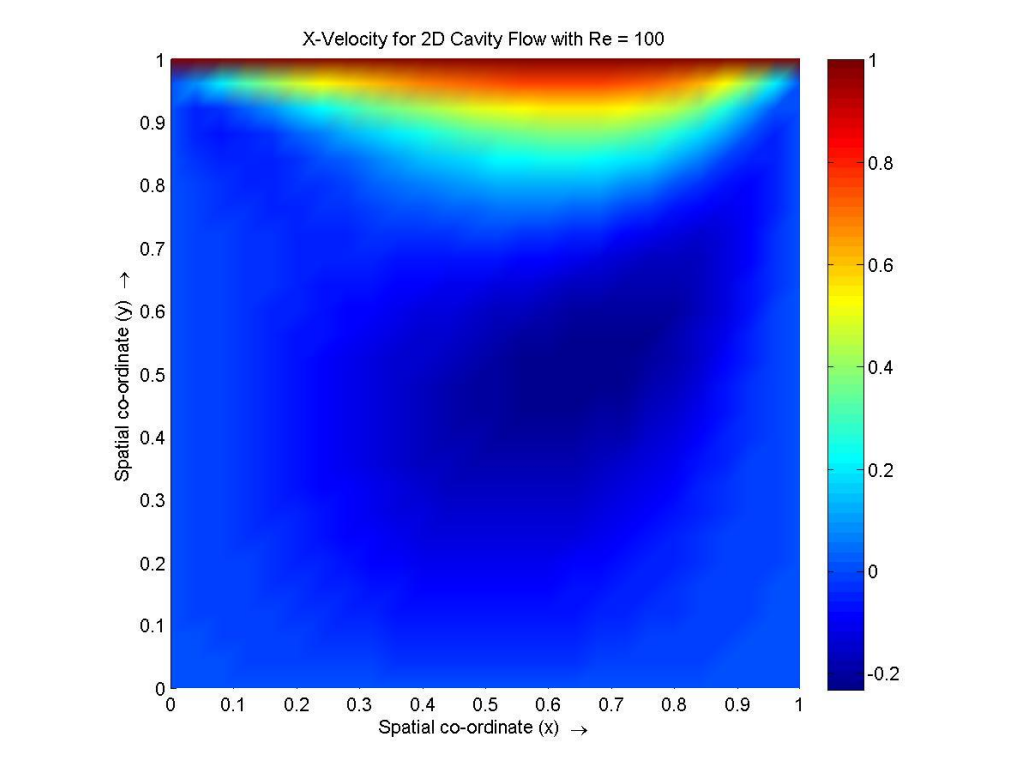


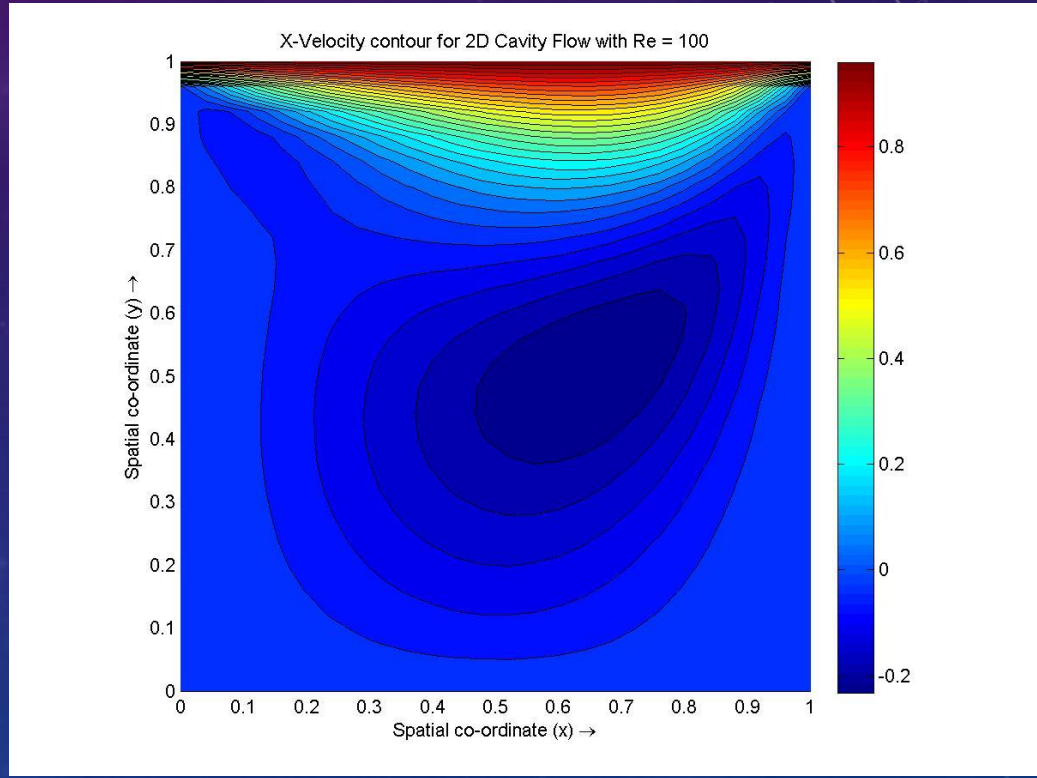
RESULTS FOR A 15X15 GRID ,WITH RE = 100



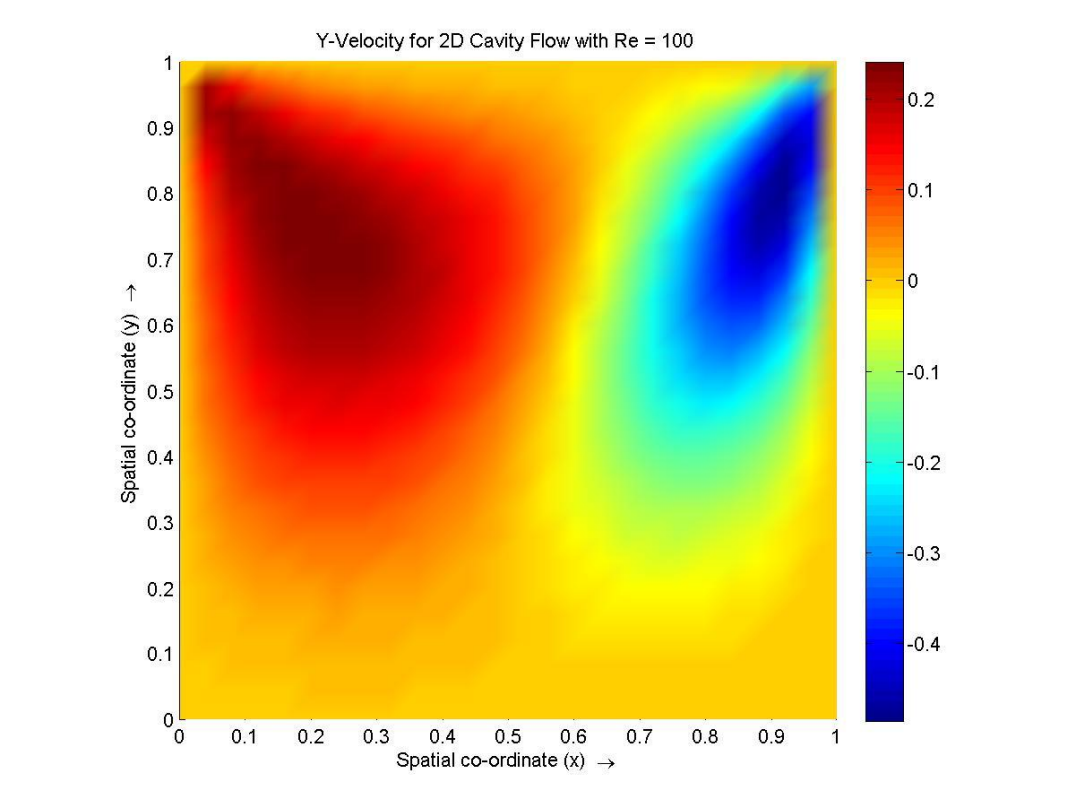


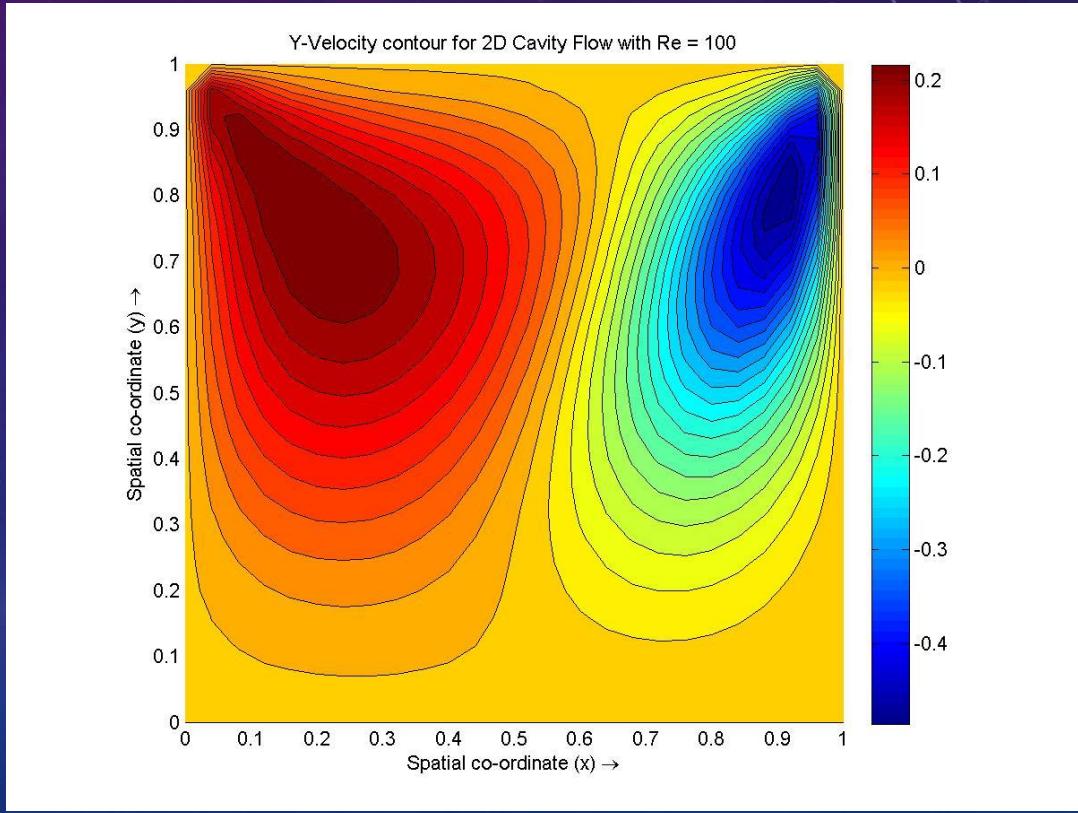
X COMPONENT OF VELOCITY – SMOOTH AND CONTOUR PLOTS



