Lid-Driven Cavity problem

Foundations of Computational Fluid Dynamics AM5630

Name - Anand Zambare

Roll no. - AM21S004

Summary

The lid-driven cavity problem has been solved using 2-D in-compressible Navier-Stokes equations in square domain. The formulation used for solving the velocity field is stream function-vorticity formulation. After getting the converged values of velocity field, pressure Poisson equation is solved to get pressure field in the domain. The different plots are created using post-processing tools in Python. The plots are compared to the reference paper of Ghia and it was found that the solution is correct for given problem. The boundary conditions used here for the stream function and the vorticity are derived using the Taylor's series and the definitions of both respectively. The boundary conditions for velocity and pressure can be taken from the physical sense. Here we have used pseudo time-marching schemes to solve and reach the steady state solution.

1 Problem Definition

Given problem is to solve for the steady state velocity and pressure field using 2-D, in-compressible Navier-Stokes equations the given domain. The size of domain is user defined (taken as L = W = 1 units) where L is the length along X-axis and W is the height along Y-axis also the no. of grid points are defined by user(Nx and Ny). We have to write a generalized code to solve a steady 2-D, in-compressible N-S equations using the vorticity and stream function formulation. The Boundary conditions are to be implemented are of Dirichlet type for velocity at all walls as well as at lid. The Neumann type boundary conditions are to be imposed for pressure and the boundary conditions for the vorticity and stream function should be derived properly. Figure 1 explains the domain for the lid-driven problem and also for boundary conditions for the primitive variables like velocity and pressure. Other boundary conditions are discussed and derived in separate section of boundary conditions.

Constants for the problem :

L = Length of domain = 1.0 meter (Taken here)
W = Height of domain = 1.0 meter (Taken here)
N_x = No. of grid points in the X direction = 51 (Taken here)
N_y = No. of grid points in the Y direction = 51 (Taken here)
Reynolds number (Re) = 100 (Taken here)
Density (ρ) = 1.0 kg/(m^3)
kinematic viscosity = 0.01 (m^2/s)
Grid spacing between points in x direction (dx) = L/(Nx-1).
Grid spacing between points in y direction (dy) = W/(Ny-1).

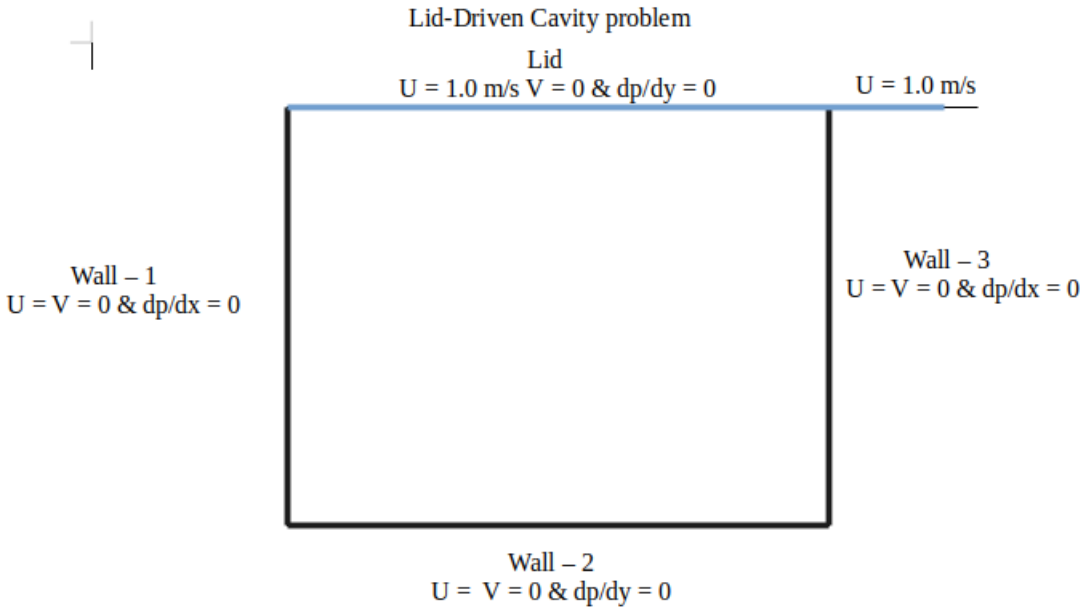

Figure 1: Problem definition and the primitive variable boundary conditions.

time step (dt) = 0.001.

