2D BACKWARD FACING LAMINAR STEP FLOW SIMULATION BY FINITE DIFFERENCE METHOD

Cihad ÇELİK[1], Devrim Bülent DANIŞMAN[2], Barış BARLAS[3],
[1]İstanbul Technical University, Faculty of Naval Architecture and Marine Engineering | celikciha@itu.edu.tr
[2]İstanbul Technical University, Faculty of Naval Architecture and Marine Engineering | bulent.danisman@itu.edu.tr
[3]İstanbul Technical University, Faculty of Naval Architecture and Marine Engineering | barlas@itu.edu.tr

ABSTRACT

In the current study, a backward-facing step flow (BFS) by finite difference discretization is solved in 2D Cartesian coordinate system. The governing equations of the problem are the incompressible Navier-Stokes equations and the continuity equation. The no-slip boundary conditions are applied using ghost cells within the solid domain. The Dirichlet and Neumann boundary conditions are implemented at the inlet and outlet of the channel, respectively. MAC (Marker and Cell) method is utilized as a numerical scheme to solve the flow. The problem is considered as a Stokes flow ($Re = 0$). Results show good agreement with the data that is calculated by the commercial software. The code written in Matlab is provided in the Appendix.

Key Words: Backward-Facing Step Flow, Finite Difference Method, Stokes Flow, MAC Method.

1. Introduction

Stokes flow which was named after George Gabriel Stokes, is a type of fluid flow where advective inertial forces are small compared with viscous forces. The Reynolds number is very low ($Re \ll 1$). This is a typical situation in flows where the fluid velocities are very slow, the viscosities are very large. In practice this type of flow occurs in the swimming of microorganisms, sperm motility, the flow of lava, painting brush problem, lubrication between plates, microelectromechanical and nanoelectromechanical systems particularly those with moving parts, and in the flow of viscous polymers. Backward-Facing Step is widely known for its application in internal flow studies. The flow separation is caused due to the sudden changes in the geometry. This creates a zone of recirculation and a point of flow reattachment. Strong adverse pressure gradients arise through this process.

Experimental [1-2] and numerical [3-7] studies of backward-facing step flow have been carried out with different flow conditions, laminar [3], transitional and turbulent in detail.

A technique [7] is first presented by Harlow & Welch namely, the marker and cell method, implemented to numerically solve the time-dependent flow of an incompressible fluid by finite difference discretization. The pressure and the velocity components as the primary variables are defined at cell centers and cell boundaries, respectively, shown in Figure 1 (a). Further investigations have been performed to understand the effect of the expansion ratios, the ratio of the channel height (H) to the inlet channel height (h), at low and moderate Reynolds numbers. It is highlighted that the total pressure loss rises with the increasing step height ($H - h$) and decrease with increasing Re number ($0 < Re < 200$) [3]. Direct numerical simulation of BFS flow has been performed at $Re = 395$ and expansion ratio 2 in order to understand the strong adverse pressure gradients attached to the step's downstream which leads to flow instabilities and defines the pressure increasing [5]. The BFS flow problem has also been investigated numerically and experimentally in the transitional flow regime, from laminar to the turbulent regime, in a water channel [2]. In the experimental part, electro-diffusion technique is implemented to measure the wall shear rate. Numerical simulations performed in FLUENT software using finite volume discretization in 2D. Numerical simulations show good agreement with the experimental ones, which depicted that the backward-facing flow structure becomes more complex while the expansion ratio increases.

In this study, a backward-facing step flow by finite difference discretization is solved in 2D Cartesian coordinate system at $Re = 0$ and the code written in Matlab is provided to the readers can be found in the Appendix. The authors believe that the readers would benefit from the code and it is ensured that it could be further developed.

2. Mathematical and Numerical Formulation

The incompressible Navier-Stokes equations that govern the incompressible viscous fluid flow in the Cartesian coordinate system can be written in dimensionless form as follows;
The momentum equations along the x-axis and y- axis, respectively;

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

$$Re\frac{\partial v}{\partial t} + Re\frac{\partial (uv)}{\partial x} + Re\frac{\partial (vv)}{\partial y} + \frac{\partial p}{\partial y} = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \tag{2}$$

The continuity equation;

$$-\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} = 0 \tag{3}$$

In these equations (u, v) represents the velocity vector components, p is the pressure and Re is the dimensionless Reynolds number.

