University of Southern California
Deparment of Aerospace and Mechanical Engineering
Los Angeles, California 90089-1191

AME 535A
Introduction to Computational Fluids Dynamics I
Instructor: Dr. Domaradzki

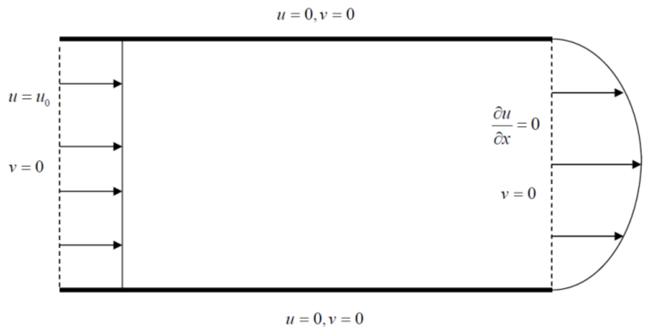
Final project
Fall 2012

Banh, Deedee

Date of submission: December 13th, 2012

Table of Content

Solving for a fully a 2-D parabolic Poiseuille flow in channel for which $\frac{\partial u}{\partial x} = 0$.



1. 2-D Navier-Stoke equations

$$\rho\left(\frac{\partial u}{\partial t} + \vec{u}.\nabla u\right) = -\nabla p + \mu\Delta\vec{u}$$
$$\nabla.\vec{u} = 0$$

Two-dimensional components:

$$\frac{\partial u}{\partial t} + \left(u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y}\right).u = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \frac{\mu}{\rho}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$
$$\frac{\partial v}{\partial t} + \left(u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y}\right).v = -\frac{1}{\rho}\frac{\partial P}{\partial y} + \frac{\mu}{\rho}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)$$
$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

Dimensionless variables:

$$x' = \frac{x}{h}, y' = \frac{y}{h}, t' = \frac{u_0 t}{h}, u' = \frac{u}{u_0}, v' = \frac{v}{u_0}, P' = \frac{P}{\rho u_0{}^2}$$
$$\frac{\partial u}{\partial t} = u_0\frac{\partial u'}{\partial t}\left(\frac{u_0}{h}\right) = \frac{u_0{}^2}{h}\frac{\partial u'}{\partial t'}$$
$$\frac{\partial}{\partial t} = \frac{\partial t'}{\partial t}\frac{\partial}{\partial t'} = \frac{u_0}{h}\frac{\partial}{\partial t'} \to u = u_0 u'$$

Dimensionless 2-D Navier-Stokes:

$$\frac{u_0{}^2}{h}\frac{\partial u'}{\partial' t} + u_0 u'\left(\frac{u_0}{h}\frac{\partial u'}{\partial x'}\right) + u_0 v'\left(\frac{u_0}{h}\frac{\partial u'}{\partial y'}\right) = -\frac{\rho u_0{}^2}{h}\frac{1}{\rho}\frac{\partial P'}{\partial x'} + \frac{\mu}{\rho}\left(\frac{u_0}{h^2}\frac{\partial^2 u'}{\partial x'^2} + \frac{u_0}{h^2}\frac{\partial^2 u'}{\partial y'^2}\right)$$
$$Re = \frac{u_0 h}{\upsilon}, \upsilon = \frac{\mu}{\rho}$$
$$\to \frac{\partial u'}{\partial' t} + u'\frac{\partial u'}{\partial x'} + v'\frac{\partial u'}{\partial y'} = -\frac{\partial P'}{\partial x'} + \frac{1}{Re}\left(\frac{\partial^2 u'}{\partial x'^2} + \frac{\partial^2 u'}{\partial y'^2}\right)$$

Similarly:

$$\rightarrow \frac{\partial v'}{\partial' t} + u' \frac{\partial v'}{\partial x'} + v' \frac{\partial v'}{\partial y'} = -\frac{\partial P'}{\partial y} + \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v'}{\partial x'^2} + \frac{\partial^2 v'}{\partial y'^2}\right)$$

$$\rightarrow \frac{\partial u'}{\partial x'} + \frac{\partial v'}{\partial y'} = 0$$

$$\boxed{(1)\ \ \frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - \frac{\partial P}{\partial x} + \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)}$$

$$\boxed{(2)\ \ \frac{\partial v}{\partial t} = -u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} - \frac{\partial P}{\partial y} + \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)}$$

$$\boxed{(3)\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0}$$

2. Exact solution for a Poiseuille flow

$$\frac{\partial u}{\partial x} = 0, \qquad u = u(y)$$

$$(3)\,\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \rightarrow \frac{\partial v}{\partial y} = 0 \rightarrow v(x) = \mathrm{const}$$

But $v = 0$ at boundary, so $v = 0$ everywhere.

$$(1)\ 0 = -\frac{\partial P}{\partial x} + \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial y^2}\right)$$

$$(2)\ 0 = -\frac{\partial P}{\partial y} \rightarrow P = P(x) \rightarrow \frac{dP}{dx} = -|C|$$

where $|C|$ is a constant representing the mean pressure gradient.