2 Governing Equations (Navier-Stokes equations)

For the given problem, to find out the steady state velocity and pressure distribution we need to solve steady 2-D in-compressible Navier-Stokes equations. Let's look at the equations now:

Continuity equation:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{1}$$

Momentum equations:

$$\frac{\partial u}{\partial t} + u \cdot \frac{\partial u}{\partial x} + v \cdot \frac{\partial u}{\partial y} = -\frac{1}{\rho} \cdot \frac{\partial p}{\partial x} + \nu \cdot (\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{2}$$

$$\frac{\partial v}{\partial t} + u \cdot \frac{\partial v}{\partial x} + v \cdot \frac{\partial v}{\partial y} = -\frac{1}{\rho} \cdot \frac{\partial p}{\partial y} + \nu \cdot (\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{3}$$

We will use Stream function vorticity formulation for solving this problem. This formulation involves more equations for solving the situation. The vorticity transport equation and pressure Poisson equation which are derived using the momentum equations only. Also the definition of vorticity is involved in this as well as stream function definition. Let's look at it one by one.
Stream Function:

$$u = \frac{\partial \psi}{\partial y} \tag{4}$$

$$v = -\frac{\partial \psi}{\partial x} \tag{5}$$

Vorticity:

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{6}$$

Equation 4 and Equation 5 can be used to define the ω i.e. vorticity in terms of stream function and it can be re arranged to get the following equation 7.

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega \tag{7}$$

Also using the momentum equations i.e. equation 2 and equation 3 we can derive the vorticity transport equation. Equation 8 is vorticity transport equation

$$\frac{\partial \omega}{\partial t} + u \cdot \frac{\partial \omega}{\partial x} + v \cdot \frac{\partial \omega}{\partial y} = \nu \cdot (\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}) \tag{8}$$

Using equation 7 and equation 8 with properly derived boundary conditions we have to solve for the vorticity and stream function in the given domain. once we have solution for both then we can solve for other primitive variable i.e. pressure P. The pressure Poisson equation is used for solving the pressure in the domain. It can be derived using the momentum equations. Equation 9 is pressure Poisson equation.

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = (\frac{\partial u}{\partial x})^2 + (\frac{\partial v}{\partial y})^2 + 2 \cdot \frac{\partial u}{\partial x} \frac{\partial v}{\partial y} \tag{9}$$

3 Boundary conditions

The boundary conditions are divided into the two parts. One part is for the primitive variables and other part is for the derived boundary conditions for vorticity and stream function.
A) Primitive boundary conditions:
i) Walls (wall 1, wall 2 and wall 3) :
we have to implement the No-slip boundary condition for velocity i.e. u = v = 0 and the normal pressure gradient is 0 for all walls.

$$u = v = 0 \tag{10}$$

$$\frac{\partial p}{\partial n} = 0 \tag{11}$$

ii) The lid i.e. the top side of the square domain, the x direction velocity is defined and y-direction velocity is 0. so v = 0 and u = U_0

B) Derived boundary conditions for the vorticity and stream function:
i) Boundary conditions for stream function -
The stream function value normally is taken as any constant at the wall. By the definition of stream function the velocities are the derivatives of stream function i.e. ψ so at wall for velocities to be zero the value of stream function should be constant. Here we can take the value to be zero at all walls. While taking the constant value for stream function we should keep in mind that the difference between the values of stream function is equal to the volume flow rate between those two stream lines. Here there is no such flow of fluid so we can take the value to be zeros at all walls. let the indexing be starting from 0 and imax and jmax are the last nodes for the x and y directions respectively. Now the boundary conditions can be written as $\psi[0, :] = \psi[:, 0] = \psi[:, \text{imax}] = 0.0$ B) Derived boundary conditions for the stream function and vorticity:
Stream function (ψ):
At walls the stream function is taken as zero. So we can write

$$\psi_{wall1} = 0.0 \tag{12}$$

$$\psi_{wall2} = 0.0 \tag{13}$$

$$\psi_{wall3} = 0.0 \tag{14}$$

For the lid at the top, there is constant velocity, so we can write the stream function as a linear function of y only as follows.