$$Re = Re_D = \frac{\rho u D}{\mu} \tag{4}$$

Where, ρ is the density, D is the hydraulic diameter of the inlet channel, that is equivalent to twice its height, μ is the dynamic viscosity. The primitive variables can be arranged as shown in Figure 1(a). The finite difference approximations to the momentum equations (1) and (2) can be written; [6-7]
The momentum equations along the x-axis;

$$Re\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} + Re\frac{(uu)_{i+1,j} - (uu)_{i-1,j}}{2\Delta x} + Re\frac{(uv)_{i,j+1} - (uv)_{i,j-1}}{2\Delta y} + \frac{p_{i,j} - p_{i-1,j}}{\Delta x} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\Delta x^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{\Delta y^2} \tag{5}$$

The momentum equations along the y-axis;

$$Re\frac{v_{i,j}^{n+1}-v_{i,j}^{n}}{\Delta t}+Re\frac{(uv)_{i+1,j}-(uv)_{i-1,j}}{2\Delta x}+Re\frac{(vv)_{i,j+1}-(vv)_{i,j-1}}{2\Delta y}+\frac{p_{i,j}-p_{i,j-1}}{\Delta y}=\frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{\Delta x^2}+\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{\Delta y^2} \quad (6)$$

The similar approximations to the continuity;

$$-\frac{u_{i+1,j}-u_{i,j}}{\Delta x}-\frac{v_{i,j+1}-v_{i,j-1}}{\Delta y}=0 \quad (7)$$

The no-slip boundary conditions can be applied using ghost cells within the solid domain as shown in Figure 1(b). The application of $u_b = 0$ requires that $u_{i+1,j} = 0$. In a similar manner, the application of $v_b = 0$ requires that $v_{i+1,j} = -v_{i,j}$.

2.1 Stokes flow

Using the above described MAC (Marker and Cell) [7], [8] scheme to solve Stokes flow ($Re = 0$) within the backward step $[0,5] \times [0,1]$. The boundary conditions can be seen in Figure 2. The computation is proceeded using the local numbering similar to that of in Figure 3.

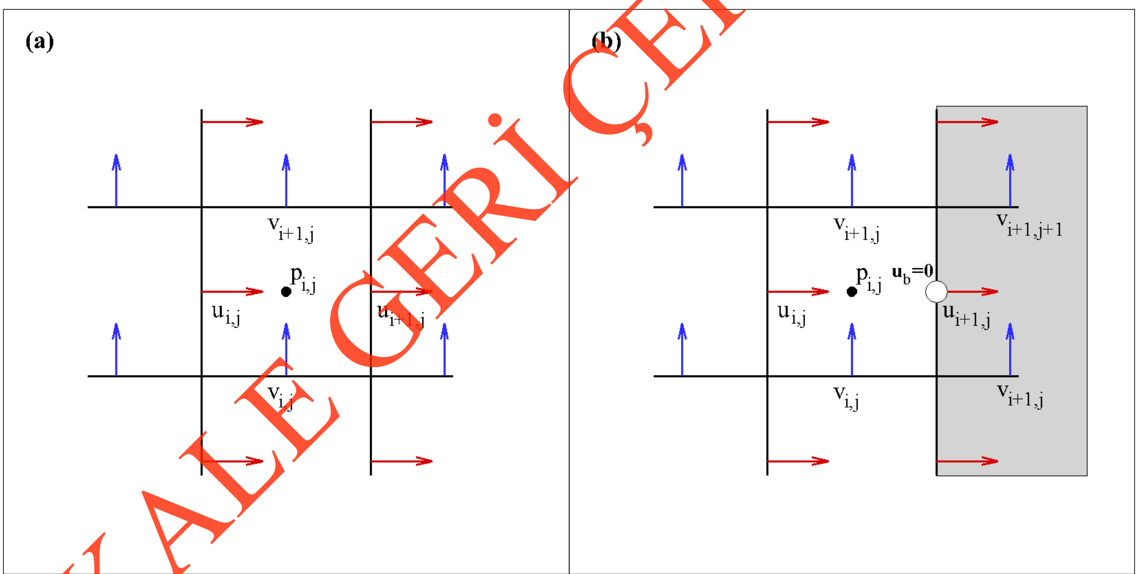

Figure 1: (a) The arrangement of primitive variables and (b) the application of no-slip boundary condition.

