Computational Fluid Dynamics

Project Report

Under guidance of

Prof. Dr.-Ing. E VON LAVANTE
Dipl.-Ing. HARUN KAYA

UNIVERSITY Of DUISBURG ESSEN
COMPUTATIONAL MECHANICS
PrudhviPriyanka Bhogavalli
Matr Nr ES03022760
March 24, 2016

Table of Contents

Project1

Given is a two-dimensional flow field consisting of a channel. The flow is assumed to be frictionless, incompressible, steady and irrotational. Thus, it can be described by either the potential function Φ or the stream function Ψ. In both cases the flow field is obtained by solving the elliptic Laplace equation for either one of these functions. The inflow velocity is 10 m/s, the working fluid is water with a density of ρ=1000 kg/m3.

Each student has to investigate an individual geometry that can be collected in Ma445 from Mr.Menze.

1. The stream function Ψ should be found by solving the Laplace equation at discrete points. The resulting function $\Psi(x,y)$ should be stored in a separate file.

2. The corresponding velocity components u and v have to be computed and stored for each point,
$u = d\Psi/dy$, $v = -d\Psi/dx$.

3. The resulting pressure distribution, given by the Bernoulli equation, has to be computed. In particular, the pressure coefficient Cp should be calculated for each point.

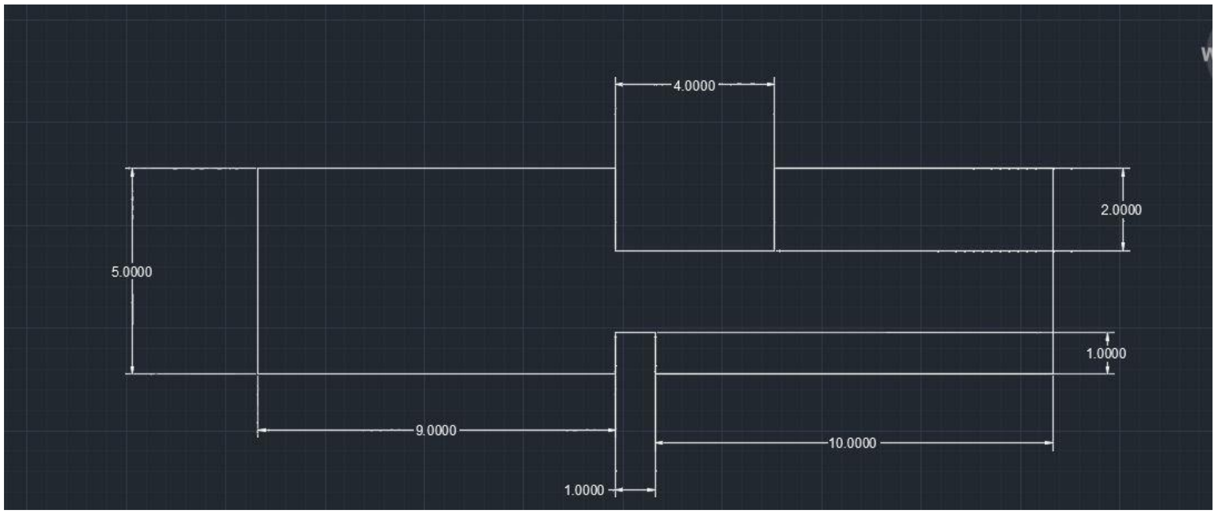
4. The results should be displayed using a proper graphic representation. In particular, following pictures should be produced:
a. Stream lines
b. Contours of the stream function
c. Contours of the velocity magnitude
d. Contours of the pressure coefficient
As an option, the velocity vectors and/or the contours of the single velocity components can be given.

The results should be summarized in an individual report. This report should include:

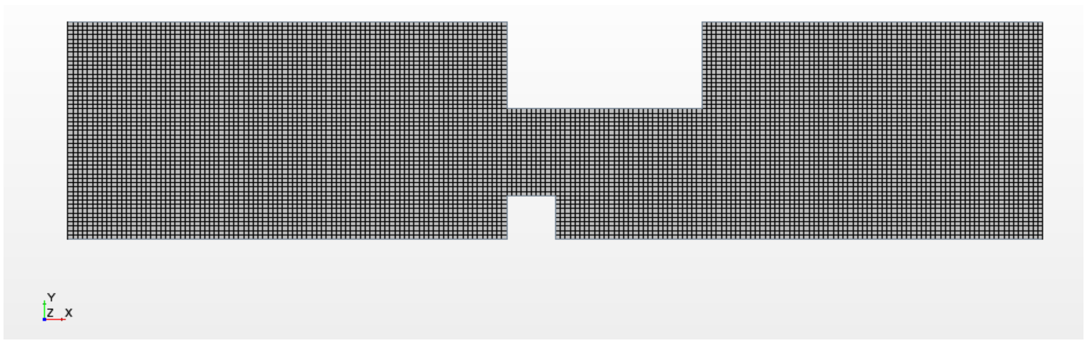
The above described pictures, a complete listing of your program, a short discussion of your own results (about 1 page) and a comparison with the corresponding results obtained using the commercial product Star-CCM+.

Star –CCM+ Results:

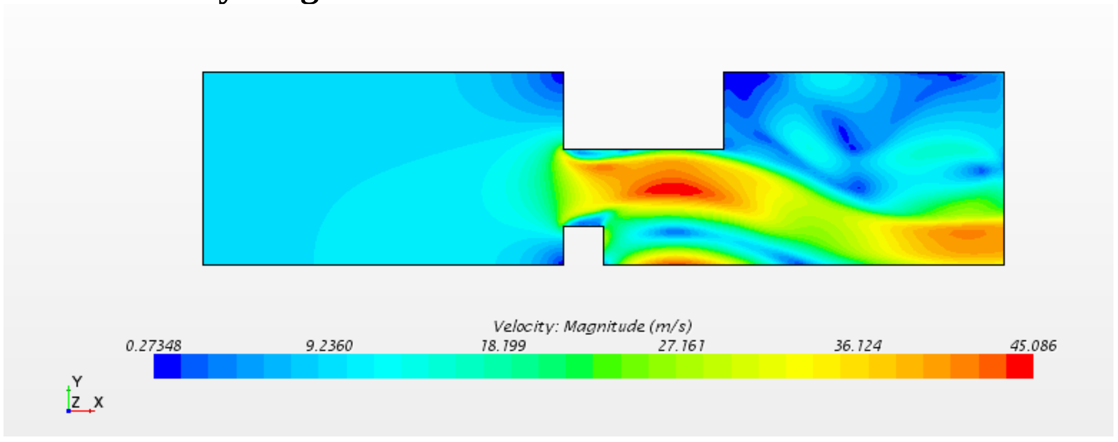
Given Body:



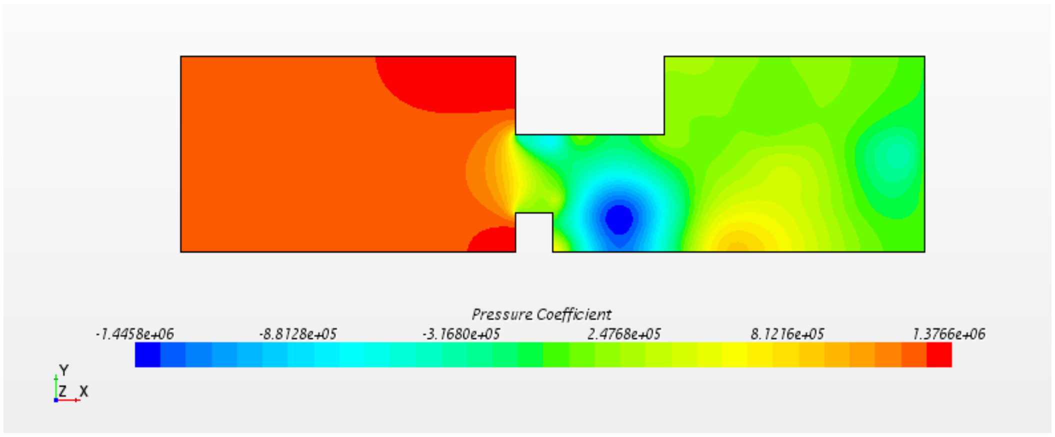
Mesh File:



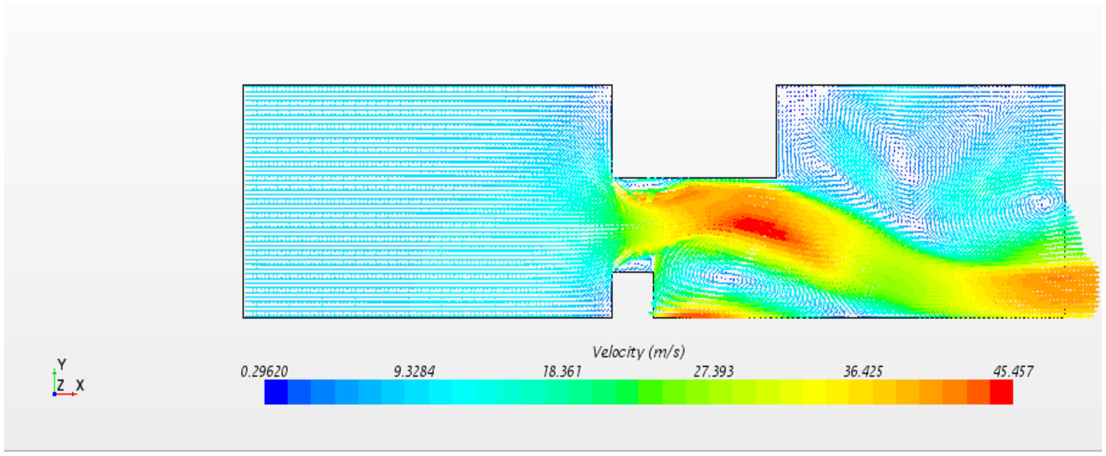
Contours of Velocity Magnitude:



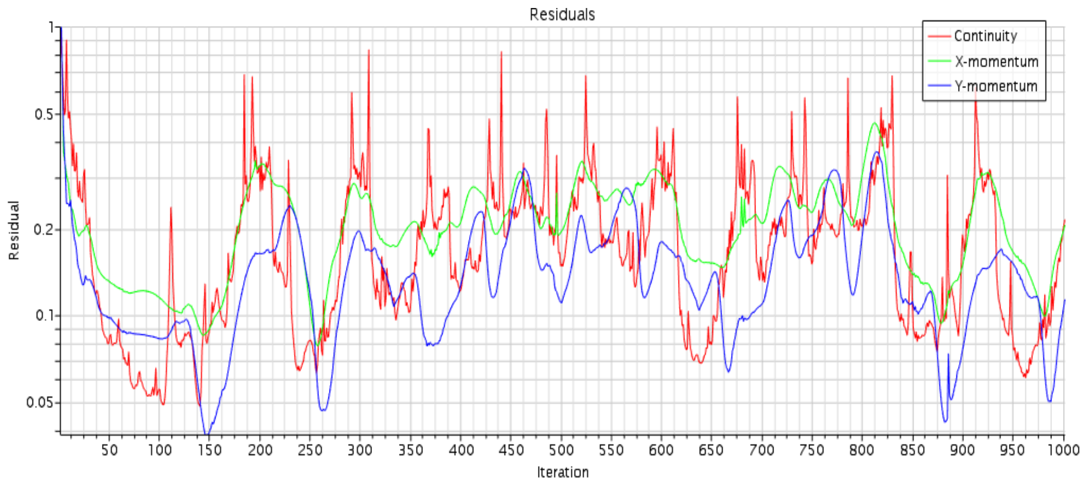
Contours of Pressure Coefficient:



Stream Lines:



Residuals:



MATLAB Code

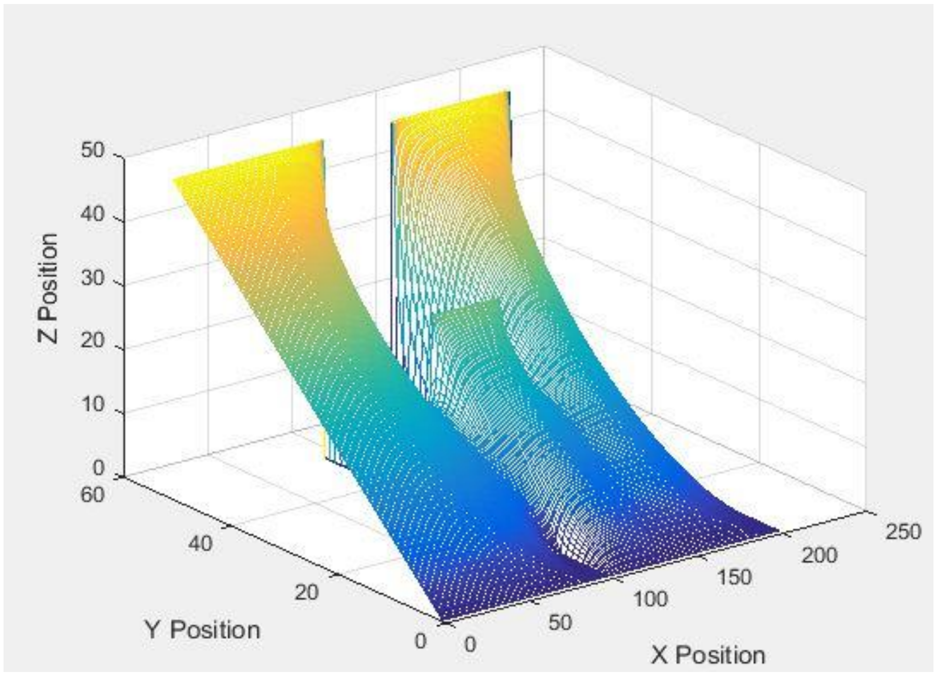
```
clear all
close all
clc
l    =  20;       %Wall Length
h    =  5;        %Wall Height
Uint =  10;      %Initial Velocity

Psi_old = zeros(51,201);% Defining a Matrix M whose values will be replaced
                      % by stream function (SF) values
u       =   zeros(51,201);   % Matrix u for velocity in x-direction
v       =   zeros(51,201);   % Matrix v for velocity in y-direction
V       =   zeros(51,201);   % Matrix V for Total velocity
Cp      =   zeros(51,201);   % Matrix for Pressure coefficient values.
P       =   zeros(51,201);   % Matrix for Pressure value at each point

for i=2:51         %Assigning inlet Psi values
Psi_old(i,1)=Psi_old(i-1,1)+10*0.1;
end

%Assigning initial boundary conditions
Psi_old(1,2:90)             =   0;
Psi_old(11,91:110)          =   Psi_old(11,1);
Psi_old(1,111:201)          =   0;
Psi_old(51,2:90)            =   Psi_old(51,1);
Psi_old(31,91:130)          =   Psi_old(31,1);
Psi_old(51,131:201)         =   Psi_old(51,1);

Psi_new   =      Psi_old; %Assigning the Psi values to the new Psi matrix
Abserr    =      1;       %Initial error value
Abstolr   =      1e-7;    %Max. Iteration deviation allowed

%The given diagram is divided into 4 sections.
while max(abs(Abserr)) > Abstolr

% For Streamlines of area 1
for i=2:50
for j=2:90
Psi_new(i,j)=0.25*(Psi_old(i-1,j)+Psi_old(i+1,j)+Psi_old(i,j-1)+Psi_old(i,j+1));
end
end

% For Streamlines of area 2
for i=11:30
for j=91:100
Psi_new(i,j)=0.25*(Psi_old(i-1,j)+Psi_old(i+1,j)+Psi_old(i,j-1)+Psi_old(i,j+1));
end
end

% for Streamlines of area 3
```

```
for i=2:30
for j=101:130
Psi_new(i,j)=0.25*(Psi_old(i-1,j)+Psi_old(i+1,j)+Psi_old(i,j-1)+Psi_old(i,j+1));
end
end

% For Streamlines of area 4
for i=2:50
for j=131:200
Psi_new(i,j)=0.25*(Psi_old(i-1,j)+Psi_old(i+1,j)+Psi_old(i,j-1)+Psi_old(i,j+1));
end
end

Abserr = Psi_new(:)-Psi_old(:); % Error calculation
Psi_old=Psi_new;
end

%Velocity and coefficient of pressure calculations
for i=2:50
for j=2:90
u(i,j)       =   (Psi_new(i,j)-Psi_new(i-1,j))/0.1;
v(i,j)       =   (Psi_new(i,j+1)-Psi_new(i,j))/0.1;
V(i,j)       =   (((u(i,j))^2)+(v(i,j))^2)^0.5;
Cp(i,j)      =   1-((((u(i,j))^2)+(v(i,j))^2)/100);
P(i,j)       =   0.5*1000*V(i,j)^2;
end
end

for i=11:30
for j=91:100
u(i,j)       =   (Psi_new(i,j)-Psi_new(i-1,j))/0.1;
v(i,j)       =   (Psi_new(i,j+1)-Psi_new(i,j))/0.1;
V(i,j)       =   (((u(i,j))^2)+(v(i,j))^2)^0.5;
Cp(i,j)      =   1-((((u(i,j))^2)+(v(i,j))^2)/100);
P(i,j)       =   0.5*1000*V(i,j)^2;
end
end

for i=2:30
for j=101:130
u(i,j)       =   (Psi_new(i,j)-Psi_new(i-1,j))/0.1;
v(i,j)       =   (Psi_new(i,j+1)-Psi_new(i,j))/0.1;
V(i,j)       =   (((u(i,j))^2)+(v(i,j))^2)^0.5;
Cp(i,j)      =   1-((((u(i,j))^2)+(v(i,j))^2)/100);
P(i,j)       =   0.5*1000*V(i,j)^2;
end
end

for i=2:50
for j=131:200
u(i,j)       =   (Psi_new(i,j)-Psi_new(i-1,j))/0.1;
v(i,j)       =   (Psi_new(i,j+1)-Psi_new(i,j))/0.1;
V(i,j)       =   (((u(i,j))^2)+(v(i,j))^2)^0.5;
Cp(i,j)      =   1-((((u(i,j))^2)+(v(i,j))^2)/100);
P(i,j)       =   0.5*1000*V(i,j)^2;
end
end
```

