Friedrich-Alexander University of Erlangen

LSTM Institute of Fluid Mechanics

Dr.-Ing. M. Münsch, S. Saha

Report for Practical Sheet 3

Students: Franciskus Xaverius Erick, 22811754, franciskus.erick@fau.de

Serouj Djabraian, 22849126, serouj.d.djabraian@fau.de

Submitted: Erlangen, February 7th 2021

Table of Contents

Table of Figures

1.1 Discretized Navier-Stokes Equations

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

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{2}$$

$$\frac{\partial v}{\partial t} + \frac{\partial v^2}{\partial y} + \frac{\partial uv}{\partial x} = -\frac{\partial p}{\partial y} + \frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \tag{3}$$

Equation (2) is discretized by using an explicit scheme in time and 2nd order CDS schemes in space. The terms are discretized as follows.

Table 1 - Discretization of x-momentum terms

$\frac{\partial u^2}{\partial x}$	$\frac{1}{\Delta x}\left(\left(\frac{u_{i,j}+u_{i+1,j}}{2}\right)^2 - \left(\frac{u_{i,j}+u_{i-1,j}}{2}\right)^2\right)$
$\frac{\partial uv}{\partial y}$	$\frac{1}{\Delta y}\left(\frac{v_{i,j}+v_{i+1,j}}{2}\times\frac{u_{i,j}+u_{i,j+1}}{2} - \frac{v_{i,j-1}+v_{i+1,j-1}}{2}\times\frac{u_{i,j}+u_{i,j-1}}{2}\right)$
$\frac{\partial^2 u}{\partial x^2}$	$\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{\Delta x^2}$
$\frac{\partial^2 u}{\partial y^2}$	$\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{\Delta y^2}$
$\frac{\partial p}{\partial x}$	$\frac{p_{i+1,j}-p_{i,j}}{\Delta x}$

Similarly, Equation (3) is also discretized using an explicit scheme in time and 2nd order CDS schemes in spaces. The terms are discretized as follows.

Table 2 - Discretization of y-momentum terms

$\frac{\partial v^2}{\partial y}$	$\frac{1}{\Delta y}\left(\left(\frac{v_{i,j}+v_{i,j+1}}{2}\right)^2-\left(\frac{v_{i,j}+v_{i,j-1}}{2}\right)^2\right)$
$\frac{\partial uv}{\partial x}$	$\frac{1}{\Delta x}\left(\frac{v_{i,j}+v_{i+1,j}}{2}\times\frac{u_{i,j}+u_{i,j+1}}{2}-\frac{v_{i,j-1}+v_{i,j}}{2}\times\frac{u_{i-1,j}+u_{i-1,j+1}}{2}\right)$
$\frac{\partial^2 v}{\partial x^2}$	$\frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{\Delta x^2}$
$\frac{\partial^2 v}{\partial y^2}$	$\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{\Delta y^2}$
$\frac{\partial p}{\partial y}$	$\frac{p_{i,j+1}-p_{i,j}}{\Delta y}$

The Navier-Stokes equations are solved by using the Predictor-Corrector Approach (PCA) which is also known as the Fractional-Step Approach. Equations (2) and (3) are represented explicitly in time while ignoring the pressure terms in order to calculate intermediate velocities u^* and v^*, as follows:

$$\frac{u^*-u}{\Delta t}=\frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)-\frac{\partial u^2}{\partial x}-\frac{\partial uv}{\partial y} \tag{4}$$

$$\frac{v^*-v}{\Delta t}=\frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}\right)-\frac{\partial v^2}{\partial y}-\frac{\partial uv}{\partial x} \tag{5}$$

As this is an explicit scheme, it is conditionally stable according to ____ with the following condition:

$$\Delta t \leq \left(\frac{1}{2}\right)^{d}.Re.\Delta x^2 \tag{6}$$

where d is the domain dimension, Re is the Reynold's number and Δx is the smallest grid size.