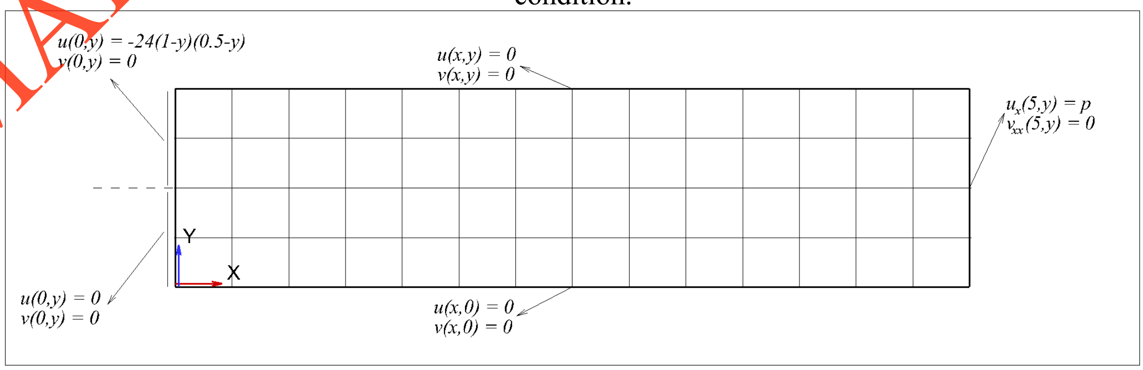

Figure 2: Computational domain and boundary conditions.

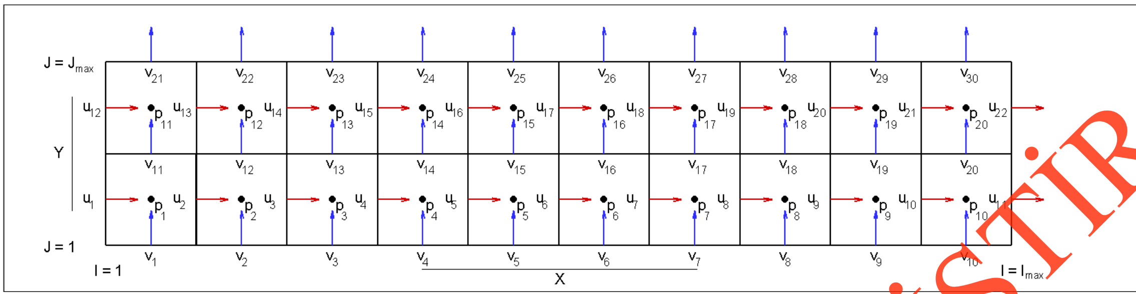

Figure 3: The local numbering of primitive variables.

2.2 Boundary conditions

The boundary conditions were given below have been implemented to the inlet, outlet, top and bottom boundaries accordingly.

- **Inlet:** Dirichlet boundary condition was applied. The u velocity profile was given as parabolic function.
  $u(0,y) = -24(1-y)(0.5-y)$ $\quad v(0,y) = 0 \quad y > 0.5$
  $u(0,y) =$ $\quad v(0,y) = 0 \quad y \leq 0.5$

- **Bottom:** Dirichlet boundary condition was applied.
  $u(x,0) =$ $\quad v(x,0) = 0$

- **Top:** Dirichlet boundary condition was applied.
  $u(x,1) =$ $\quad v(x,1) = 0$

- **Outlet:** Neumann boundary condition was implemented.
  $\frac{\partial u}{\partial x} = p \quad at\ x = 5$ $\quad \frac{\partial^2 v}{\partial x^2} = 0 \quad at\ x = 5$

3. Coding

The coefficients matrix A as depicted in Figure 4 includes the coefficients of u and v velocities and pressures in the X-Momentum, Y-Momentum and Continuity equations respectively. The matrix A is coded by considering the boundary conditions. Also, the right hand side matrix is defined according to the given boundary values. Finally, u and v velocities in the direction of X and Y with the pressure values defined in the cell centers is calculated by the matrix multiplication of inverse of A and the right hand side matrix. Pseudo code is found below.
A11: Coefficients of u velocities in the X-Momentum equation.
A12: 0
A13: Coefficients of pressures in the X-Momentum equation.
A21: 0
A22: Coefficients of v velocities in the Y-Momentum equation.
A13: Coefficients of pressures in the Y-Momentum equation.
A31: Coefficients of u velocities in the Continuity equation.