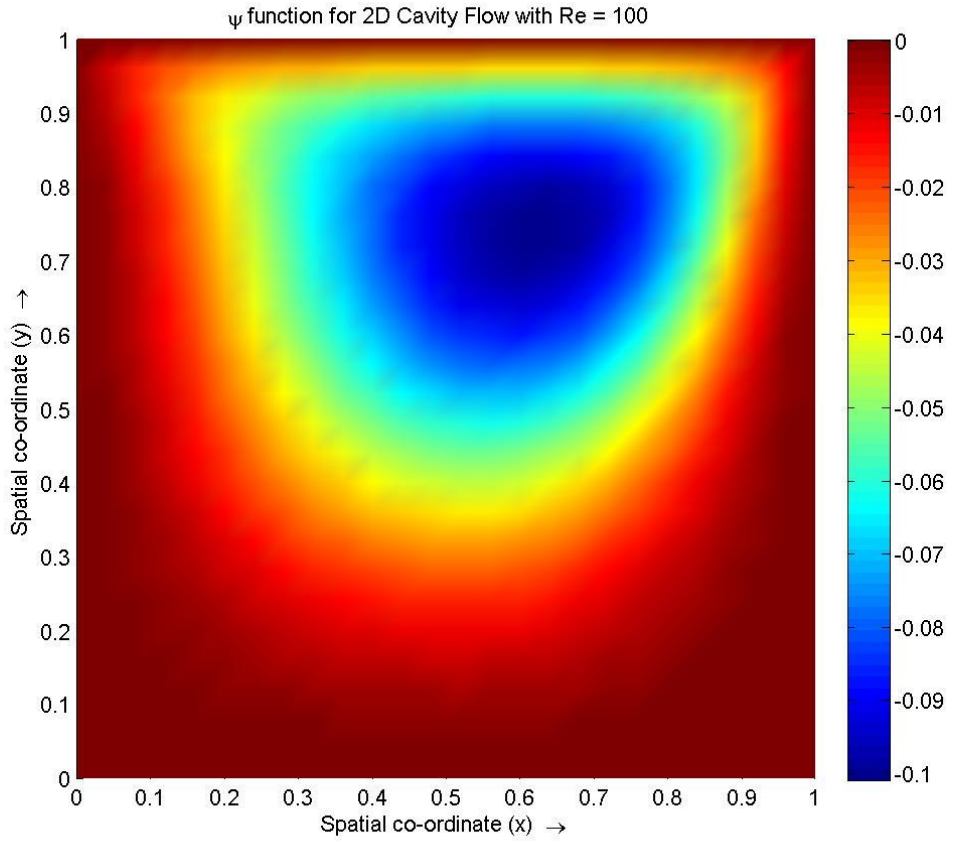


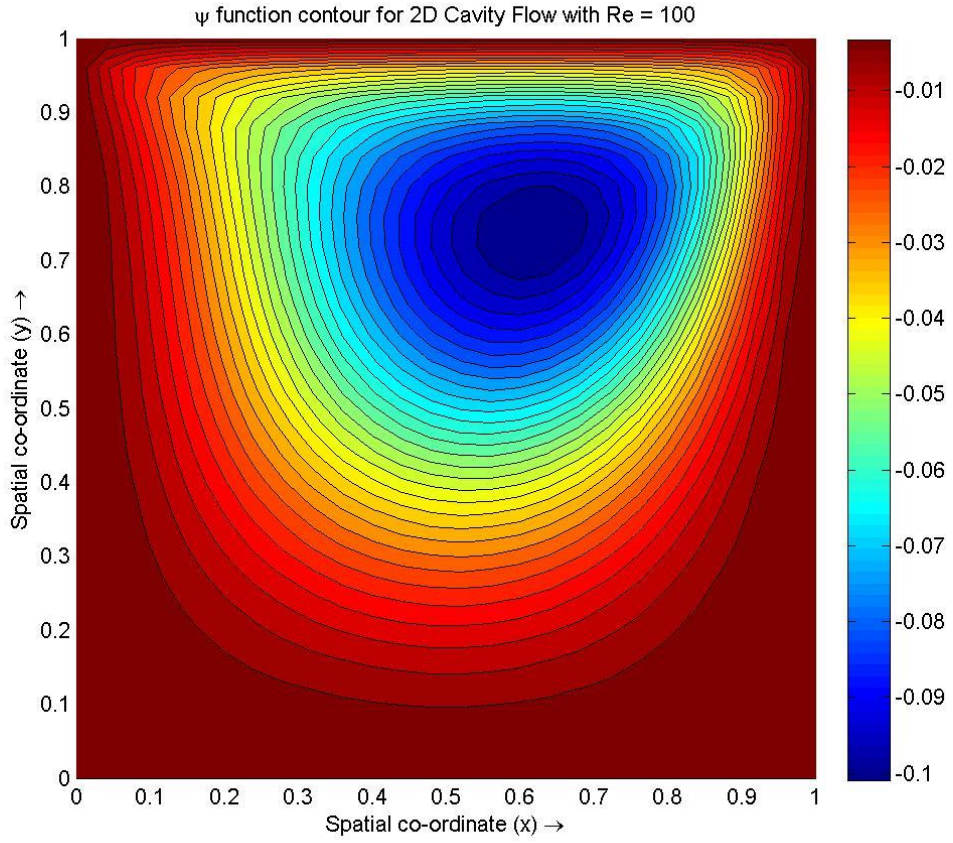
Y COMPONENT OF VELOCITY – SMOOTH AND CONTOUR PLOTS



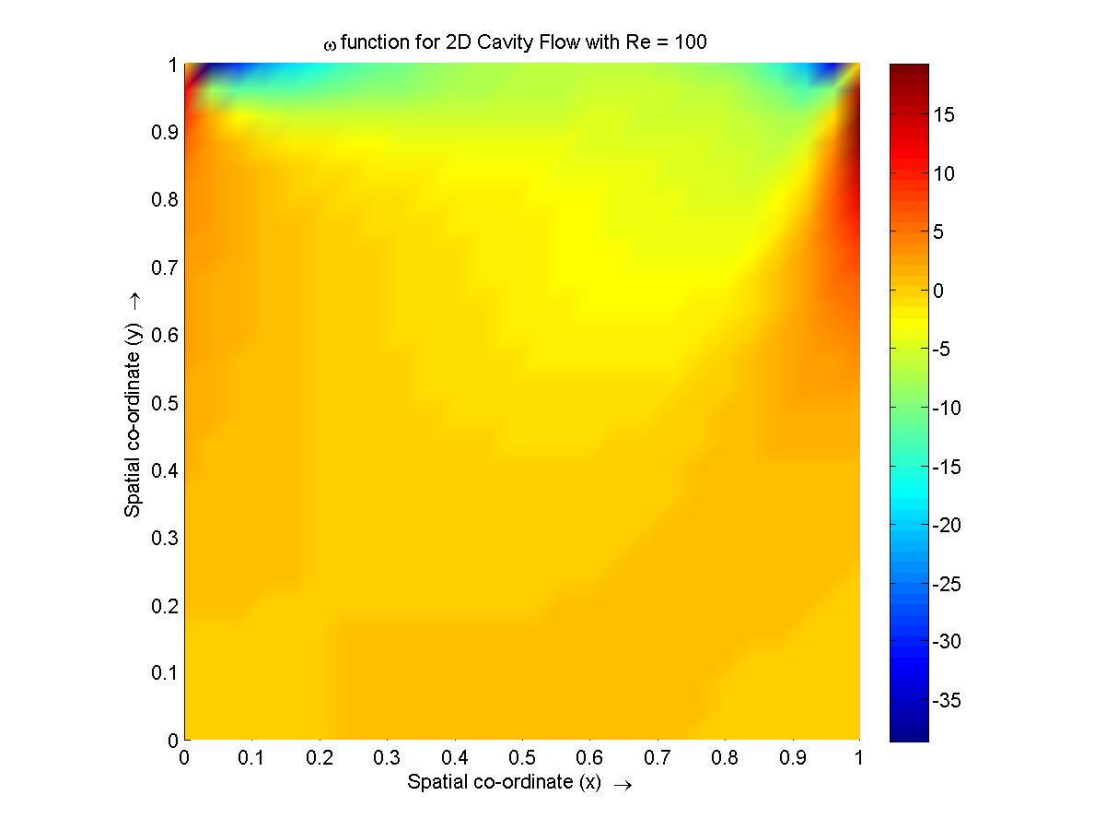


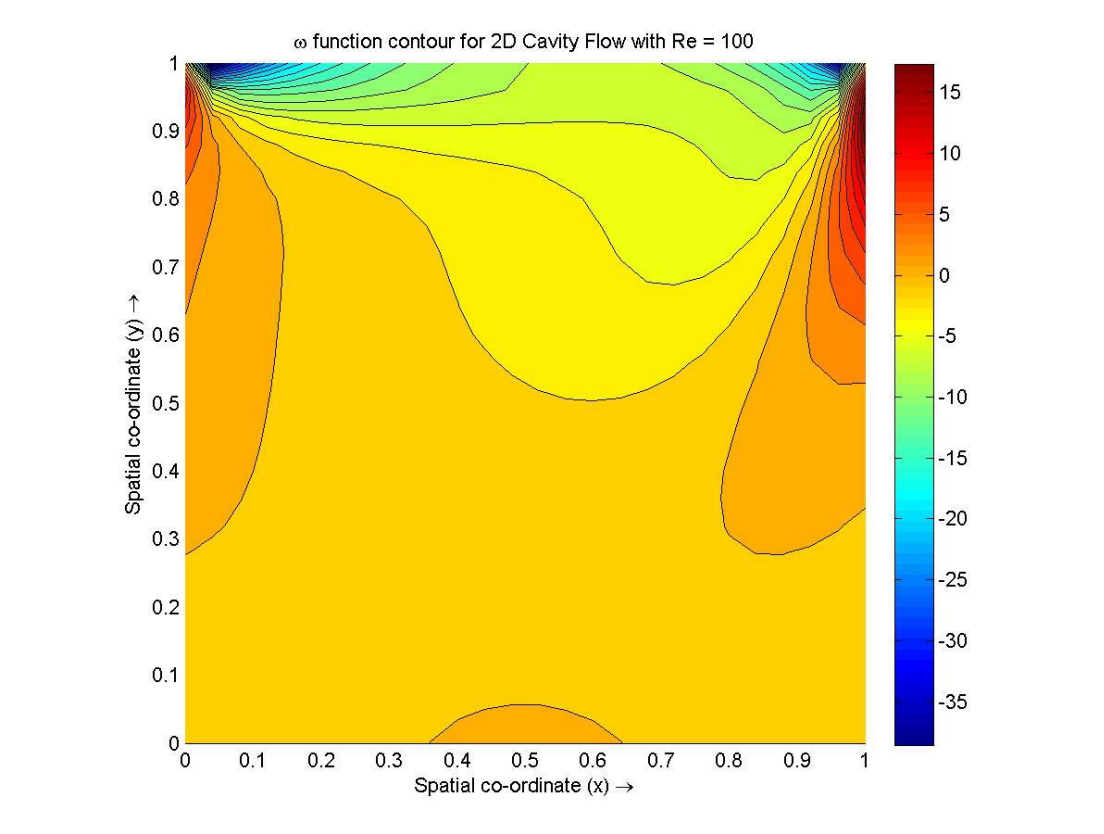
PSI FUNCTION – SMOOTH AND CONTOUR



