UNIVERSITY OF ALBERTA

SEYED EHSAN SEYED BOLOURI

1567349

E-mail: seyedbol@ualberta.ca

Project Description:

The main goal of the project is to obtain the pressure and velocity distribution of a liquid flowing inside a porous media.

`

CONTENTS

`

PART A

The first step to solve any FEM problem is generating a reasonable mesh. In this case I decided to use GMSH. Figure 2 shows that is a quadrilateral mesh that has 8 elements in y direction and 9 elements in x direction. So, it satisfies the condition that was mentioned in the question.

A short description of the algorithm used to generate and plot the mesh:

1. Defining nodal location of boundaries
2. Connecting nodes on the boundaries and getting figure 1

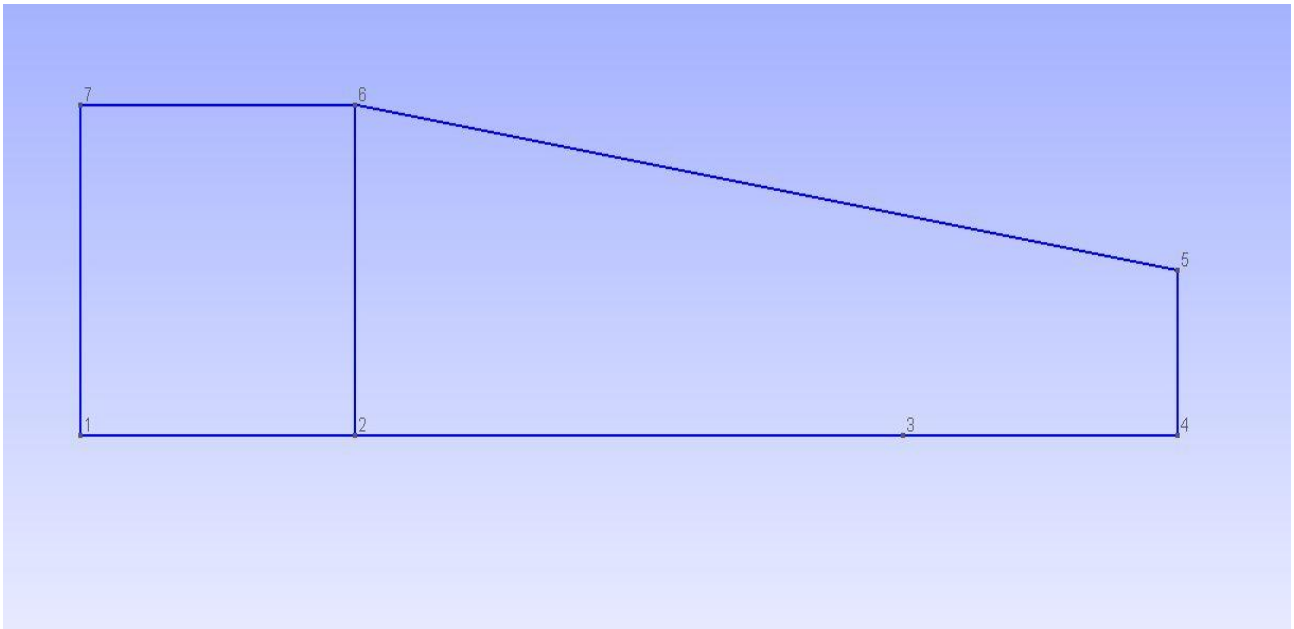


Figure 1

3. Defining two planes surface between (1,2,6,7) and (2,4,5,6)
4. Dividing the body with the help of transfinite surface tool
5. Dividing each edge with the help of transfinite line tool
6. Recombining the body
7. Using 2D tool to generate the mesh

`

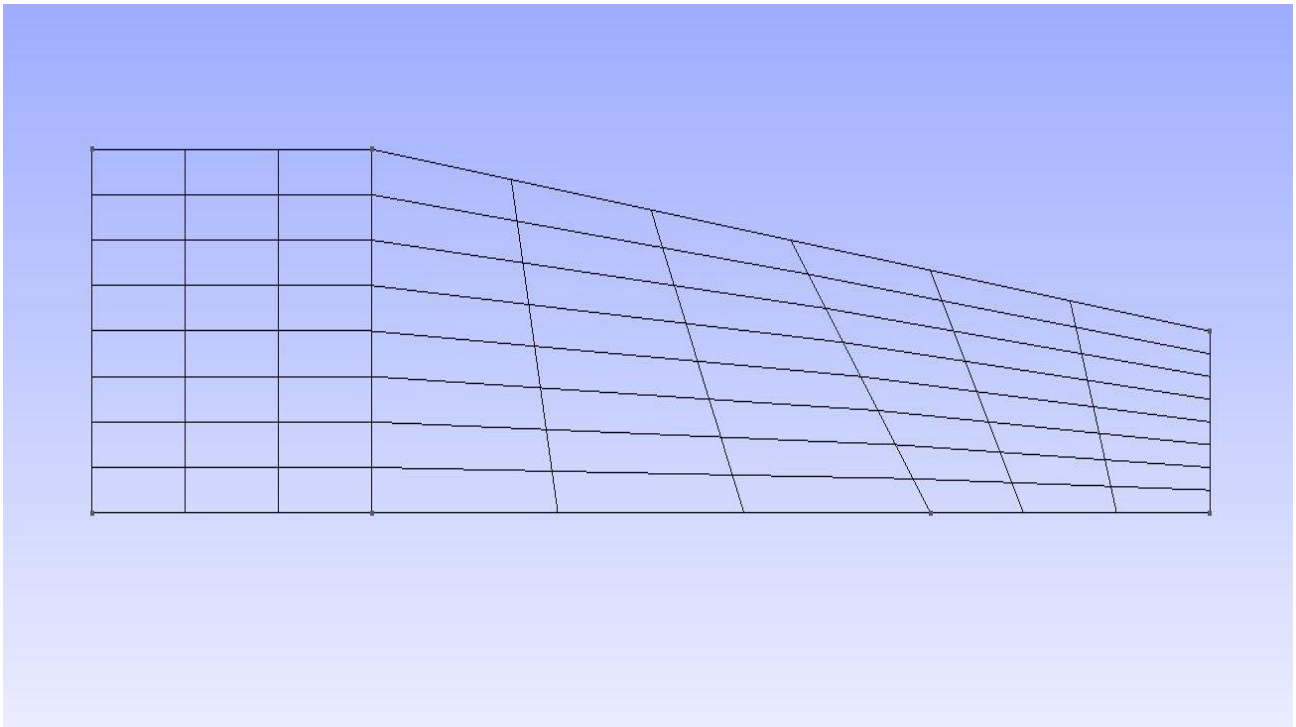


Figure 2

In order to work with (.msh) file in MATLAB, I divided the file that was generated by GMSH into 2 parts. The first part is Nodes.txt that contains location of each node and the second part is Elements.txt that contains connectivity matrix.

```
6 0.25 0.25 0
7 0 0.25 0
8 0.08333333333312526 0.25 0
9 0.1666666666664469 0.25 0
10 0.8750000000009216 0.1458333
11 0.75000000000076 0.166666666
```

Figure 3

For instance, in Figure 3 and line 8 we can see the node number in the first column, location of the node in x direction in the second column, location of the node in y direction in the third column and location of the node in z direction in the fourth column.

`

66	3	2	0	10	9	49	41	6
67	3	2	0	10	49	50	40	41
68	3	2	0	10	50	51	39	40
69	3	2	0	10	51	52	38	39
70	3	2	0	10	52	53	37	38
71	3	2	0	10	53	54	36	37
72	3	2	0	10	54	55	35	36

Figure 4

In Figure 4 and line 68 we can see the element number in the first column, the first node that creates the element in the 6th column, the second node that creates the element in the 7th column, the third node that creates the element in the 8th column and the fourth node that creates the element in the 9th column.

By using the code visualizationofmesh.m in (part a) folder, we can generate Figure 5 in MATLAB.

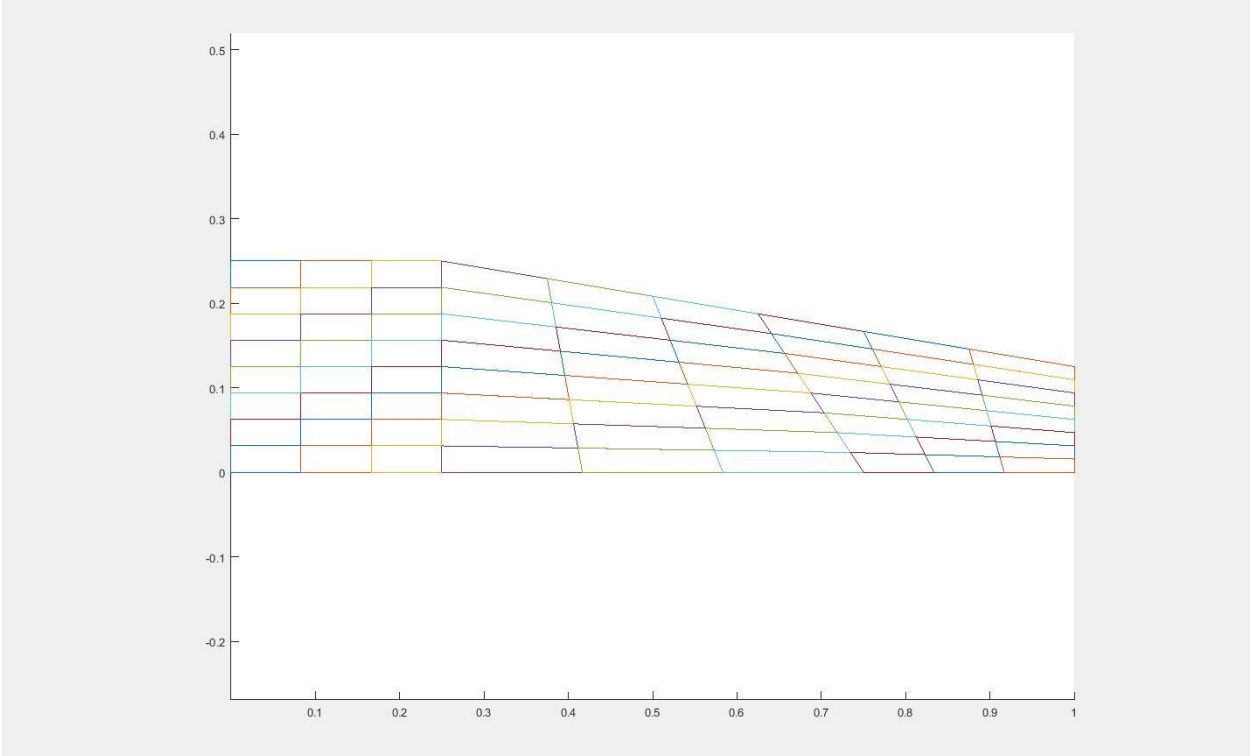


Figure 5

`

Part B:

Governing equations are:

$$\nabla . u = 0$$

$$u = \left(-\frac{k}{\mu}\right)(\nabla p - \rho g)$$

By plugging the first equation in the second one we can omit (u) and transform two equations into 1 equation.

$$\nabla . [\left(-\frac{k}{\mu}\right)(\nabla p - \rho g)\ ] = 0$$

To get the weak form we have to multiply the above equation by the weighted function (v) and take the integral.

$$\int \nabla . \left[\left(-\frac{k}{\mu}\right)(\nabla p - \rho g)\right] v d\Omega = 0$$

Now we have to do integration by parts for the equation above:

$$\int n . (\frac{k}{\mu} \nabla p\ v)\ d\tau \text{ - } \int (\frac{k}{\mu}) \nabla p\ \nabla v\ d\Omega \text{ - } \int\ n . (\frac{k}{\mu} \rho\ g\ v)\ d\tau + \int\ (\frac{k}{\mu})\ \rho\ g\ \nabla v\ d\Omega\ =\ 0$$

The term for boundary condition is:

$$\int n . (\frac{k}{\mu})(\ \nabla p - \rho g) v d\tau$$

According to $u.n = 0$ on the boundaries, we can conclude that we have no Neuman boundary conditions on the impermeable walls. Consequently, the only boundary conditions that are important for us are at inlet and outlet.