$$\frac{d\psi}{dy} = u \tag{15}$$

so by integrating we can write it as follows:

$$\psi_{lid} = U \cdot y \tag{16}$$

So these are the four boundary conditions on the stream function. Now let's look at the vorticity function.
ii) Vorticity :
Vorticity boundary conditions are derived using Taylor's series as follows:

$$\psi_{wall+1} = \psi_{wall} + \Delta x \cdot \frac{\partial \psi}{\partial x}_{wall} + \frac{(\Delta x)^2}{2!} \cdot \frac{\partial^2 \psi}{\partial x^2} + ... \tag{17}$$

we have to use the definition of stream function and use it to define the vorticity boundary conditions. In general for wall since the velocities are zero the first derivative vanishes and we will get the second order derivative as vorticity. so the equation (17) can be re arranged as follows:

$$\omega_{wall} = \frac{2 \cdot (\psi_{wall} - \psi_{wall+1})}{(\Delta x)^2} \tag{18}$$

This is defined for the bottom wall i.e. wall 2, similarly we can utilize the Taylor's series and get boundary condition for wall 1 as well as wall 3.

$$\omega_{wall} = \frac{2 \cdot (\psi_{wall} - \psi_{wall+1})}{(\Delta y)^2} \tag{19}$$

$$\omega_{wall} = \frac{2 \cdot (\psi_{wall} - \psi_{wall-1})}{(\Delta y)^2} \tag{20}$$

Equation 18, equation 19 and equation 20 give the boundary conditions for the vorticity at wall 2 , wall 1 and wall 3 respectively

4 Numerical Formulation

We will use the central difference scheme to calculate the first as well as second order derivative. The generalized formula for first and second derivative with respect to x and y is given as follows:

$$\frac{\partial f}{\partial x} = \frac{f_{i+1,j} - f_{i-1,j}}{2 \cdot \Delta x} \tag{21}$$

$$\frac{\partial f}{\partial y} = \frac{f_{i,j+1} - f_{i,j-1}}{2 \cdot \Delta y} \tag{22}$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{f_{i+1,j} - 2 \cdot f_{i,j} + f_{i-1,j}}{(\Delta x)^2} \tag{23}$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{f_{i,j+1} - 2 \cdot f_{i,j} + f_{i,j-1}}{(\Delta y)^2} \tag{24}$$

The functions are defined in the code for finding out the first derivative w.r.t x as well as y and second derivative w.r.t x as well as y using equation 21 - equation 24. Using the Euler's forward difference scheme we can define the time derivative as follows

$$\frac{\partial f}{\partial t} = \frac{f_{i,j}^{n+1} - f_{i,j}^{n}}{\Delta t} \tag{25}$$

Now using the equation 21-25 we have to formulate the equation 4, 5, 7 and 8. These equations are solved in iterative manner to get the required converged values of the v, u, ψ and ω .
Equation 4 and 5 gives velocity using the stream function formulation.

$$u_{i,j} = \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2 \cdot \Delta y} \tag{26}$$

$$v_{i,j} = -\frac{\psi_{i+1,j} - \psi_{i-1,j}}{2 \cdot \Delta x} \tag{27}$$

The equation 7 is discritized as follows

$$-w_{i,j} = \frac{\psi_{i+1,j} - 2 \cdot \psi_{i,j} + \psi_{i-1,j}}{(\Delta x)^2} + \frac{\psi_{i,j+1} - 2 \cdot \psi_{i,j} + \psi_{i,j-1}}{(\Delta y)^2} \tag{28}$$

The equation 8 is discritized as follows

$$\frac{\omega_{i,j}^{n+1} - \omega_{i,j}^{n}}{\Delta t} + u_{i,j} \cdot \frac{\omega_{i+1,j} - \omega_{i-1,j}}{2 \cdot \Delta x} + v_{i,j} \cdot \frac{\omega_{i,j+1} - \omega_{i,j-1}}{2 \cdot \Delta y} = \nu \cdot \left(\frac{\omega_{i+1,j} - 2 \cdot \omega_{i,j} + \omega_{i-1,j}}{(\Delta x)^2} + \frac{\omega_{i,j+1} - 2 \cdot \omega_{i,j} + \omega_{i,j-1}}{(\Delta y)^2} \right) \tag{29}$$