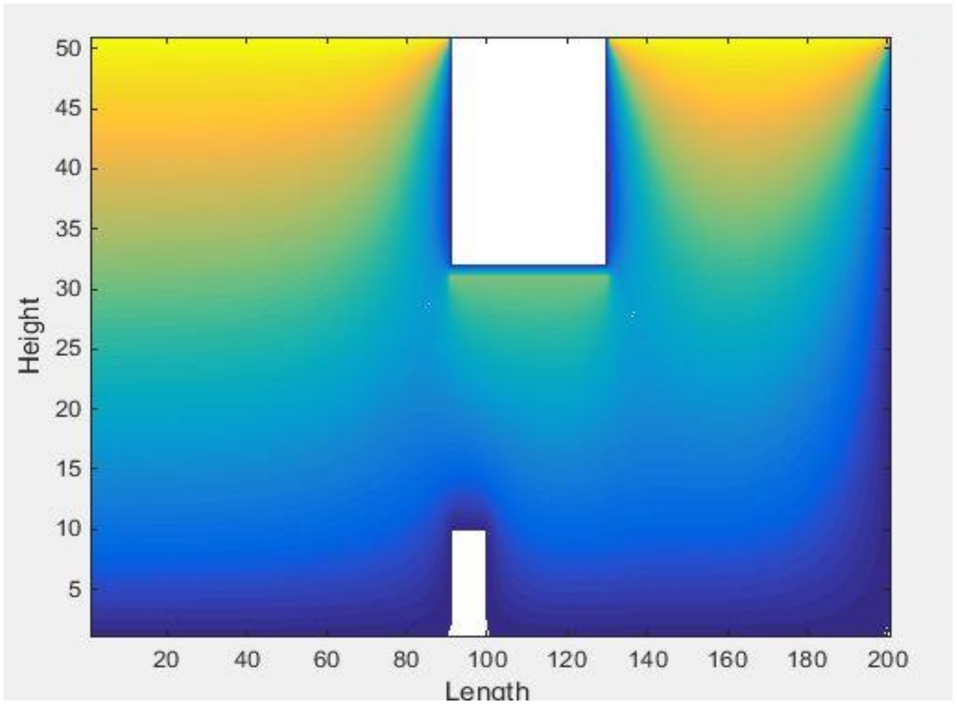
```
% Plotting the graph with MESH
figure(1);
mesh(Psi_new);
xlabel('X Position');
ylabel('Y Position');
zlabel('Z Position');

% Plotting the graph with contour streamlines
figure(2)
contour(Psi_new,5000);
xlabel('Length ');
ylabel('Height');

% Plotting graph contour of resultant velocity
figure(3)
hold on
contour(V,5000);
colormap jet
xlabel('Length');
ylabel('Height')

% Plotting graph with contour of pressure coefficient
figure(4)
hold on
contour(Cp,5000);
xlabel('Length');
ylabel('Height');
```

Matlab Results:

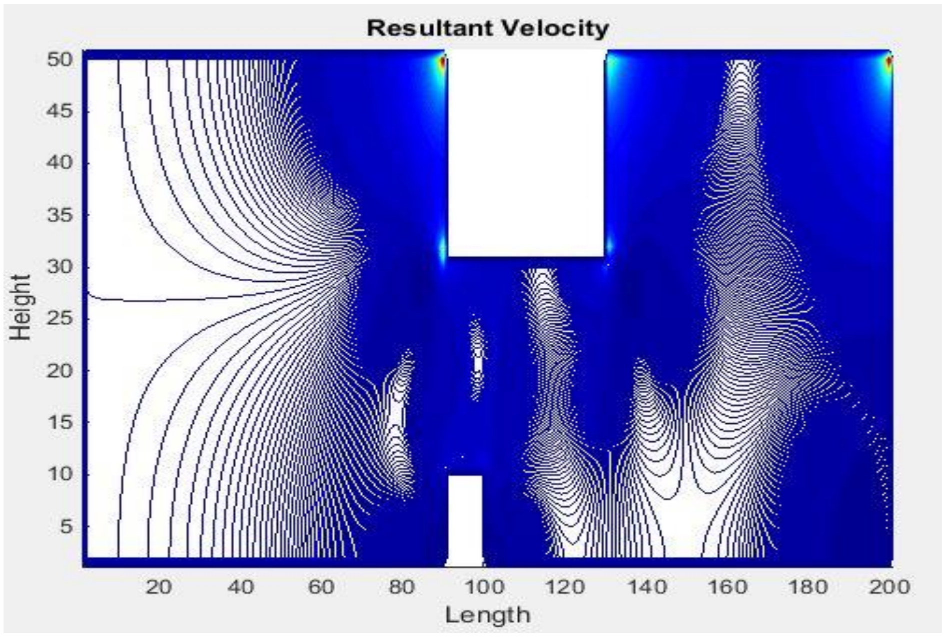
Mesh File:



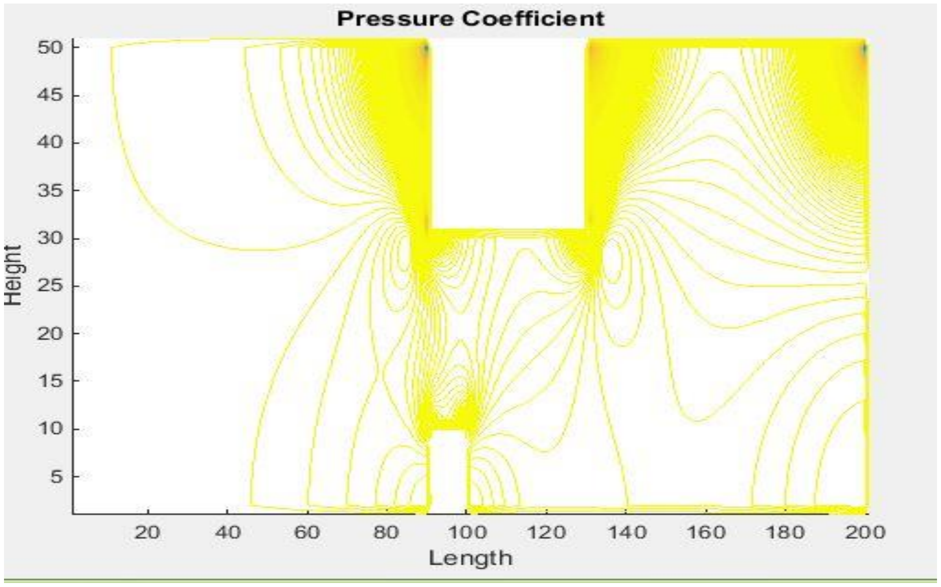
Stream Lines:



Contours of Velocity Magnitude:



Contours of the Pressure magnitude:



Results & Conclusion:

- After having a look at above figures and plots, it is clear that there is a difference in results in MATLAB and STAR-CCM.
- Star CCM+ works with physical specifications, as we set physics and also define different parameters in the commercial software, but in MATLAB we don’t define the physical values.
- In MATLAB programming different grids are used that is also one reason for different results.
- High pressure before the notch and low pressure after the notch due to high velocity at outlet.
- Streamlines are better in MATLAB when compared to Commercial software
- From the plots it is clear that velocity is more at outlet compared to inlet

Comparison between Commercial software and MATLAB:

- The sudden sharp contraction of duct area is the cause of increasing the velocity at the region which causes losses .The values of velocity magnitude show a change on the angle which suddenly changes the flow direction and hence shows observable changes at this regions, and also the increase of v-component value affected by the large increase of the flow velocity at this point.
- The pressure at the step will suddenly decrease because the flow energy has been used to increase the velocity at this point, which may cause pressure bubbles under certain physical conditions.
- From the results streamlines drawn in STAR CCM+ are not as clear as in MATLAB as Numerical method does not show any vortex because streamlines are function of mathematical means around the nodes.

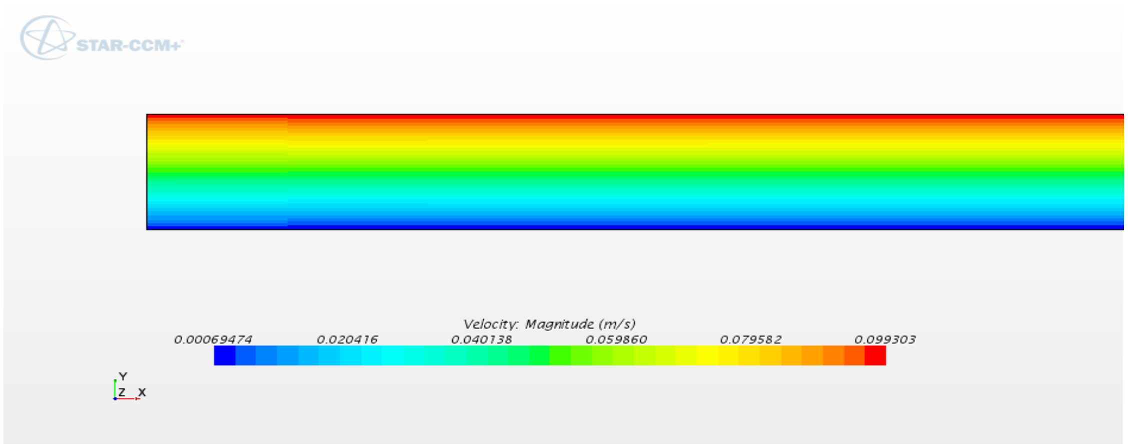
Project2:

Given is the first Stokes problem in which the unsteady flow between two horizontal plates is to be calculated. At t = 0 (starting time) both plates are fixed in space. The plates are 1 m long and the distance (height) is h = 0.01 m. The fluid between the plates is glycerin, ρ =1280.84 kg/m³, μ = 0.8943 Pa s. For t > 0 the upper plate is suddenly moving with U = 0.1 m/s to the right and will remain for all times t > 0
at this speed. The distribution of axial velocity u (in x direction along the plates) as a function of time and space, u (t,y) has to be determined. It was shown that the dimensionless representation of the problem u'(t',h) is computationally much cheaper and therefore recommended.
1. For the parabolic equation, the axial velocity component u has to solved numerically. For this purpose, the explicit method and the implicit method has to be used. The results for certain time periods should be shown and compared graphically.
2. The results have to be to be discussed in a written report. In particular, the explicit and the implicit method have to be compared. The respective results of u(t,h) should be re plotet for different dimensionless times.
3. The own numerical solution has to compared to the solution obtained witch StarCCM+. In particular, the accuracy and quality of the results should be discussed.

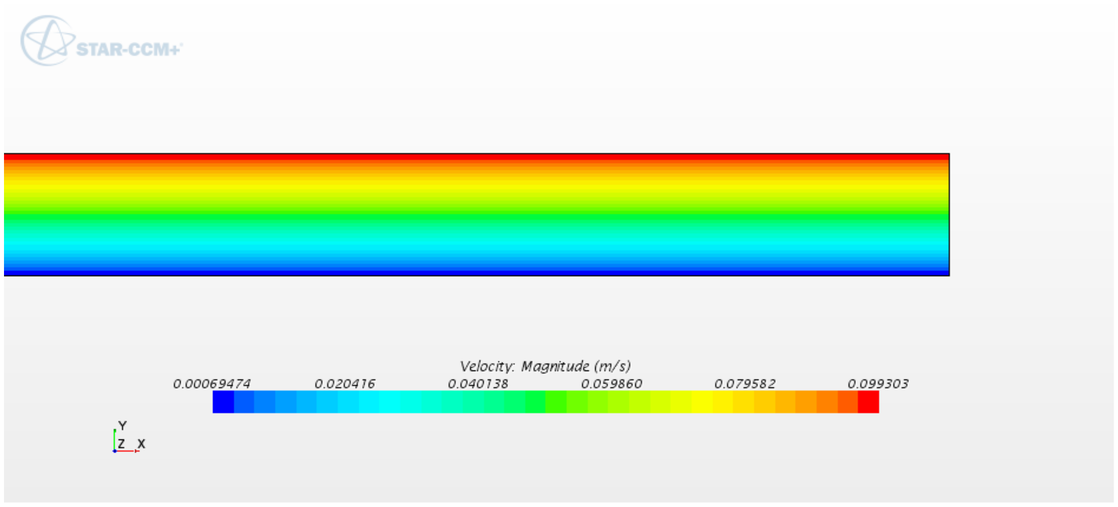
Solution:
Given values are:
Type of fluid: **Glycerin**
Distance between the plates (H): **0.8 m**
Velocity of Upper Plate (Uw): **0.1 m/s**
In this problem, two plates are arranged horizontally whose distance between them is given. Here one plate is fixed and the other one moves along the fixed plate.
During the process the temperature and the density of the fluid remains unchanged for the type of given fluid whereas the flow of the fluid is assumed to be incompressible.