Based on the equations above left hand side equation is:

$$\int (\frac{k}{\mu}) \nabla p\ \nabla v\ d\Omega$$

And right hand side equation is:

$$\int (\frac{k}{\mu})\ \rho\ g\ \nabla v\ d\Omega$$

`

Shape functions for each element in Galerkian methods are:

$$P = \sum \psi_j P_j$$

$$v = \sum \psi_i v_i$$

By plugging them in the right hand side and left hand side equation we have:

Left hand side for each element= $\int (k/\mu)\nabla\psi_i \nabla\psi_j \, d\Omega$

Right hand for each element = $\int (k/\mu)\, \rho\, g\, \nabla\psi_i \, d\Omega$

Based on what we have on the question, we can use isoperimetric linear functions. Consequently, each element has 4 nodes and we need shape function at each node. (shapefunction.m)

$$\psi_1 = \left(\frac{1}{4}\right)(1-x)(1-y)$$

$$\psi_2 = \left(\frac{1}{4}\right)(1+x)(1-y)$$

$$\psi_3 = \left(\frac{1}{4}\right)(1+x)(1+y)$$

$$\psi_4 = \left(\frac{1}{4}\right)(1-x)(1+y)$$

By plugging them in right hand side and left hand side we have:

$$K_{ij}^e = \int \left(\frac{k(X(x,y),Y(x,y))}{\mu}\right)\left[\left(J^{-1}(1,1)\frac{\partial\psi_i}{\partial x} + J^{-1}(1,2)\frac{\partial\psi_i}{\partial y}\right) * \left(J^{-1}(1,1)\frac{\partial\psi_j}{\partial x} + J^{-1}(1,2)\frac{\partial\psi_j}{\partial y}\right) + \left(J^{-1}(2,1)\frac{\partial\psi_i}{\partial x} + J^{-1}(2,2)\frac{\partial\psi_i}{\partial y}\right) * \left(J^{-1}(2,1)\frac{\partial\psi_j}{\partial x} + J^{-1}(2,2)\frac{\partial\psi_j}{\partial y}\right)\right] * \det(J)dxdy$$

$$f_i^e = \int \left(\frac{k(X(x,y),Y(x,y))}{\mu}\right)\left[\left(J^{-1}(2,1)\frac{\partial\psi_i}{\partial x} + J^{-1}(2,2)\frac{\partial\psi_i}{\partial y}\right)\right] \rho\, g\, \det(J)dxdy$$

$X(x,y) = \sum_{i=1}^{4} X_i \psi_i$ (mapx.m)

$Y(x,y) = \sum_{i=1}^{4} Y_i \psi_i$ (mapy.m)

`

Now we need to use quadrature points to calculate the integral:

$$J = \begin{bmatrix} \frac{\partial X}{\partial x} & \frac{\partial Y}{\partial x} \\ \frac{\partial X}{\partial y} & \frac{\partial Y}{\partial y} \end{bmatrix}$$ (jacobian.m)

$$J^{-1} = \frac{1}{\det(J)} \begin{bmatrix} \frac{\partial Y}{\partial y} & -\frac{\partial Y}{\partial x} \\ -\frac{\partial X}{\partial y} & \frac{\partial X}{\partial x} \end{bmatrix}$$

$$F_{1(x,y)} = \left(\frac{k(X(x,y),Y(x,y))}{\mu}\right)\left[\left(J^{-1}(1,1)\frac{\partial \psi_i}{\partial x} + J^{-1}(1,2)\frac{\partial \psi_i}{\partial y}\right) * \left(J^{-1}(1,1)\frac{\partial \psi_j}{\partial x} + J^{-1}(1,2)\frac{\partial \psi_j}{\partial y}\right) + \left(J^{-1}(2,1)\frac{\partial \psi_i}{\partial x} + J^{-1}(2,2)\frac{\partial \psi_i}{\partial y}\right) * \left(J^{-1}(2,1)\frac{\partial \psi_j}{\partial x} + J^{-1}(2,2)\frac{\partial \psi_j}{\partial y}\right)\right] * \det(J)$$ (line 61, main code)

$$F_{2(x,y)} = \left(\frac{k(X(x,y),Y(x,y))}{\mu}\right)\left[\left(J^{-1}(2,1)\frac{\partial \psi_i}{\partial x} + J^{-1}(2,2)\frac{\partial \psi_i}{\partial y}\right)\right] \rho\, g\, \det(J)$$ (line 50, main code)

For 4 quadrature points:

$$W(1) = .347854, W(2) = .652145, W(3) = .652145, W(4) = .347854$$

$$P(1) = -.861136, P(2) = -.339981, P(3) = .339981, P(4) = .861136$$

$$K_{ij}^{e} = \sum_{i=1}^{4}\sum_{j=1}^{4} F1_{(P(i),P(j))} W(i) W(j)$$

$$f_i^{e} = \sum_{i=1}^{4} F1_{(P(i))} W(i)$$

(intquad.m)

After using the summation of quadrature points we have a 4by4 left hand side matrix and 4by1 right hand side matrix.

`

Boundary conditions:

Part C:

Neuman boundary conditions:

We have no Neuman boundary conditions on impermeable walls because:

$$\int n.\left(\frac{k}{\mu}\right)(\nabla p - \rho g)v d\tau = u.n = 0$$

Drichlet boundary conditions: (line80-110, main code)

$$P(inlet) = 121.3\ Kpa\ \&\ P(outlet) = 101.3Kpa$$

Part E:

Neuman boundary conditions:

We have Neuman boundary condition at inlet and we can calculate N_i^e with the help of four quadrature points:

$$\int n.\left(\frac{k}{\mu}\right)(\nabla p - \rho g)v d\tau = \int (n.u)v d\tau = \int (10^{-3})\psi_i(x, y = 1)(\frac{l}{2})dx$$

$$N_i^e = \sum_{i=1}^{4} F1_{(P(i))}W(i)$$

Drichlet boundary conditions:

$$P(outlet) = 101.3Kpa$$

`

Part C:

For the first case, I used 8*9 cells to show the results for pressure, X velocity and Y velocity.

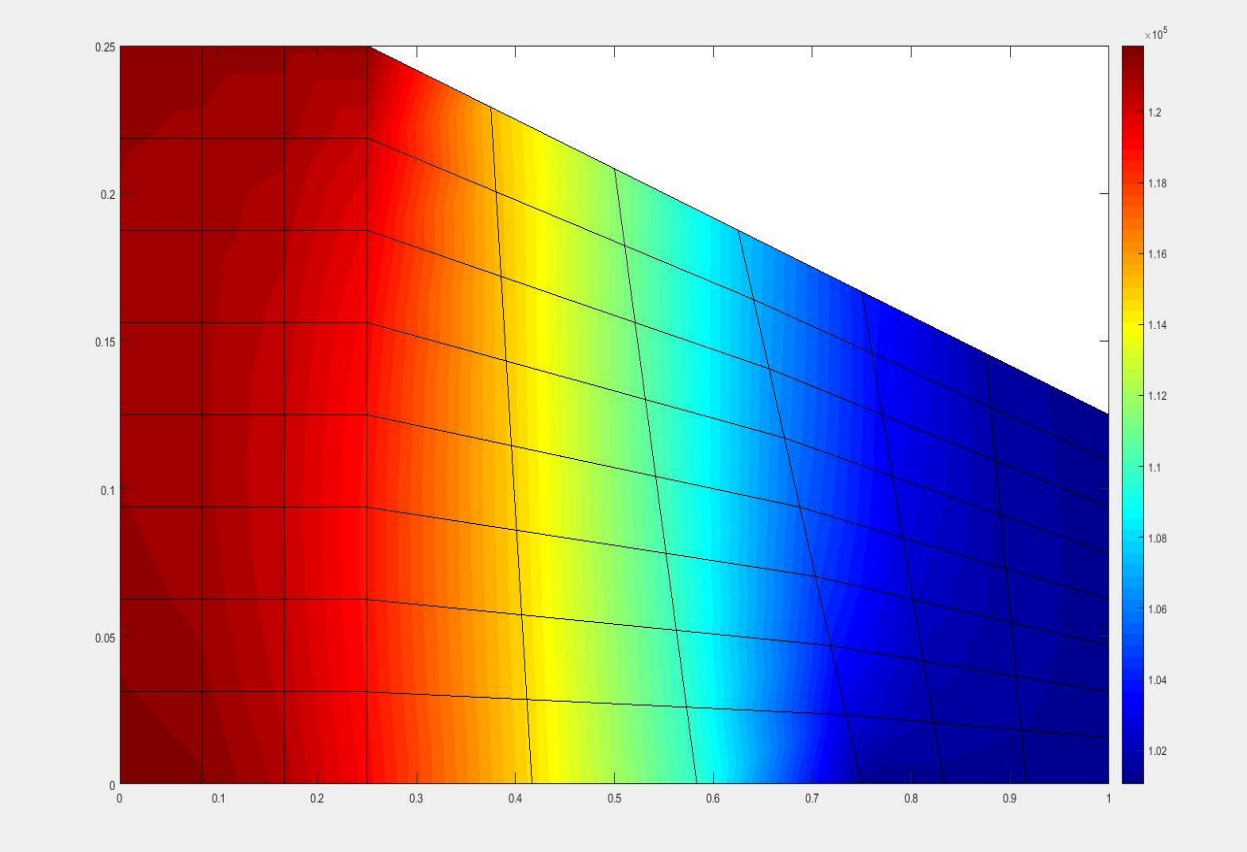


Figure 6(pressure distribution)

Gradient of pressure is the reason that water flows inside the porous media. According to the boundary conditions that were mentioned in the question all the circumferential boundary conditions are impermeable walls and would not let the flow exit from them. Consequently, it is rational that we see the maximum pressure at inlet and minimum pressure at outlet. It seems that Figure 6 satisfies this reasoning.

`

We have to calculate a secondary variable for this problem like velocity. Based on the nature of Fem, we don't need to have similar gradient of pressure for each node that is common among adjacent elements. So, it is likely to get one or two or four values for each node after calculating the gradient of pressure. In order to get a unique value, I took the average of values for each node. With the help of this idea we can plot the results for each node easily. P and Q are locations in real coordinate.

By plugging in the values and derivatives after mapping in the velocity equation we obtain:

$$u_y = \left(-\frac{k(P,Q)}{\mu}\right)(J^{-1}(2,1)\frac{\partial \psi_i}{\partial x} + J^{-1}(2,2)\frac{\partial \psi_i}{\partial y} - \rho g)$$

$$u_x = \left(-\frac{k(P,Q)}{\mu}\right)(J^{-1}(1,1)\frac{\partial \psi_i}{\partial x} + J^{-1}(1,2)\frac{\partial \psi_i}{\partial y})$$

`

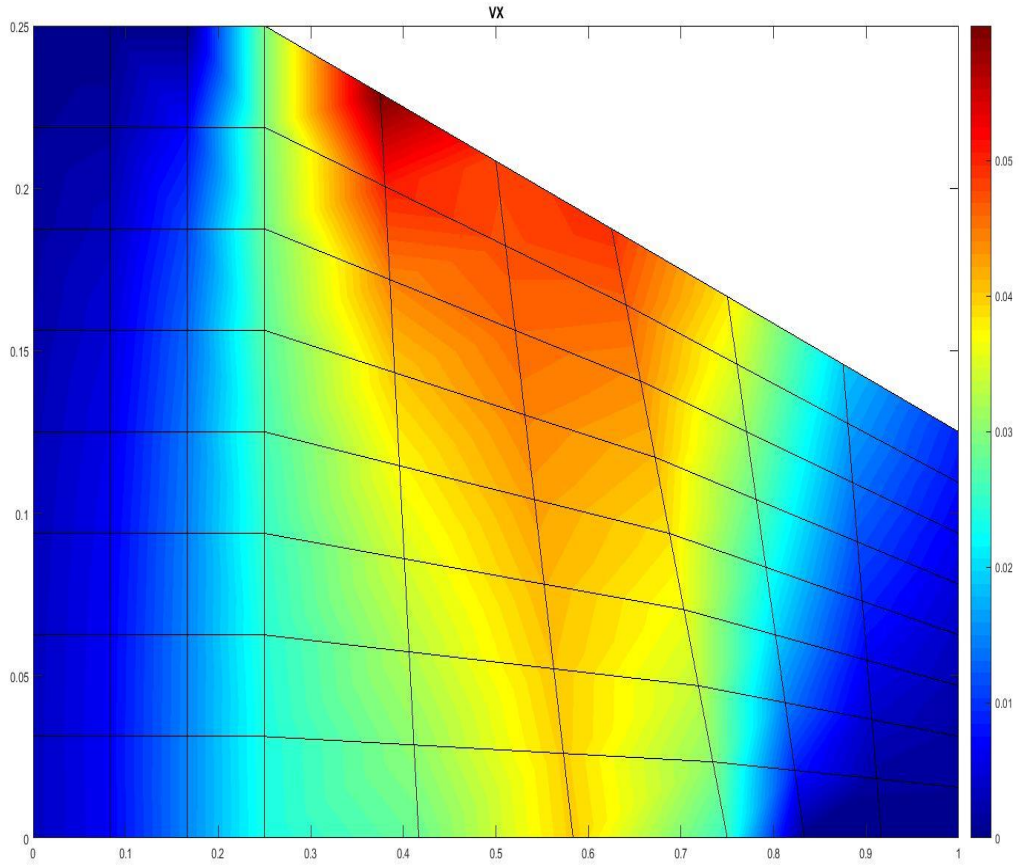


Figure 7(X velocity)

Figure 7 demonstrates the X velocity. It seems reasonable that we have the maximum velocity at steep boundary because we don't have X velocity at inlet and outlet and also, vertical impermeable walls $(u.n = 0)$.

