Exercise 4 - Optional Problem
Finite Elements in Fluids
Master of Science in Computational Mechanics 2016

Paris Dilip Mulye

June 4, 2016

PART A

The strong form of given problem is as follows,

$$-\nabla\cdot(\mu\nabla\rho)+\bar{\boldsymbol{u}}\cdot\nabla\rho=s(\bar{\boldsymbol{u}}) \qquad \rho_{\Gamma_1,\Gamma_2}=1 \qquad \rho_{\Gamma_4}=0 \tag{1}$$
$$-\nabla\cdot(\nu\nabla)\bar{\boldsymbol{u}}-\nabla p=0 \qquad u_{\Gamma_1,\Gamma_2}=1 \qquad v_{\Gamma_1,\Gamma_2}=0 \qquad p(0,0)=0 \tag{2}$$
$$\nabla\cdot\bar{\boldsymbol{u}}=0 \tag{3}$$

Where,

$$\nu(\rho)=\nu_0+\nu_0\left(\frac{1}{1+e^{-10(\rho-0.5)}}\right)$$
$$s(\bar{\boldsymbol{u}})=\frac{1}{1+e^{-10(|\bar{\boldsymbol{u}}|-0.5)}}$$

Method Summary

Since the coupled nature of equations, a global matrix assembly would be done with DoFs ρ, u, v, p. To simplify, a fixed point iteration method would be used to tackle the nonlinearity of problem. Also, to further simplify, element type Q1-Q1 would be used which is bilinear in ρ, u, v and p. This reduces the complexity since the nodes for all DoFs coincide which simplifies Gaussian point integrations for coupled terms. It is important to note that this element does not satisfy LBB stability condition. Space stabilization for ρ is done using SUPG method.

Weak Form

Equation 1

All weak forms are derived with the fact that Γ_N is zero wherever Dirichlet conditions are not specified. Therefore, flux on Neumann are not inserted into the equation. Also, fluxes on Dirichlet conditions are not needed for solving the system (they can be obtained from eigenvalues). Therefore, flux terms could be skipped in the derivation of weak form. Also, for brevity of representation, integration signs are omitted from all terms. (1) can be reduced to weak form as follows for Galerkin Form, $(w_i = N_i)$

$$-w\nabla\cdot(\mu\nabla\rho)+w\bar{\boldsymbol{u}}\cdot\nabla\rho=ws(\bar{\boldsymbol{u}})$$
$$\mu(\nabla w\cdot\nabla\rho)+\bar{\boldsymbol{u}}\cdot(\nabla w\rho)=ws(\bar{\boldsymbol{u}})$$
$$\mu(\nabla N_i\cdot\nabla N_j)\rho^h+\bar{\boldsymbol{u}}\cdot(\nabla N_iN_j)\rho^h=N_is(\bar{\boldsymbol{u}})$$
$$(K_{rho}+C^T)\rho^h=S$$

Equation 2

(2) can be converted into weak form, with same assumptions for the flux above, the bold values of shape functions, indicate that they are for velocity, $\boldsymbol{N_i} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} N_i$

$$\begin{aligned} -\bar{\boldsymbol{w}} \cdot \nabla \cdot (\nu \nabla)\bar{\boldsymbol{u}} - \bar{\boldsymbol{w}} \cdot \nabla p &= 0 \\ \nu \nabla \bar{\boldsymbol{w}} : \nabla \bar{\boldsymbol{u}} - \bar{\boldsymbol{w}} \cdot \nabla p &= 0 \\ \nu \nabla \boldsymbol{N_i} : \nabla \boldsymbol{N_j} \bar{\boldsymbol{u}}^h - \boldsymbol{N_i} \cdot \nabla N_j p^h &= 0 \\ K \bar{\boldsymbol{u}}^h + G^T p^h &= 0 \end{aligned}$$

Equation 3

(3) can be converted into weak form. with same assumptions for the flux above,

$$\begin{aligned} w \nabla \cdot \bar{\boldsymbol{u}} &= 0 \\ \nabla w \cdot \bar{\boldsymbol{u}} &= 0 \\ -\nabla N_i \cdot \boldsymbol{N_j} \bar{\boldsymbol{u}}^h &= 0 \\ G \bar{\boldsymbol{u}}^h &= 0 \end{aligned}$$

Assembly of Matrices

All three equations can be assembles into a single system as follows, (Dirichlet BCs yet to implement)

$$\begin{bmatrix} K_{rho} + C^T & 0 & 0 \\ 0 & K & G^T \\ 0 & G & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \bar{\boldsymbol{u}} \\ p \end{bmatrix} = \begin{bmatrix} S \\ 0 \\ 0 \end{bmatrix}$$