Equations (1) is discretized in order to get Poisson's equation as follows:

$$\frac{P_{i+1,j} - 2P_{i,j} + P_{i-1,j}}{\Delta x^2} + \frac{P_{i,j+1} - 2P_{i,j} + P_{i,j-1}}{\Delta y^2} = \frac{1}{\Delta t}\left(\frac{u^*{}_{i,j} - u^*{}_{i-1,j}}{\Delta x} + \frac{v^*{}_{i,j} - v^*{}_{i,j-1}}{\Delta y}\right) \tag{7}$$

Equation (7) is solved explicitly while performing several iterations until a desired error is reached for the difference between pressure from one iteration to the next. These iterations improve our results dramatically since it helps stabilizing our coding and get accurate results for pressure.

After calculating the pressure, it is required to calculate the real velocities and a time step of (n+1). The equations to calculate these are the followings:

$$u_{i,j} = u^*{}_{i,j} - \frac{\Delta t}{\Delta x}\left(P_{i,j} - P_{i-1,j}\right) \tag{8}$$

$$v_{i,j} = v^*{}_{i,j} - \frac{\Delta t}{\Delta y}\left(P_{i,j} - P_{i,j-1}\right) \tag{9}$$

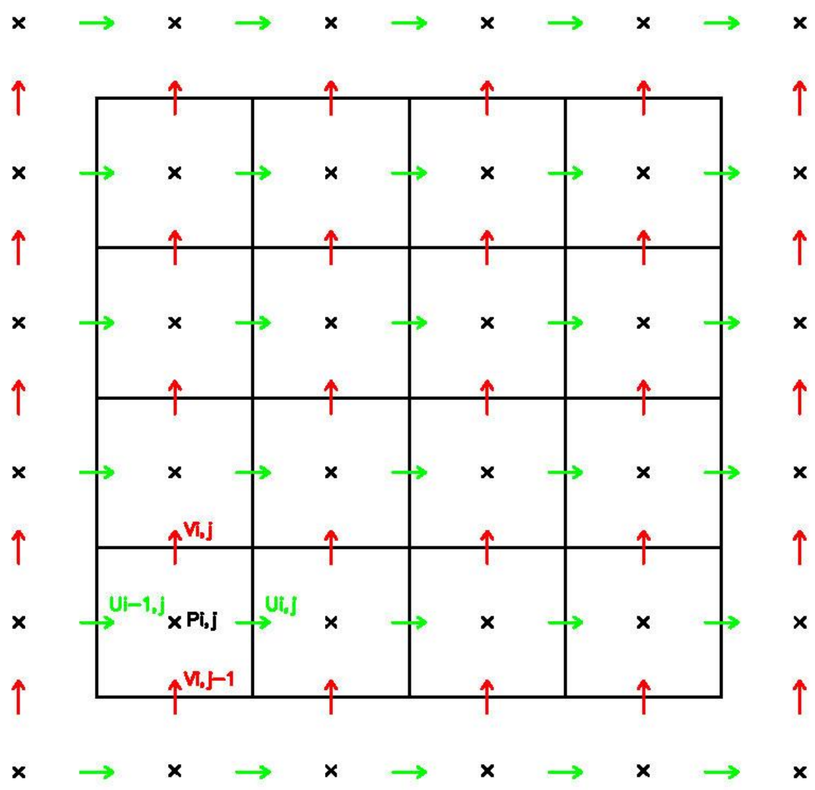

Figure 1 - Staggered grid configuration

For the boundary conditions, u(x,y,t)→1 at the top of our domain which represents our the lid, where the other boundary conditions of the velocities are set to zero. Moreover, interpolation is required between "ghost-cells" and our velocities. Thus, we set our BC's as follows:

$$\mathrm{u}(:,1) = -\mathrm{u}(:,2) \ \ (\text{Bottom Side})$$

$$u(:,Ny+2) = 2 - u(:,Ny+1) \ \ (Top\ Lid)$$

$$\mathrm{v}(1,:) = -\mathrm{v}(2,:) \ \ (\text{Right Side})$$

$$v(Nx+2,:) = -v(Nx+1,:) \ \ (Left\ Side)$$

For the BC's of the pressure, it is required to have a pressure gradient of zero between the pressure at the wall and the "ghost" pressure. Thus we set them equal to each other.