`

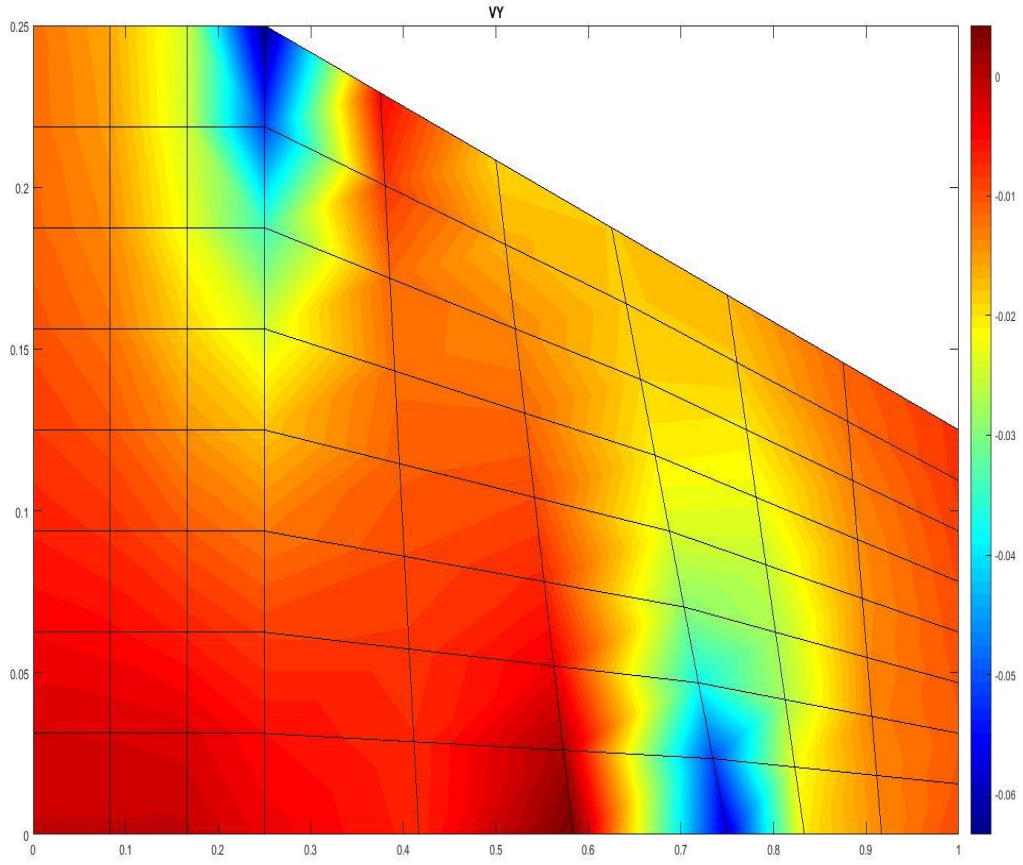


Figure 8(Y velocity)

In Figure 8, velocity is almost zero at horizontal boundary and it seems explainable according to $u.n = 0$. Also, we know that we have Y velocity on steep boundary. So, it would justify the maximum velocity at the intersections of top boundaries and bottom boundaries.

`

Part D :

(a)

For part C I used 8*9 elements to demonstrate the results for pressure and velocity. In part D, I did h-refinement to investigate the accuracy of my results.

In the first case h-2 refinement means using 16*18 elements.

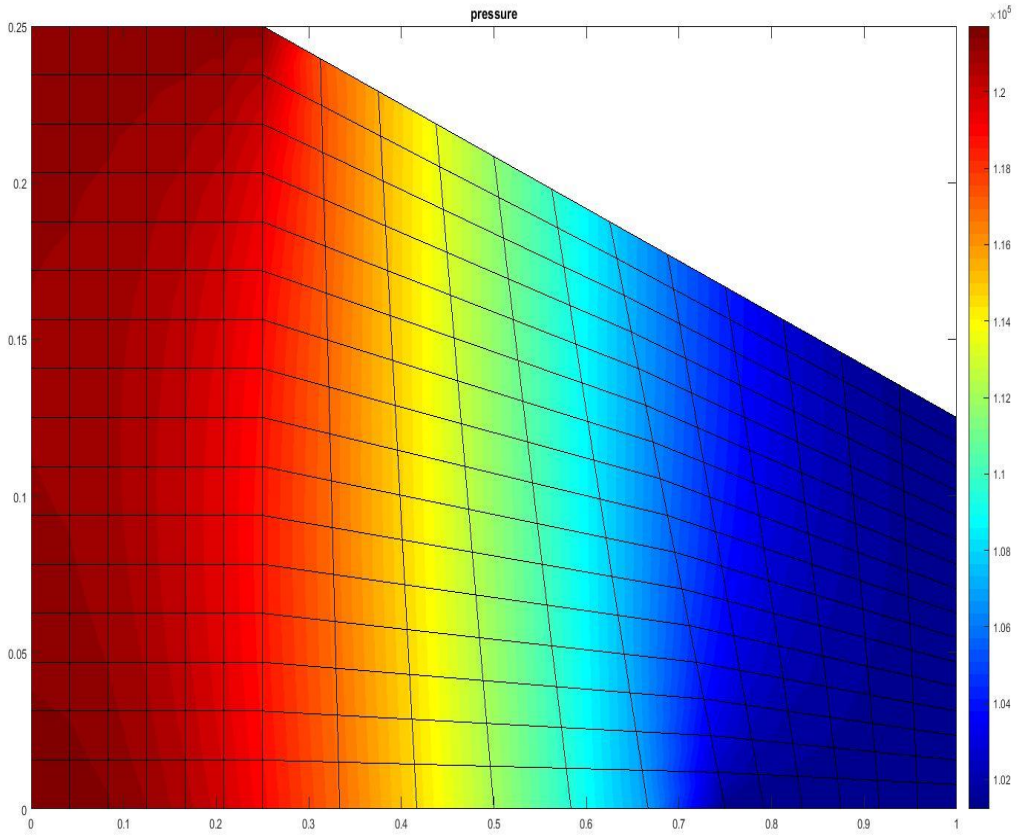


Figure 9(pressure distribution)

`

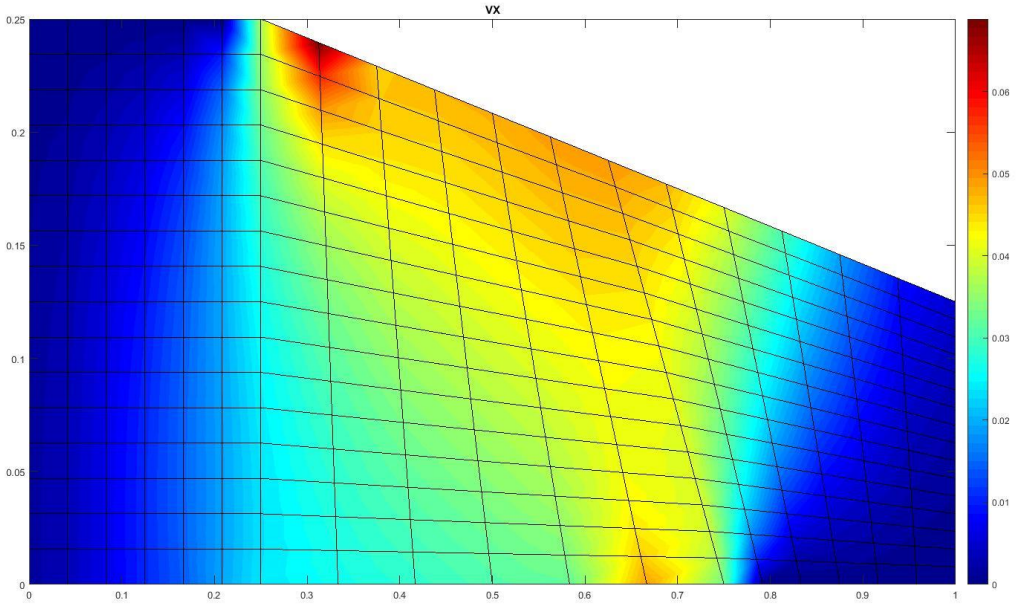


Figure 10(X velocity distribution)

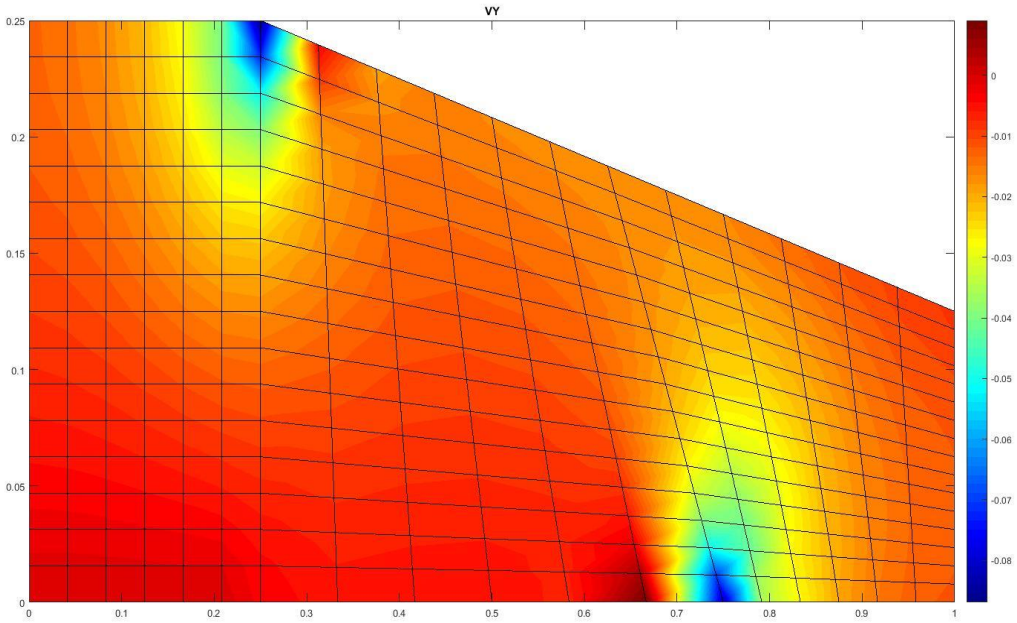


Figure 11(Y velocity distribution)

`

In the second case h-3 refinement means using 24*27 elements.

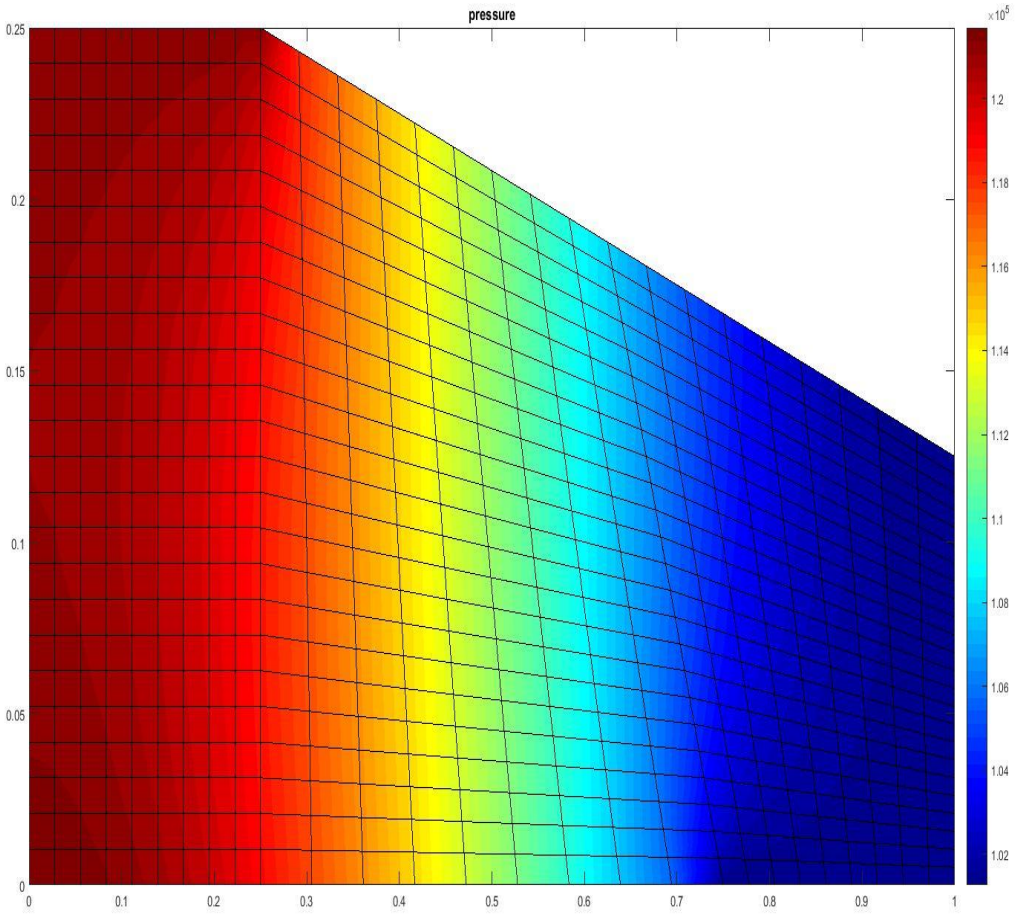


Figure 12(pressure distribution)