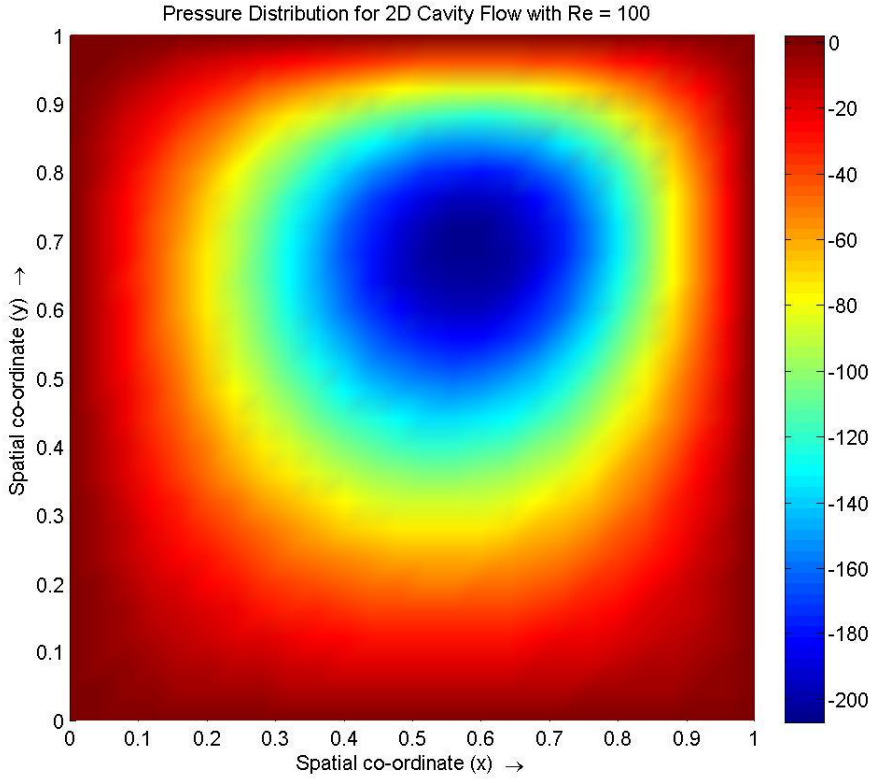


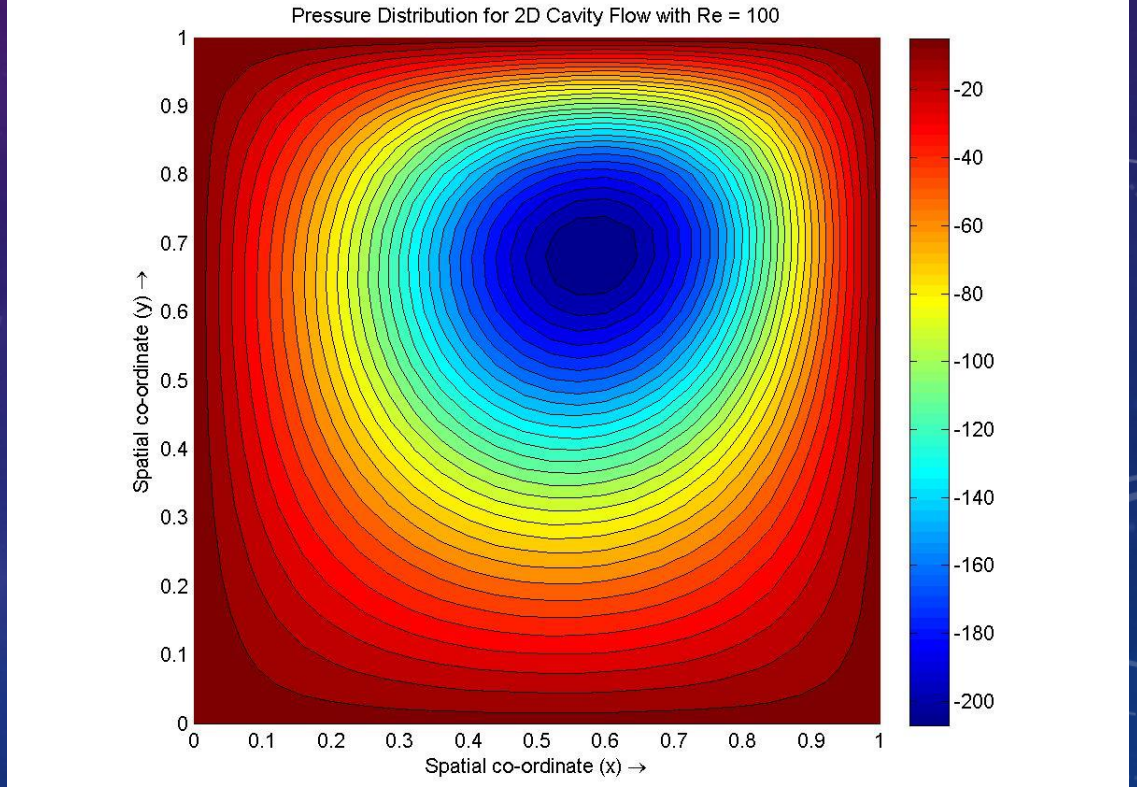
VORTICITY – SMOOTH AND CONTOUR



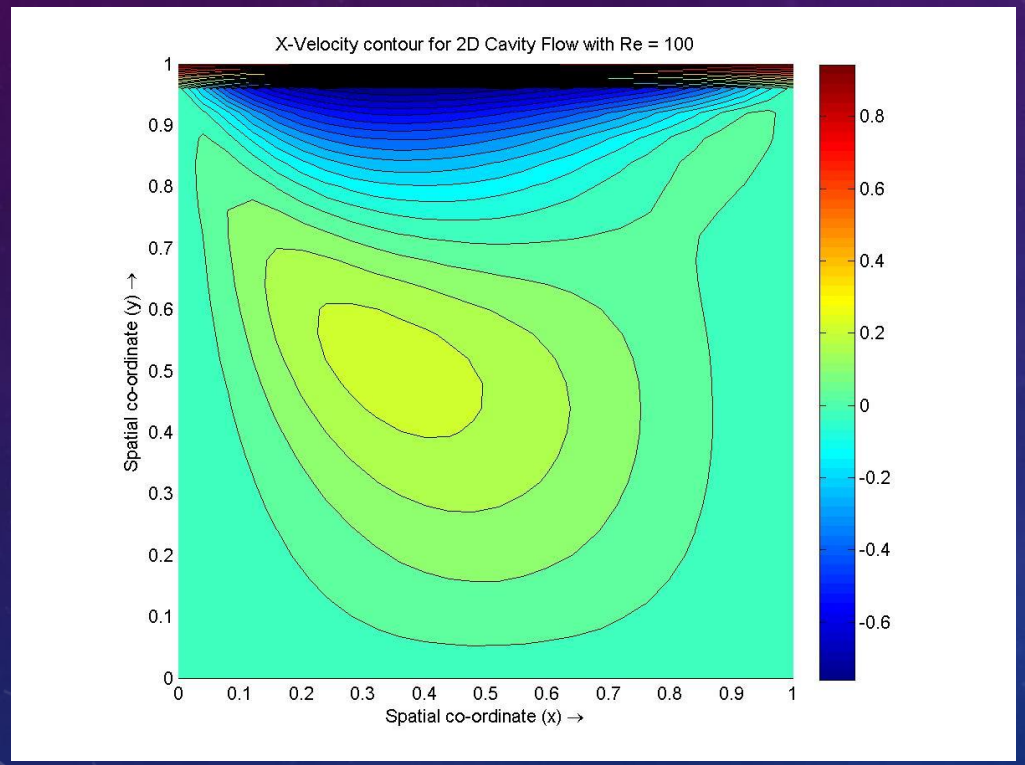


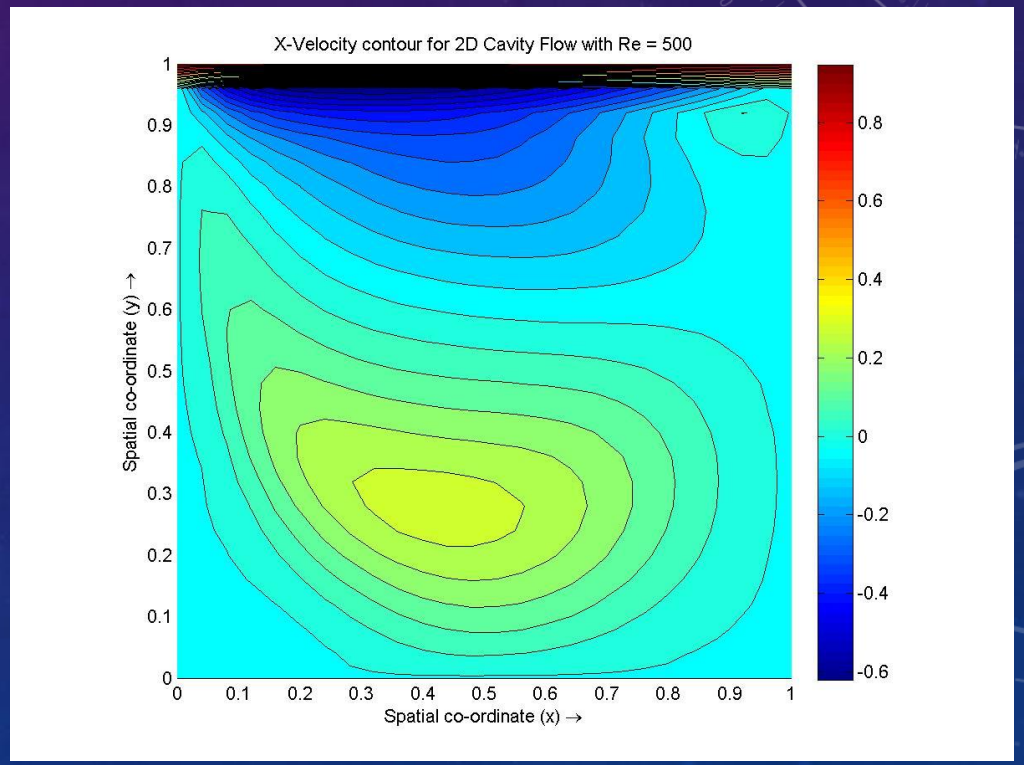
PRESSURE DISTRIBUTION – SMOOTH AND CONTOUR