Star CCM+ Simulations:
Geometry:

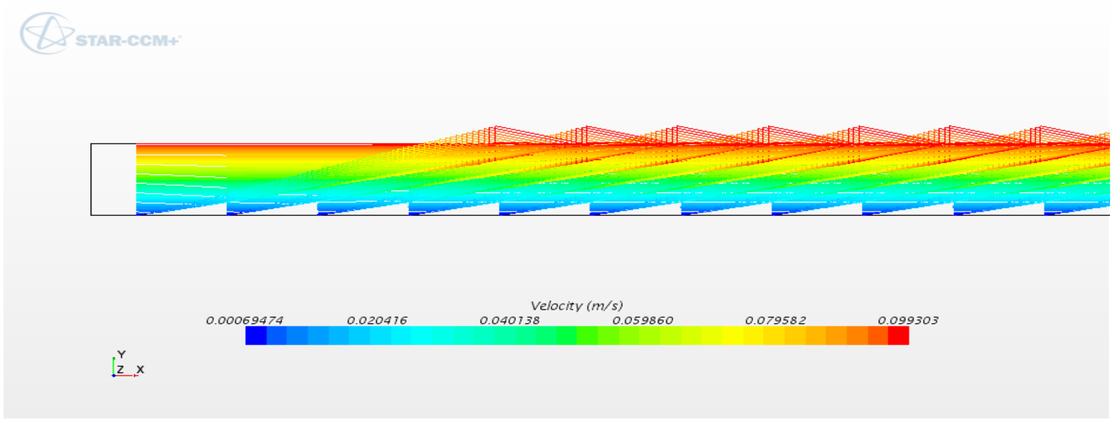
Scalar Scene (Inlet):



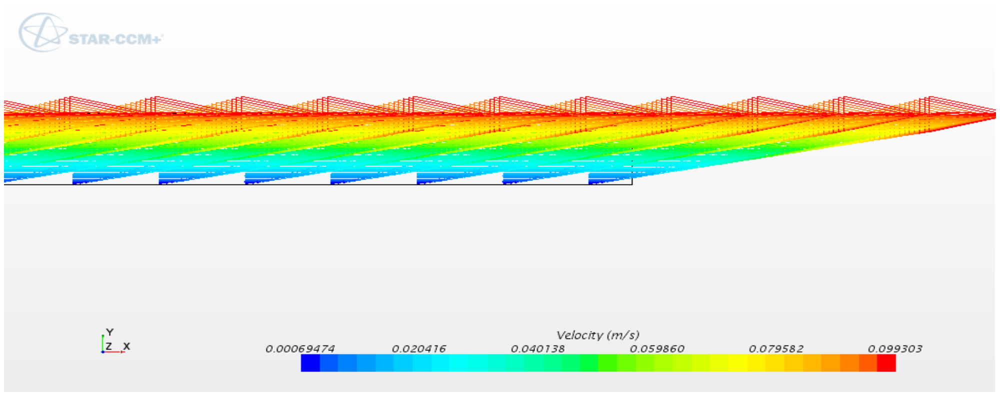
Scalar Scene (Outlet):



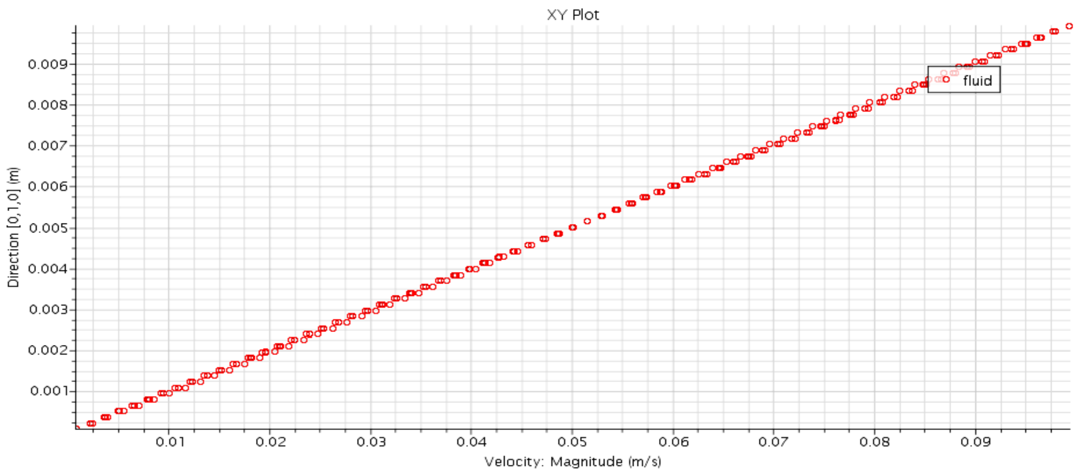
Vector Scene (Inlet):



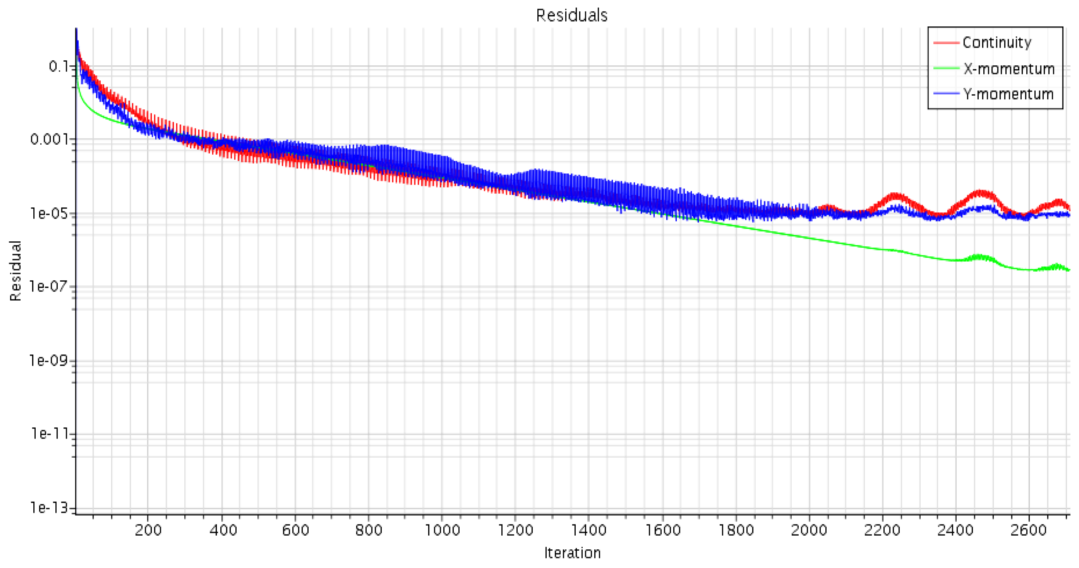
Vector Scene (Outlet):



XY-Plot: Velocity magnitude (pressure inlet) Vs position:



Residuals:



