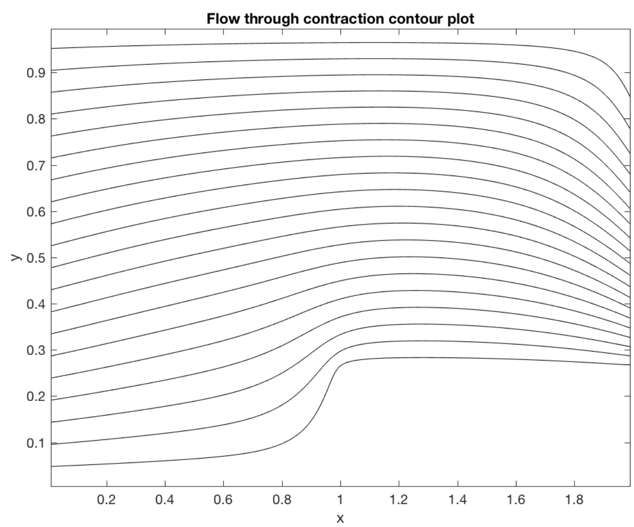
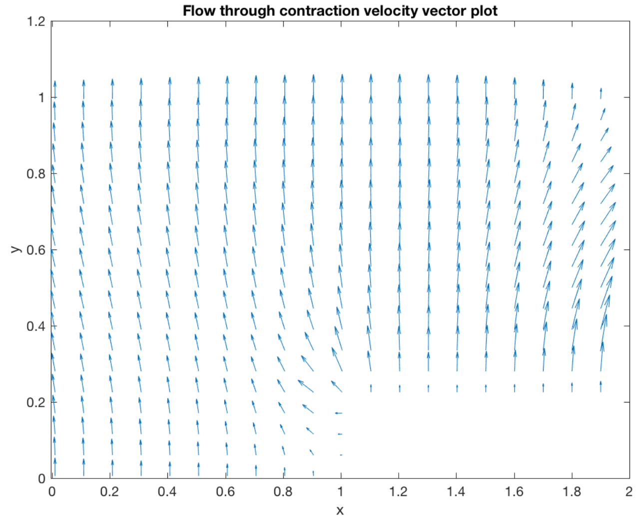
Caleb Logemann
AER E 546 Fluid Mechanics and Heat Transfer I
Homework 6

1. In order to determine the last boundary condition, we must realize that the flow in must be the same as the flow out. That is
$$\int_0^1 \Psi(0, y) \, \mathrm{d}y = \int_{1/4}^1 \Psi(2, y) \, \mathrm{d}y$$
Also in order to be continuous $\Psi(2, 1/4) = 0$ and $\Psi(2, 1) = -1$. We can achieve these three conditions with a quadractic polynomial $p(y) = ay^2 + by + c$. We have the following equations.
$$\begin{aligned} -1 &= a + b + c \\ 0 &= \frac{1}{16}a + \frac{1}{4}b + c \\ -\frac{1}{2} &= \frac{1}{3}\left(1 - \frac{1}{64}\right)a + \frac{1}{2}\left(1 - \frac{1}{16}\right)b + \left(1 - \frac{1}{4}\right)c \end{aligned}$$
Solving these gives
$$\begin{aligned} a &= \frac{16}{9} \\ b &= -\frac{32}{9} \\ c &= \frac{7}{9} \end{aligned}$$
or a boundary condition of
$$\Psi(2, y) = \frac{16}{9}y^2 - \frac{32}{9}y + \frac{7}{9}$$

```
function [u, k, res] = sor3(lambda, u, x, y, b, tol, maxIter)
    [I, J] = size(u);
    uold = u;
    k = 0;
    mstop = 1;
    res = zeros(1,maxIter);
    residual0 = 0;
    while(mstop && k < maxIter)
        k = k + 1;
        for i = 1:I
            for j = 1:J
                if (i == I) % right boundary x = 2
                    uip1j = (16/9)*y(j)^2 - (32/9)*y(j) + 7/9;
                    uim1j = u(i-1, j);
                elseif (i == 1) % left boundary x = 0
                    uip1j = u(i+1,j);
                    uim1j = -y(j);
                else
                    uip1j = u(i+1,j);
                    uim1j = u(i-1, j);
                end

                if (j == J) % top boundary, y = 1
                    uijp1 = -1;
                    uijm1 = u(i,j-1);
                elseif (j == 1) % bottom boundary y = 0
                    uijp1 = u(i,j+1);
```

```
                    uijm1 = 0;
                else
                    uijp1 = u(i,j+1);
                    uijm1 = u(i,j-1);
                end

                if ( x(i) >= 1 && y(j) <= 1/4)
                    u(i, j) = 0;
                else
                    u(i, j) = (1 - lambda)*u(i, j) + lambda*(uip1j + uim1j + b^2*(uijp1
                        ↪  + uijm1))/(2*(1 + b^2));
                end
            end
        end

        deltaU = u - uold;
        residual = sqrt(sum(sum(deltaU.^2))/(I*J));
        res(k) = residual;
        if (k == 1)
            residual0 = residual;
        else
            if (residual/residual0 <= tol)
                mstop = 0;
            else
                uold = u;
            end
        end
    end
    res = res(1:k);
end
```