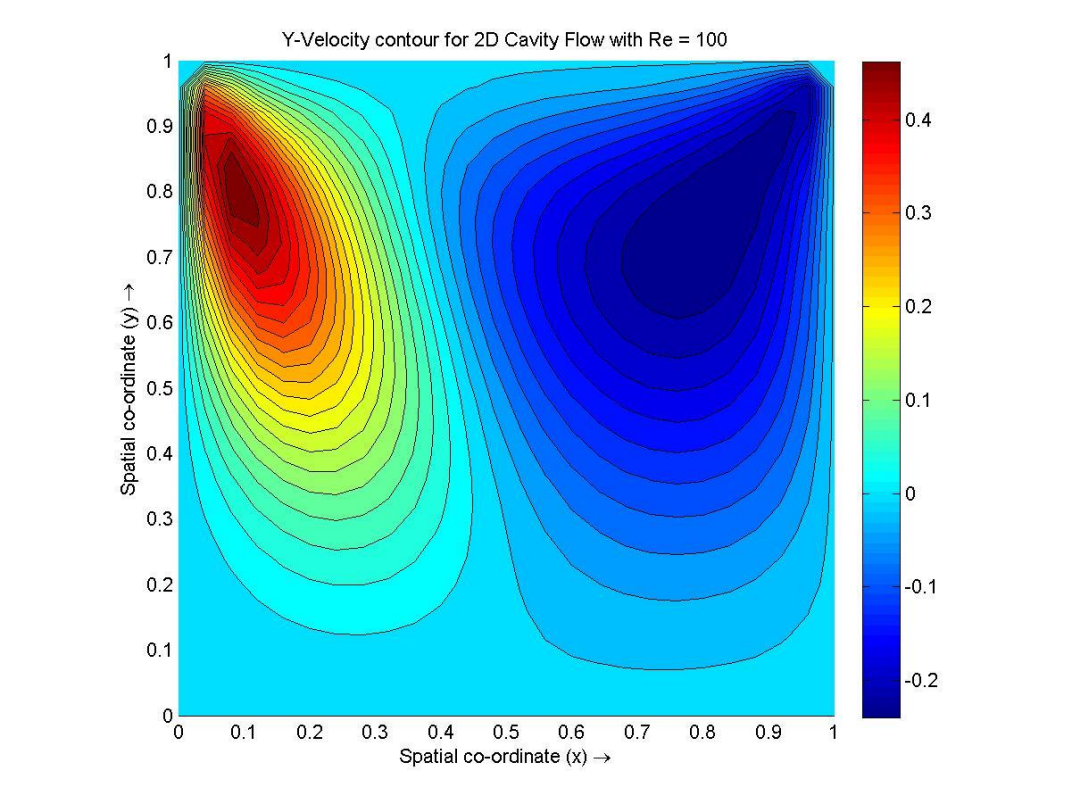


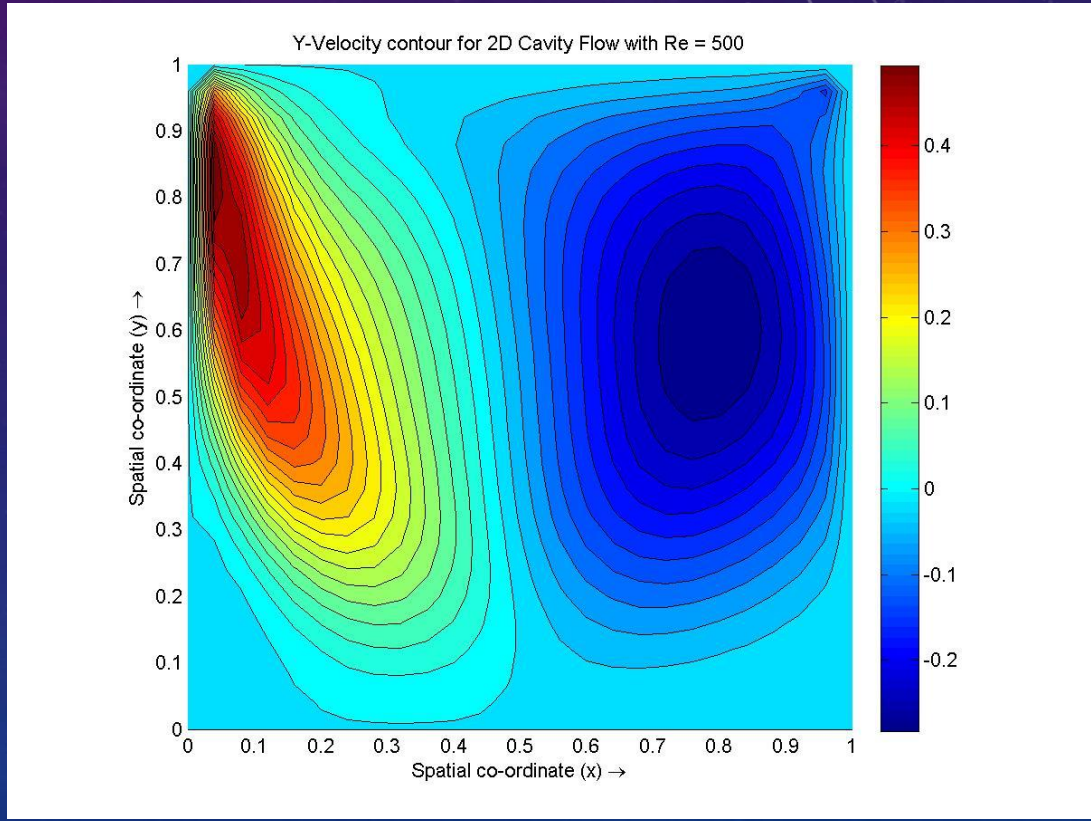
COMPARISON OF RESULTS FOR DIFFERENT RE NUMBERS FOR A 25X25 GRID



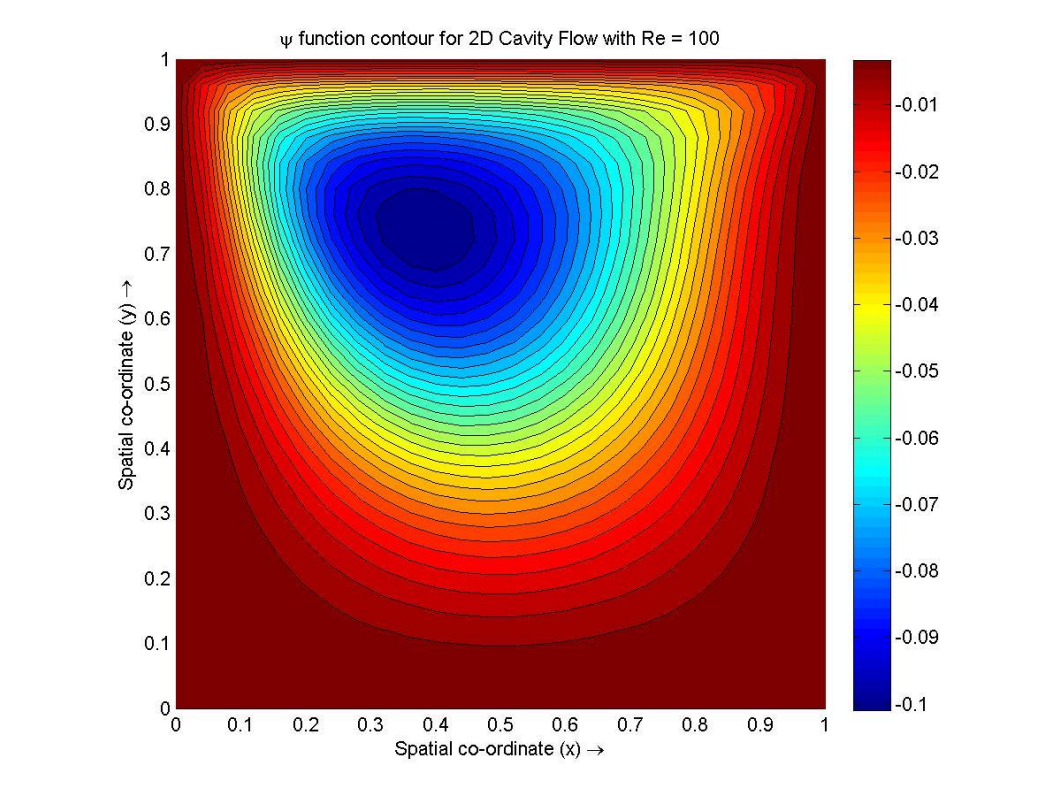


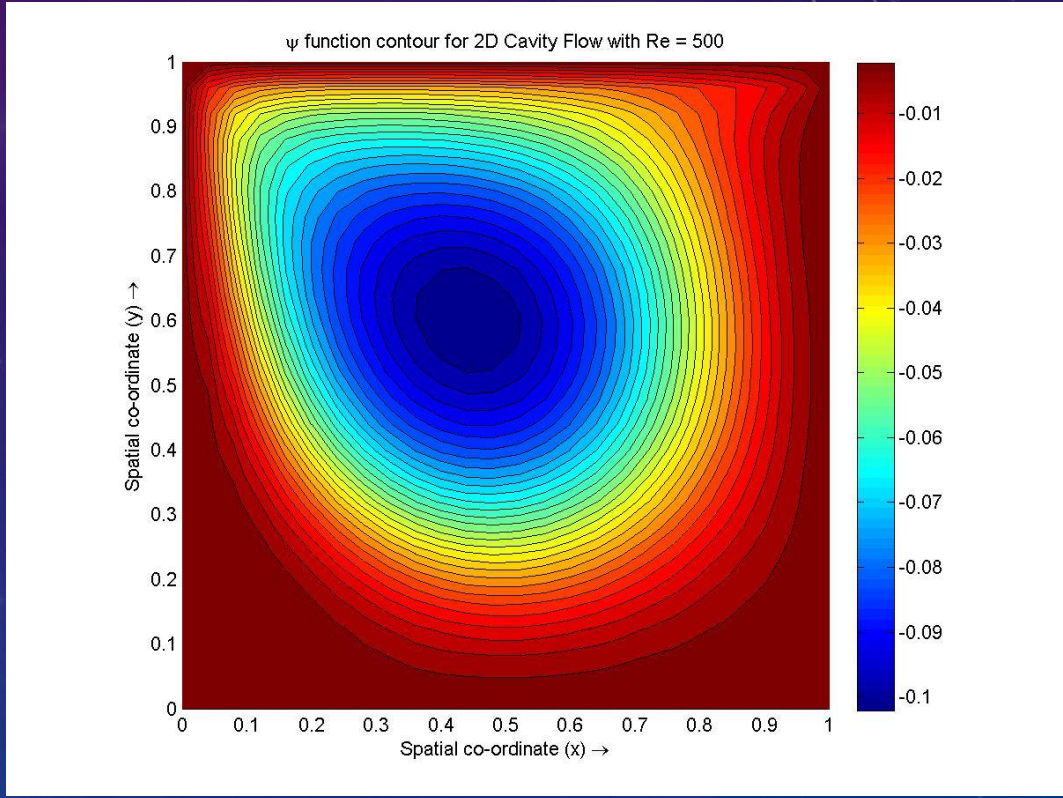
Y COMPONENT OF VELOCITY



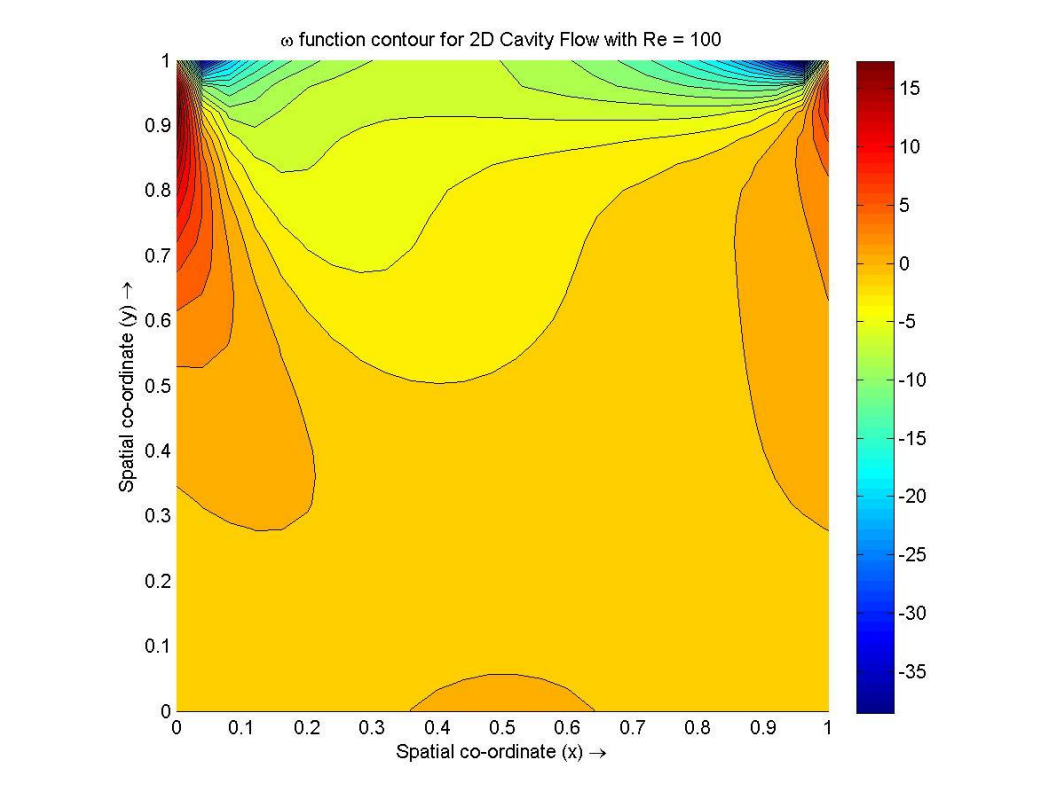


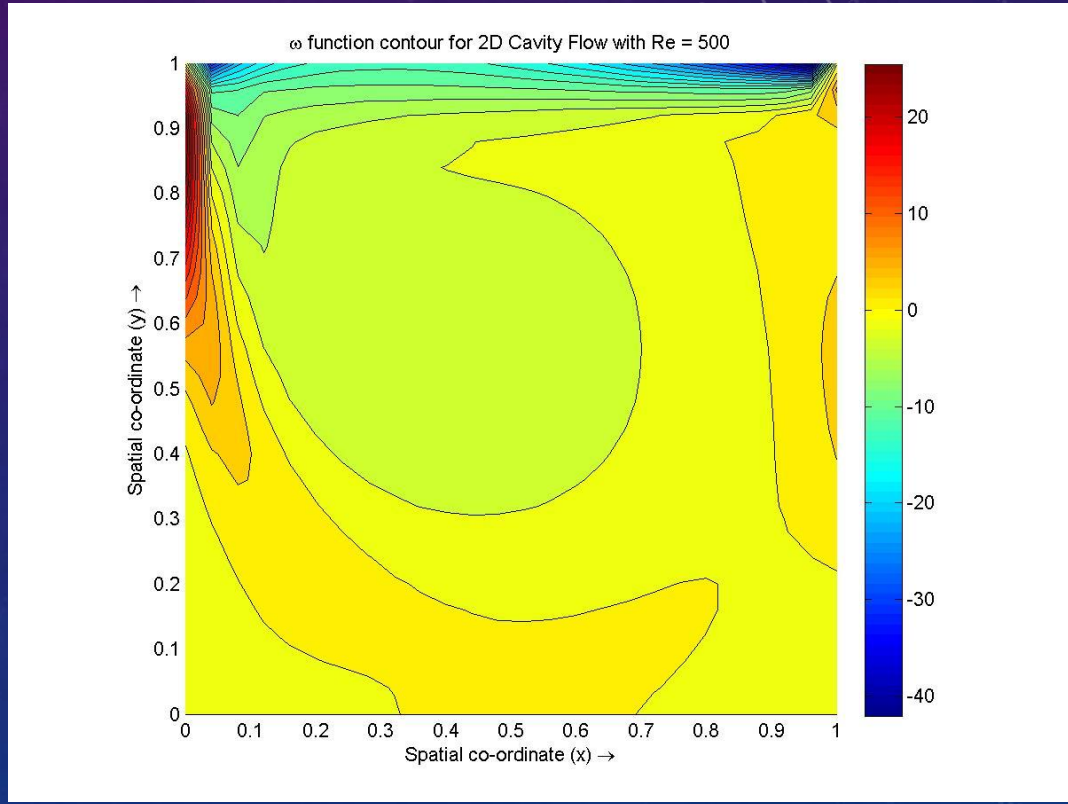
STREAM FUNCTION



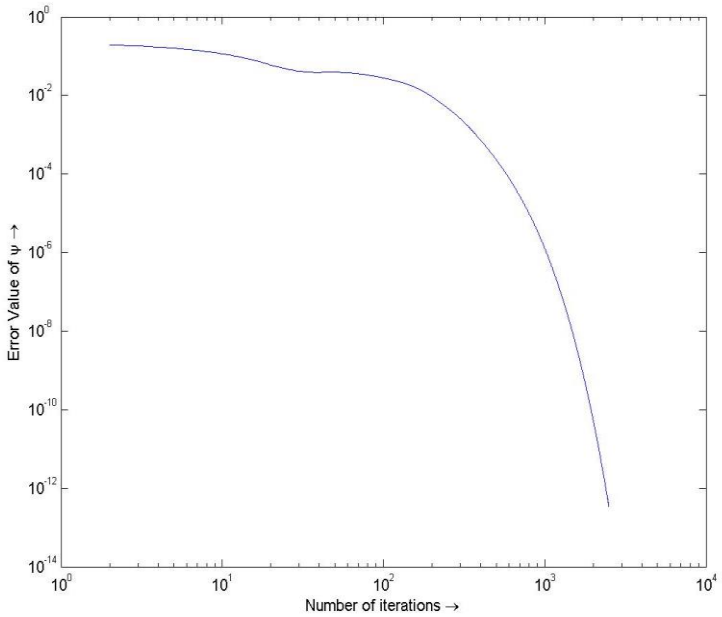


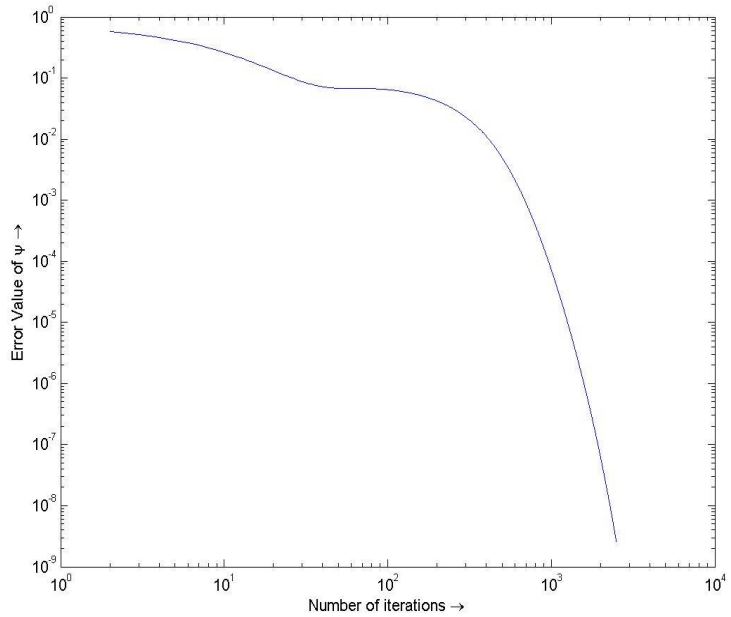
VORTICITY

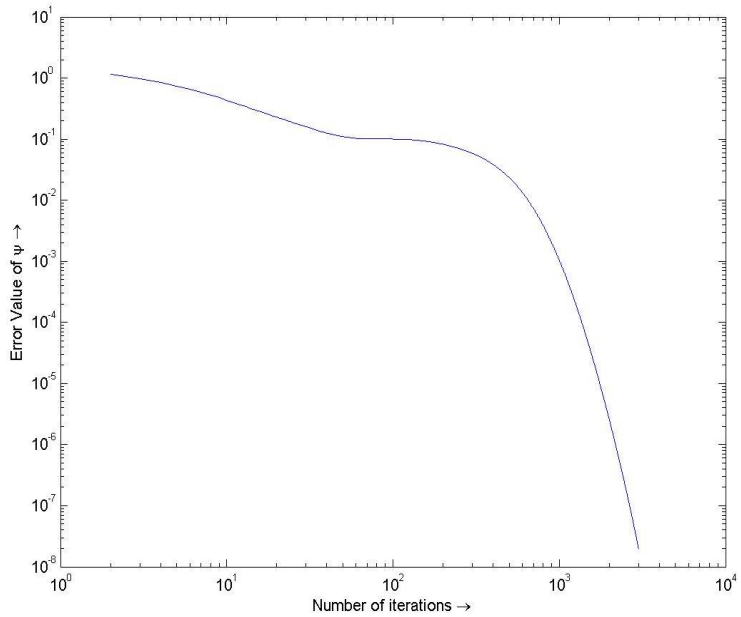