From equation (1), $\frac{\partial P}{\partial x}$ is a function of x, and $\frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial y^2}\right)$ is a function of x and y. For these two to be equal,

$$\frac{dP}{dx} = \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial y^2}\right) = \mathrm{constant} \rightarrow \frac{dP}{dx} = \mathrm{constant}$$

Solving for the u(y):

$$\frac{\partial^2 u}{\partial y^2} = \mathrm{Re}\frac{dP}{dx}$$

$$\frac{\partial u}{\partial y} = \mathrm{Re}\frac{dP}{dx}y + c_1$$

$$u(y) = \mathrm{Re}\frac{dP}{dx}\left(\frac{y^2}{2}\right) + c_1 y + c_2$$

Apply boundary conditions:

$$u(0,y) = u_0$$
$$\frac{\partial u}{\partial x}(x_{max},y) = 0$$
$$u(x,h) = 0$$
$$u(x,-h) = 0$$
$$\rightarrow c_1 = 0, c_2 = -\frac{Re}{2}\frac{dP}{dx} = \frac{Re}{2}|C|$$

$$\boxed{u(y) = \frac{Re}{2}\ |C|(1-y^2)}$$

3. Conservation of mass

$$q = \frac{Q}{W} = \int u(y)dy = \frac{Re}{2}\ |C|\int_{-1}^{1}(1-y^2)dy = \frac{Re}{2}\ |C|\left(y-\frac{y^3}{3}\right)_{-1}^{1} = \frac{Re}{2}\ |C|\left(\frac{4}{3}\right) = \frac{2}{3}Re|C|$$
$$q_{in} = q_{out}$$
$$q_{in} = u_{in} * 2h = 2 = \frac{2}{3}Re|C|$$
$$\boxed{|C| = \frac{3}{Re}}$$

4. Numerical code

No-slip boundary conditions:

$$u(x,1) = 0, \qquad u(x,-1) = 0$$
$$u(0,y) = 1, \qquad \frac{\partial u}{\partial x}(x_{max},y) = 0$$
$$v(0,y) = 0, \qquad v(x_{max},y) = 0$$

General discretization using second-order finite difference method for spatial terms:

$$\frac{\partial u}{\partial x} = \frac{u_{j+1,k} - u_{j-1,k}}{2\Delta x}, \quad \frac{\partial^2 u}{\partial x^2} = \frac{u_{j+1,k} - 2u_{j,k} + u_{j-1,k}}{(\Delta x)^2}$$
$$\frac{\partial u}{\partial y} = \frac{u_{j,k+1} - u_{j,k-1}}{2\Delta y}, \quad \frac{\partial^2 u}{\partial y^2} = \frac{u_{j,k+1} - 2u_{j,k} + u_{j,k-1}}{(\Delta y)^2}$$
$$\frac{\partial v}{\partial x} = \frac{v_{j+1,k} - v_{j+1,k}}{2\Delta x}, \quad \frac{\partial^2 v}{\partial x^2} = \frac{v_{j+1,k} - 2v_{j,k} + v_{j-1,k}}{(\Delta x)^2}$$
$$\frac{\partial v}{\partial y} = \frac{v_{j,k+1} - v_{j,k-1}}{2\Delta y}, \quad \frac{\partial^2 v}{\partial y^2} = \frac{v_{j,k+1} - 2v_{j,k} + v_{j,k-1}}{(\Delta y)^2}$$

Method of solution

1. Nonlinear step

$$\frac{\partial u}{\partial t} = F_u,$$
$$\frac{\partial v}{\partial t} = F_v,$$
$$F_u = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} + \frac{3}{Re}$$
$$F_v = -u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y}$$

First time-step is generated using Euler method:

$$\frac{u^* - u^n}{\Delta t} = F_u^n = \frac{3}{2}\left[-u_{j,k}^n\left(\frac{u_{j+1,k}^n - u_{j-1,k}^n}{2\Delta x}\right) - v_{j,k}^n\left(\frac{u_{j,k+1}^n - u_{j,k-1}^n}{2\Delta y}\right) + \frac{3}{Re}\right]$$
$$\frac{v^* - v^n}{\Delta t} = F_v^n = \frac{3}{2}\left[-u_{j,k}^n\left(\frac{v_{j+1,k}^n - v_{j-1,k}^n}{2\Delta x}\right) - v_{j,k}^n\left(\frac{v_{j,k+1}^n - v_{j,k-1}^n}{2\Delta y}\right)\right]$$

Time advancement using Adams-Bashforth method for second time-step and so-on:

$$\boxed{\frac{u^* - u^n}{\Delta t} = \frac{3}{2}F_u^n - \frac{1}{2}F_u^{n-1}}$$
$$\boxed{\frac{v^* - v^n}{\Delta t} = \frac{3}{2}F_v^n - \frac{1}{2}F_v^{n-1}}$$

2. Pressure step

$$\frac{\partial u}{\partial t} = -\frac{\partial p}{\partial x}$$
$$\frac{\partial v}{\partial t} = -\frac{\partial p}{\partial y}$$
$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

Time discretization

$$\frac{u^{**} - u^*}{\Delta t} = -\nabla p$$

Taking divergence and assume incompressible:

$$\frac{\partial u^{**}}{\partial x} + \frac{\partial v^{**}}{\partial y} = 0$$

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = \frac{1}{\Delta t}\left(\frac{\partial u^*}{\partial x} + \frac{\partial v^*}{\partial y}\right)$$

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

$$\boxed{\frac{p_{j+1,k} - 2p_{j,k} + p_{j-1,k}}{(\Delta x)^2} + \frac{p_{j,k+1} - 2p_{j,k+}p_{j,k-1}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}}$$

Use one-sided formula for the divergences inside the domain:

$$\boxed{(\mathrm{RHS})^*_{j,k} = \frac{1}{\Delta t}D_{j,k} = \frac{1}{\Delta t}\left(\frac{u^*{}_{j,k} - u^*{}_{j,k-1}}{\Delta x} + \frac{v^*{}_{j,k} - v^*{}_{j,k-1}}{\Delta y}\right)}$$

For internal pressure points:

$$\frac{p_{j+1,k} + p_{j-1,k}}{(\Delta x)^2} - 2p_{j,k}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + \frac{p_{j,k+1} + p_{j,k-1}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}$$

On boundary:

Neumann boundary on all boundaries for p:

$$\frac{\partial p}{\partial x} = 0 \text{ at } x = 0, x = x_{max}$$

$$\frac{\partial p}{\partial y} = 0, \text{at } y = +1, y = -1$$

$$\frac{\partial p}{\partial x} = 0 \text{ at } x = 0, \qquad 2\frac{p_{j+1,k}}{(\Delta x)^2} - 2p_{j,k}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + \frac{p_{j,k+1} + p_{j,k-1}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}$$

$$\frac{\partial p}{\partial x} = 0 \text{ at } x = x_{max} \qquad 2\frac{p_{j-1,k}}{(\Delta x)^2} - 2p_{j,k}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + \frac{p_{j,k+1} + p_{j,k-1}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}$$

$$\frac{\partial p}{\partial y} = 0 \text{ at } y = +1, \qquad \frac{p_{j+1,k} + p_{j-1,k}}{(\Delta x)^2} - 2p_{j,k}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + 2\frac{p_{j,k+1}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}$$

$$\frac{\partial p}{\partial y} = 0 \text{ at } y = -1, \qquad \frac{p_{j+1,k} + p_{j-1,k}}{(\Delta x)^2} - 2p_{j,k}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + 2\frac{p_{j,k-1}}{(\Delta y)^2} = (RHS)^*_{j,k}$$

Values of pressure p at corner points are approximated in terms of the values of the neighboring points:

$$At\ p_{1,1}: 0.5\frac{p_{2,1}}{(\Delta x)^2} - 2p_{1,1}\left(\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right) + 0.5\frac{p_{1,2}}{(\Delta y)^2} = (\mathrm{RHS})^*_{j,k}$$

u^{**} and v^{**} can be found using the velocity components from the convection step:

$$u^{**} = u^{*} - \Delta t \frac{\partial p}{\partial x}$$

$$v^{**} = v^{*} - \Delta t \frac{\partial p}{\partial y}$$

$$\boxed{u_{j,k}^{**} = u_{j,k}^{*} - \Delta t \left(\frac{p_{j+1,k} - p_{j-1,k}}{2\Delta x}\right)}$$

$$\boxed{v_{j,k}^{**} = \mathrm{v}_{\mathrm{j,k}}^{*} - \Delta \mathrm{t} \left(\frac{\mathrm{p}_{\mathrm{j,k+1}} - \mathrm{p}_{\mathrm{j,k-1}}}{2\Delta \mathrm{y}}\right)}$$

3. Viscous steps

First viscous step

$$\frac{\partial \mathrm{u}}{\partial \mathrm{t}} = \frac{1}{\mathrm{Re}} \frac{\partial^2 \mathrm{u}}{\partial \mathrm{x}^2}$$

$$\frac{\partial \mathrm{v}}{\partial \mathrm{t}} = \frac{1}{\mathrm{Re}} \frac{\partial^2 \mathrm{v}}{\partial \mathrm{x}^2}$$

Using Crank-Nicolson Scheme

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

$$\mathrm{u}^{***} - \frac{\Delta \mathrm{t}}{\mathrm{Re}} \frac{1}{2} \frac{\partial^2 \mathrm{u}^{***}}{\partial \mathrm{x}^2} = \mathrm{u}^{**} + \frac{\Delta \mathrm{t}}{\mathrm{Re}} \frac{1}{2} \frac{\partial^2 \mathrm{u}^{**}}{\partial \mathrm{x}^2}$$

$$\boxed{\begin{cases} \mathrm{u}_{\mathrm{j,k}}^{***} - \dfrac{\Delta \mathrm{t}}{\mathrm{Re}} \dfrac{1}{2} \left(\dfrac{\mathrm{u}_{\mathrm{j+1,k}}^{***} - 2\mathrm{u}_{\mathrm{j,k}}^{***} + \mathrm{u}_{\mathrm{j-1,k}}^{***}}{(\Delta \mathrm{x})^2}\right) = (\mathrm{RHS}_{\mathrm{u}})_{\mathrm{j,k}}^{**} \\ (\mathrm{RHS}_{\mathrm{u}})_{\mathrm{j,k}}^{**} = \mathrm{u}_{\mathrm{j,k}}^{**} + \dfrac{\Delta \mathrm{t}}{\mathrm{Re}} \dfrac{1}{2} \left(\dfrac{\mathrm{u}_{\mathrm{j+1,k}}^{**} - 2\mathrm{u}_{\mathrm{j,k}}^{**} + \mathrm{u}_{\mathrm{j-1,k}}^{**}}{(\Delta \mathrm{x})^2}\right) \end{cases}}$$

Same for v^{***}

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

$$\boxed{\begin{cases} \mathrm{v}_{\mathrm{j,k}}^{***} - \dfrac{\Delta \mathrm{t}}{\mathrm{Re}} \dfrac{1}{2} \left(\dfrac{\mathrm{v}_{\mathrm{j+1,k}}^{***} - 2\mathrm{v}_{\mathrm{j,k}}^{***} + \mathrm{v}_{\mathrm{j-1,k}}^{***}}{(\Delta \mathrm{x})^2}\right) = (\mathrm{RHS}_{\mathrm{v}})_{\mathrm{j,k}}^{**} \\ (\mathrm{RHS}_{\mathrm{v}})_{\mathrm{j,k}}^{**} = \mathrm{v}_{\mathrm{j,k}}^{**} + \dfrac{\Delta \mathrm{t}}{\mathrm{Re}} \dfrac{1}{2} \left(\dfrac{\mathrm{v}_{\mathrm{j+1,k}}^{**} - 2\mathrm{v}_{\mathrm{j,k}}^{**} + \mathrm{v}_{\mathrm{j-1,k}}^{**}}{(\Delta \mathrm{x})^2}\right) \end{cases}}$$

Second viscous:

$$\frac{\partial u}{\partial t} = \frac{1}{Re}\frac{\partial^2 u}{\partial y^2}$$

$$\frac{\partial v}{\partial t} = \frac{1}{Re}\frac{\partial^2 v}{\partial y^2}$$

using the Crank-Nicolson method.