A_{32}: Coefficients of v velocities in the Continuity equation.
A_{33}: 0

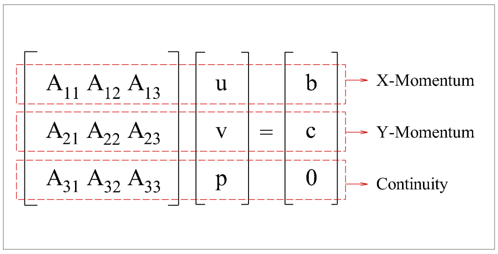

Figure 4: The coefficients matrix structure

Table 1: Pseudo code for Stokes flow

```
1 - Define computational domain dimensions ([0,5] × [0,1])
2 - Define number of nodes along X and Y directions (Imax & Jmax)
3 - Create sparse matrix A which includes coefficients of X, Y – Momentums and Cont. equation
4 - Create X-Momentum coefficients in matrix A
for i = 1:Imax
  for j = 1:Jmax-1
    if i = 1 (Inlet boundary)
      if cc > 0.5
        Dirichlet boundary condition
      else
        No-slip boundary condition
      end if
    else if i = Imax (Outlet boundary)
      Out-flow boundary condition
    else
      Calculate pressure coefficients location in matrix A
      if j = 1 (Bottom boundary)
        No-slip boundary condition
      else if j = Jmax-1 (Top boundary)
        No-slip boundary condition
      else
        Inner cells
      end if
    end if
  end for
end for
5 - Create Y-Momentum coefficients in matrix A
for j = 1:Jmax
  for i = 1:Imax-1
    if j = 1 (Bottom boundary)
      No-slip boundary condition
    else if j = Jmax (Top boundary)
      No-slip boundary condition
    else
      Calculate pressure coefficients location in matrix A
      if i = 1 (Inlet boundary)
        No-slip boundary condition
      else if I = Imax-1 (Outlet boundary)
```

```
        Out-flow boundary condition
      else
        Inner cells
      end if
    end if
  end for
end for
6 - Create continuity equation coefficients in matrix A
for i = Imax-1
  for j = Jmax-1
    Inner cells
  end for
end for
7 – Calculate velocities in the direction of X-Y and pressures
```

4. Results

Backward-facing step flow has been solved with continuity and incompressible Navier-Stokes equations as governing equations. Finite difference method with the MAC scheme was implemented to compute the u, v velocities in the X-Y directions and pressure values in the cell centers. u velocity distribution can be seen in Figure 5. u velocity profile at X=3 was compared with the data calculated in FLUENT. It can be seen from Figure 6 that the numerical code shows good agreement with the verified data. v velocity and dynamic pressure distribution are depicted in Figure 7 and 8. The vertical velocity changes dominantly occur around the inlet boundary because of the geometrical discontinuity. The computations were proceeded with 101 and 21 finite difference nodes along the X and Y directions respectively. The comparison between the number of finite difference nodes on streamlines can be seen in Figure 9 and 10. Table 1 shows the comparison of error value for different number of finite difference nodes. The absolute error value has been decreased by increasing the nodes number.

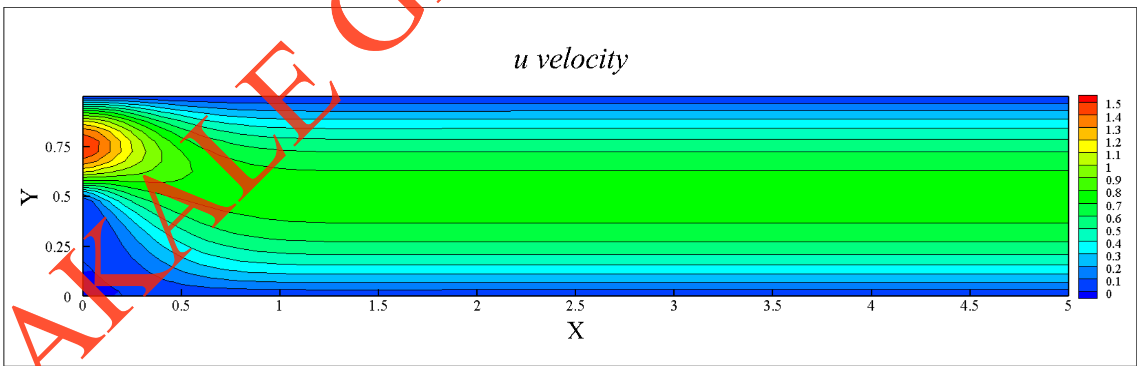

Figure 5: u velocity distribution with 101 and 21 finite difference nodes along the X and Y directions respectively.