5 Algorithm, Pseudo code or Flow chart

The algorithm for this problem is as shown in the figure 2 as below. we start with importing the necessary libraries. Define the constants for the problem like length, width, no.of grid points in x as well as y direction. Also defining the x and y vector which is like a mesh for given problem. Next step is to initialize the u, v and p i.e. velocity, pressure, omega (ω) and psi (ψ). After the initialization, we need to define the functions for calculating the derivatives i.e first and second derivatives with respect to x and y. Also define the boundary conditions using functions. Now the important step is to solve for psi using equation 28 and impose the boundary conditions on ψ. Next step is to calculate

the velocities using the equation 26 and equation 27. Next step is to go to the vorticity transport equation. This is used to calculate the next value of the ω using the equation 29. Now go back to equation 28 and calculate the psi and respective velocities and check for the convergence. If converged then go for solving pressure Poisson equation. Pressure can be solved from this. Complete algorithm is shown in the figure 2.

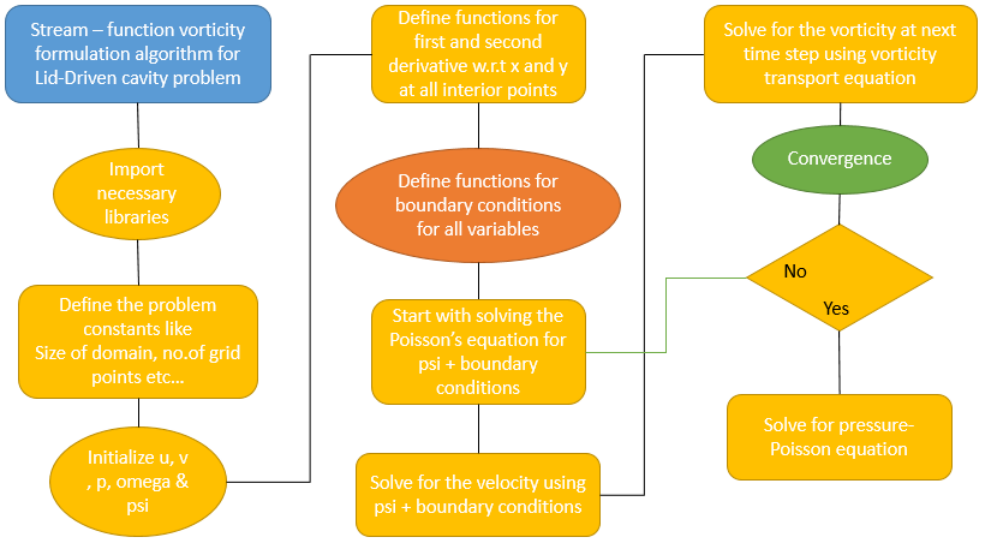

Figure 2: Flow Chart for the psi-omega formulation.

6 Results and discussion

Using the above algorithm with time step being 0.001 and the no.of steps to be 500, we have simulated the problem for 0.5 seconds. We can plot pressure contours, velocity vectors and the stream lines using the post processing techniques available in the Python. Figure 3 shows the velocity vector plots in the domain. It clearly shows that the velocity is equal to the top velocity at the lid surface and it's varying through out the domain. We can also see that there is formation of the primary as well as secondary vortex inside the domain. This can be better visualized using the streamline plots for the flow field. Stream lines and path lines coincide when the flow gets steady so we can say that the vortex motion of fluid can be traced using the streamlines. Figure 4 is the stream line plot for the given problem at t = 0.5 seconds. In the plot we can see the primary vortex developed in the center or nearby area of domain. Secondary small vortices can be seen in the corner of the domain. The plots generated are compared to the Ghia's paper for the verification of the solution obtained. We can see the similar plots generated in the Ghia's paper.

The additional plots asked in the question are vorticity contours and the u component of the velocity right at the center of the domain i.e. for x = 0.5 and the v velocity plot right at the center of the domain i.e for y = 0.5.The figure 5 is the vorticity distribution inside the domain. The color plot shows clearly the nature as of elliptic solution i.e. the solution for the Poisson's equation. Figure 6 and figure 7 are the required plot of y vs u at x = 0.5 and v vs x at the y = 0.5. The nature of plot

u vs y at x = 0.5 and v vs x at y = 0.5 can be compared with the reference paper and it was found that the solution is matching.