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

For u^{n+1},

$$\boxed{\begin{cases} u_{j,k}^{n+1} - \dfrac{\Delta t}{Re}\dfrac{1}{2}\left(\dfrac{u_{j,k+1}^{n+1} - 2u_{j,k}^{n+1} + u_{j,k-1}^{n+1}}{(\Delta y)^2}\right) = (RHS_u)_{j,k}^{***} \\ (RHS_u)_{j,k}^{***} = u_{j,k}^{***} + \dfrac{\Delta t}{Re}\dfrac{1}{2}\left(\dfrac{u_{j,k+1}^{***} - 2u_{j,k}^{***} + u_{j,k-1}^{***}}{(\Delta y)^2}\right) \end{cases}}$$

For v^{n+1},

$$\boxed{\begin{cases} v_{j,k}^{n+1} - \dfrac{\Delta t}{Re}\dfrac{1}{2}\left(\dfrac{v_{j,k+1} - 2v_{j,k} + v_{j,k-1}}{(\Delta y)^2}\right) = (RHS_v)_{j,k}^{***} \\ (RHS_v)_{j,k}^{***} = v_{j,k}^{***} + \dfrac{\Delta t}{Re}\dfrac{1}{2}\left(\dfrac{v_{j,k+1}^{***} - 2v_{j,k}^{***} + v_{j,k-1}^{***}}{(\Delta y)^2}\right) \end{cases}}$$

4. Code Organization:

main.m	where it calls on user defined variables and run the program
u_f.m	Calculates the non-linear step for u-component
v_f.m	Calculates the non-linear step for v-component
euler.m	Calculates the first iteration using Euler method
CONVEC.m	Solve non-linear step using Adams-Bashforth method
PRES.m	Calculates the pressure step and update u and v components
matrix.m	Initialize pressure gradient matrix with Neumann boundary condition
VISC.m	Solve the two viscous steps using Crank-Nicholson method
fact.m	Same as given in Fletcher.
solve.m	Same as given in Fletcher.
banfac.m	Same as given in Fletcher.
bansol.m	Same as given in Fletcher.
exactvelocity.m	Exact solution for u-component and the flow rate
artificialpressure.m	Generate an artificial solution for pressure gradient to test matrix.m

1. General parameters:

Parameter	Description
nx	Number of grids in x-direction
ny	Number of grids in y-direction
re	Reynolds number
h	Half the height of channel
xmax	Length of channel
itmax	Maximum number of iteration
uin	Flow input
dx	Δx
dy	Δy
dt	Δt
c	CFL condition

2. Running the code:

Go to main.m and change the variables as desired.
Run MATLAB within the main.m directory

To test subroutine matrix.m using the artificial exact solution, uncomment the following code in main.m:

```
>> nx=[11 21 31];
>> ny=nx;
>> for i=1:3
>>     rms=PressureExact(nx(i),ny(i));
>>     fprintf('Mesh: %dx%d \nrms: %.4e\n',nx(i),ny(i),rms);
>> end
```

Results:

```
Mesh: 11x11
rms: 9.4150e-003
Mesh: 21x21
rms: 2.3240e-003
Mesh: 31x31
rms: 1.0290e-003
```

3. Code:

main.m

```
%% User Inputs

nx=31;
ny=21;
re=10;
xmax=3;
h=1;
itmax=50;
uin=1;
dt=0.01;

%%
dx=xmax/(nx-1);
dy=h*2/(ny-1);
c=dt*uin/dx;
if (c>1)
    display('Error! CFL Condition > 1');
    break;
end
u=zeros(ny,nx);
v=zeros(ny,nx);
p=zeros(ny,nx);

x=0:dx:xmax;
y=-1:dy:1;

%Exact Solution
[uex qex]=exactvelocity(nx,ny);

%Initiate velocity u-component
u=uex;

%%
firstiter=1;
iter=0;

while (iter<=itmax)
    % Non-linear step
     if (firstiter==1)
        [u,v,fu,fv] = euler(u,v,nx,ny,dx,dy,re,dt,uin);
         firstiter=firstiter+1;
     else
         [u,v,fu,fv] = CONVEC(u,v,nx,ny,dx,dy,re,dt,fu,fv,uin);
     end
    % Pressure
```

```
    [p u v]=PRES(u,v,p,nx,ny,dx,dy,dt,uin,iter);
    % Viscous step
     u=VISC(u,nx,ny,dx,dy,re,dt,uin,1);
     v=VISC(v,nx,ny,dx,dy,re,dt,uin,2);
    %--------------------------------------
    iter=iter+1;
end

%%
%RMS
%For u-component
sum=0;
for k=2:ny-1
    error=uex(k)-u(k,nx);
    sum=sum+(error*error);
end
nn=(ny+1)/2;
an=ny-2;
u_rms=sqrt(sum/an);

%For the flow rate
i=0;
ii=0;
q=zeros(nx);
for k=1:nx
    ii=ii+1;
    for j=1:ny
        sum=i+u(j,ii);
        if (j==1)
            i=0;
        else
            i=sum;
        end
    end
    q(ii)=i*dy;
end
sum=0;
for j=1:nx
    error=qex(j)-q(j);
    sum=sum+(error*error);
end
an=nx;
q_rms=sqrt(sum/an);

fprintf('nx=%d; ny=%d; re=%d;\n',nx,ny,re)
fprintf('xmax=%d; h=%d; itmax=%d; uin=%d; dt=%.3e\n\n',xmax,h,itmax,uin,dt)
fprintf('RMS for u-component at the outflow is: %.4e\n',u_rms);
fprintf('RMS for the flow rate: %.4e\n\n',q_rms);
%%
% Plots

% Vertical Components
figure(1)
plot(u(:,2),y,'-x',u(:,(nx+1)/2),y,'-x',u(:,nx-1),y,'-x',u(:,nx),y,'-
x',uex(:,nx),y,'LineWidth',1.4)
legend('j=2','j=(nx+1)/2','j=nx-1','j=nx','exact solution');
xlabel('u'),ylabel('y');title('Vertical Profiles for u');
```

```
figure(2)
plot(v(:,2),y,'--',v(:,(nx+1)/2),y,v(:,nx-1),y,v(:,nx),y,'LineWidth',1.4)
legend('j=2','j=(nx+1)/2','j=nx-1','j=nx');
xlabel('v'),ylabel('y');title('Vertical Profiles for v');

% Along the centerline
figure(3)
plot(x,u(nn,:),xmax,uex(nn,nx),'o','LineWidth',2)
xlabel('x'),ylabel('u');title('Centerline for u');
figure(4)
plot(x,v((ny+1)/2,:),'LineWidth',2)
xlabel('x'),ylabel('v');title('Centerline for v');
figure(5)
plot(x,q,x,qex,'--','LineWidth',2)
xlabel('x'),ylabel('q');title('Centerline for q');

%% To test subroubtine matrix.m, uncomment the code below

% nx=[11 21 31];
% ny=nx;
% for i=1:3
%     rms=artificialpressure(nx(i),ny(i));
%     fprintf('Mesh: %dx%d \nrms: %.4e\n',nx(i),ny(i),rms);
% end

%%
```