$$\mathrm{pt}(1,:) = \mathrm{pt}(2,:)$$

$$pt(:,1) = pt(:,2)$$

$$pt(Nx+2,:) = pt(Nx+1,:)$$

$$pt(:,Ny+2) = pt(:,Ny+1)$$

1.2 Results and Discussion

a) The number of nodes in the x and y directions are chosen as 11 points, thus $\Delta x = \Delta y = 0.1$. At a Reynold's number of 0.1 and a domain dimension of 1, our stability criterion as mentioned previously yields that the Navier-Stokes equations are stable for a $\Delta t \leq 0.0005$. In our simulation we have chosen a time step of 0.0001. This is due to the fact that our pressure term is treated explicitly, in addition to the convective and diffusive velocity terms which are also explicitly, thus a very small time step size is required for our solution to converge and obtain accurate results.

b) For a Reynold's number of $Re = 0.1$, the figures (2) and (3) are generated for ranging $t = 0{:}\,0.02{:}\,0.1$. It can be observed in figure (2) that there are 2 main pressure concentration points which are located at the top of domain where we have our moving lid, the first point is a very high pressure point and the second one is a very low pressure point. This is phenomenon that allows the flow to form a cavity from a high pressure point to a low pressure point. Figure (3) demonstrates the formation of this cavity, where the flow is subject to a rotation around a certain point which is our cavity.