`

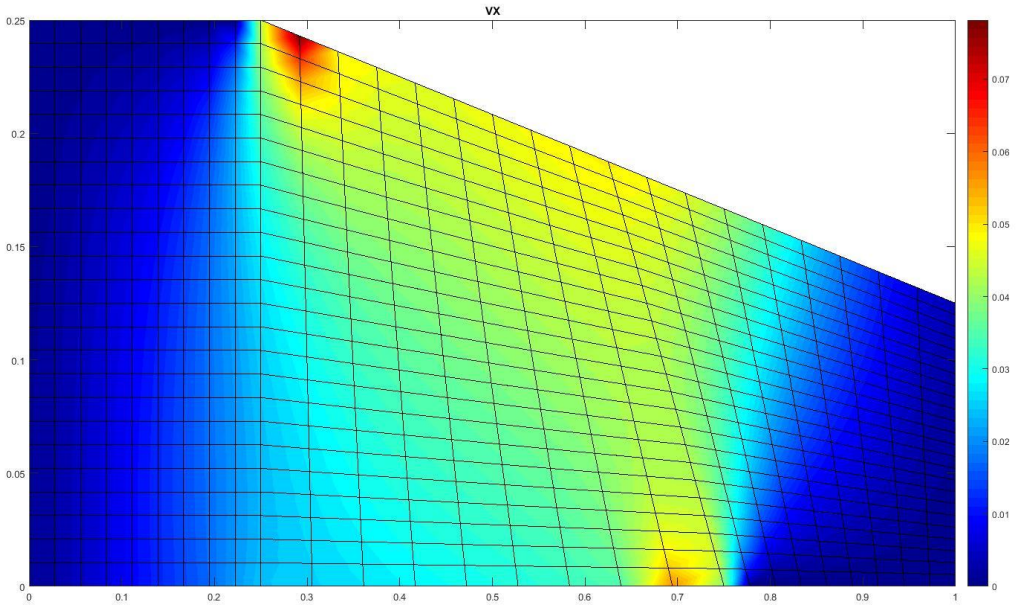


Figure 13(X velocity distribution)

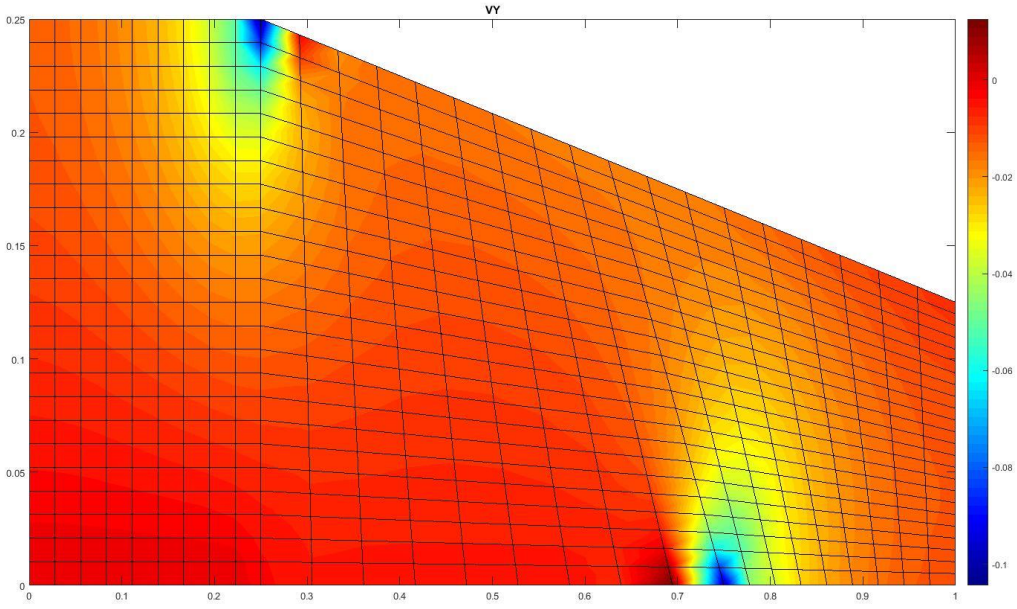


Figure 14(Y velocity distribution)

`

In the third case h-4 refinement means using 32*36 elements:

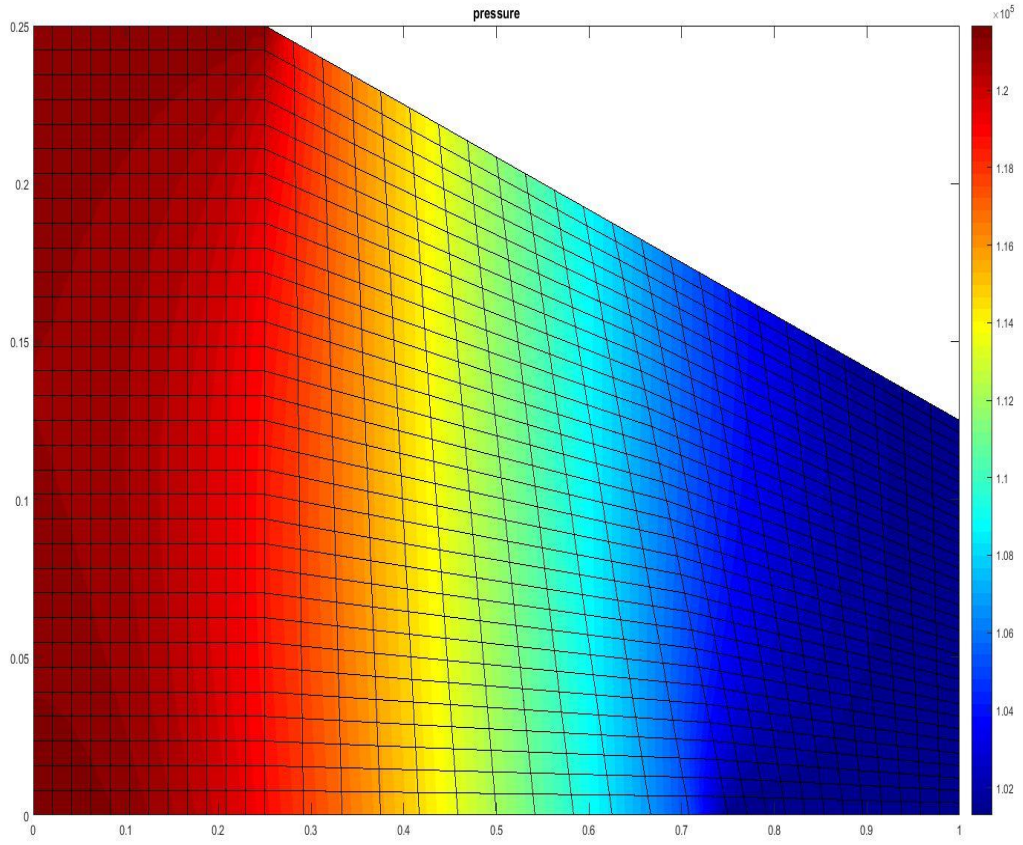


Figure 15(pressure distribution)

`

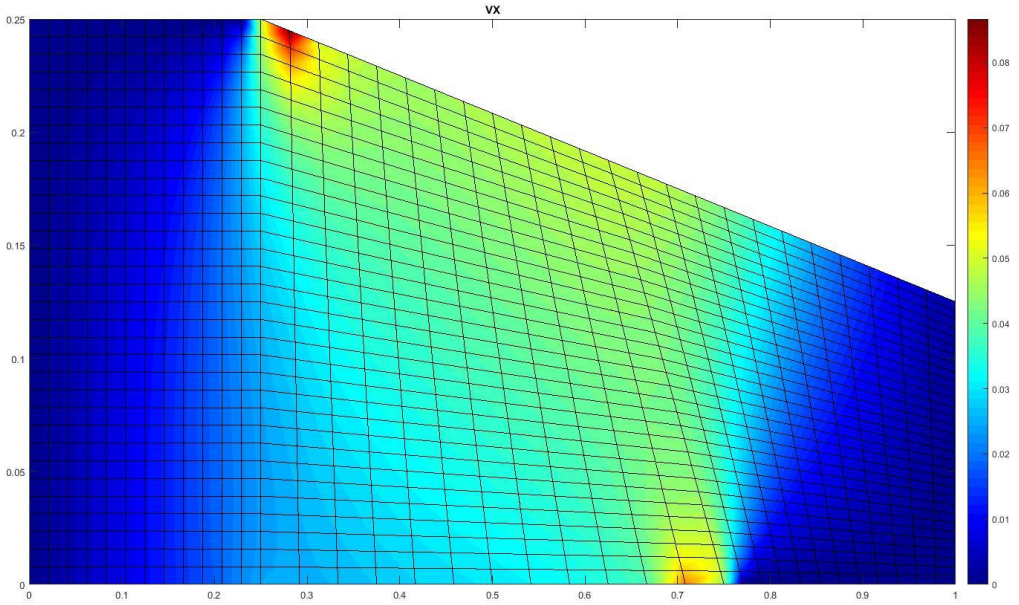


Figure 16(X velocity distribution)

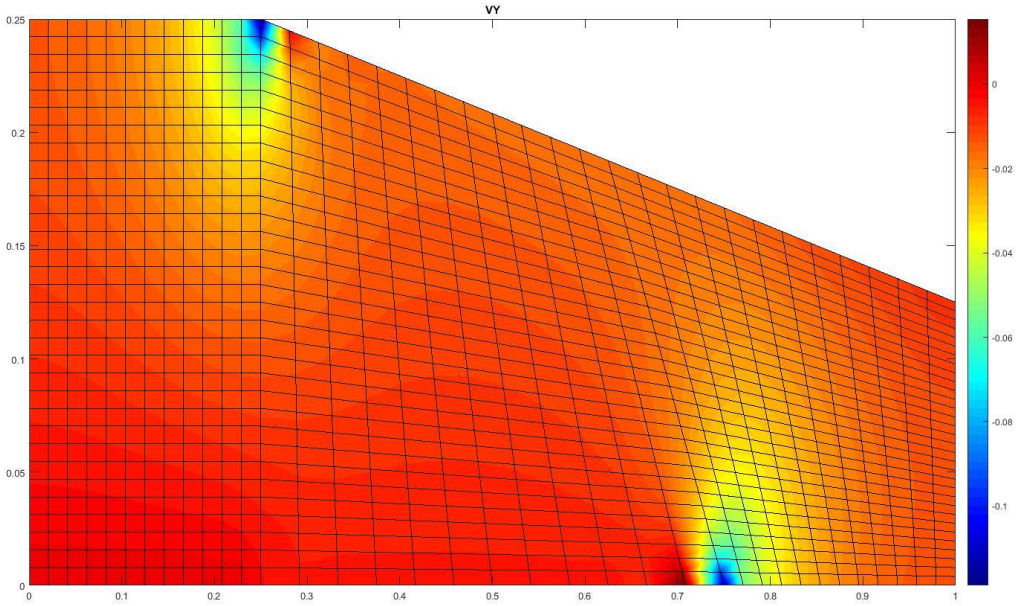


Figure 17(Y velocity distribution)

`

We have to calculate the value of pressure and velocity for (0.5, 0.1) for different level of refinement:

Mesh refinement level: 1 point location: element 86th crucial nodes: 60, 67, 66, 59

Mesh refinement level: 2 point location: element 229th crucial nodes: 197, 198

Mesh refinement level: 3 point location: element 460th crucial nodes: 415, 416, 392, 393

Mesh refinement level: 4 point location: element 739th crucial nodes: 682, 683

In the first case point is in the middle of element 86th .So, we can calculate the pressure and velocity by taking average of four corner nodes.

P (1) = [(1.1478+1.1011+1.1036+1.1489)*10^5]/4 = 1.1253*10^5

VX (1) = [(0.0347+0.0413+0.0427+0.0371)]/4 = 0.03896

VY (1) = [-(0.0095+0.0070+0.0095+0.0109)]/4 = -0.0092

In the second case point is on the edge of element 229th .So, we can calculate the pressure and velocity by taking average of two nodes on the edge.