euler.m

```
function [u,v,fu,fv] = euler(u,v,nx,ny,dx,dy,re,dt,uin)
%First-step: use Euler method to initiate Fu and Fv
fu = u_f(u,v,nx,ny,dx,dy,re);
fv = v_f(u,v,nx,ny,dx,dy);
u = u + dt*fu; %update u
v = v + dt*fv; %update v
u(2:ny-1,1)=uin;
end
```

CONVEC.m

```
function [u,v,fu,fv] = CONVEC(u,v,nx,ny,dx,dy,re,dt,fu1,fv1,uin)
% Second-step and so on use
% Adam Bashford to update
% velocity u and v
fu = u_f(u,v,nx,ny,dx,dy,re);
fv = v_f(u,v,nx,ny,dx,dy);
u = u + dt*(3/2*fu-1/2*fu1); %Ustar for nonlinear step
v = v + dt*(3/2*fv-1/2*fv1); %Vstar for nonlinear step
u(2:ny-1,1)=uin;
end
```

u_f.m

```
function [u_f] = u_f(u,v,nx,ny,dx,dy,re)
% Calculate the Nonlinear step
% for Dudt=Fu
u_f=u;
cx=1/2/dx;
cy=1/2/dy;
```

```
for j=2:ny-1 %rows
    for k=2:nx %cols
        if (k==nx)
            u(j,k+1)=u(j,k-1);
        end
        a=u(j,k)*(u(j,k+1)-u(j,k-1))*cx;
        b=v(j,k)*(u(j+1,k)-u(j-1,k))*cy;
        c=3/re;
        u_f(j,k)=-a-b+c;
    end
end
```

v_f.m

```
function [v_f] = v_f(u,v,nx,ny,dx,dy)
% Calculate the Nonlinear step
% for Dvdt=Fv
v_f=v;
cx=1/2/dx;
cy=1/2/dy;
for j=2:ny-1 %rows
    for k=2:nx-1 %cols
        a=u(j,k)*(v(j,k+1)-v(j,k-1))*cx;
        b=v(j,k)*(v(j+1,k)-v(j-1,k))*cy;
        v_f(j,k)=-a-b;
    end
end
```

PRES.m

```
function [p u v]=PRES(u,v,p,nx,ny,dx,dy,dt,uin,iter)

n=nx*ny;
cx=1/(2*dx);
cy=1/(2*dy);
D=zeros(ny,nx);
for j=2:ny-1 %rows
    for k=2:nx-1 %cols
        D(j,k)=1/dt*((u(j,k)-u(j,k-1))/dx+(v(j,k)-v(j-1,k))/dy);
    end
end
%%
rhs(1:n)=0;
A=matrix(ny,nx,dx,dy);
for j=1:ny
    for k=1:nx
        l=j+(k-1)*ny;
        rhs(l)=D(j,k);
        rhs(l)=rhs(l)-A(l,n);
    end
end
rhs(l)=1;
A(:,l)=0;
A(l,:)=0;
A(l,l)=rhs(l);
jpvt(1:n)=0;
ww=1;
if(mod(iter,ww)==0 || iter==1)
```

```
    [AJ,jpvt]=fact(n,A,jpvt);
end
[rd]=solve(n,AJ,jpvt,rhs);
%%
zz=0;
for i=1:nx
    for j=1:ny
        zz=zz+1;
        p(j,i)=rd(zz);
    end
end
%%
for j=2:ny-1
    for k=2:nx
        if (k==nx)
            p(:,nx+1)=p(:,nx-1);
        end
        u(j,k)=u(j,k)-dt*cx*(p(j,k+1)-p(j,k-1));
    end;
end;
for j=2:ny-1
    for k=2:nx-1
        v(j,k)=v(j,k)-dt*cy*(p(j+1,k)-p(j-1,k));
    end
end
u(2:ny-1,1)=uin;

end
```