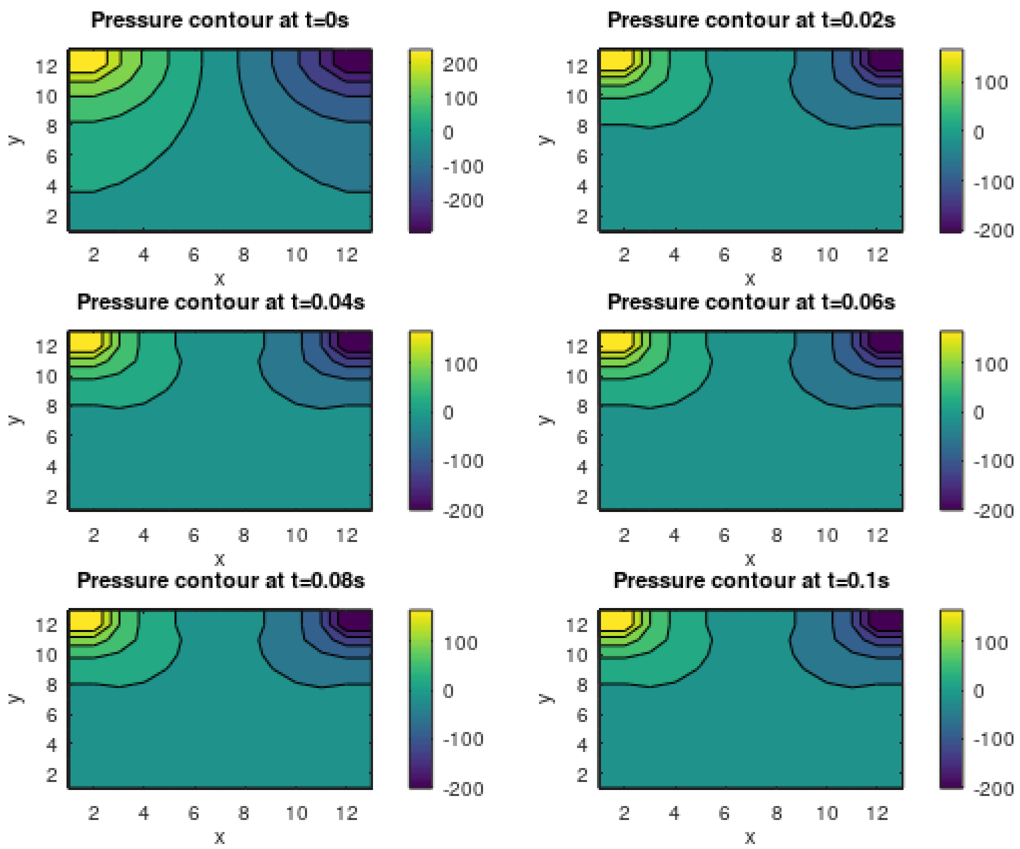

Figure 2 – Representation of Pressure Contours at different times

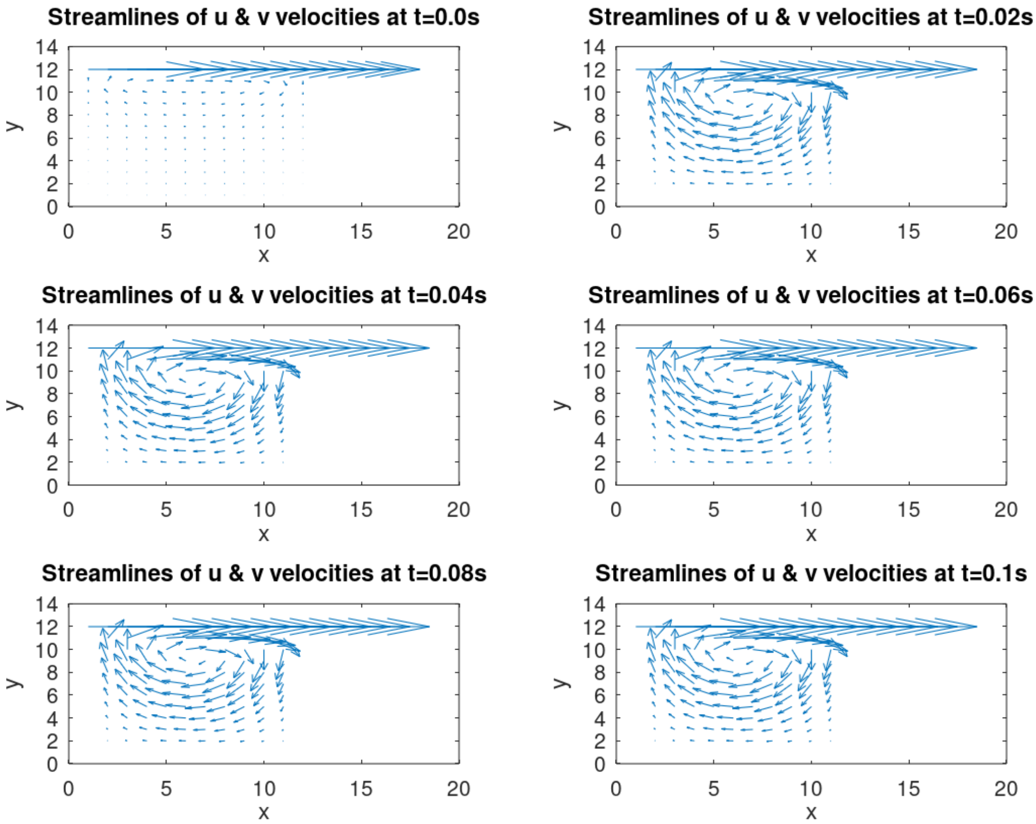

Figure 3 – Representation of Streamlines at different times

c) Figures (4) & (5) represents the change of u & v velocities with respect to their position in y at x=0.5. It can be noticed that the value of the u velocity is zero at the bottom of our domain where y=0, then it subjected to a negative velocity until reaching a minimum of -0.2 at the middle of our domain, where we have a flow in the opposite direction of our top lid. Then the u-velocity starts increasing again until reaching a value of 0, which the location of our cavity.
For the v-velocity, it can be noted that it is subjected to an increase from its 0 value which is the BC's at the bottom of our domain up to a maximum value of approximately 0.05 and then it decreases again until reaching zero where we have our second BC for v.
There is only a slight difference between 2 different simulations Re=0.1 and Re=1, since the change in Reynold's number is not that high. But in figures (6) & (7), additionally we have provided with a simulation at Re=500 and number of nodes=41 in both directions. This simulation is better at pointing out the different velocity contours in the flow along with the streamlines plot.