P (2) = [(1.1260+1.1252)*10^5]/2 = 1.1256*10^5

VX (2) = [(0.0378+0.0368)]/2 = 0.0373

VY (2) = [-(0.0098+0.0090)]/2 = -0.0094

In the third case point is in the middle of element 460th .So, we can calculate the pressure and velocity by taking average of four corner nodes.

P (3) = [(1.1185+1.1178+1.1332+1.1327)*10^5]/4 = 1.1256*10^5

VX (3) = [(0.0377+0.0371+0.0364+0.0357)]/4 = 0.0367

VY (3) = [-(0.0097+0.0092+0.0097+0.0092)]/4 = -0.0094

In the fourth case point is on the edge of element 739th .So, we can calculate the pressure and velocity by taking average of two nodes on the edge.

P (4) = [(1.01259+1.0254)*10^5]/2 = 1.1257*10^5

VX (4) = [(0.0367+0.0362)]/4 = 0.0365

VY (4) = [-(0.0092+0.0096)]/4 = -0.0094

`

We can see the trend of convergence in Figure 18, 19 and 20.

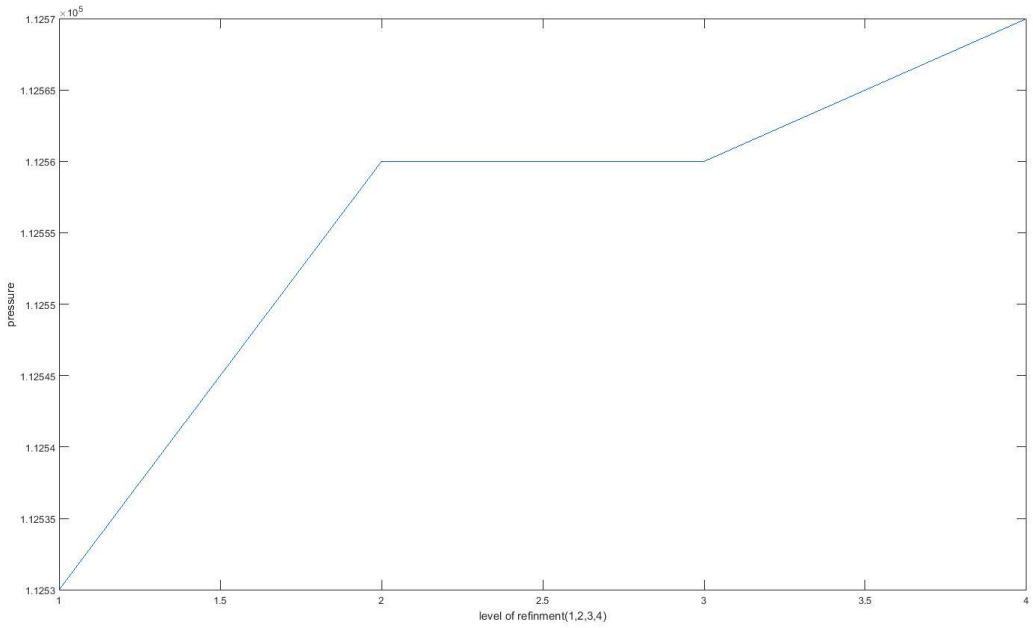


Figure 18(pressure)

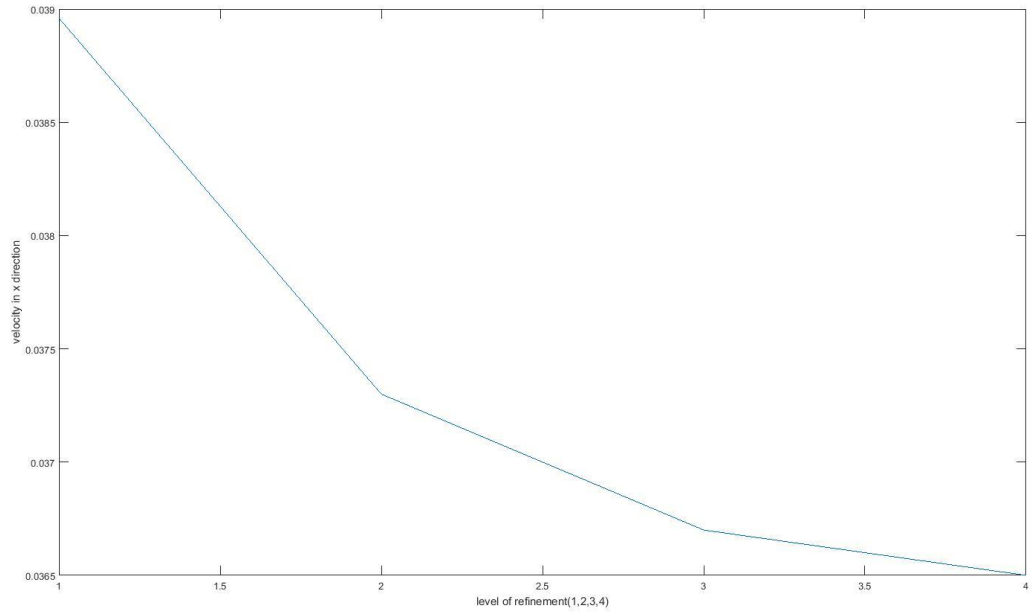


Figure 19(velocity in X direction)

`

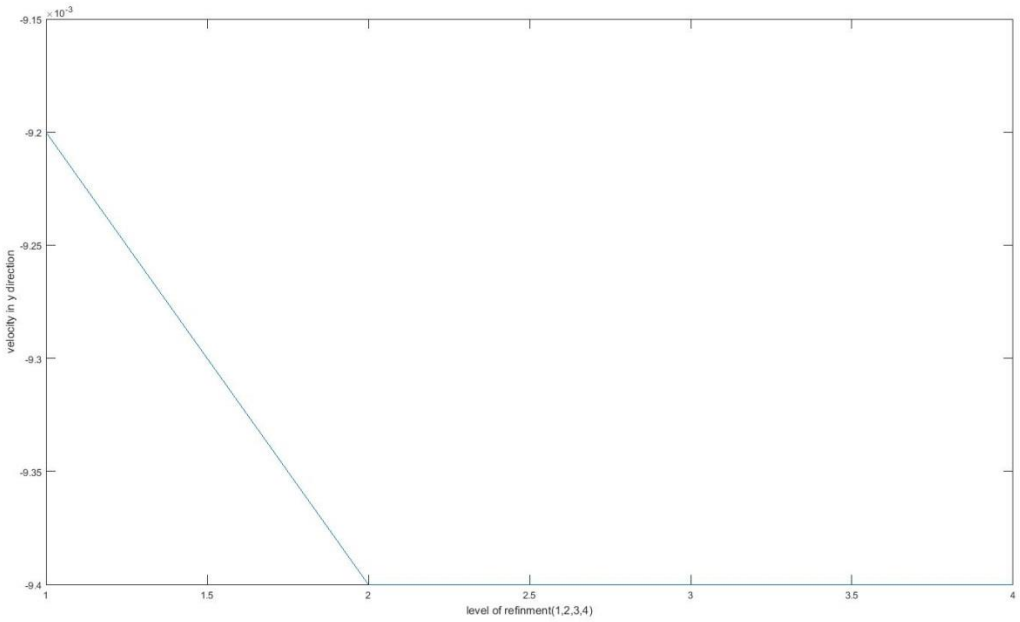


Figure 20(velocity in Y direction)

- ✓ Based on what we see in the above figures. It seems reasonable to use the mesh with 72 elements because it is much faster corresponding to running time, and there is not much difference in values of pressure and velocity. In fact, rate of convergence in all different cases are so fast it does not seem necessary to refine the mesh at all. Of course, Using 16 quadrature points (4*4) are one of the factors that help us to get a good rate of convergence.

`

(b)

I did p-refinement for the second order Lagrangian element. However, I still used the first order mapping. Consequently, shape functions are:

$$\psi_1 = \left(\tfrac{1}{4}\right)(x^2 - x)(y^2 - y)$$

$$\psi_2 = \left(\frac{1}{2}\right)(1 - x^2)(y^2 - y)$$

$$\psi_3 = \left(\frac{1}{4}\right)(x^2 + x)(y^2 - y)$$

$$\psi_4 = \left(\frac{1}{2}\right)(x^2 - x)(1 - y^2)$$

$$\psi_5 = (1 - x^2)(1 - y^2)$$

$$\psi_6 = \left(\frac{1}{2}\right)(x^2 + x)(1 - y^2)$$

$$\psi_7 = \left(\frac{1}{4}\right)(x^2 - x)(y^2 + y)$$

$$\psi_8 = \left(\frac{1}{2}\right)(1 - x^2)(y^2 + y)$$

$$\psi_9 = \left(\frac{1}{4}\right)(x^2 + x)(y^2 + y)$$

In this part I had to change my connectivity matrix that gmsh gave me and define a new connectivity matrix to use the above shape functions. (line 34-43, main code and line 20-30 velocity contour)

`

In this case my mesh has 72 elements and 323 nodes:

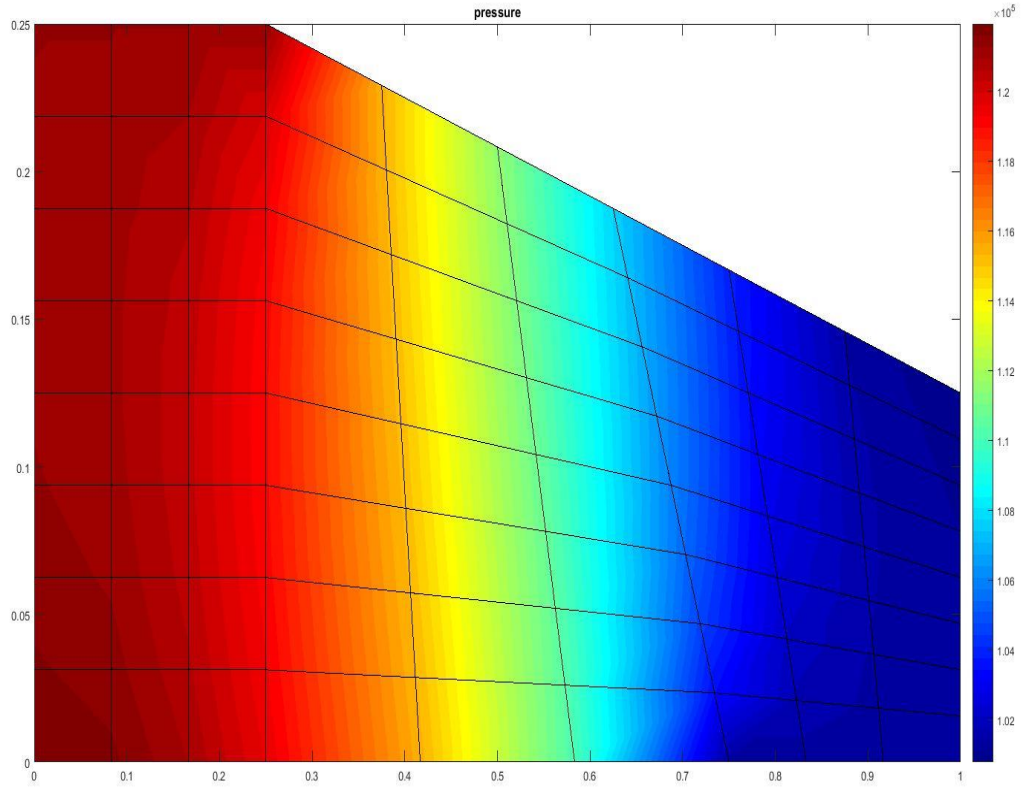


Figure 21(pressure distribution)

`

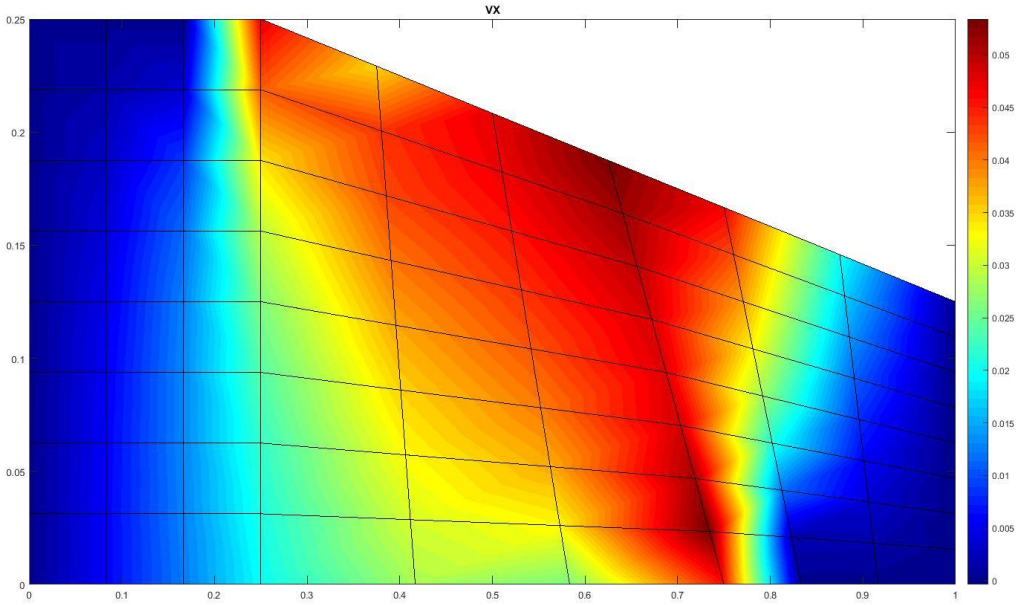


Figure 22(Velocity in X direction)