MATLAB Code

Explicit

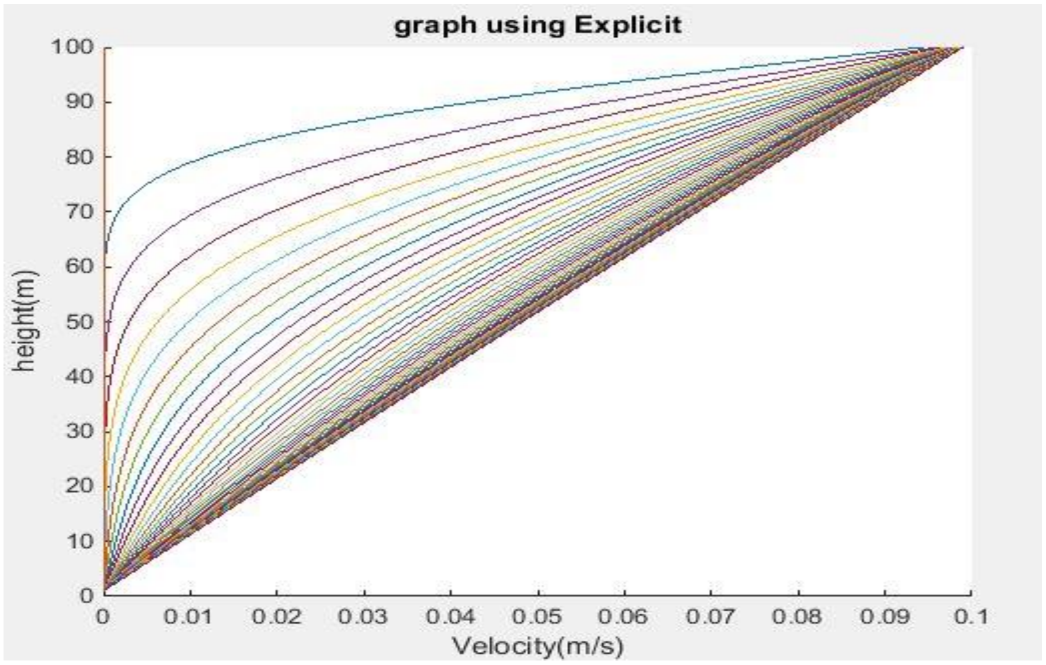
```
clear all
close all
%Matlab code for Explicit
Up       = 0.1;            %Velocity of Upper Plate
Dg       = 1280.84;        %Density of glycerin
Vi       = 0.8943;         %viscosity of glycerin
dp       = 0.01;           %Distance between grid points
d        = 0.01;           %Distance between plates
l        = 1;              %Length of the plate
D        = 1:1:101;
Ymin     = 1;              %Minimun y coordiante point
Ymax     = 100;            %Maximum Y coordinate point
M        = zeros(Ymax+1); %Assigning null matrix
Mi       = zeros(Ymax+1);
M(1)     = 0;          %Initial speed at bottom of plate for first time step
M(101)   = 0.1;        %Value of speed at top of the plate for initial Time step
Mi(101)  = 0.1;        %Initial speed at bottom of plate for Second time step
Re       = (Up*Dg*d)/Vi;     %formula of Reynolds Number
Ts       = Re*(dp^2)/2;       %calculating timestep
AV       = Ts/(Re*dp^2);      %calculating alpha value
SN       = 100;               %Stopping Number
CT       = 0;                 %Loop counter

for T = 1:4500 % Time step for iteration
for i = Ymin+1:Ymax %Loop for 2nd order Equation
Mi(i) = M(i)+AV*(M(i+1)-2*M(i)+M(i-1)); M(i) = Mi(i);
end
if (T == SN)
figure(1)
hold on
plot(Mi,D); % velocity verses height
CT = CT+1;
SN=SN+100;
end
end
title('graph using Explicit')
ylabel('height(m)');
xlabel('Velocity(m/s)')
ylim([0 100]);
```

Implicit

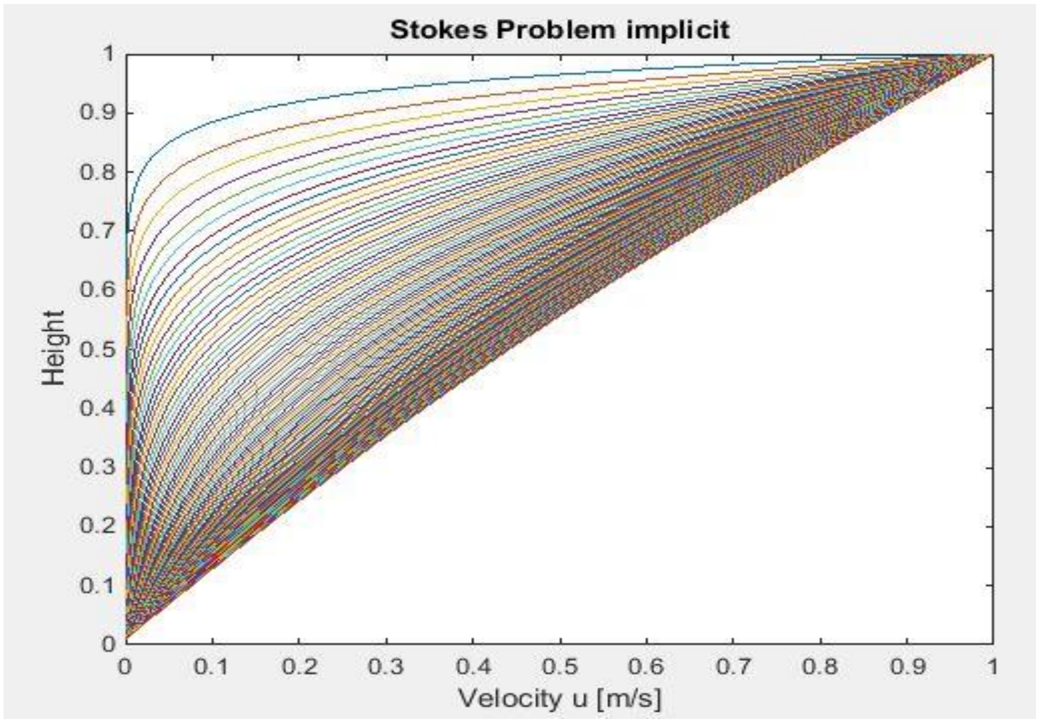
```
clear all
close all
%code for implicit
Up          =       0.1;          %velocity of the plate
dp          =      0.01;          % Spatial Increment in y—direction
l           =         1 ;         %length of the Plate
d           =       0.1;          %Distance between plates
de          =      0.01;
em          =    1/de—2;
Vi          =    0.08943;         %vicosity of glycerin
Dg          =    1280.84;         %density of Glycerin
Re          = (Up)*Dg*d/Vi;
Ts          = 50*0.5*Re*de^2;     %timestep calculator
AV          = Ts/(Re*de^2);       %alpha calculator
er          = 0.000001;
EM          = er*ones(em,1);
M(1,1)      = 1+2*AV;             %tridiagonal Matrix M
M(2,1)      = —AV;
M(1,2)      = —AV;
M(em,em)    = 1+2*AV;

for i = 2:(em—1)
M(i,i)      = 1+2*AV;
M(i+1,i)    = —AV;
M(i,i+1)    = —AV;
end

Z(em,1)=AV;
L = diag(ones(1,em));
R(1,1)  = M(1,1);

for i = 1:em—1
L(i+1,i)     =   M(i+1,i)/R(i,i);
R(i+1,i+1)   =   M(i+1,i+1)—(L(i+1,i)*M(i,i+1));
R(i,i+1)     =   M(i,i+1);
end

Ia = (inv(R)*inv(L)); LM(:,1)  = Ia*Z;
for a = 1:800
d(:,a)       =   LM(:,a)+Z;
LM(:,a+1)    =   Ia*d(:,a);
if LM(:,a+1)—LM(:,a) <= EM;
a;
break
end
end

vs           = [zeros(1,a+1);LM ;ones(1,a+1)];
y            = 1:1:100;
D            = vs(:,1:1:100);
plot(D,(y)/100)
title('Stokes Problem implicit');
xlabel ('Velocity u [m/s]');
ylabel ('Height');
```

Matlab Results:

Explicit:



Implicit:



Results & Conclusion:

The second project part is executed in both STAR CCM+ and MATLAB and
the plots are presented.