7 Python code

```
from numpy import *
import matplotlib.pyplot as plt
L = 1.0  square domain
Nx = Ny = 51  No.of grid points
U = 1.0  velocity of plate
nu = 0.01  Kinematic viscocity
rho = 1.0  Kg per m^3
dt = 0.0001
grid
dx = L/(Nx - 1)
dy = L/(Ny - 1)
x = linspace(0.0, L, Nx)
y = linspace(0.0, L, Ny)
X, Y = meshgrid(x, y)
deferror(f1, f2) :
err = 0
forminrange(0, Ny) :
forqinrange(0, Nx) :
err = err + abs(f1[i, j] - f2[i, j])
returnerr
initialization
w_pre = zeros((Ny, Nx))
chi_pre = zeros((Ny, Nx))
chi_new = zeros((Ny, Nx))
w_new = zeros((Ny, Nx))
wewillsolveonlysteadycasehere
for_inrange(15000) :
foriiinrange(1, Ny - 1) :
forjinrange(1, Nx - 1) :
chi_new[i, j] = 0.25*(chi_pre[i+1, j]+chi_pre[i-1, j]+chi_pre[i, j+1]+chi_pre[i, j-1]+(dx*dx*w_pre[i, j]))
chi_new[0, :] = 0.0
chi_new[:, 0] = 0.0
chi_new[:, -1] = 0.0
chi_new[-1, :] = U
letusconsiderFasdefinedinformulation
F1 = zeros((Ny, Nx))
foriiinrange(1, Ny - 1) :
forjinrange(1, Nx - 1) :
F1[i, j] = nu * (w_pre[i + 1, j] + w_pre[i - 1, j] + w_pre[i, j + 1] + w_pre[i, j - 1] - (4 * w_pre[i, j]))/(dx * *2)
F2 = zeros((Ny, Nx))
foriiinrange(1, Ny - 1) :
forjinrange(1, Nx - 1) :
F2[i, j] = (chi_new[i + 1, j] - chi_new[i - 1, j]) * ((w_pre[i, j + 1] - w_pre[i, j - 1])/(4 * dx * dx))
F3 = zeros((Ny, Nx))
```

$for iin range(1, Ny-1):$
$for jin range(1, Nx-1):$
$F3[i,j] = (chi_new[i,j+1] - chi_new[i,j-1]) * ((w_pre[i+1,j] - w_pre[i-1,j])/(4*dx*dx))$
$F = F1 + F3 - F2$
$for iin range(1, Ny-1):$
$for jin range(1, Nx-1):$
$w_new[i,j] = w_pre[i,j] + (dt * F[i,j]) w_new[0,:] = 2*(chi_new[0,:] - chi_new[1,:])/(dx**2)$
$w_new[:,0] = 2*(chi_new[:,0] - chi_new[:,1])/(dy**2)$
$w_new[:,-1] = 2*(chi_new[:,-1] - chi_new[:,-2])/(dy**2)$
$w_new[-1,:] = 2*(chi_new[-1,:] - chi_new[-2,:])/(dx**2)$

$\mathrm{w}_p re = w_new$
$chi_pre = chi_new$
$u = zeros((Ny, Nx))$
$v = zeros((Ny, Nx))$
$for iin range(1, Ny-1):$
$for jin range(1, Nx-1):$
$u[i,j] = (chi_new[i+1,j] - chi_new[i-1,j])/(2*dy)$
$for iin range(1, Ny-1):$
$for jin range(1, Nx-1):$
$v[i,j] = (chi_new[i,j+1] - chi_new[i,j-1])/(2*dy)$
$u[0,:] = 0.0$
$u[:,0] = 0.0$
$u[:,-1] = 0.0$
$u[-1,:] = U$
$v[0,:] = 0.0$
$v[:,0] = 0.0$
$v[:,-1] = 0.0$
$v[-1,:] = 0.0$
$Z1 = u[:,25]$
$Z2 = v[25,:]$
$plt.figure(1)$
$plt.contourf(X, Y, u, cmap =' jet')$
$plt.colorbar()$
$plt.figure(2)$
$plt.plot(Z1, y)$
$plt.figure(3)$
$plt.plot(x, Z2)$
$plt.figure(4)$
$plt.streamplot(X, Y, u, v, color =' black')$
$plt.show()$