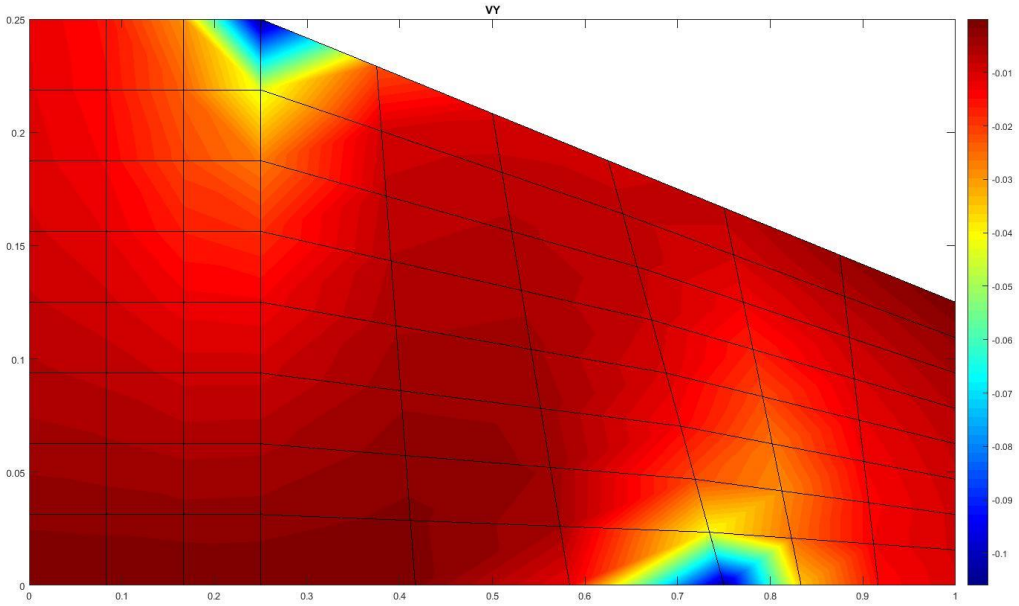


Figure 22(Velocity in Y direction)

`

Pressure and velocity values are at point (0.5, 0.1) for linear and quadratic Lagrangian elements are:

Linear characteristics are just like part C:

P (1) = [(1.1478+1.1011+1.1036+1.1489)*10^5]/4 = 1.1253*10^5

VX (1) = [(0.0347+0.0413+0.0427+0.0371)]/4 = 0.03896

VY (1) = [-(0.0095+0.0070+0.0095+0.0109)]/4 = -0.0092

And for quadratic case: (node 231)

P (2) = (1.1304)*10^5

VX (2) =0.0385

VY (2) =-0.0037

- ✓ We see that in this case, error for pressure and X velocity is less than 1%. But, it seems that we have bigger error for Y velocity. However, the rate of convergence still seems very fast and in my opinion there is no need to use higher order Lagrangian elements.

`

Part E :

According to what we discussed in page 10 about Neuman boundary conditions, I calculated the boundary integral for every point at inlet. (line 80-99 main code)

For the case of $u_y = 10^{-3}$ results may not be reasonable in physical sense because the value of velocity is very small at the inlet and may cause some weird parts in pressure and velocity contours. By using bigger values for velocity at inlet it is likely that we get more accurate and rational results.

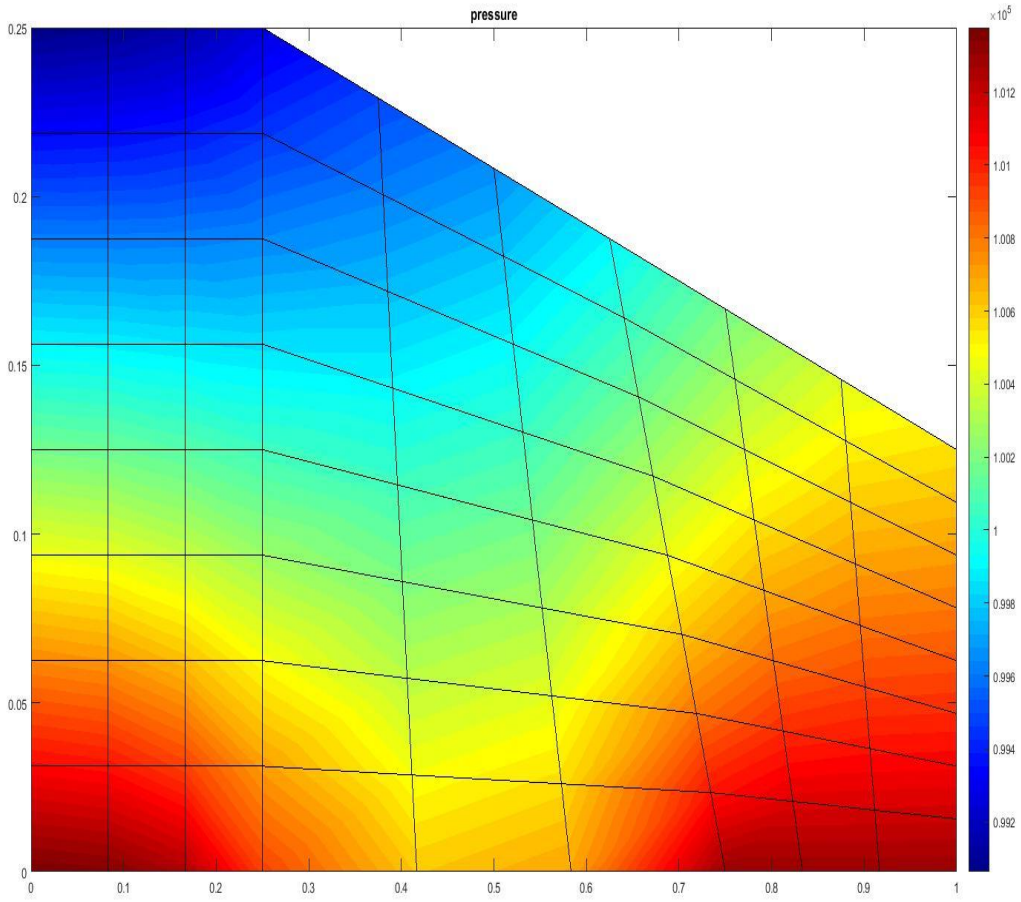


Figure 23(pressure for u=10^-3)

`

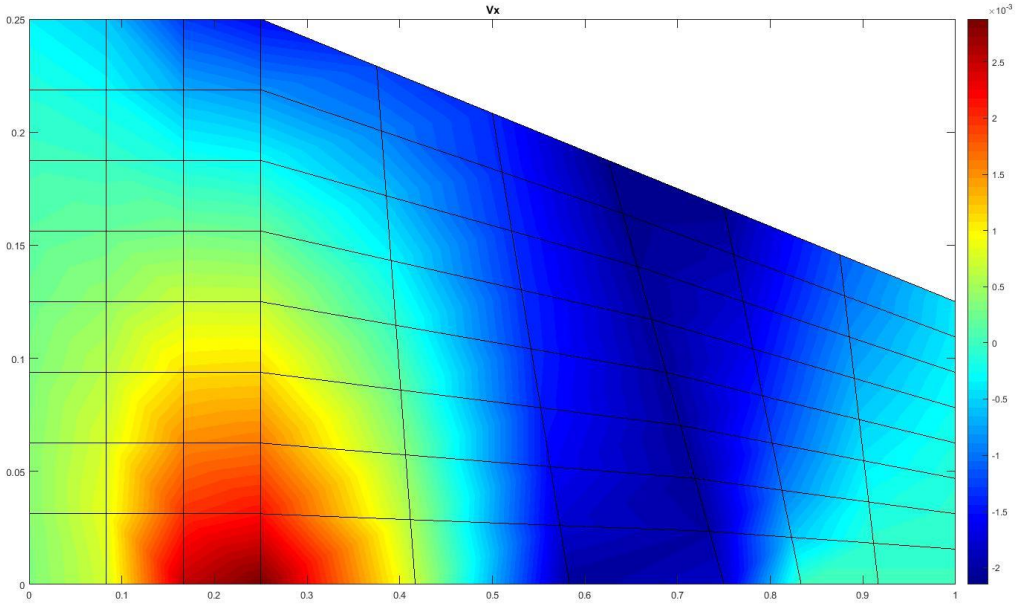


Figure 24(velocity in X direction for u=10^-3)

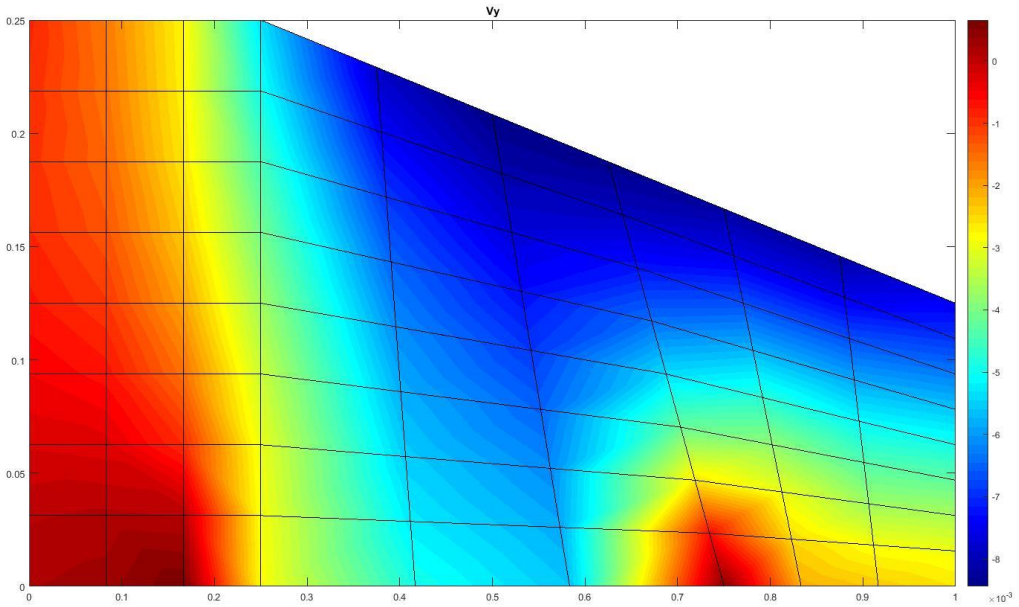


Figure 25(velocity in Y direction for u=10^-3)