matrix.m

```
function A=matrix(ny,nx,dx,dy)

ccx=1/(dx*dx);
ccy=1/(dy*dy);
n=nx*ny;

A(1:n,1:n)=0;
ib=ny;
it=ny+1;
for j=1:ny
    for k=1:nx
        l=j+(k-1)*ny;
        A(l,l)=-2*(ccx+ccy);

        xr=l+ny;
        if(xr<=n)
            A(l,xr)=ccx;
            if(l<=ny)
                A(l,xr)=2*ccx;
            end
        end

        xl=l-ny;
        if(xl>0)
            A(l,xl)=ccx;
            if(l>=n-ny+1)
                A(l,xl)=2*ccx;
```

```
            end
        end

        yr=l+1;
        if(yr<=n)
            A(l,yr)=ccy;
            if(yr==it && it<n)
                A(yr,it+1)=2*ccy;
                A(yr,it-1)=0;
                it=it+ny;
            end
        end

        yl=l-1;
        if(yl>0)
            A(l,yl)=ccy;
            if(l==ib && ib<n)
                A(l,ib+1)=0;
                A(l,ib-1)=2*ccy;
                ib=ib+ny;
            end
        end
    end; %k
end; %j
A(1,2)=2*ccy;
A(l,l-1)=2*ccy;
```

exactvelocity.m

```
function [uex qex]=exactvelocity(nx,ny)
dy=2/(ny-1);
y=-1:dy:1;
for j=1:ny
    for k=1:nx
        uex(j,k)=3/2*(1-y(j)^2);
    end
end
sum=0;
for j=1:ny
    sum=sum+uex(j,1);
    for k=1:nx
        qex(k)=sum*dy;
    end
end
```

artificialpressure.m

```
function rms=artificialpressure(nx,ny)

dx=1/(nx-1);
dy=1/(ny-1);

n=nx*ny;

count=1;
N=zeros(ny,nx);
```

```
for i=1:nx
    for j=1:ny
        N(j,i)=count;
        count=count+1;
    end
end

x=0:dx:1;
y=0:dy:1;
A=matrix(ny,nx,dx,dy);
for j=1:ny
    for k=1:nx
       pex(j,k)=cos(pi*x(k))*cos(pi*y(j));
       Dex(j,k)=-2*pi^2*cos(pi*x(k))*cos(pi*y(j));
    end;
end;
pex=reshape(pex,[],1);
for j=1:ny
    for k=1:nx
        l=j+(k-1)*ny;
        rhs(l)=Dex(j,k);
        rhs(l)=rhs(l)-A(l,n);
    end
end
rhs(l)=1;
A(:,l)=0;
A(l,:)=0;
A(l,l)=rhs(l);

jpvt(1:n)=0;
[AJ,jpvt]=fact(n,A,jpvt);
[rd]=solve(n,AJ,jpvt,rhs);
zz=0;
for i=1:nx
    for j=1:ny
        zz=zz+1;
        p(j,i)=rd(zz);
        moddex(j,i)=rhs(zz);
    end
end
sum=0;
for j=1:n
    sum = sum+ (pex(j)-p(j))^2;
    err(j)=pex(j)-p(j);
end
rms=sqrt(sum/n);
% x=1:n;
% plot(x,err)
```

5. Results

1. Three different Reynolds numbers

For Re=1:

```
nx=11; ny=11; re=1;
xmax=50; h=1; itmax=50; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 1.2556e-002
RMS for the flow rate: 1.2147e-001
```

For Re=10:

```
nx=11; ny=11; re=10;
xmax=50; h=1; itmax=50; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 4.8929e-004
RMS for the flow rate: 6.8779e-002
```

For Re=100:

```
nx=11; ny=11; re=100;
xmax=50; h=1; itmax=50; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 1.8595e-005
RMS for the flow rate: 6.4006e-002
```

2. Uniform grid comparison

For Re=1:

```
nx=31; ny=21; re=1;
xmax=3; h=1; itmax=50; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 5.1890e-002
RMS for the flow rate: 1.1907e-001
```

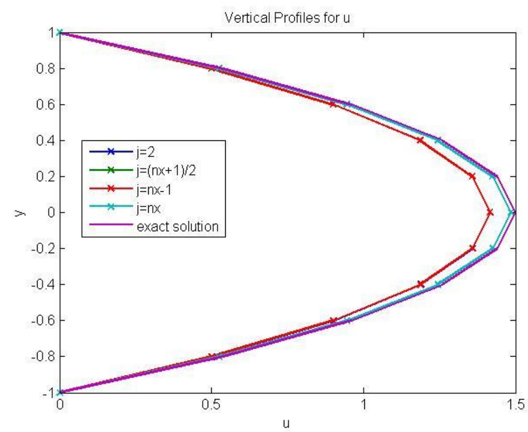
For Re=10:

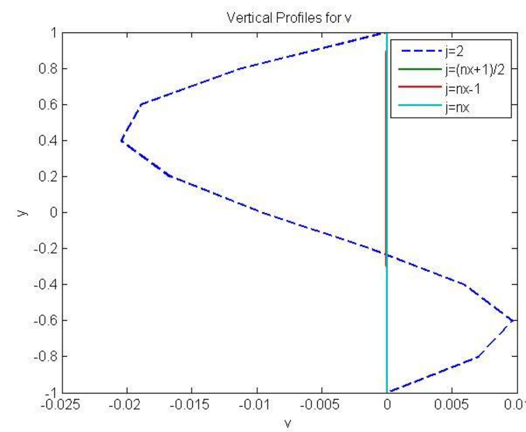
```
nx=31; ny=21; re=10;
xmax=3; h=1; itmax=50; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 1.6094e-002
RMS for the flow rate: 4.5697e-002
```