ω function contour for 2D Cavity Flow with Re = 100
Spatial co-ordinate (y) →
Spatial co-ordinate (x) →
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
15
10
5
0
-5
-10
-15
-20
-25
-30
-35


ω function contour for 2D Cavity Flow with Re = 500
Spatial co-ordinate (y) →
Spatial co-ordinate (x) →
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
20
10
0
-10
-20
-30
-40

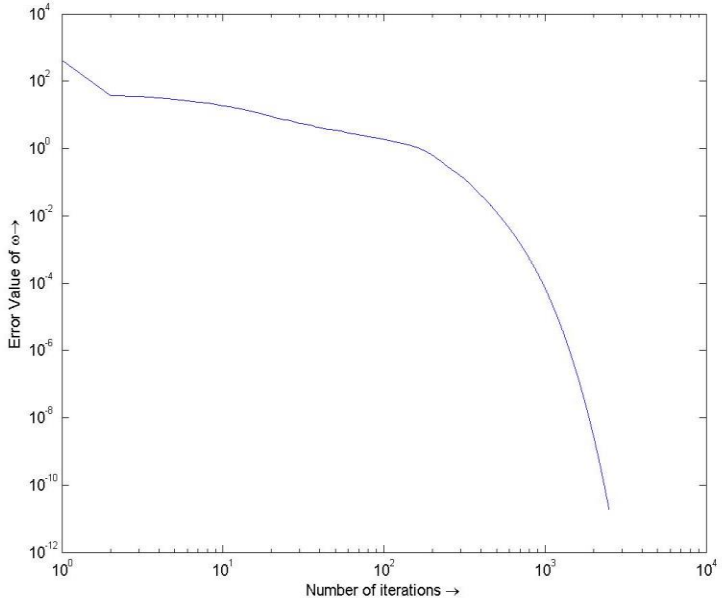
RESULTS FOR DIFFERENT GRID SIZES ,WITH RE = 500 STREAM FUNCTION (15X15, 25X25,35X35)