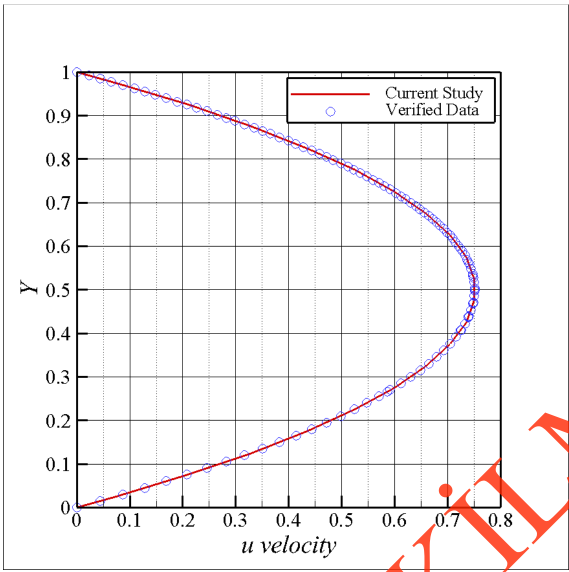

Figure 6: u velocity comparison between the current numerical study and data by Fluent at X=3.

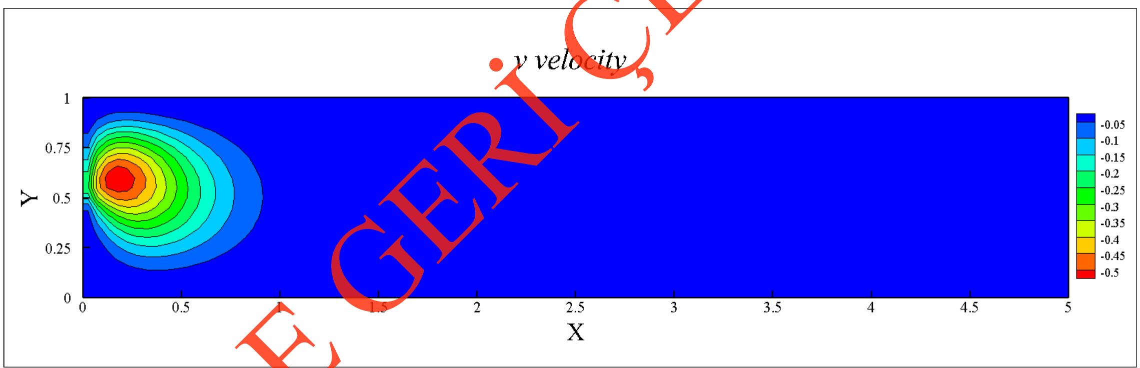

Figure 7: v velocity distribution with 101 and 21 finite difference nodes along the X and Y directions respectively.

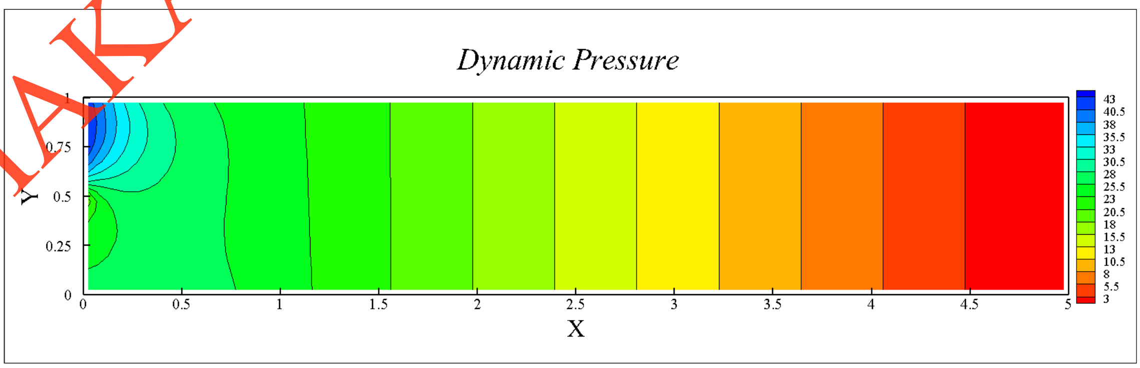

Figure 8: Dynamic pressure distribution with 101 and 21 finite difference nodes along the X and Y directions respectively.

Table 1. Comparison of error value for different number of finite difference nodes.