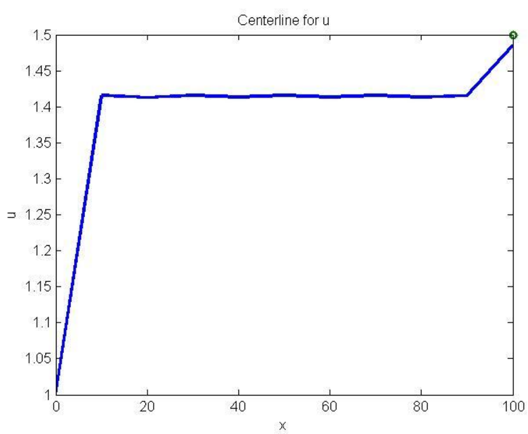
3. Printout and plots

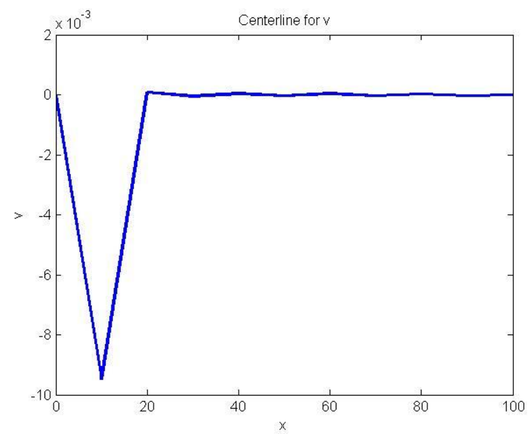
I. Re=1

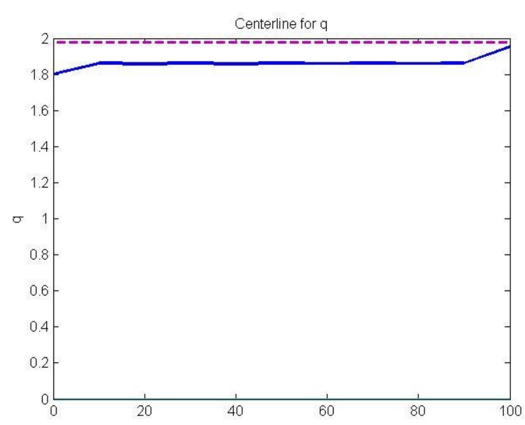
```
nx=11; ny=11; re=1;
xmax=100; h=1; itmax=1000; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 1.5414e-002
RMS for the flow rate: 1.2125e-001
```





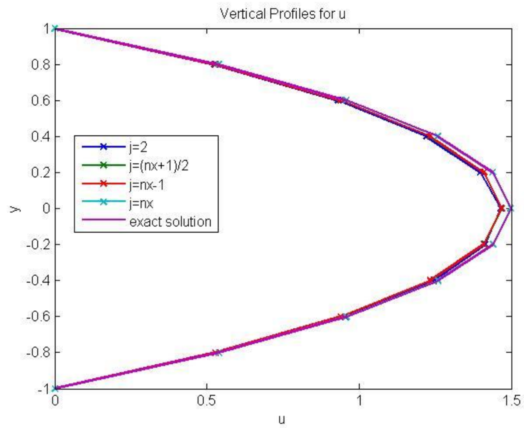


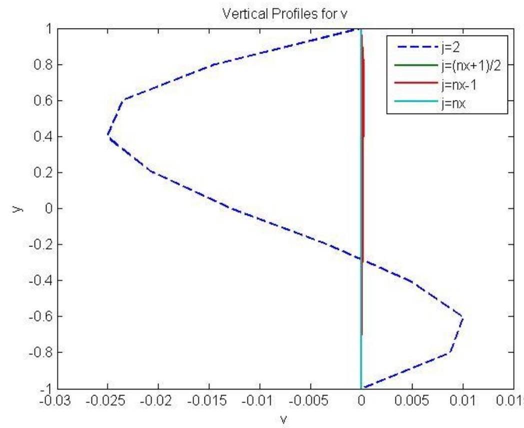


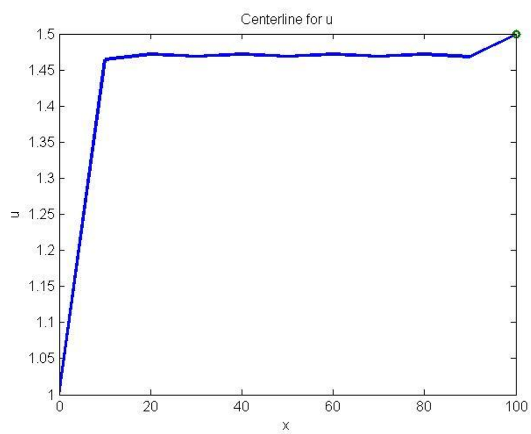


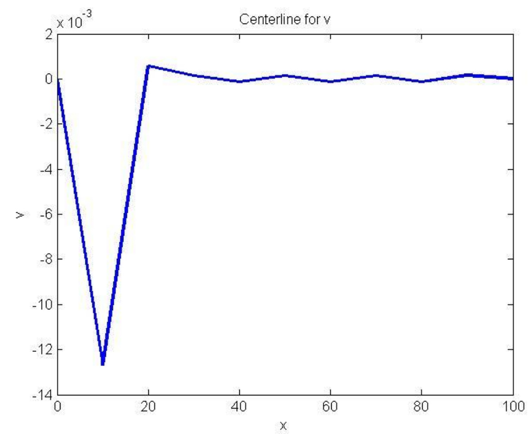
II. Re=10

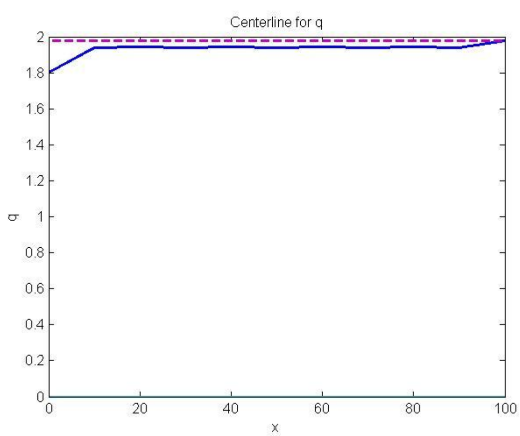
```
nx=11; ny=11; re=10;
xmax=100; h=1; itmax=1000; uin=1; dt=1.000e-002

RMS for u-component at the outflow is: 1.5504e-003
RMS for the flow rate: 6.5397e-002
```