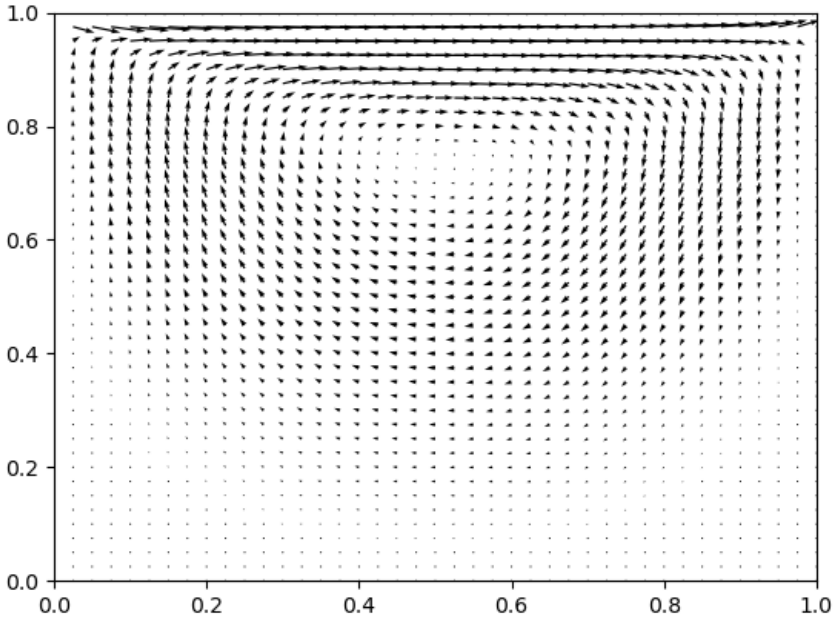

Figure 3: Velocity vectors in the domain.

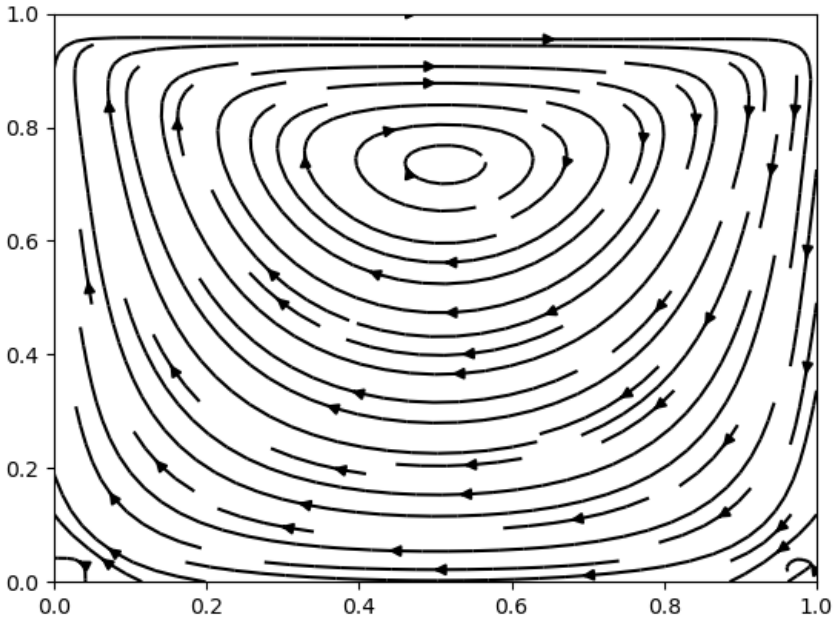

Figure 4: Streamlines showing the primary and secondary vortices.