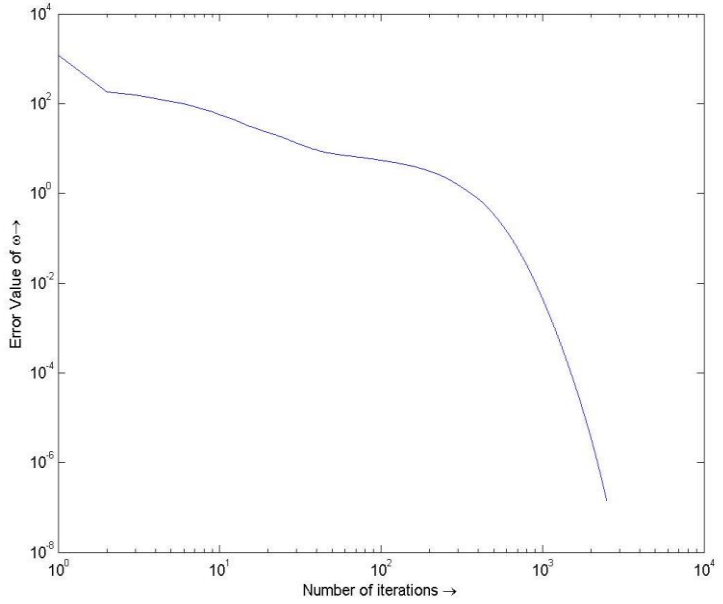


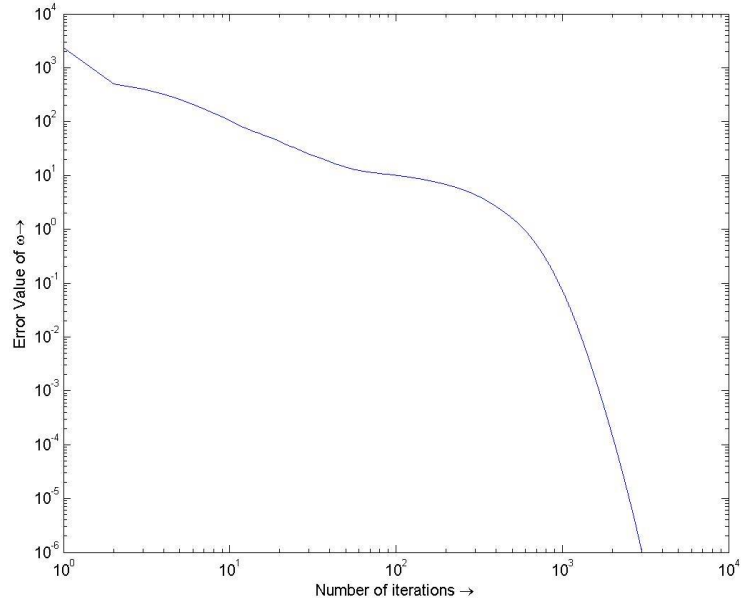




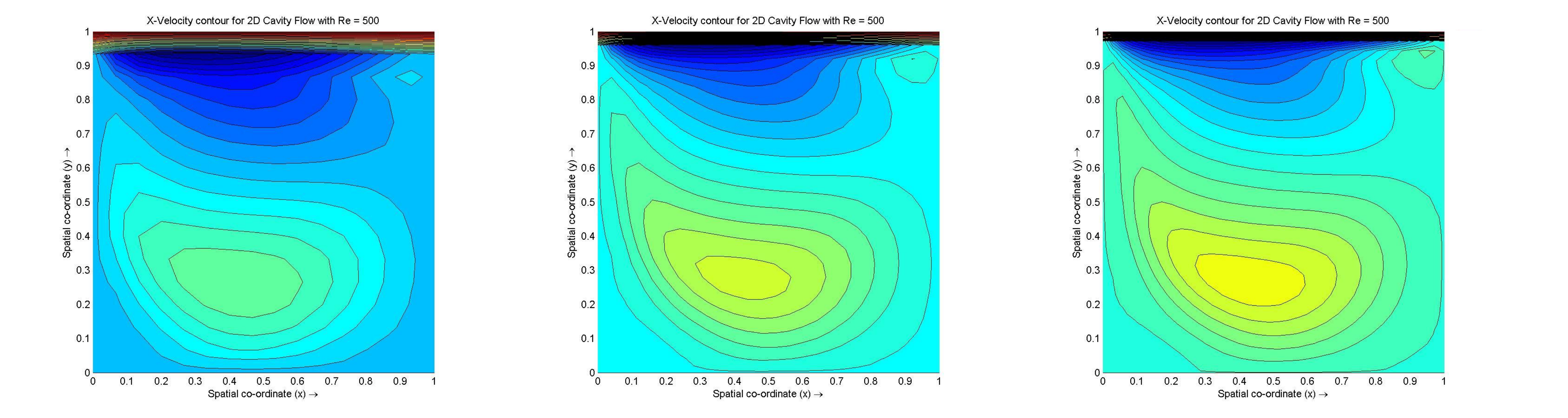
VORTICITY


Error Value of ω→
Number of iterations →

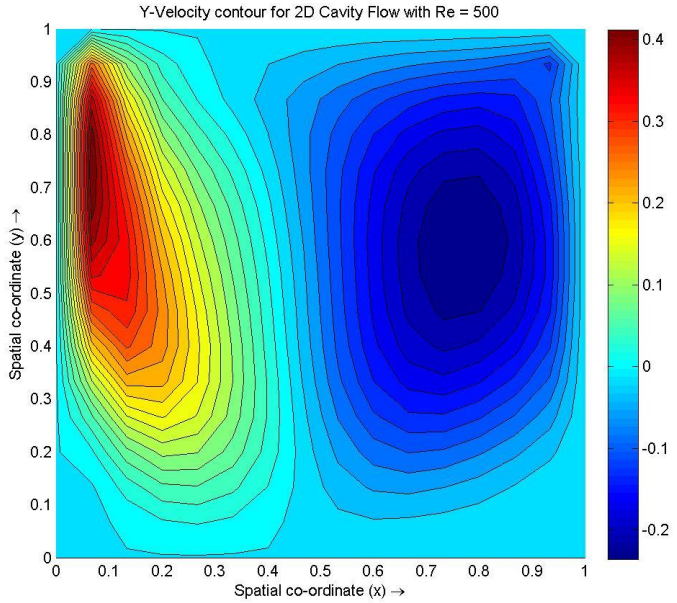

Error Value of ω→
Number of iterations →

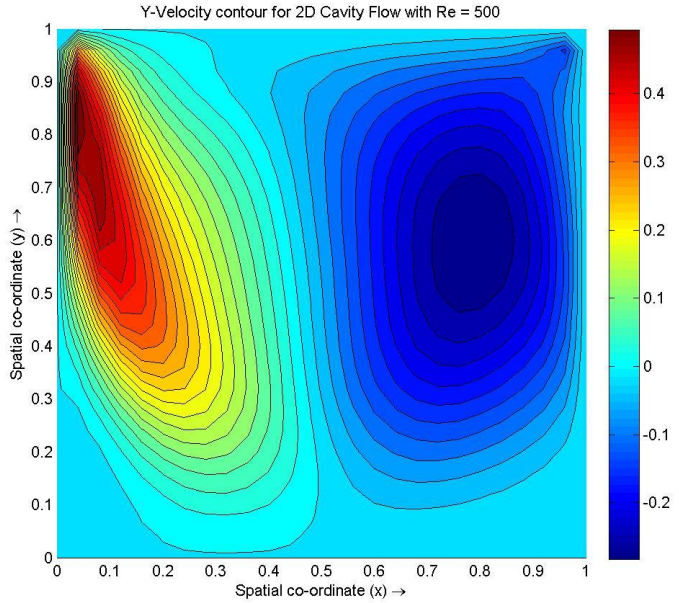

Error Value of ω→
Number of iterations →

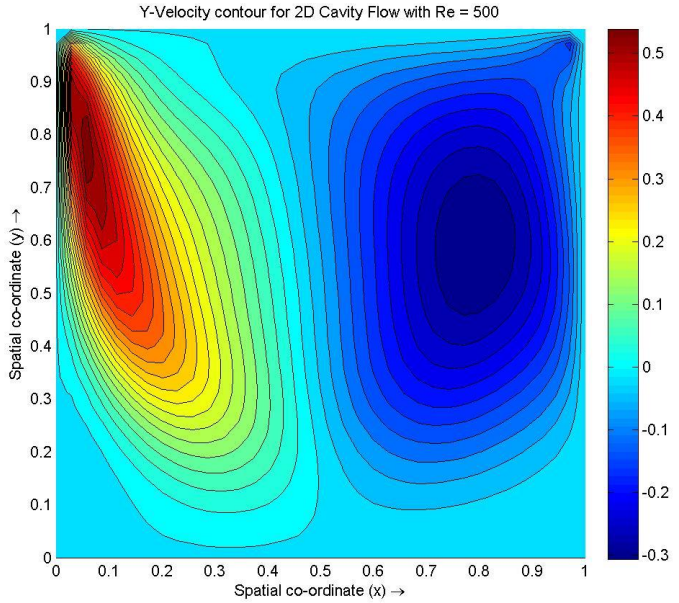
X COMPONENT OF VELOCITY



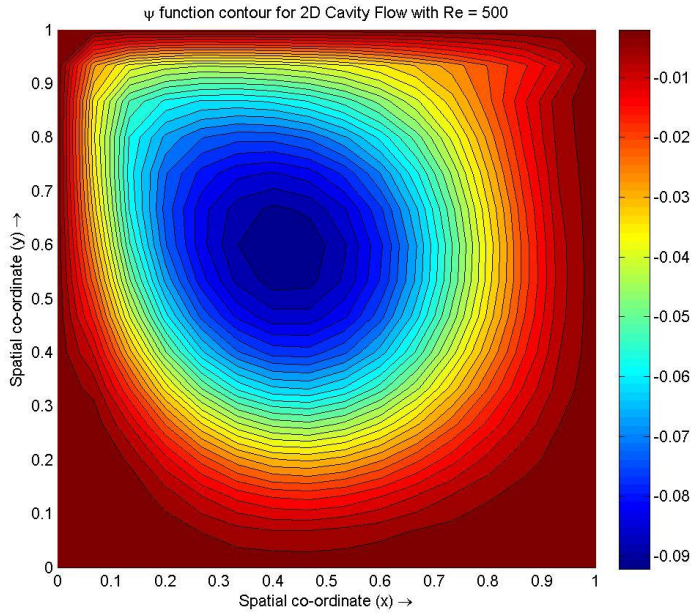
Y COMPONENT OF VELOCITY

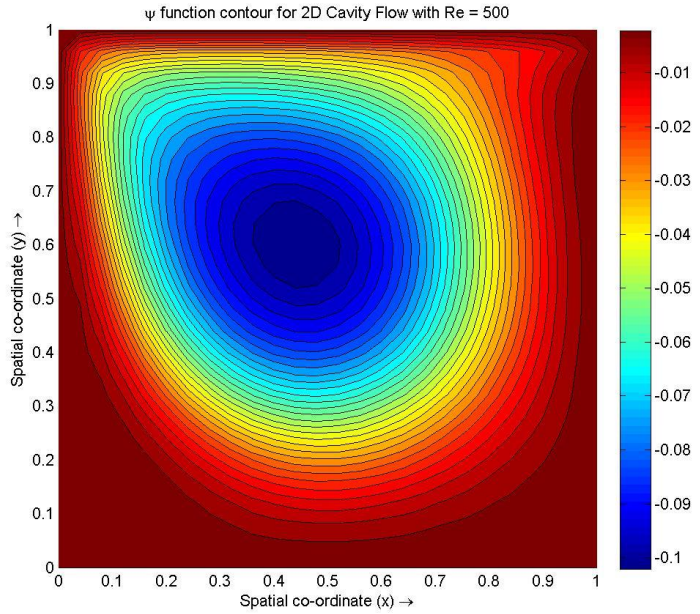


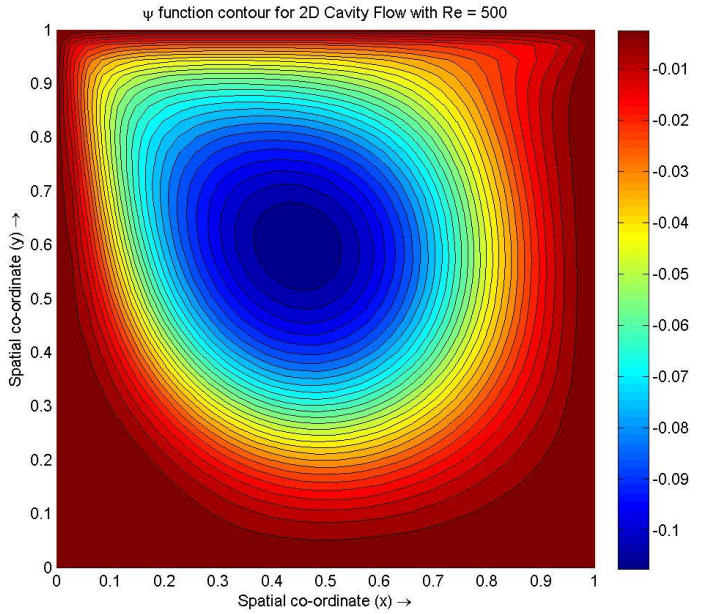