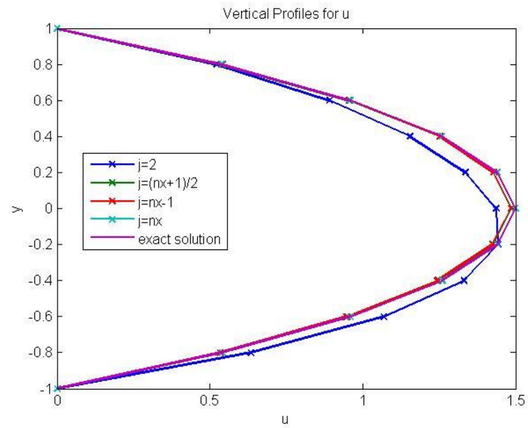


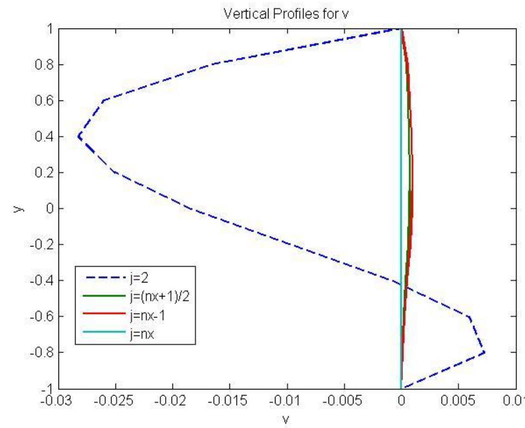


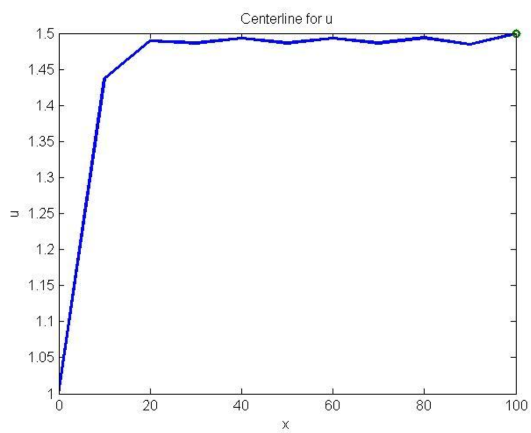


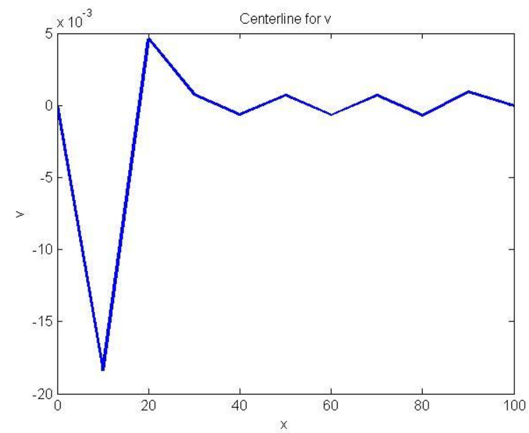
III. Re=100

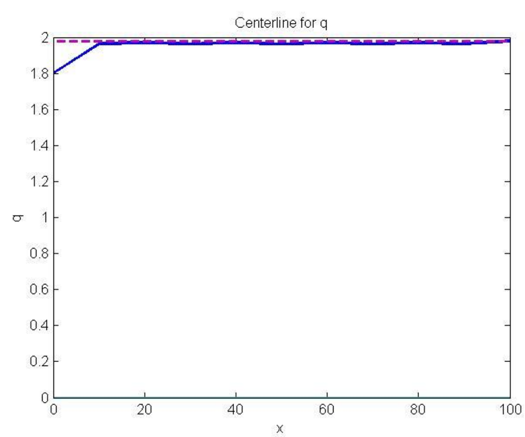
```
nx=11; ny=11; re=100;
xmax=100; h=1; itmax=1000; uin=1; dt=1.000e-002
RMS for u-component at the outflow is: 7.0226e-005
RMS for the flow rate: 5.5796e-002
```











6. Comparison

Keep the following variables constant:

`xmax=50; h=1; itmax=100; uin=1; dt=1.000e-002`

Reynolds Number	Rms error 7x7	Rms error 11x11	Rms error 21x21
1	1.4762e-002	1.5472e-002	1.7599e-002
10	5.6380e-004	7.0701e-004	8.8539e-004
100	2.2861e-005	3.0566e-005	5.8321e-005

Keep the following variables constant:

`xmax=10; h=1; itmax=100; uin=1; dt=1.000e-002`

Reynolds Number	Rms error Xmax=10	Rms error Xmax=30	Rms error Xmax=60
1	3.9955e-002	1.8192e-002	1.4990e-002
10	7.3374e-003	1.0645e-003	6.5490e-004
100	7.8237e-004	6.9501e-005	2.4631e-005

Looking at the results, it is easy to see that the finer grids give the best results which are comparable to the analytical solution; but it takes longer for the program to compute. The flow has better accuracy if the channel is efficiently long, allowing the flow to fully develop. The Reynolds number also plays an important role in achieving the results. High Reynolds number gives better accuracy. This is to be expected since high Reynolds number diminishes the viscous affect from the diffusion terms and the flow reaches its maximum faster.