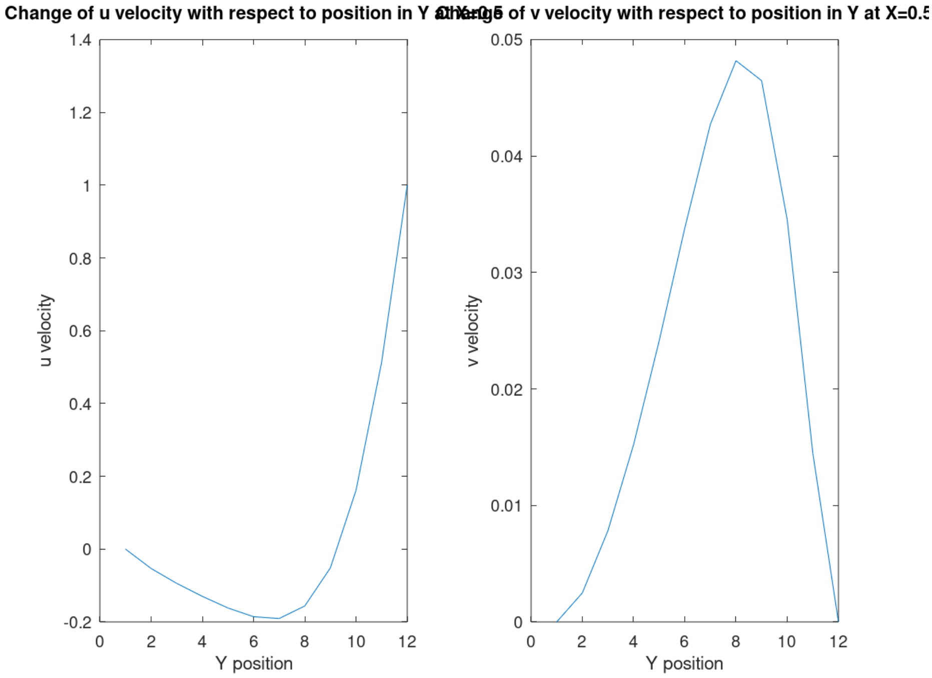

Figure 4 - Change of u & velocities at Re=0.1

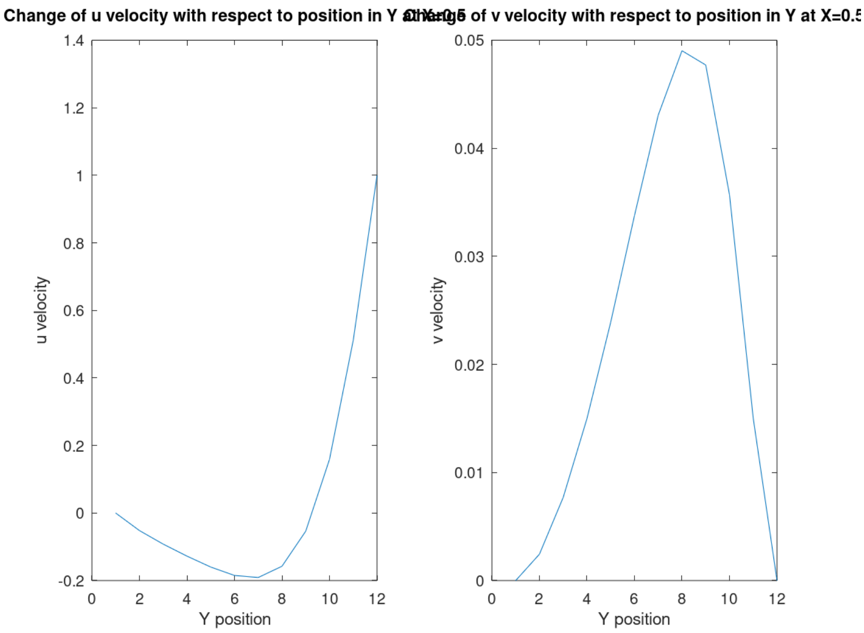

Figure 5 - Change of u & v velocities at Re=1

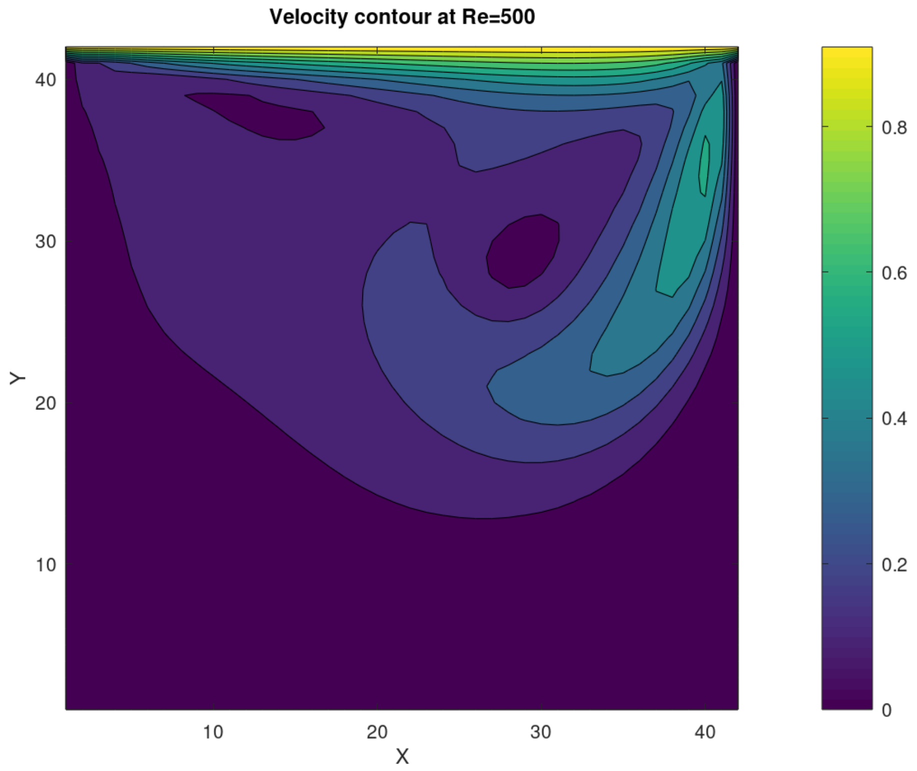

Figure 6 - Velocity contour at Re=500 at T=5s with a higher grid resolution

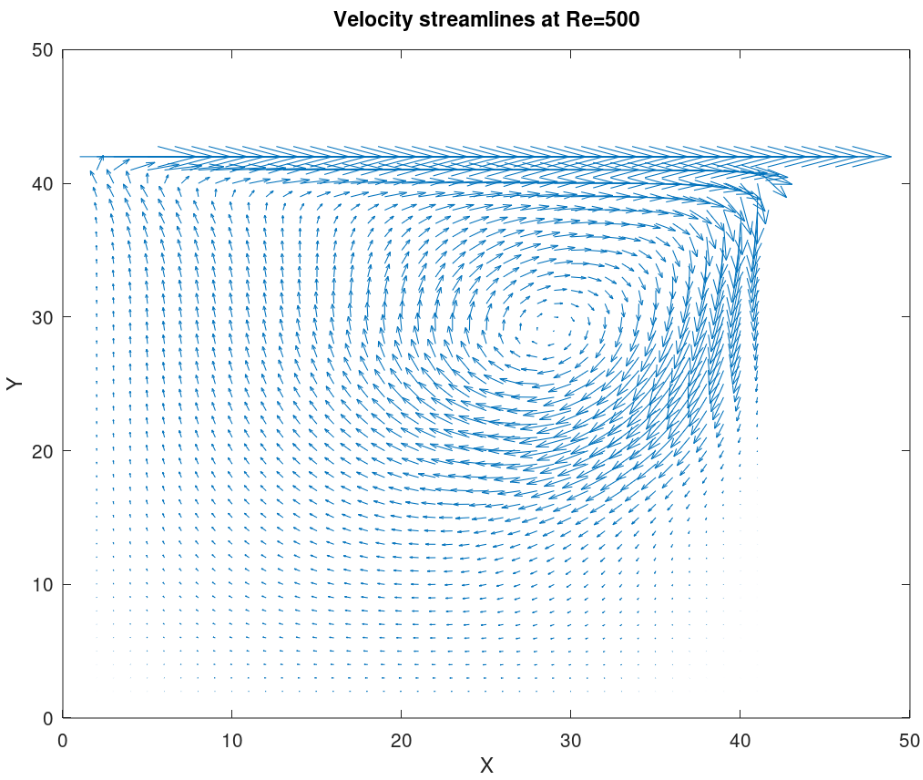

Figure 7 - Streamlines at Re=500 at T=5s with a higher grid resolution

Literature References

[1] M. Münsch, *Numerical Methods in Thermo-Fluid Dynamics I ; Numerical Solution of the Navier-Stokes Equations*, Numerical Methods in Thermo-Fluid Dynamics I, **2021**, Friedrich-Alexander Universität Erlangen, Microsoft Power Point Presentation

[2] Johnston, H., & Liu, J. G. (2004). Accurate, stable and efficient Navier–Stokes solvers based on explicit treatment of the pressure term. *Department of Mathematics, University of Michigan, Ann Arbor, MI 48109, USA.*