Dirichlet BCs

The given BCs were applied in Lagrange form, $A\_DirBC\lambda = b\_DirBC$. The pressure BC is also included in this BC matrix. This makes the full system as follows,

$$\begin{bmatrix} K_{rho} + C^T & 0 & 0 & (A\_DirBC)^T \\ 0 & K & G^T & \\ 0 & G & 0 & \\ (A\_DirBC) & & & 0 \end{bmatrix} \begin{bmatrix} \rho \\ \bar{\boldsymbol{u}} \\ p \\ \lambda \end{bmatrix} = \begin{bmatrix} S \\ 0 \\ 0 \\ (b\_DirBC) \end{bmatrix}$$

Note that, the (A_DirBC) matrix spans three columns for $\rho, \bar{\boldsymbol{u}}$ and p respectively.

Calculation of Matrices

Since, iterative scheme has been implemented, the solution at k is known at iteration = k+1. The function `Stokes_system` uses this global solution to calculate the elemental solution and sends it to the function `EleMatStokes`. This function calculates the elemental matrices and returns them to the parent function for assembly. The relevant code is mentioned below. The below mentioned quantities are calculated at every gauss point in the element. Also, it is important to note that the SUPG correction term has been added to K_{rho} and S.

```
%obtain density at gauss point using previous solution
    rho_ig = N(ig,:)*sole(1:nedofP);

%calculate nu at gauss point
    nu_ig = nu + nu*(1+exp(-10*(rho_ig-0.5)))^-1;
```

```
%obtain velocity at gauss point
    u_ig_x = N(ig,:)*sole(nedofP+1:2:nedofP+nedofV);
    u_ig_y = N(ig,:)*sole(nedofP+2:2:nedofP+nedofV);

%obtain magnitude of velocity
    u_mag = sqrt(u_ig_x^2+u_ig_y^2);

%calculate source term at gauss point
    s_ig = (1+exp(-10*(u_mag-0.5)))^-1;

% calculate stiffness for stokes
    Ke = Ke + nu_ig*(Nx'*Nx+Ny'*Ny)*dvolu;

%calculate G for stokes
    Ge = Ge - N_ig'*dN*dvolu;

%source for stokes (zero in present case)
    x_ig = N_ig(1:ngeom)*Xe;
    f_igaus = SourceTerm(x_ig);
    fe = fe + Ngp'*f_igaus*dvolu;

%calculate Peclet, tau for correction
    tol = 1e-15; %to avoid division by zero
    h = Xe(1,1)-Xe(2,1);
    Pe = u_mag*h/(2*nu_ig+tol);
    tau = h*(1 + 9/Pe^2)^(-1/2)/(2*u_mag+tol);

%K for density
    Krhoe = Krhoe + (mu*(nx.'*nx + ny.'*ny)+...
    tau*(u_ig_x*nx+u_ig_y*ny)'*(u_ig_x*nx+u_ig_y*ny))*dvolu;

%C for density
    Ce = Ce + N_ig'*(u_ig_x*nx+u_ig_y*ny)*mu;

%source term S for density
    Se = Se + (N_ig+tau*(u_ig_x*nx+u_ig_y*ny))'*s_ig*dvolu;
```

Solver

Based on the above derivation, the iterative solver has been implemented as follows,

```
while true
    [K,G,f,C,Krho,S] = Stokes_system(X,T,XP,TP,referenceElement,nu,mu,sol);
    [ndofP,ndofV] = size(G);

    Atot = [Krho+C.'                zeros(nodes,2*nodes)      zeros(nodes,nodes)
            zeros(2*nodes,nodes)    K                         G.'
              zeros(nodes,nodes)    G                         zeros(nodes,nodes)];

    Atot = [Atot,       A_DirBC.'
            A_DirBC,    zeros(nDir,nDir)];
    Btot = [S;zeros(2*nodes,1);zeros(nodes,1);b_DirBC];
    solnew = Atot\Btot;
    rel_error = abs(norm(solnew)-norm(sol))/norm(solnew)*100;
    if rel_error < 1
        break
    else
        sol = solnew;
    end
end
```