Exact	**5x11**	**11x51**	**21x101**
0.75000000000	0.75000002188	0.74999999943	0.74999999974
Error 10^6(%)	2.92	0.08	0.03

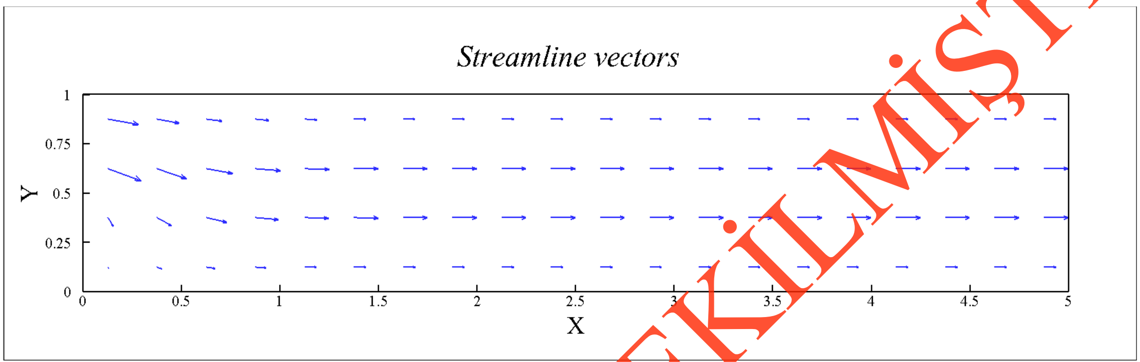

Figure 9: Streamlines vectors with 21 and 5 finite difference nodes along the X and Y directions respectively.

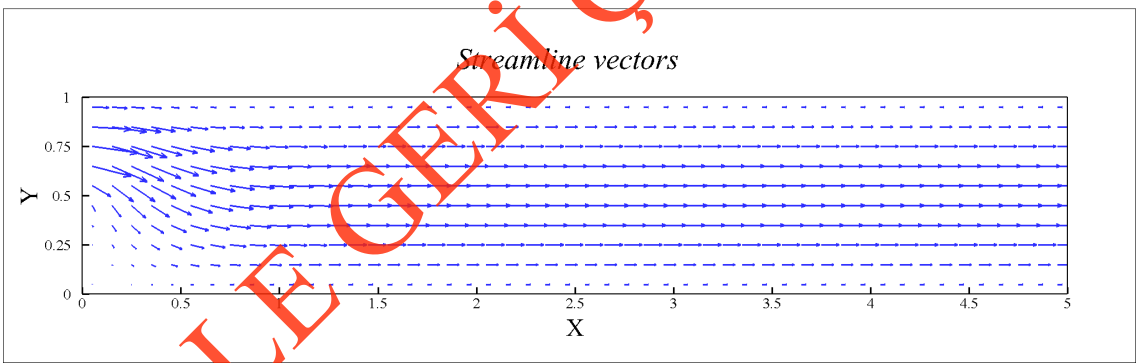

Figure 10: Streamlines vectors with 51 and 11 finite difference nodes along the X and Y directions respectively.

According to the provided parabolic function, the maximum u velocity is 1.5 in inlet section. In Figure 6, u velocity profile can be seen at section X=3 that is twice the inlet section. Here, the maximum velocity of u is 0.75, which shows that the problem provides the conservation of mass.

5. Conclusions

In this study, a backward-facing step flow by well-known finite difference discretization is solved in 2D Cartesian coordinate system using the incompressible Navier-Stokes momentum equations and the continuity equation. The convective terms in the momentum equations is discretized by using second order finite difference formulations, while pressure and time discretization is of first order. In the continuity equation the discretization in the main flow direction is of the first order and the cross flow is of the second order. The problem is considered as a Stokes flow. A Matlab code is written and compared with the results of Fluent software. It can be seen from the results that the numerical code shows good agreement with the verified data. In the future, a 3D simulation of backward-facing step flow with and without the viscosity effect will be examined by finite difference and finite volume methods. Although the second order discretization could be problematic in 3D flow problems, higher order discretization along with averaging and smoothing methods will be planned to utilized.

References:

[1] Armaly, B. F., Durst, F., Pereira, J. C. F., & Schönung, B. (1983). Experimental and theoretical investigation of backward-facing step flow. *Journal of fluid Mechanics*, *127*, 473-496.

[2] Tihon, J., Pěnkavová, V., Havlica, J., & Šimčík, M. (2012). The transitional backward-facing step flow in a water channel with variable expansion geometry. *Experimental Thermal and Fluid Science*, *40*, 112-125.

[3] Biswas, G., Breuer, M., & Durst, F. (2004). Backward-facing step flows for various expansion ratios at low and moderate Reynolds numbers. *J. Fluids Eng.*, *126*(3), 362-374.