Appendix

```
%%% Navier-Stokes Solver using the Predictor-Corrector Approach (PCA) %%%
clear all
close all
clc

Nx=11;                  %nb of nodes in x-direction
Ny=11;                  %nb od nodes in y-direction
L=1;                    %Dimension of domain
T=0.1;                  %Total time
dt=0.0001;              %Time step size
nt=T/dt;                %Total number of timesteps
Re=0.1;                 %Reynold's number
iterations=50000;       %Maximum number of iterations for Pressure correction
dx=L/(Nx-1);
dy=L/(Ny-1);
[X,Y]=meshgrid(0:dx:L,0:dy:L);
u=zeros(Nx+1,Ny+2);
v=zeros(Nx+2,Ny+1);
p=zeros(Nx+2,Ny+2);
u_star=zeros(Nx+1,Ny+2);
v_star=zeros(Nx+2,Ny+1);
u_final=zeros(Nx+1,Ny+1);
v_final=zeros(Nx+1,Ny+1);
figure 1;
for k=1:nt              %Iterating over time
  k
    if (rem(k,5)==0)
      %contourf(rot90(rot90(rot90(p))));     %Pressure Contour
      %contourf(flipud(rot90(uv_final)));    %Velocity Contour
      %colorbar;
      quiver(flipud(rot90(u_final)),flipud(rot90(v_final)),5);
%Streamlines
      drawnow;
    endif
##      if (k==2)
##        h1 = subplot( 321);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
##        title (h1,'Pressure contour at t=0s');
##        xlabel(h1,'x');
##        ylabel(h1,'y');
##      endif
##      if (k==200)
##        h2 = subplot( 322);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
```

```
##        title (h2,'Pressure contour at t=0.02s');
##        xlabel(h2,'x');
##        ylabel(h2,'y');
##      endif
##      if (k==400)
##        h3 = subplot( 323);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
##        title (h3,'Pressure contour at t=0.04s');
##        xlabel(h3,'x');
##        ylabel(h3,'y');
##      endif
##      if (k==600)
##        h4 = subplot( 324);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
##        title (h4,'Pressure contour at t=0.06s');
##        xlabel(h4,'x');
##        ylabel(h4,'y');
##      endif
##      if (k==800)
##        h5 = subplot( 325);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
##        title (h5,'Pressure contour at t=0.08s');
##        xlabel(h5,'x');
##        ylabel(h5,'y');
##      endif
##      if (k==1000)
##        h6 = subplot( 326);
##        contourf(rot90(rot90(rot90(p))));
##        colorbar;
##        title (h6,'Pressure contour at t=0.1s');
##        xlabel(h6,'x');
##        ylabel(h6,'y');
##      endif

    %%% BC's for momentum equations %%%
    u(:,1)=-u(:,2);            %Bottom
    u(:,Ny+2)=2-u(:,Ny+1);      %Lid
    v(1,:)=-v(2,:);             %Right
    v(Nx+2,:)=-v(Nx+1,:);       %Left
    %%% X-momentum equation %%%
    for i=2:Nx
      for j=2:Ny+1
        diffusion = (1/Re)*((u(i+1,j)-2*u(i,j)+u(i-1,j))/dx^2+(u(i,j+1)-
2*u(i,j)+u(i,j-1))/dy^2);
        con_x = (-0.25)*((u(i+1,j)+u(i,j)).^2-(u(i,j)+u(i-1,j)).^2)/dx ;
        con_y = (-0.25)*((u(i,j+1)+u(i,j)).*(v(i+1,j)+v(i,j))-
(u(i,j)+u(i,j-1)).*(v(i+1,j-1)+v(i,j-1)))/dy;
        u_star(i,j)=u(i,j)+dt*(diffusion + con_x + con_y);
      endfor
    endfor

    %%% Y-momentum equation %%%
    for i=2:Nx+1
      for j=2:Ny
```

```
        diffusion = (1/Re)*((v(i+1,j)-2*v(i,j)+v(i-1,j))/dx^2+(v(i,j+1)+-
2*v(i,j)+v(i,j-1))/dy^2);
        con_x = (-0.25)*((u(i,j+1)+u(i,j)).*(v(i+1,j)+v(i,j))-(u(i-
1,j+1)+u(i-1,j)).*(v(i,j)+v(i-1,j)))/dx;
        con_y = (-0.25)*((v(i,j+1)+v(i,j)).^2-(v(i,j)+v(i,j-1)).^2)/dy;
        v_star(i,j)=v(i,j)+dt*(diffusion + con_x + con_y);
      endfor
    endfor
    pt=p;         %Temporary Pressure Matrix to minimize the error
    for it=1:iterations                      %Iterate until a desired error
is reached
      %it
      %%% BC's for pressure %%%
      pt(1,:)=pt(2,:);
      pt(Nx+2,:)=pt(Nx+1,:);
      pt(:,1)=pt(:,2);
      pt(:,Ny+2)=pt(:,Ny+1);

      %%% Poisson's equation %%%
      i=2:Nx+1; j=2:Ny+1;
          pt(i,j) = 0.25*(pt(i+1,j) + pt(i-1,j) + pt(i,j+1) + pt(i,j-1) -
(dx/dt)*(u_star(i,j)-u_star(i-1,j)+v_star(i,j)-v_star(i,j-1)));
      err=max(max(pt-p));
      p=pt;
      if err<10^-5
          break;
      endif
    endfor
    %%% Velocities u and v at timestep (n+1) %%%
    u(2:Nx,2:Ny+1) = u_star(2:Nx,2:Ny+1)-(dt/dx)*(p(3:Nx+1,2:Ny+1)-
p(2:Nx,2:Ny+1));
    v(2:Nx+1,2:Ny) = v_star(2:Nx+1,2:Ny)-(dt/dy)*(p(2:Nx+1,3:Ny+1)-
p(2:Nx+1,2:Ny));

    %%% Velocity correction to our actual grid points %%%
    u_final(1:Nx+1,1:Ny+1)=0.5*(u(1:Nx+1,2:Ny+2)+u(1:Nx+1,1:Ny+1));
    v_final(1:Nx+1,1:Ny+1)=0.5*(v(2:Nx+2,1:Ny+1)+v(1:Nx+1,1:Ny+1));
    uv_final=sqrt(u_final.^2+v_final.^2);
endfor
```