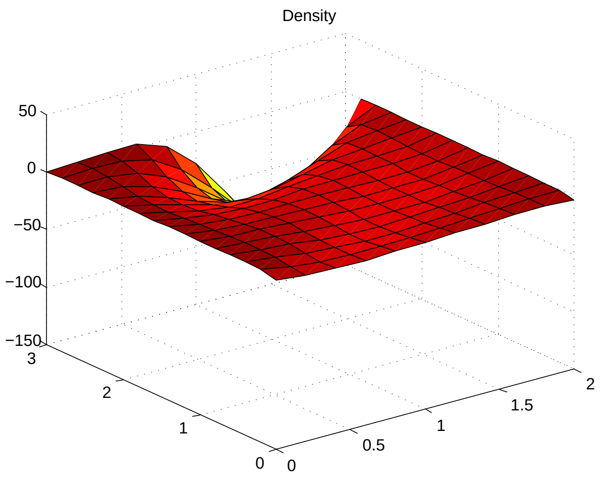

Figure 1: Case 1, Density

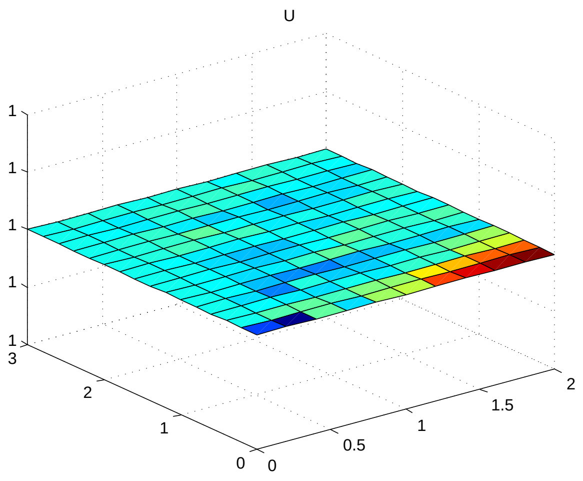

Figure 2: Case 1, Velocity in X Direction

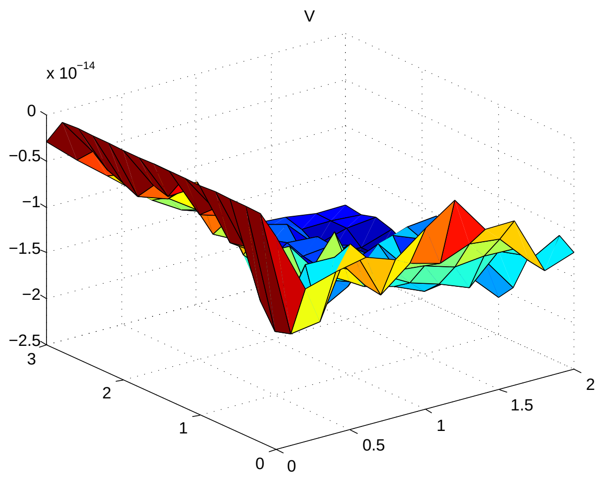

Figure 3: Case 1, Velocity in Y Direction

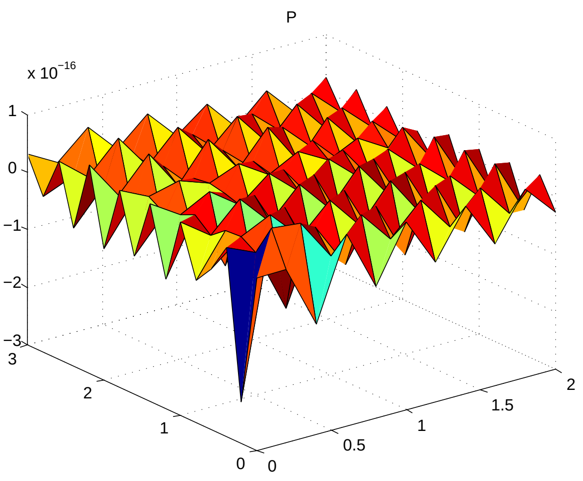

Figure 4: Case 1, Pressure

Solution: Case 1

The solution for μ = 1e-3 and ν_0 = 1e-4, is as shown in Figure 1 to Figure 4. This is a convection dominated problem since $|\bar{\boldsymbol{u}}| \approx 1$. The density variation had spurious oscillations but they disappear as the SUPG stabilization is implemented, as expected. The velocity in X has a constant value of 1 throughout domain. The velocity in Y has almost 0 solution. Pressure is 0 as well. The solution of stokes equation seems a trivial solution. The oscillations in pressure plot are due to two facts, numerical error (the error is of the order of 1e-18) and the fact that Q1-Q1 element does not satisfy the LBB stability condition.