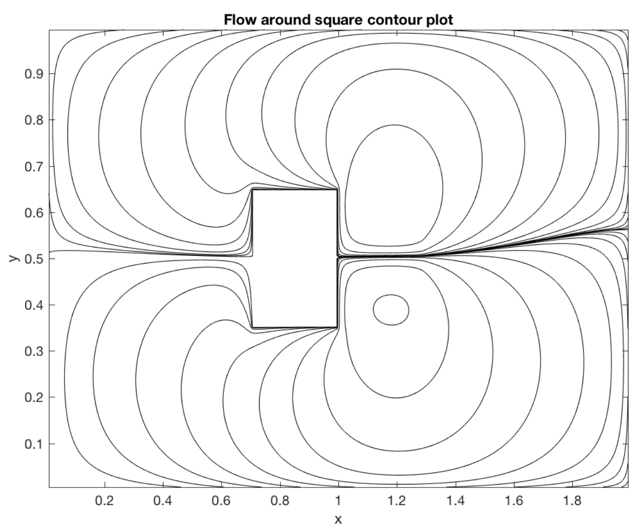


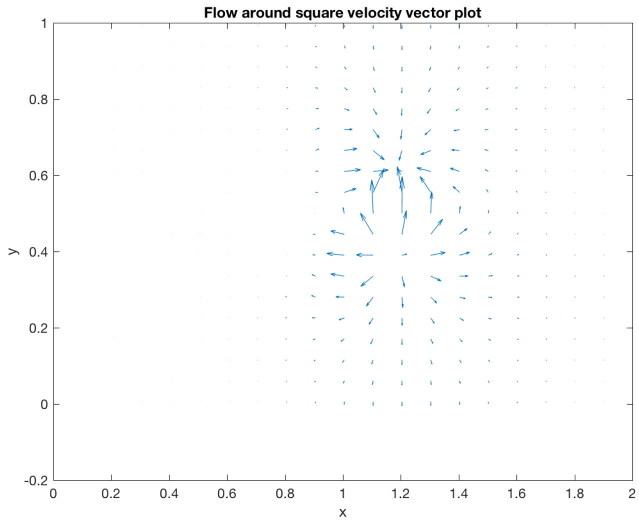


2. The following function runs Gauss-Seidel with SOR on the given problem.

```
function [u, k, res] = sor4(lambda, u, x, y, bt, tol, maxIter)
    [I, J] = size(u);
    uold = u;
    k = 0;
    mstop = 1;
    res = zeros(1,maxIter);
    residual0 = 0;
    while(mstop && k < maxIter)
        k = k + 1;
        for i = 1:I
            for j = 1:J
                if (i == I) % right boundary x = 2
                    uip1j = -y(j) + 0.5;
                    uim1j = u(i-1, j);
                elseif (i == 1) % left boundary x = 0
                    uip1j = u(i+1,j);
                    uim1j = -y(j) + 0.5;
                else
                    uip1j = u(i+1,j);
                    uim1j = u(i-1, j);
                end

                if (j == J) % top boundary, y = 1
                    uijp1 = -0.5;
                    uijm1 = u(i,j-1);
                elseif (j == 1) % bottom boundary y = 0
                    uijp1 = u(i,j+1);
                    uijm1 = 0.5;
                else
                    uijp1 = u(i,j+1);
                    uijm1 = u(i,j-1);
                end

                w = 0;
                if (x(i) >= 1 && x(i) <= 1.3)
                    if (y(j) >= 0.35 && y(j) <= 0.5)
                        w = 50;
                    elseif (y(j) >= 0.5 && y(j) <= 0.65)
```

```
                        w = -50;
                    end
                end

                if ( x(i) >= 0.7 && x(i) <= 1 && y(j) >=0.35 && y(j) <= 0.65)
                    u(i, j) = 0;
                else
                    u(i, j) = (1 - lambda)*u(i, j) + lambda*(uip1j + uim1j + bt^2*(
                        ↪ uijp1 + uijm1))/(2*(1 + bt^2)) + lambda*w;
                end
            end
        end

        deltaU = u - uold;
        residual = sqrt(sum(sum(deltaU.^2))/(I*J));
        res(k) = residual;
        if (k == 1)
            residual0 = residual;
        else
            if (residual/residual0 <= tol)
                mstop = 0;
            else
                uold = u;
            end
        end
    end
    res = res(1:k);
end
```




Flow around square velocity vector plot
1
0.8
0.6
y 0.4
0.2
0
-0.2
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x