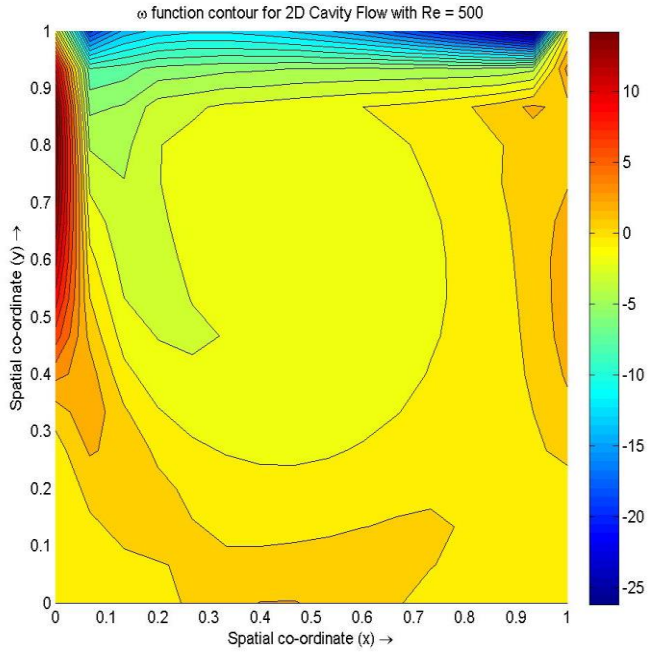
PSI FUNCTION – SMOOTH AND CONTOUR

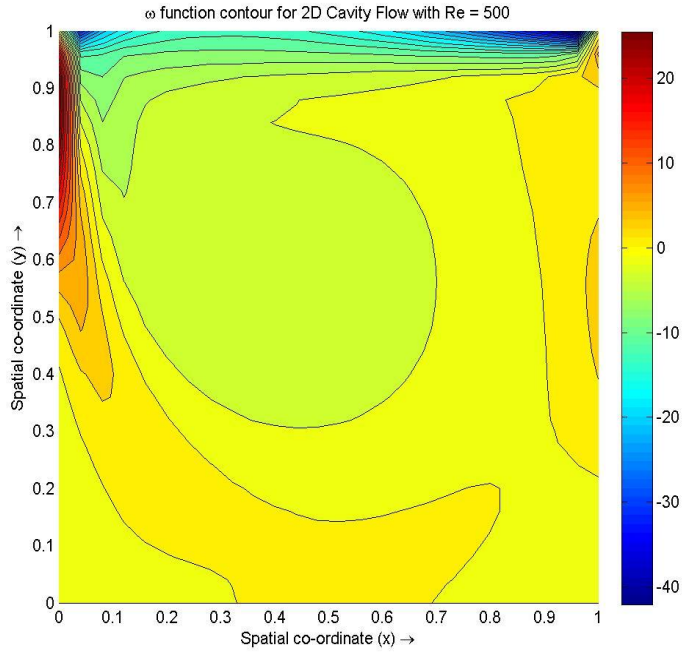


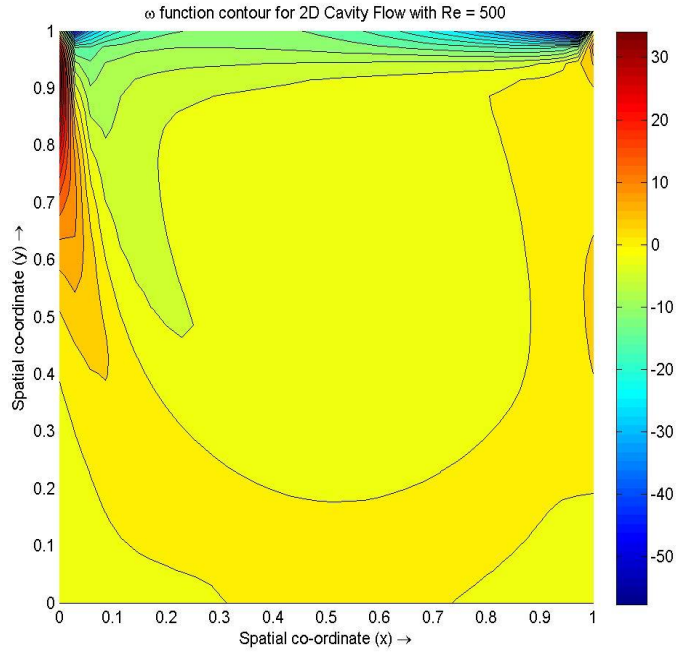




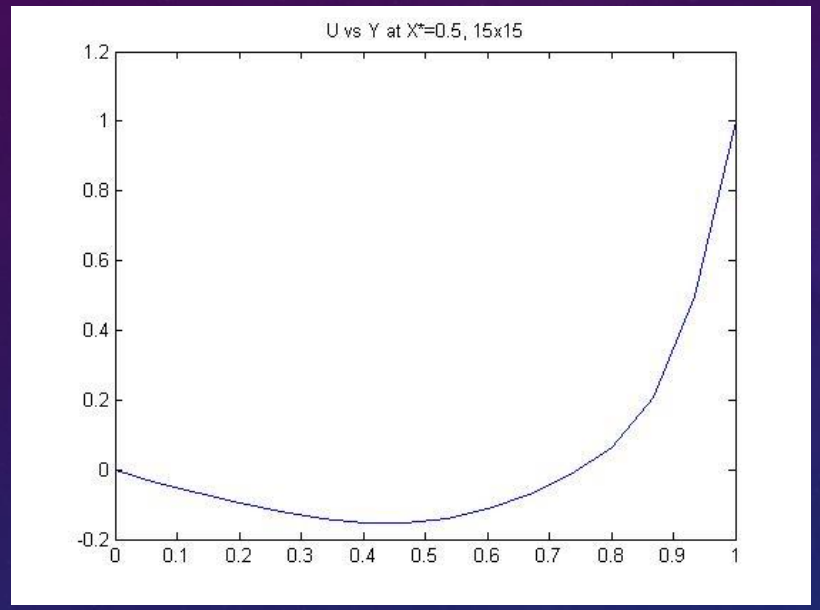
VORTICITY – SMOOTH AND CONTOUR

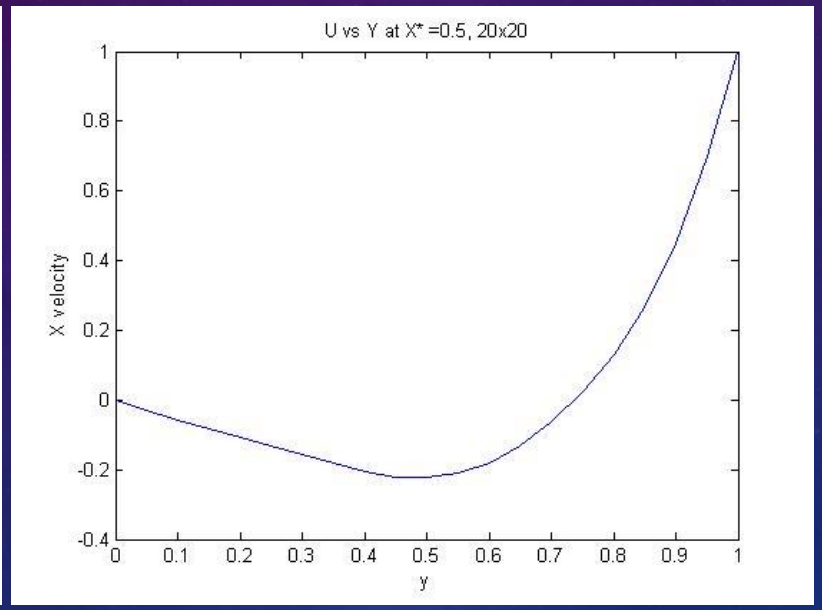


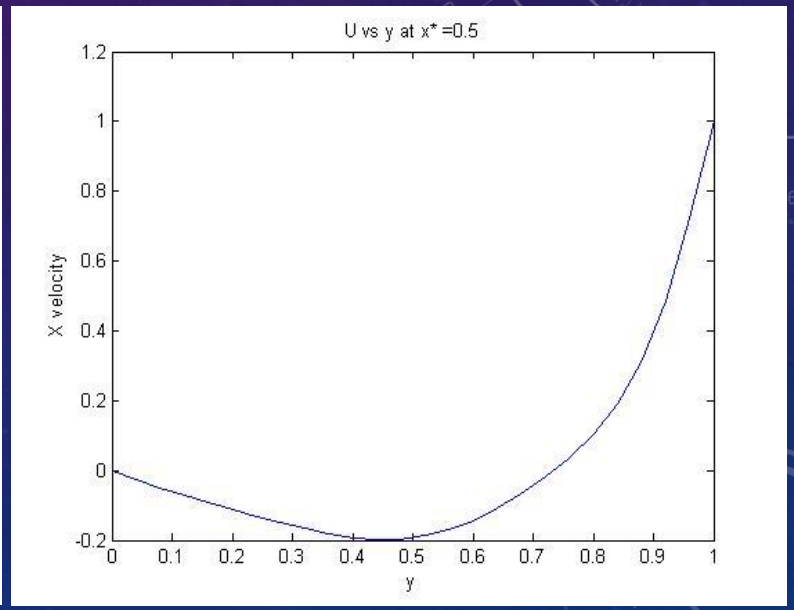




U VS Y AT X*=0.5, FOR RE =100 AND DIFFERENT GRID SIZES (15X15,20X20 &25X25)

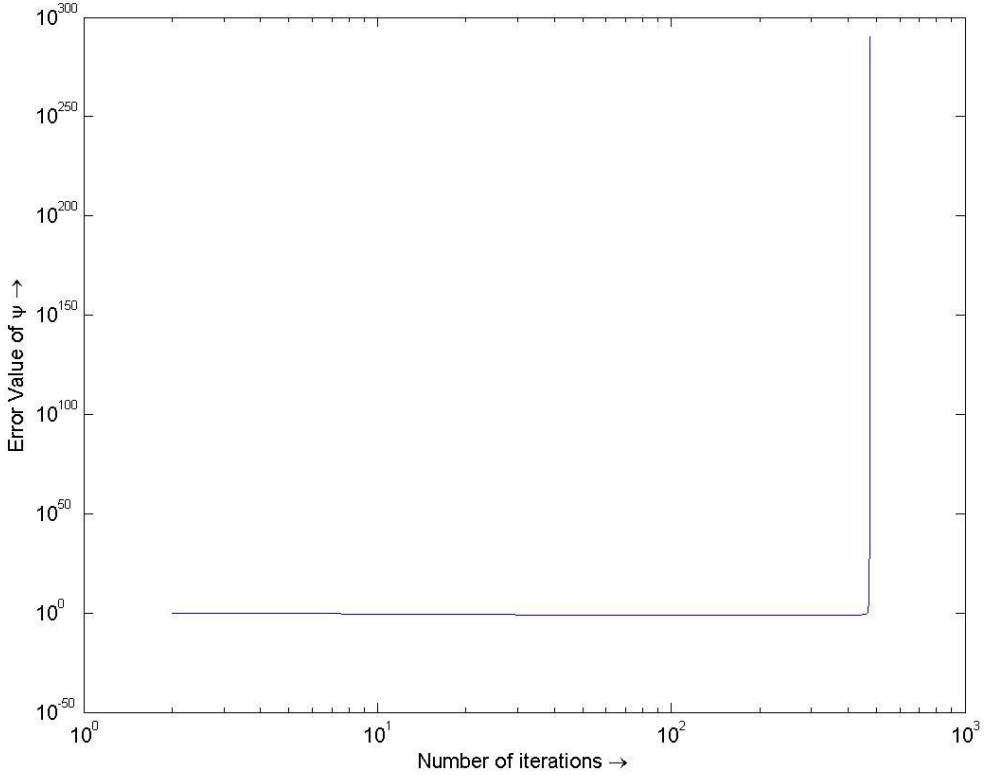


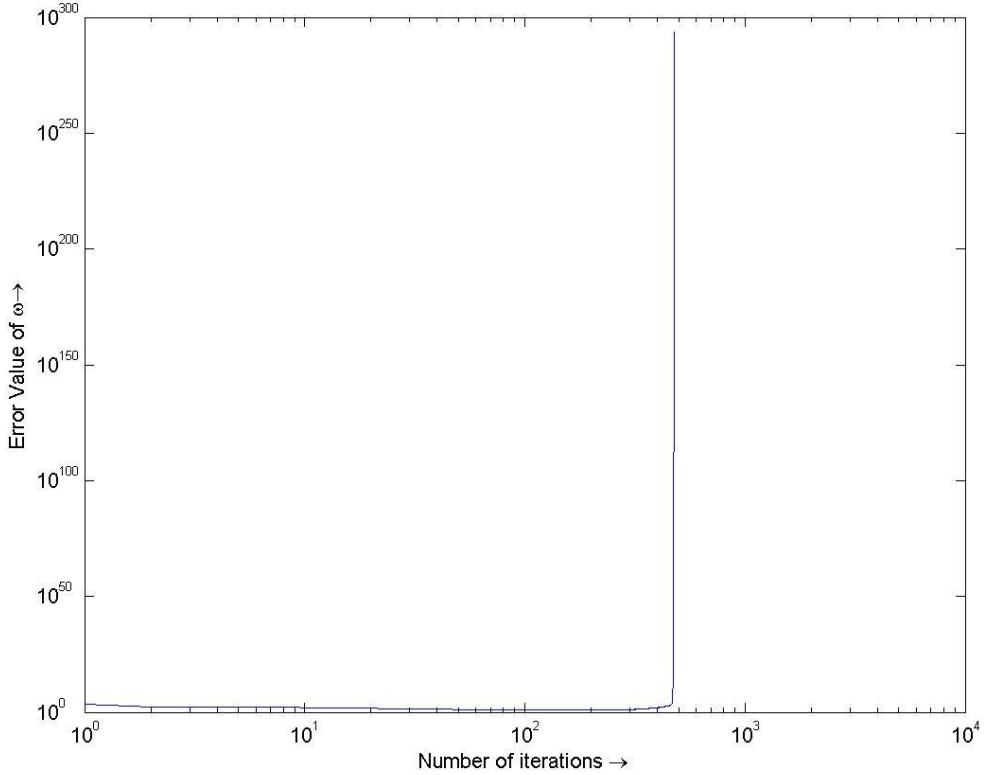




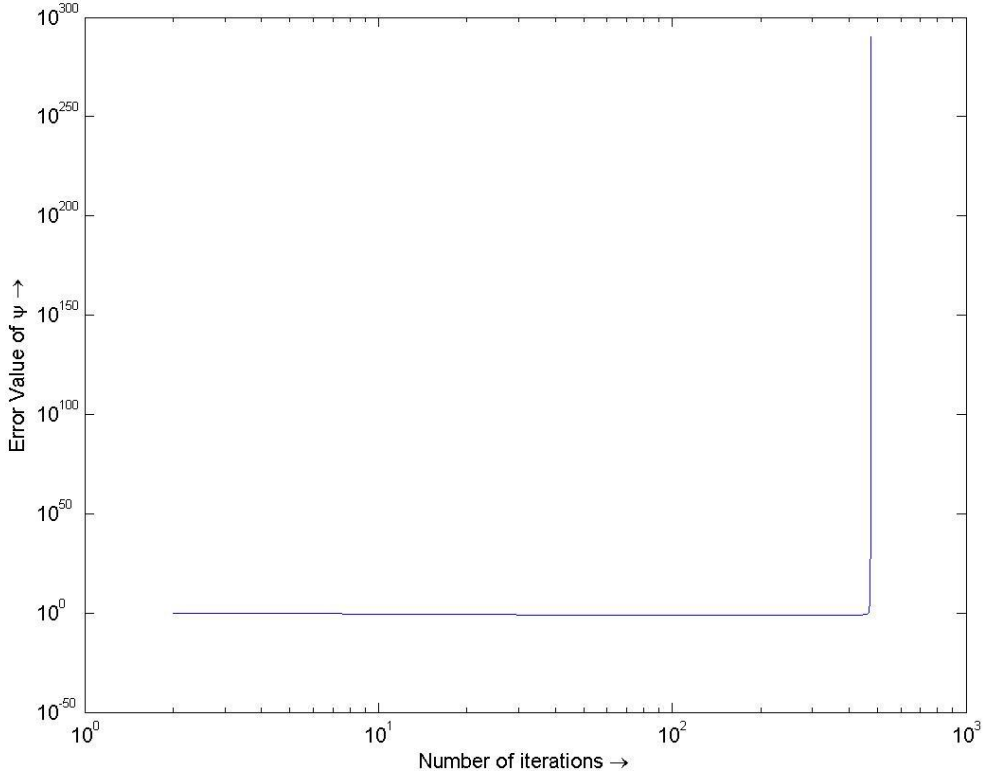
The consistency is maintained.