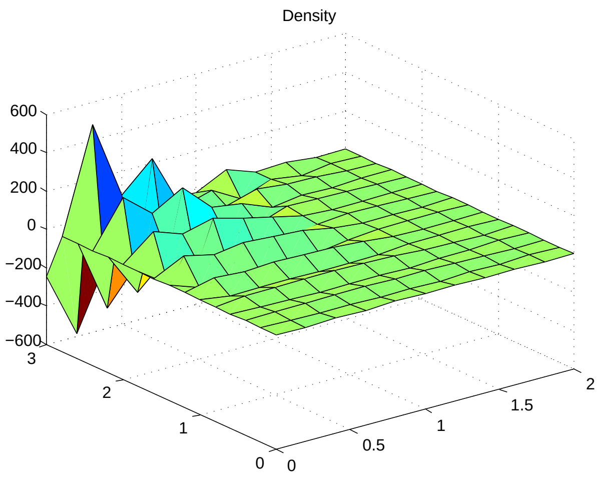

Figure 5: Case 2, Density

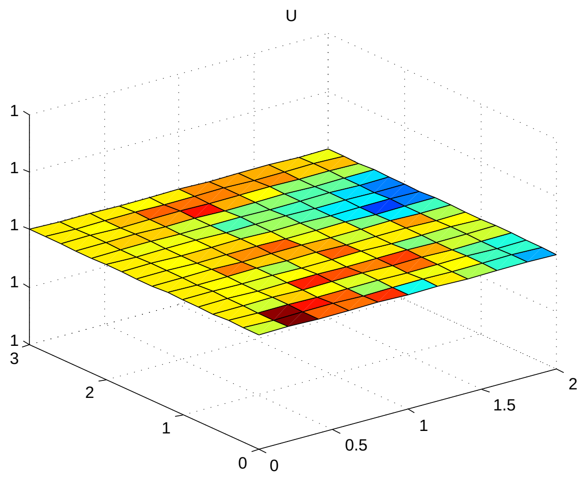

Figure 6: Case 2, Velocity in X Direction

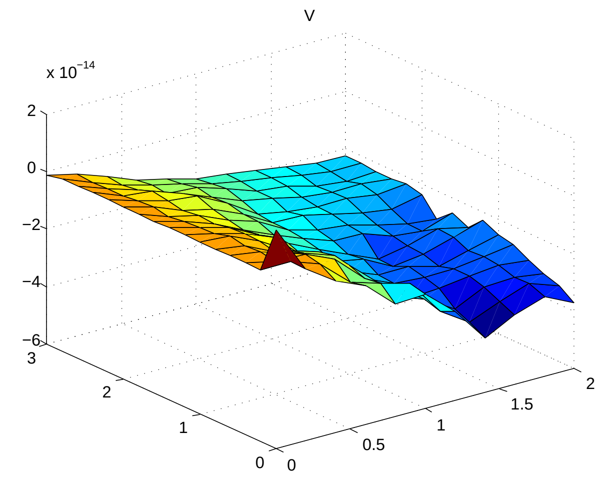

Figure 7: Case 2, Velocity in Y Direction

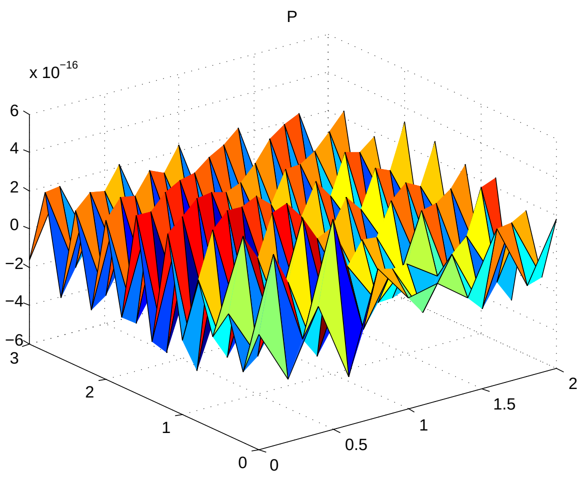

Figure 8: Case 2, Pressure

Solution: Case 2

The solution for $\mu = 1$ and $\nu_0 = $ 1e-4, is as shown in Figure 5 to Figure 8. This is a convection-diffusion problem since $|\bar{\boldsymbol{u}}| \approx \mu \approx 1$. The velocity in X has a constant value of 1 throughout domain. The velocity in Y has almost 0 solution. Pressure is 0 as well. The solution of stokes equation seems a trivial solution. The oscillations in pressure plot can be explained same as Case 1.

Conclusion

From the solution plots, it is clear that Stokes solution seems correct but trivial. It is difficult to deviate from this stable solution, even after changing the initial solution guess.

PART B

Not Attempted.

THE END