`

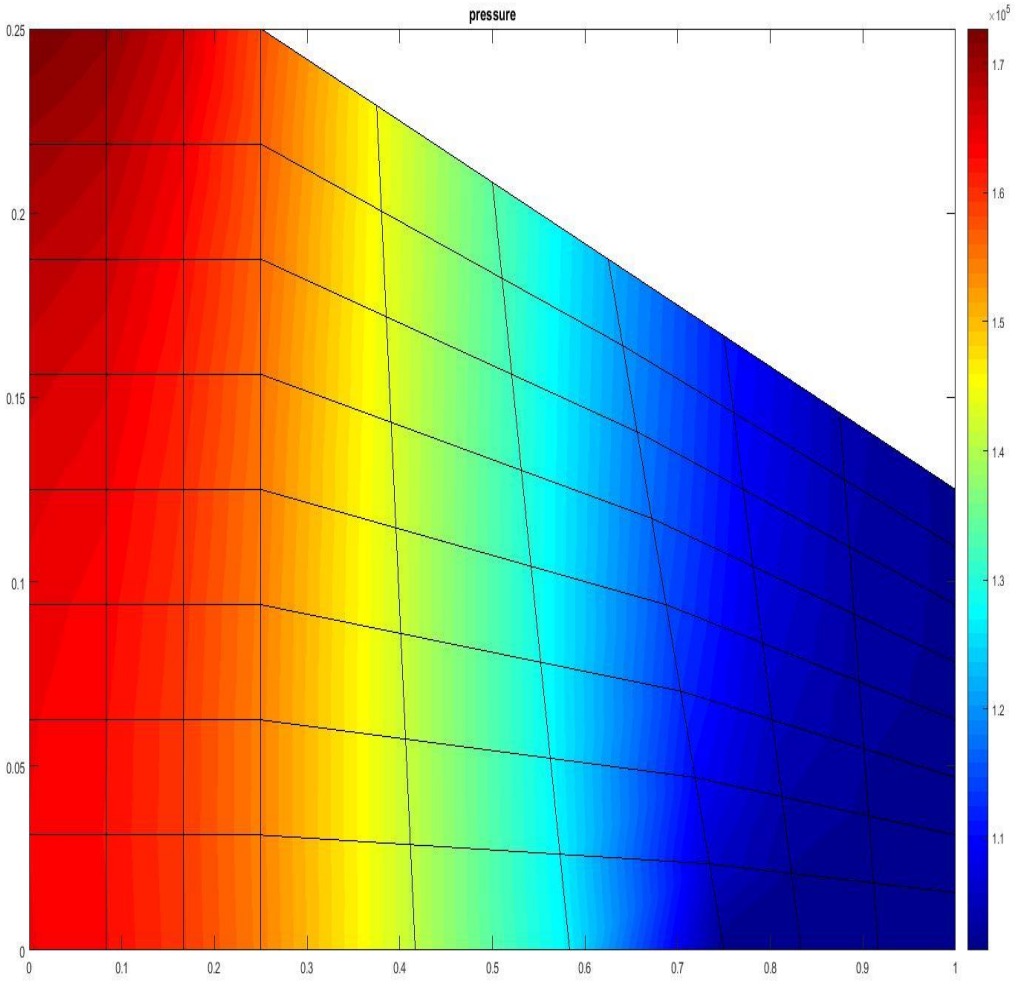


Figure 26(pressure for u=10^-1)

`

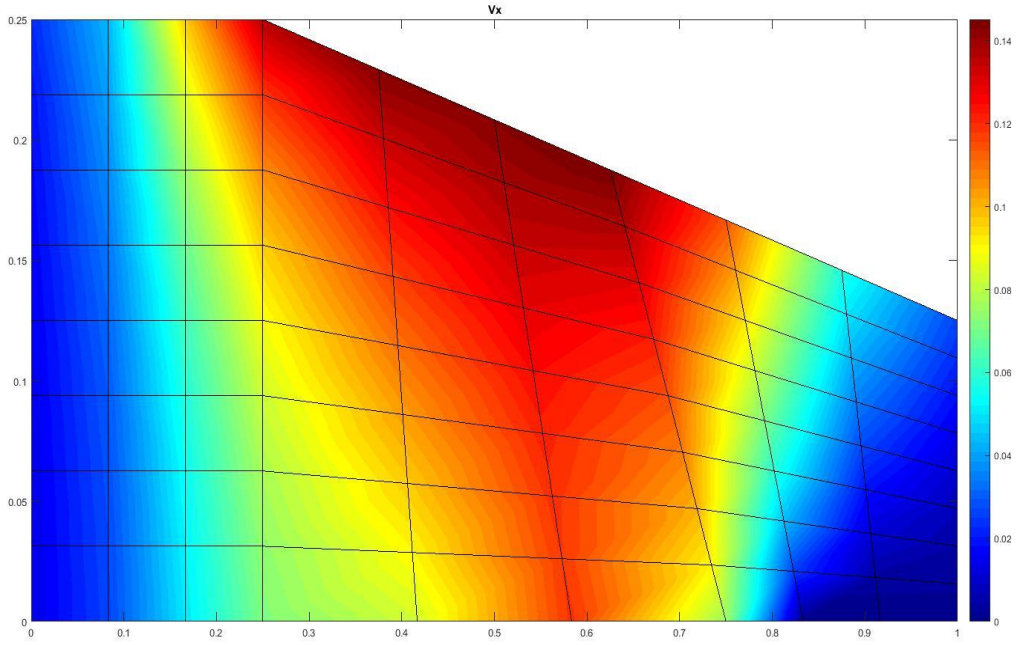


Figure 27(velocity in X direction for u=10^-1)

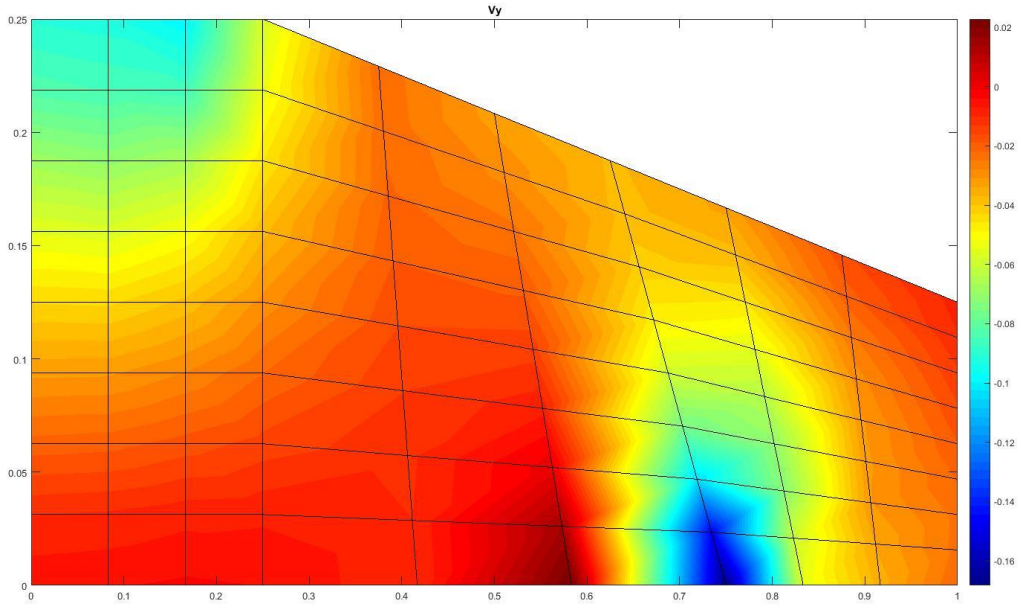


Figure 28(velocity in Y direction for u=10^-1)

`

MATLAB

Editor - D:\FEM PROJECT\mesh\part-c\maincode.m

maincode.m

```
clc ;
clear all ;


%%%%%%%%%%%%%coefficients
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

g=-9.81 ; miu=0.001 ; ro=1000 ; p0=121300 ; pinfinity=101300 ;
global x y Ex Ey connectivity ;
syms x y ;
global nnod nrelm ;

%%%%%%%%%%%organizing elements and nodes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load Elements.txt
load Nodes.txt

nrelm=size(Elements,1);
nnod=size(Nodes,1);
Ex=zeros(nrelm,4);
Ey=zeros(nrelm,4);
connectivity(:,1)=sort(1:nrelm);

for i=1:nrelm

    connectivity(i,1:4)=Elements(i,6:9); %%%%creating connectivity matrix
    Ex(i,:)=[Nodes(connectivity(i,1),2) Nodes(connectivity(i,2),2) Nodes(connectivity(i,3),2) Nodes(connectivity(i,
    Ey(i,:)=[Nodes(connectivity(i,1),3) Nodes(connectivity(i,2),3) Nodes(connectivity(i,3),3) Nodes(connectivity(i,

end
```

Command Window

>>

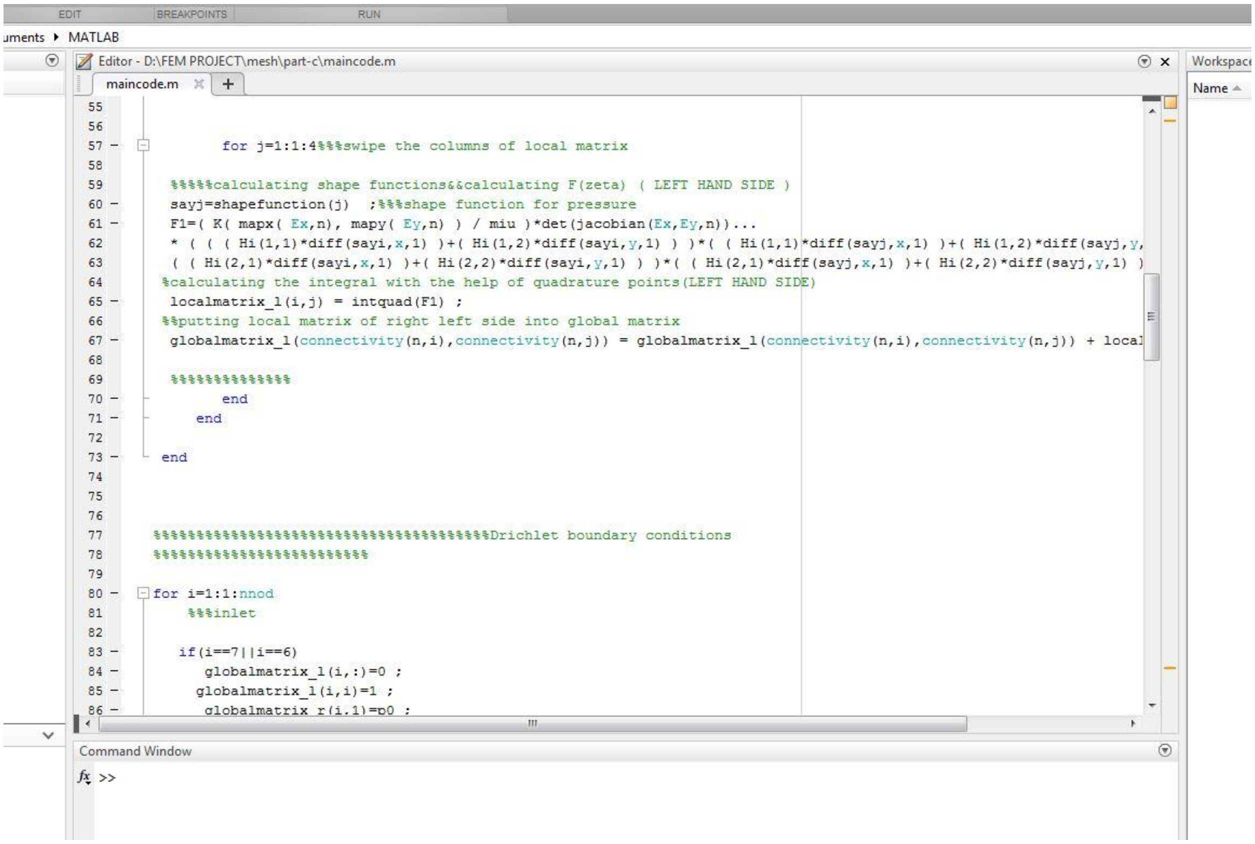
Editor - D:\FEM PROJECT\mesh\part-c\maincode.m