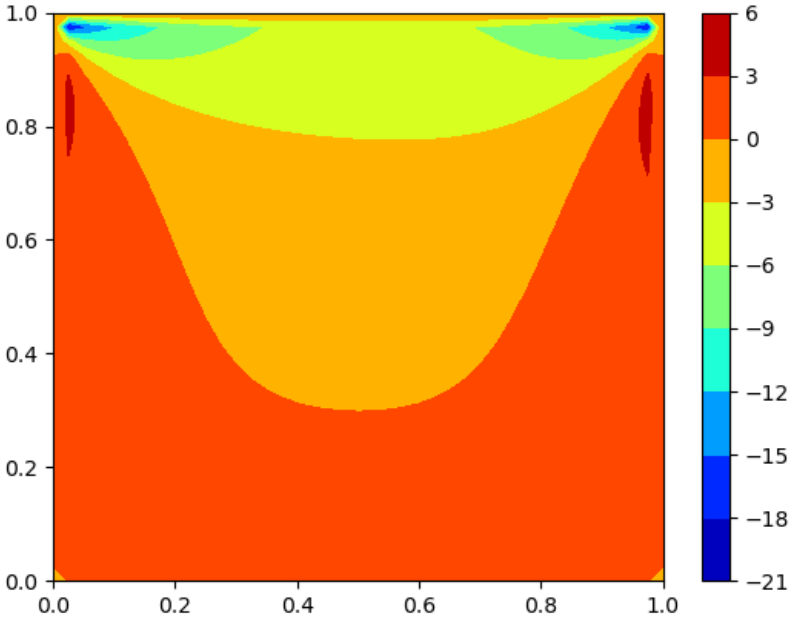

Figure 5: Vorticity distribution in the domain.

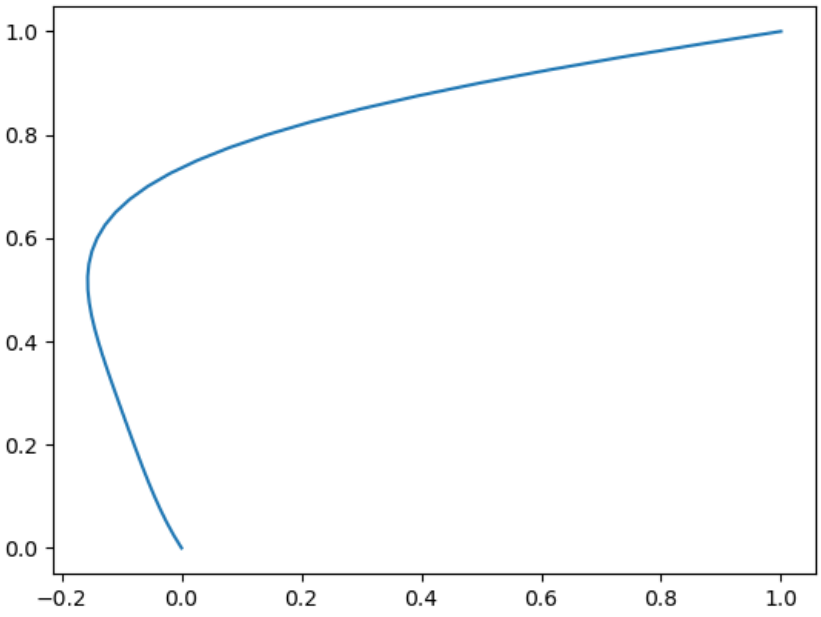

Figure 6: u velocity at x = 0.5.

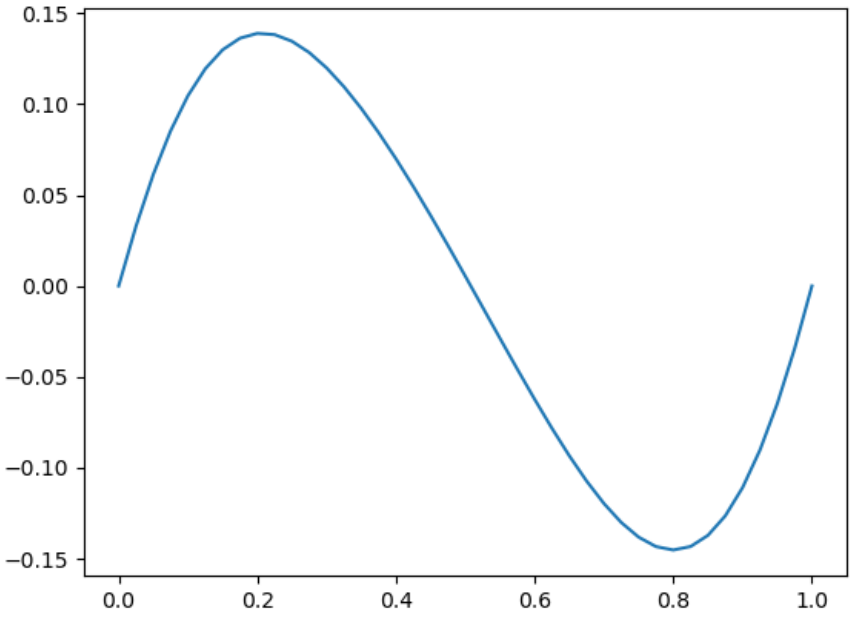

Figure 7: v velocity at y = 0.5