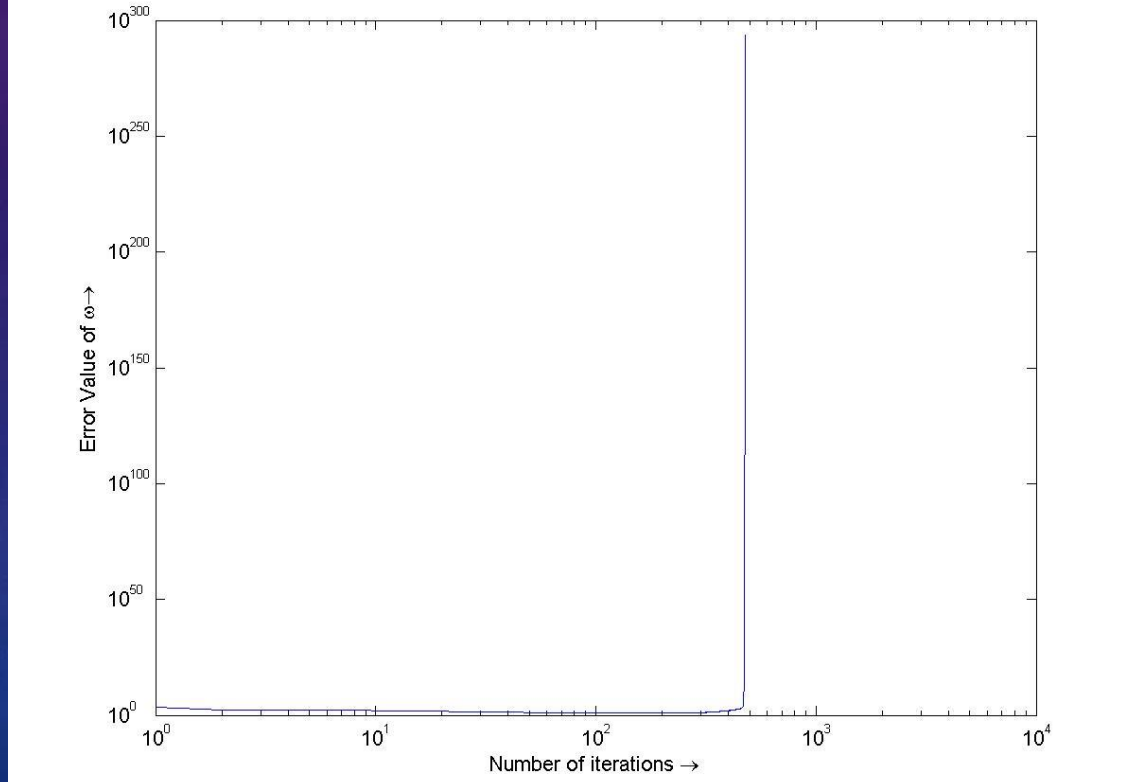
RESULTS FOR A 100X100 GRID SIZE ,WITH RE = 500 NOT CONVERGENT!





RESULTS FOR A 35X35 GRID SIZE ,WITH RE = 1000 NOT CONVERGENT!





But it converges for Re = 500!

INFERENCES AND CONCLUSIONS

- The stream function always converges faster than the vorticity function as per the formulation
- U velocity and vorticity contours show the presence of two minor vortices in the lower edges of the wall
- A big vortex (region of prominent vorticity) is found in the middle of the box as well
- V velocities are maximum close to wall, thus contributing for the vorticity as well
- As grid size increases, details are captured in a finer sense, while the overall characteristics are maintained.
- Convergence is not guaranteed for the same grid size for different Reynolds numbers
- Convergence is not guaranteed for the same Reynolds numbers and for different grid sizes

GMRES METHOD : OUTLINE

- Introduction
  - Related terminologies
  - Iterative methods
- Krylov iterative methods
  - Arnoldi, FOM, IOM, GCR etc
- GMRES method
- Variations of GMRES
  - GMRES with GS, Householder, Restarted, Truncated
- Comparison with other methods

INTRODUCTION

- RELATED TERMINOLOGIES
- ABOUT ITERATIVE METHODS
- KRYLOV SUBSPACES

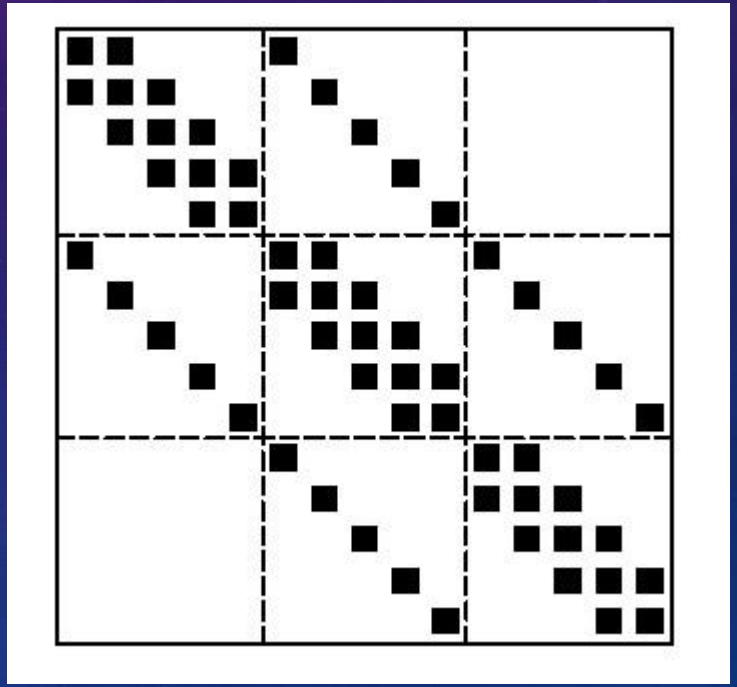
RELATED TERMINOLOGIES

- Problem : To solve $\boldsymbol{A}x = \boldsymbol{b}$
- Sparse matrix – A matrix with very few non-zero entities
- Projection methods – Give a good approximation even when $\boldsymbol{b}$ does not lie in column space of $\boldsymbol{A}$ (or $\boldsymbol{A}$ is singular)
- Residual Vs Error – We work with the residual as exact solution is not known

Residual -> $|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_{\boldsymbol{num}}|$

Error -> $|x_{exact} - x_{num}|$

- Krylov subspaces – Polynomial approximation of A^{-1}



A sparse matrix

PROJECTION METHODS

- Grahm - Schmidt
- Modified GS
- Householder algorithm

KRYLOV SUBSPACE METHODS

- Arnoldi algorithm – Uses GS orthonormalization on Krylov subspace
- FOM (Full orthonormalization method)
- IOM and DIOM (Incomplete orthonormalization, when m is too large)
- GMRES - Modified version of Arnoldi algorithm, improves accuracy

GMRES METHODS

OUTLINE

1. GMRES GENERAL ALGORITHM
2. GMRES WITH MODIFIED GS ORTHONORMALIZATION
3. ISSUES WITH GMRES
4. RESTARTED AND TRUNCATED GMRES

GMRES : BASIC ALGORITHM

- **Algorithm**:

1. Choose any x_0 and compute $r_0 = b - Ax_0$

and set $v_1 = \frac{r_0}{\beta}$ $where\ \beta = |r_0|$.

2. Iterate : for j = 1,2….k… until satisfied do

$h_{i,j} = (Av_j, v_i)$ for i = 1,2…j

$\widehat{v_{j+1}} = Av_j - \sum_{i=1}^{j} h_{i,j}\, v_i$

$h_{j+1,j} = |v_{j+1}|$

and $v_{j+1} = \frac{\widehat{v_{j+1}}}{h_{j+1,j}}$ if $h_{j+1,j}$ is not zero.

3. Get the approximate solution

$x_k = x_0 + V_k y_k$ where y_k minimizes the residual.

MINIMIZING THE RESIDUAL

- **Minimizing the residual**:

Any vector x_k in $x_0 + K_m$ can be written as

$x_k = x_0 + V_m y$ where V_m is the matrix containing m vectors of K_m, $\{v_1, Av_1, \dots, A^{m-1}v_1\}$.

So the residual becomes:

$$J(y) = |b - Ax| = |b - A(x_0 + V_m y)|$$

Now, $b - Ax_0 = b - A(x_0 + V_m y)$

$$= r_0 - A V_m y$$

$$= \beta v_1 - V_{m+1}\widetilde{H_m} y$$

$$= V_{m+1}(\beta e_1 - \widetilde{H_m} y)$$

As vectors of V_{m+1} are orthonormal, we get

$$J(y) = \left|\beta e_1 - \widetilde{H_m} y\right|$$

Hence y_k can be found which minimizes residual.

SOLVING LEAST-SQUARES PROBLEM

- To solve the least squares problem $|\beta e_1 - \widetilde{H_m} y|$, we transform the Hessenberg matrix $\widetilde{H_m}$ into an upper triangular matrix R by multiplying it with series of appropriate rotations. The row matrix $\bar{g}_0 = \beta e_1$ is also appropriately transformed into $\bar{g}_m$ and the problem reduces to

$$\mathrm{J(y)} = \min\left|\beta e_1 - \widetilde{H_m} y\right| = \min|\bar{g}_m - \bar{R}_m y|$$

which can be solved as: $y_m = {R^{-1}}_m g_m$

and the residual satisfies: $J(y) = g_m[m+1,1]$

GMRES WITH MODIFIED GS

- The GMRES algorithm can also be used more effectively by implementing the modified GS ortho-normalization process. The only changes in algorithm are reflected below:

Step 2: Iterate : for j = 1,2….k… until satisfied do

define $w_j = Av_j$

For i = 1,2…j

$$h_{i,j} = (w_j, v_i)$$

$$w_j = w_j - \sum_{i=1}^{j} h_{i,j}\, v_i$$

End

$h_{j+1,j} = |w_j|$

and $v_{j+1} = \frac{w_j}{h_{j+1,j}}$ if $h_{j+1,j}$ is not zero.

ISSUES WITH PRACTICAL IMPLEMENTATION

- As the algorithm does not provide approximate solution at each step, it is difficult to know when to stop.
- GMRES becomes impractical when m is too large due to cost of computing and storage.
- To overcome these problems, GMRES can be either restarted after every *k* steps or use truncation to run the Quasi-GMRES versions (QGMRES and DQGMRES)

RESTARTED GMRES

- Algorithm :

1. Start with v_1 of norm 1 and compute the V_k matrix using the earlier algorithm. (For a certain k)
2. Find $x_k = x_0 + V_k y_k$, if satisfied stop, otherwise $x_0 = x_k$ and follow the same steps again till satisfied.

Note that this algorithm, as with GMRES breaks only at the convergence where numerical solution is the exact solution.

TRUNCATED ALGORITHM

- Algorithm:

Use the following incomplete orthogonalization instead of Arnoldi algorithm as following:

For $j = 1,2 \ldots m$ Do

Compute $w = Av_j,$

For $i = \max\{1, j - k + 1\}, \ldots j$ Do

$h_{i,j} = (w, v_i)$

$w = w - h_{ij} v_i$

EndDo

QGMRES

- …(Continued)

  Compute $h_{j+1,j} = |w|$ and $v_{j+1} = w/h_{j+1,j}$

  EndDo

Comments:

- Quasi –GMRES is similar to the IOM algorithm.
- It saves computational cost but not the storage cost.

COMPUTATIONAL COSTS

- In assessing the computational cost of the algorithm, we ignore the computation required for the least squares solution y_k as k is much smaller than N.
- The major contributors to computational cost are then

1. Arnoldi vectors computation : $k(k+1)N + kNz$ multiplications where Nz is the number of non zero elements.
2. Formulation of $x_0 + V_k y_k$ which requires kN multiplications

Hence the total multiplications now become $k(k+2)N + kNz$ with average as $(k+2)N + Nz$.

COMPARISON WITH OTHER ALGORITHMS

- GCR and ORTHODIR require on an average $\frac{1}{2}(3k+5)N + Nz$ multiplications to produce the same result.
- Hence GMRES is always less expensive compared to GCR and ORTHODIR, and is almost $\frac{1}{3}^{rd}$ when k is large.
- For restarted versions, results are shown below:

Method	Multiplications	Storage
GCR	((3m+5)/2)N+Nz	(2m+l)N
GMRES	(m+3+1/m)N+NZ	(m+2)N

CONCLUSIONS

- GMRES method involves less computation and less storage memory
- It converges fast and is extremely useful for sparse matrix equations
- It breaks down only at the point of convergence hence it is numerically stable

REFERENCES

- References - Lid Driven Cavity Problem, Vaidehi Ambatipudi (for the reference graphs)
- Saad, Yousef, and Martin H. Schultz. "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems." *SIAM Journal on scientific and statistical computing* 7.3 (1986): 856-869.
- Saad, Yousef “Iterative methods for Sparse linear systems” 2nd Edition, January 3rd 2000.

THANK YOU