maincode.m

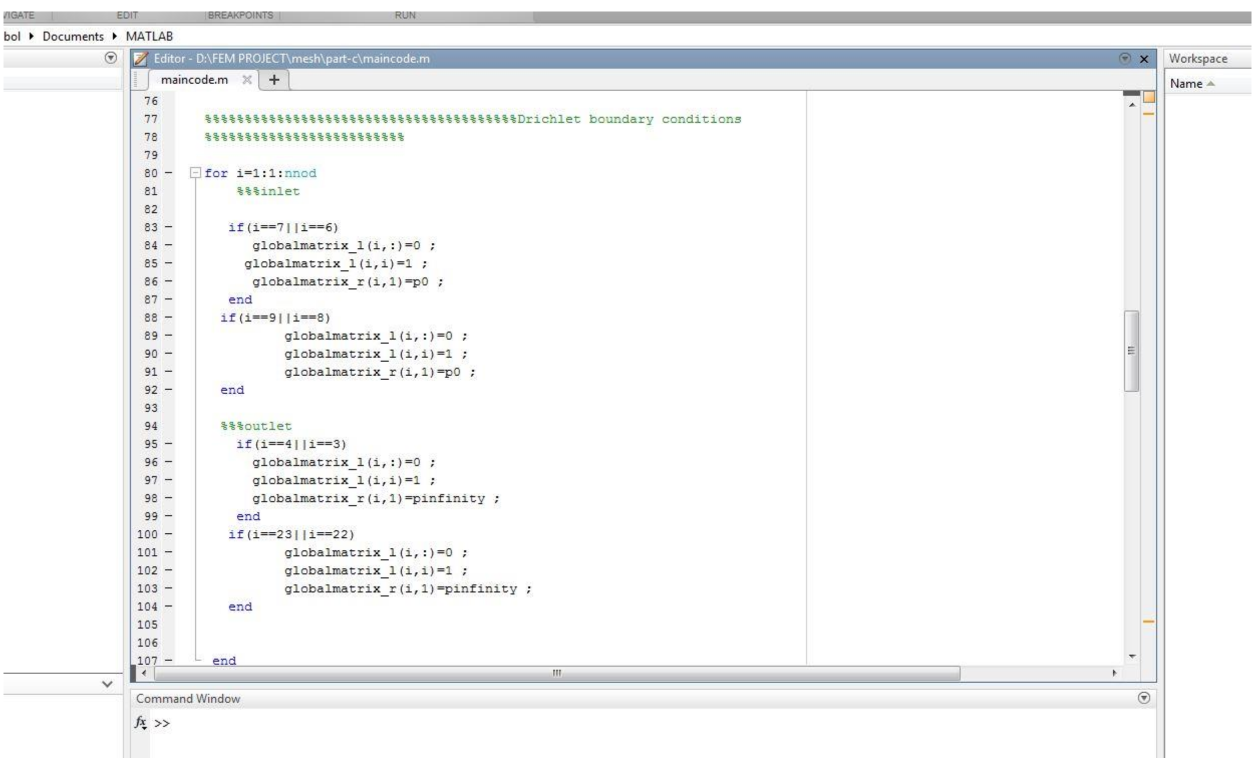
```
    Ex(i,:)=[Nodes(connectivity(i,1),2) Nodes(connectivity(i,2),2) Nodes(connectivity(i,3),2) Nodes(connectivity(i,
    Ey(i,:)=[Nodes(connectivity(i,1),3) Nodes(connectivity(i,2),3) Nodes(connectivity(i,3),3) Nodes(connectivity(i,

end



%%%%%%%%%%%%%%%%%%creating local matrix (linear 4*4)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

localmatrix_l = zeros(4,4) ;%%left
localmatrix_r = zeros(4,1) ;%%right
globalmatrix_l = zeros(nnod,nnod) ;%%left
globalmatrix_r = zeros(nnod,1) ;%%right


for n=1:1:nrelm%%%%swipe the elements

    for i=1:1:4%%%swipe the rows of local matrix

  sayi=shapefunction(i)  ;%%shape function
  Hi=inv(jacobian(Ex,Ey,n)) ;
  F2=(  K( mapx( Ex,n), mapy( Ey,n) ) / miu ) * g * ro * ( Hi(2,1)*diff(sayi,x,1)+Hi(2,2)*diff(sayi,y,1) )*det(jacc
  %calculating the integral with the help of quadrature points(RIGHT HAND SIDE)
  localmatrix_r(i,1,n) =intquad(F2) ;
  %%putting local matrix of right hand side into global matrix
  globalmatrix_r(connectivity(n,i)) = globalmatrix_r(connectivity(n,i)) + localmatrix_r(i,1) ;


        for j=1:1:4%%%swipe the columns of local matrix

  %%%%%calculating shape functions&&calculating F(zeta) ( LEFT HAND SIDE )
```

Command Window

>>



```
        for j=1:1:4%%%swipe the columns of local matrix

 %%%%%calculating shape functions&&calculating F(zeta) ( LEFT HAND SIDE )
 sayj=shapefunction(j)  ;%%%shape function for pressure
 F1=( K( mapx( Ex,n), mapy( Ey,n) ) / miu )*det(jacobian(Ex,Ey,n))...
 * ( ( ( Hi(1,1)*diff(sayi,x,1) )+( Hi(1,2)*diff(sayi,y,1) ) )*( ( Hi(1,1)*diff(sayj,x,1) )+( Hi(1,2)*diff(sayj,y,
 ( ( Hi(2,1)*diff(sayi,x,1) )+( Hi(2,2)*diff(sayi,y,1) ) )*( ( Hi(2,1)*diff(sayj,x,1) )+( Hi(2,2)*diff(sayj,y,1) )
%calculating the integral with the help of quadrature points(LEFT HAND SIDE)
 localmatrix_l(i,j) = intquad(F1) ;
%%putting local matrix of right left side into global matrix
 globalmatrix_l(connectivity(n,i),connectivity(n,j)) = globalmatrix_l(connectivity(n,i),connectivity(n,j)) + local

 %%%%%%%%%%%%%%
        end
     end

 end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Drichlet boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:1:nnod
    %%%inlet

   if(i==7||i==6)
      globalmatrix_l(i,:)=0 ;
     globalmatrix_l(i,i)=1 ;
      globalmatrix_r(i,1)=p0 ;
```

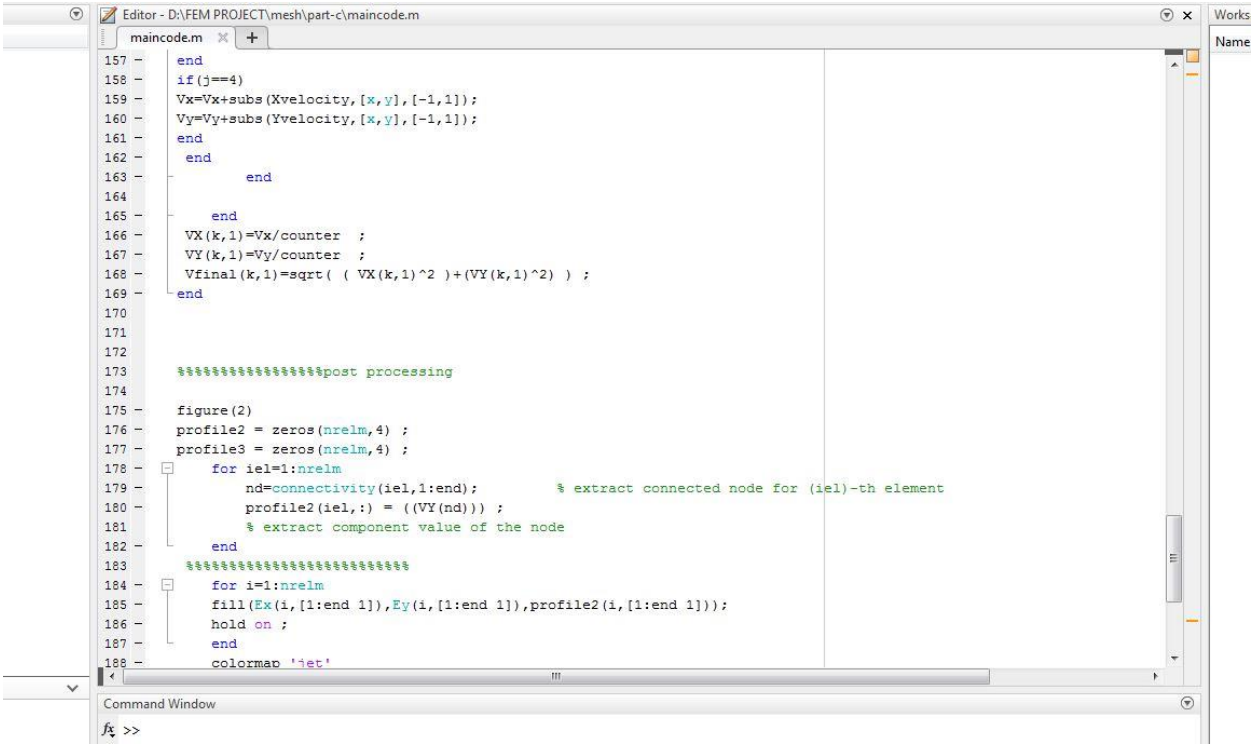


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Drichlet boundary conditions
%%%%%%%%%%%%%%%%%%%%%%%%%%

for i=1:1:nnod
    %%%inlet

   if(i==7||i==6)
      globalmatrix_l(i,:)=0 ;
     globalmatrix_l(i,i)=1 ;
      globalmatrix_r(i,1)=p0 ;
   end
  if(i==9||i==8)
          globalmatrix_l(i,:)=0 ;
          globalmatrix_l(i,i)=1 ;
          globalmatrix_r(i,1)=p0 ;
  end

  %%%outlet
    if(i==4||i==3)
      globalmatrix_l(i,:)=0 ;
      globalmatrix_l(i,i)=1 ;
      globalmatrix_r(i,1)=pinfinity ;
    end
   if(i==23||i==22)
          globalmatrix_l(i,:)=0 ;
          globalmatrix_l(i,i)=1 ;
          globalmatrix_r(i,1)=pinfinity ;
   end


 end
```

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%post processing the pressure
global Pressure ;
global profile ;
Pressure = globalmatrix_l\globalmatrix_r ;
profile = zeros(nrelm,4) ;

    for iel=1:nrelm
        nd=connectivity(iel,1:end);          % extract connected node for (iel)-th element
        profile(iel,:) = (Pressure(nd)) ;         % extract component value of the node
    end
    figure(1)
    for i=1:nrelm
    fill(Ex(i,[1:end 1]),Ey(i,[1:end 1]),profile(i,[1:end 1]));
    hold on ;
    end
    colormap 'jet'
    title('pressure')


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%velocity calculation
VX=zeros(nnod,1) ;
VY=zeros(nnod,1) ;
Vfinal=zeros(nnod,1) ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for k=1:1:nnod
    counter=0 ;
    Vx=0;
    Vy=0;
    for n=1:1:nrelm
        for j=1:1:4
if( connectivity(n,j)== k )
```

```matlab
        for j=1:1:4
if( connectivity(n,j)== k )
counter=counter+1 ;
A = (profile(n,1)*shapefunction(1))+(profile(n,2)*shapefunction(2))+(profile(n,3)*shapefunction(3))+(profile(n,4)*s
Hi=inv(jacobian(Ex,Ey,n)) ;
Xvelocity= -(  K( Ex(n,j),Ey(n,j))/miu)* ( Hi(1,1)*diff(A,x,1)+Hi(1,2)*diff(A,y,1) ) ;
Yvelocity= -(  K( Ex(n,j),Ey(n,j)) / miu )* ( (Hi(2,1)*diff(A,x,1)+Hi(2,2)*diff(A,y,1)) - (ro*g) ) ;
if(j==1)
Vx=Vx+subs(Xvelocity,[x,y],[-1,-1]);
Vy=Vy+subs(Yvelocity,[x,y],[-1,-1]);
end
if(j==2)
Vx=Vx+subs(Xvelocity,[x,y],[1,-1]);
Vy=Vy+subs(Yvelocity,[x,y],[1,-1]);
end
if(j==3)
Vx=Vx+subs(Xvelocity,[x,y],[1,1]);
Vy=Vy+subs(Yvelocity,[x,y],[1,1]);
end
if(j==4)
Vx=Vx+subs(Xvelocity,[x,y],[-1,1]);
Vy=Vy+subs(Yvelocity,[x,y],[-1,1]);
end
 end
         end

     end
 VX(k,1)=Vx/counter  ;
 VY(k,1)=Vy/counter  ;
 Vfinal(k,1)=sqrt( ( VX(k,1)^2 )+(VY(k,1)^2) ) ;
end
```


`


Editor - D:\FEM PROJECT\mesh\part-c\maincode.m
maincode.m
end
if(j==4)
Vx=Vx+subs(Xvelocity,[x,y],[-1,1]);
Vy=Vy+subs(Yvelocity,[x,y],[-1,1]);
end
end
end
end
VX(k,1)=Vx/counter ;
VY(k,1)=Vy/counter ;
Vfinal(k,1)=sqrt((VX(k,1)^2)+(VY(k,1)^2)) ;
end
%%%%%%%%%%%%%%%%%post processing
figure(2)
profile2 = zeros(nrelm,4) ;
profile3 = zeros(nrelm,4) ;
for iel=1:nrelm
nd=connectivity(iel,1:end); % extract connected node for (iel)-th element
profile2(iel,:) = ((VY(nd))) ;
% extract component value of the node
end
%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:nrelm
fill(Ex(i,[1:end 1]),Ey(i,[1:end 1]),profile2(i,[1:end 1]));
hold on ;
end
colormap 'jet'
Command Window
>>


maincode.m
%%%%%%%%%%%%%%%%%post processing
figure(2)
profile2 = zeros(nrelm,4) ;
profile3 = zeros(nrelm,4) ;
for iel=1:nrelm
nd=connectivity(iel,1:end); % extract connected node for (iel)-th element
profile2(iel,:) = ((VY(nd))) ;
% extract component value of the node
end
%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:nrelm
fill(Ex(i,[1:end 1]),Ey(i,[1:end 1]),profile2(i,[1:end 1]));
hold on ;
end
colormap 'jet'
title('Vy')
%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%
for iel=1:nrelm
nd=connectivity(iel,1:end); % extract connected node for (iel)-th element
profile3(iel,:) = ((VX(nd))) ; % extract component value of the node
end
%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(3)
for i=1:nrelm
fill(Ex(i,[1:end 1]),Ey(i,[1:end 1]),profile3(i,[1:end 1]));
hold on ;
end
colormap 'jet'
title('Vx')

`