[4] Chen, L., Asai, K., Nonomura, T., Xi, G., & Liu, T. (2018). A review of Backward-Facing Step (BFS) flow mechanisms, heat transfer and control. *Thermal Science and Engineering Progress*, *6*, 194-216.

[5] Pont-Vílchez, A., Trias, F. X., Gorobets, A., & Oliva, A. (2019). Direct numerical simulation of backward-facing step flow at Re=395 and expansion ratio 2. *Journal of Fluid Mechanics*, *863*, 341-363.

[6] Erturk, E. (2008). Numerical solutions of 2-D steady incompressible flow over a backward-facing step, Part I: High Reynolds number solutions. *Computers & Fluids*, *37*(6), 633-655.

[7] Harlow, F. H., & Welch, J. E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids*, *8*(12), 2182-2189.

[8] McKee, S., Tomé, M. F., Ferreira, V. G., Cuminato, J. A., Castelo, A., Sousa, F. S., & Mangiavacchi, N. (2008). The MAC method. *Computers & Fluids*, *37*(8), 907-930.

Appendix

Matlab code for the backward step flow by finite difference method in 2D.

```
% Prepared by Cihad Çelik - 508192006
% Backward Step Flow by Finite Difference Method in 2D
% 29-01-2021

clear all
clc

% Computational domain dimensions
H = 1; % Height of the solution domain (Y direction)
W = 5; % Width of the solution domain (X direction)

Nodes = menu('# of nodes on Y and X axis','5x21','11x51','21x101','Other');
if Nodes ==1; Jmax = 5;
elseif Nodes ==2; Jmax = 11;
elseif Nodes ==3; Jmax = 21;
elseif Nodes ==4; Jmax = input...
('Define # of nodes on Y axis (# of nodes on X axis calculated automatically): ');
end
dx = H/(Jmax-1);
dx2 = dx*dx;
Imax = W/dx+1;

t_u = (Jmax-1)*Imax; % number of u velocities
t_v = (Imax-1)*Jmax; % number of v velocities
t_p = (Jmax-1)*(Imax-1); % number of pressure point
t_uvp = (Jmax-1)*Imax+(Imax-1)*Jmax+(Jmax-1)*(Imax-1); % number of u+v+P

i=[]; j=[]; s=[];
b = t_uvp;
A = sparse(i, j, s, t_uvp, b);
RHS = sparse(i, j, s, t_uvp, 1);

x_axis = 0:dx:W;
x_axis_center = (dx/2):dx:(W-dx/2);
y_axis = 0:dx:H;
yac = (dx/2):dx:(H-dx/2); % y_axis_center

%% X - Momentum
s=0;
for i = 1:Imax
    for j = 1:Jmax-1
        m = (j-1)*Imax+i;
        if i == 1
            s=s+1;
            if yac(s) <= 0.5
                A(m,m) = 1;
                RHS(m) = 0;
            else
                % Dirichlet boundary condition
                A(m,m) = 1;
                RHS(m) = -24*(1-yac(s))*(0.5-yac(s));
            end
        elseif i == Imax
            n = ((j-1)*(Imax-1)+i-1)+(Jmax-1)*Imax+(Imax-1)*Jmax;
            % Out-flow boundary condition
            A(m,m) = -1/dx; A(m,m-1) = 1/dx;
            A(m,n) = 1;
            RHS(m) = 0;
        else
            n = ((j-1)*(Imax-1)+i)+(Jmax-1)*Imax+(Imax-1)*Jmax;
            if j == 1
                A(m,m) = 5/dx2; A(m,m+1) = -1/dx2; A(m,m-1) = -1/dx2; ...
                    A(m,m+Imax) = -1/dx2;
```

```
                A(m,n) = 1/dx; A(m,n-1) = -1/dx;
                RHS(m) = 0/dx2;
            elseif j == Jmax-1
                A(m,m) = 5/dx2; A(m,m+1) = -1/dx2; A(m,m-1) = -1/dx2; ...
                    A(m,m-Imax) = -1/dx2;
                A(m,n) = 1/dx; A(m,n-1) = -1/dx;
                RHS(m) = 0/dx2;
            else
                A(m,m) = 4/dx2; A(m,m+1) = -1/dx2; A(m,m-1) = -1/dx2; ...
                    A(m,m+Imax) = -1/dx2; A(m,m-Imax) = -1/dx2;
                A(m,n) = 1/dx; A(m,n-1) = -1/dx;
                RHS(m) = 0/dx2;
            end
        end
    end
end

%% Y - Momentum
for j = 1:Jmax
    for i = 1:Imax-1
        m = (j-1)*(Imax-1)+i+(Jmax-1)*Imax;
        if j == 1
            A(m,m) = 1;
            RHS(m) = 0;
        elseif j == Jmax
            A(m,m) = 1;
            RHS(m) = 0;
        else
            n = (j-1)*(Imax-1)+i+(Jmax-1)*Imax+(Imax-1)*Jmax;
            if i == 1
                A(m,m) = 5/dx2; A(m,m+1) = -1/dx2; A(m,m-(Imax-1)) = ...
                    -1/dx2; A(m,m+(Imax-1)) = -1/dx2;
                A(m,n) = 1/dx; A(m,n-(Imax-1)) = -1/dx;
                RHS(m) = 0/dx2;
            elseif i == Imax-1
                % Out-flow boundary condition
                A(m,m) = 1/dx2; A(m,m-1) = -2/dx2; A(m,m-2) = 1/dx2;
                RHS(m) = 0/dx2;
            else
                A(m,m) = 4/dx2; A(m,m+1) = -1/dx2; A(m,m-1) = -1/dx2; ...
                    A(m,m+(Imax-1)) = -1/dx2; A(m,m-(Imax-1)) = -1/dx2;
                A(m,n) = 1/dx; A(m,n-(Imax-1)) = -1/dx;
                RHS(m) = 0/dx2;
            end
        end
    end
end

%% Continuity
for i = 1:Imax-1
    for j = 1:Jmax-1
        m = (j-1)*(Imax-1)+i+(Jmax-1)*Imax+(Imax-1)*Jmax;
        m1 = (j-1)*(Imax-1)+i+(j-1);
        n1 = (j-1)*(Imax-1)+i+(Jmax-1)*Imax;
        A(m,m1) = -1/dx;
        A(m,m1+1) = 1/dx;
        A(m,n1) = -1/dx;
        A(m,n1+(Imax-1)) = 1/dx;
        RHS(m) = 0/dx;
    end
end

t_uv = t_u+t_v; % total number of u and v velocities

x = A\RHS;

%% Plot
u_vel = x(1:t_u);
u_vel = full(u_vel);
```

```
v_vel = x(t_u+1:t_uv);
v_vel = full(v_vel);

P = x(t_uv+1:t_uvp);
P = full(P);

x_axis = 0:dx:W;
x_axis_center = (dx/2):dx:(W-dx/2);
y_axis = 0:dx:H;
y_axis_center = (dx/2):dx:(H-dx/2);

for i = 1:Imax
    for j = 1:Jmax-1
        m = (j-1)*Imax+i;
        u_vel_grid(j,i) = u_vel(m);
    end
end

for j = 1:Jmax
    for i = 1:Imax-1
        m = (j-1)*(Imax-1)+i;
        v_vel_grid(j,i) = v_vel(m);
    end
end

for i = 1:Imax-1
    for j = 1:Jmax-1
        m = (j-1)*(Imax-1)+i;
        P_grid(j,i) = P(m);
    end
end

figure('Name','u velocity','NumberTitle','off')
[X,Y] = meshgrid(x_axis,y_axis_center);
contourf(X,Y,u_vel_grid,10)
xlabel('x');
ylabel('y');
title('u velocity Stokes flow');
colorbar

figure('Name','v velocity Stokes flow','NumberTitle','off')
[X,Y] = meshgrid(x_axis_center,y_axis);
contourf(X,Y,v_vel_grid,10)
xlabel('x');
ylabel('y');
title('v velocity');
colorbar

figure('Name','Pressure Stokes flow','NumberTitle','off')
[X1,Y1] = meshgrid(x_axis_center,y_axis_center);
contourf(X1,Y1,P_grid,10)
xlabel('x');
ylabel('y');
title('Pressure');
colorbar

for i = 1:Imax-1
    for j = 1:Jmax-1
        u_cen(j,i) = (u_vel_grid(j,i+1)+u_vel_grid(j,i))/2;
    end
end

for j = 1:Jmax-1
    for i = 1:Imax-1
        v_cen(j,i) = (v_vel_grid(j+1,i)+v_vel_grid(j,i))/2;
    end
end

figure('Name','Stream Stokes flow','NumberTitle','off')
```

```
[mx,my]=meshgrid(x_axis_center,y_axis_center);
XY = stream2(x_axis_center,y_axis_center,u_cen,v_cen,mx,my);
streamline(XY);
quiver(x_axis_center,y_axis_center,u_cen,v_cen);
grid on
```

MAKALE GERİ ÇEKİLMİŞTİR