- From the plots it is clear that there are differences in results in both software's.
- From the results of commercial software, we can conclude that Velocity is transmitted through layers.
- Running time of explicit solution is comparatively more compared to implicit.
- As the lower plate is fixed, and upper plate moves with a velocity, it is clear that velocity distribution between layers is linear.
- In the star CCM+ the solver is dealing at a time with five complex equations Where as in MATLAB numerical methods I am dealing with the Euler equation

Project3

Die Ausbreitung einer schwachen Druckwelle wird näherungsweise durch die Wellengleichung 1.-Ordnung beschrieben:

$$pt + c\ px = 0 \quad (1)$$

wobei c>0 eine konstante Ausbreitungsgeschwindigkeit ist, die in diesem Fall die Schallgeschwindigkeit ist, c0 = 310 m/s. Das Problem ist durch die Anfangsverteilung des Druckes definiert, p0 =7.000, p = p0 (1 + cos(2π x/L)) . Die zu simulierende Domäne ist L = 10 m lang.

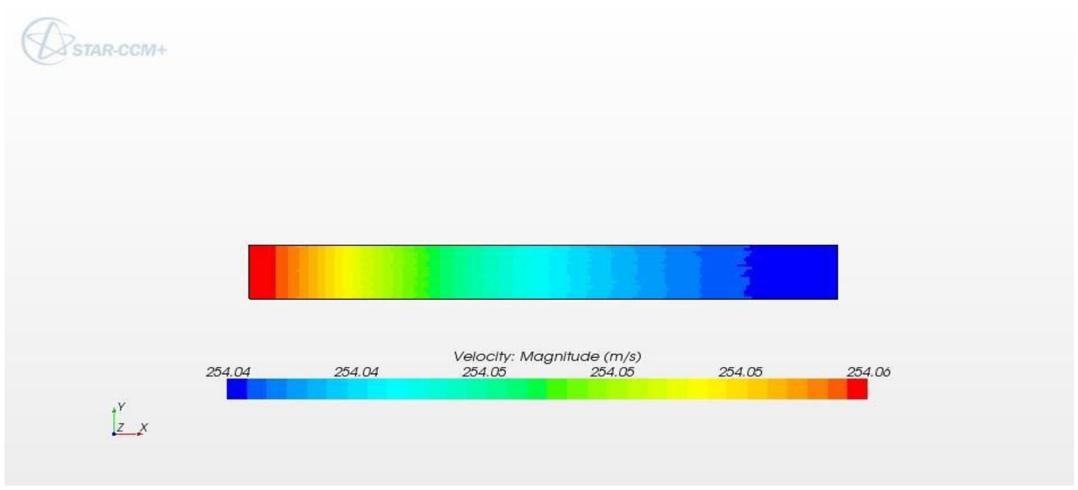
1. Das Problem soll in geeigneter Form dimensionslos gemacht werden. Die Bezugsdichte kann als ρ = 1.22 kg/m3 angenommen werden.
2. Es soll die hyperbolische Gleichung (1), die den Druck als Funktion von Zeit und Raum, p'(x',t'), beschreibt, numerisch gelöst werden. Hierzu soll eine explizite upwind Methode verwendet werden.Die Ergebnisse für verschiedene Zeiten (mindestens 4) sollen in eine Datei geschrieben werden, um später graphisch nachbearbeitet zu werden.
3. Die eigene numerische Lösung soll mit der Lösung des Star CD CCM+ verglichen werden. Insbesondere soll die Genauigkeit und Qualität der Ergebnisse diskutiert werden. Achtung: es müssen dimensionsbehaftete Daten verglichen werden, da CCM+ nur mit Dimensionen funktioniert. Insbesondere sollen anhand von Abbildungen folgende Fragen beantwortet werden:
a) Welche Methode ist dissipativer?
b) Wie ist die Dispersion der beiden Methoden (Art und Größe)

4. Der Teil 2. soll für den nichtlinearen (aber realistischeren) Fall c = c(p) wiederholt werden. Dabei gilt c/c0 = (p/p0)^((κ-1)/2κ).
5.Alle Ergebnisse sollen mit der (bekannten) analytischen Lösung verglichen werden.

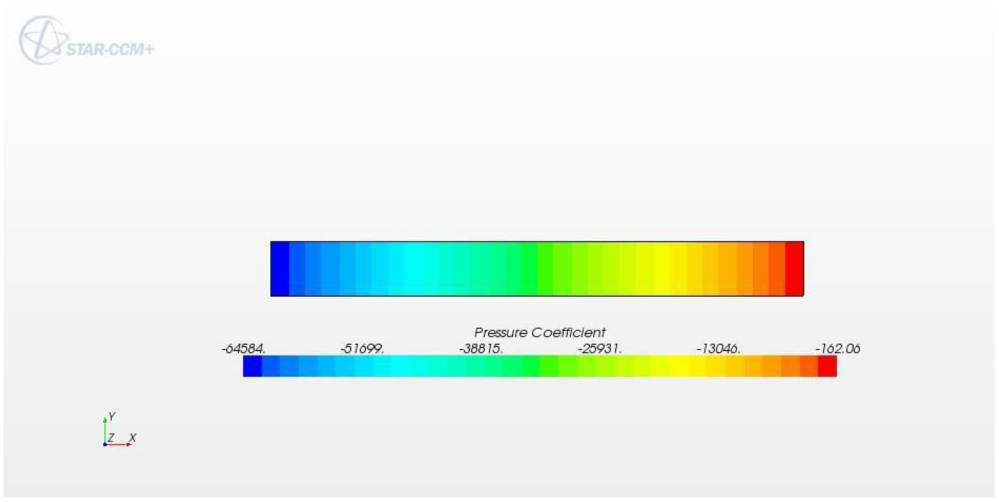
Report:

The following plots are obtained from Star –CCM+.

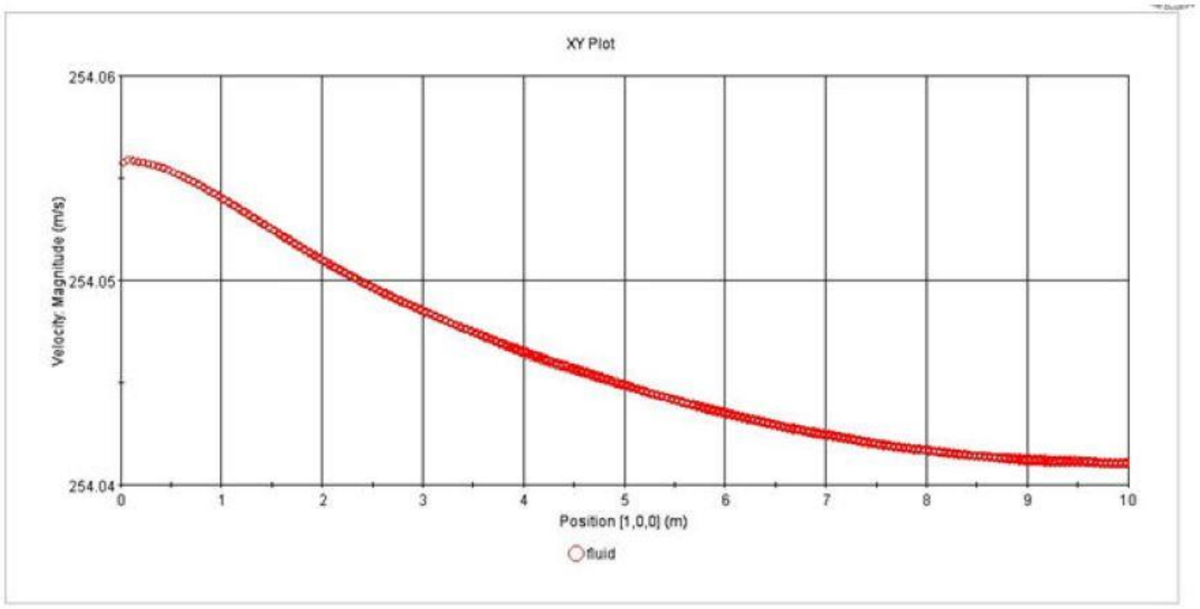
Velocity magnitude:



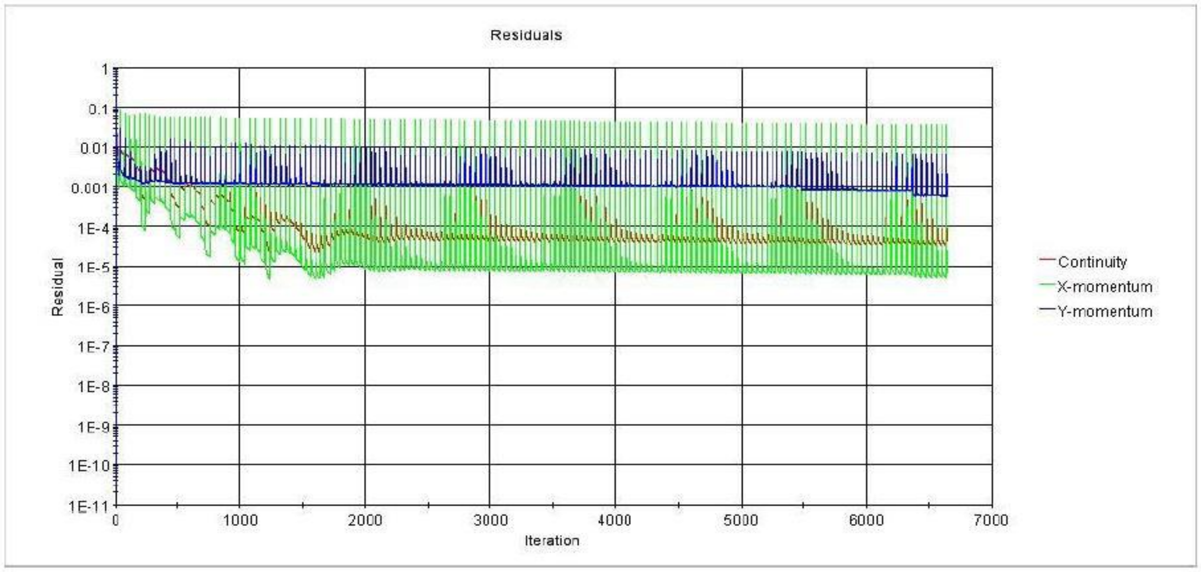
Pressure Coefficient:



XY-Plot:



Residuals:

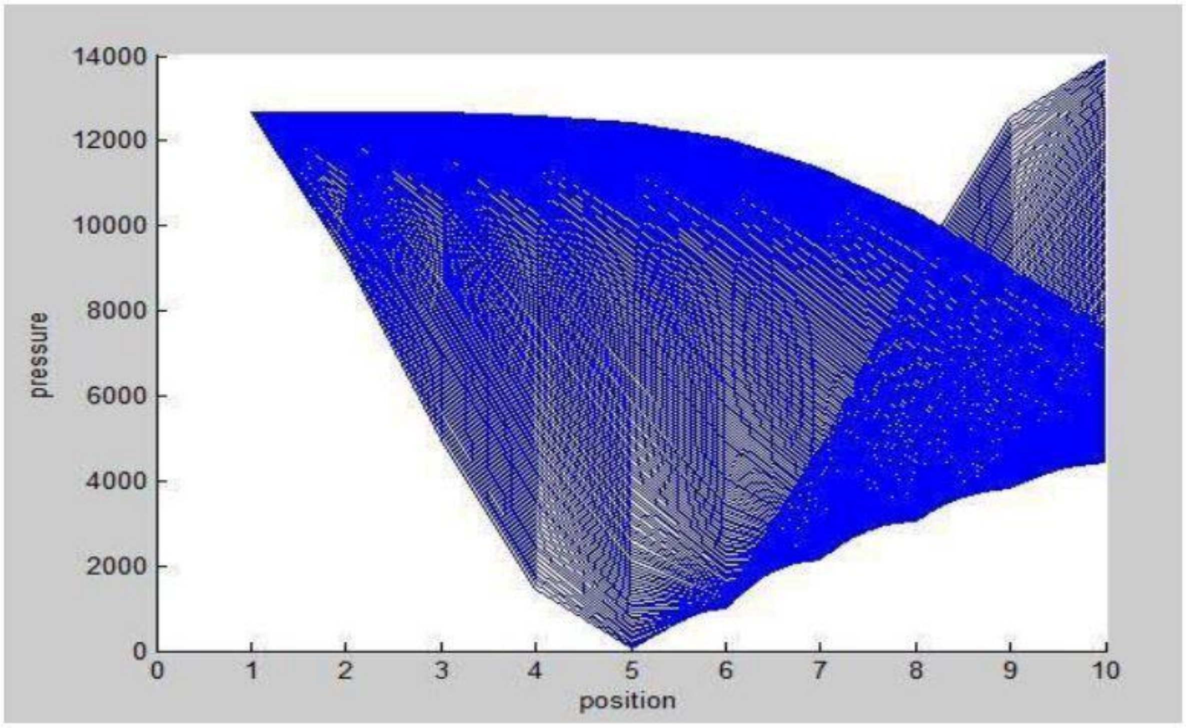


Matlab Code:

```
clear all
close all
%intializing variables
c           =    310; %Velocity of sound in air
l           =     10; %length of coloumn
pre         =    7000;
pa          =    zeros();
pb          =    zeros();
pa(1,1)     =    14000;

for q=1:1:10;

pa(q,1)     =   pre*(1+cos(6.28*(q/10)));

end
t           =           0.1; % chosen time interval
x           =       0.00001; % spacing between pointspoints spacing
R           =           t/x;
V           =           c/R;
for j=1:250
for i=2:10

pb(i,1)    =    pa(i,1)-V*(pa(i,1)-pa(i-1,1));

end

pa(i,1)    =     pb(i,1);

figure(1)
hold on
plot(pa)
xlabel('position'); ylabel('pressure'); xlim([0 10])
end
```

On executing the above code in MATLAB I got following plots

Position Vs Pressure:



Results & Conclusion:

The third project part is executed in both STAR CCM+ and MATLAB and the plots are presented.

- We can observe the pressure drop along the fluid domain uniformly in the steady state simulation
- Along the characteristic curve upwind scheme are used to numerically simulate the direction of propagation in a flow field.
- The results of MATLAB are dissipative

Project 4

Given is a quasi-one-dimensional nozzle. The divergent part of the nozzle is given by following equation:

$$\frac{A}{A_t} = 1 + 0.2223.\left(\frac{x}{L} - 1.5\right)^2 \quad 1.5 \leq \frac{x}{L} \leq 3.0$$

It should be assumed that the flow is purely supersonic. The inflow (beginning of computational domain) is located at x/L = 1.6; the inflow Mach number is M=1.2.

The flow in the nozzle should be simulated using the upwind method of 1st order. Before starting, pay attention to the formulation of physical and numerical boundary conditions. Use for this task the method of characteristics (wave propagation) as a guide. Give a brief discussion of the procedure.

Determine the stability limits and their current implementation.

The resulting distributions of density, velocity, total energy and pressure should be displayed in an appropriate way and discussed.

Bonus problem: repeat above problem using an implicit method.

Solution:

Numerical boundary conditions of supersonic wave:

Beginning of supersonic flow: x/L = 1.6

Given equation:

$$\frac{A}{A_t}=1+0.2223\cdot\left(\frac{x}{L}-1.5\right)^2 \qquad 1.5\leqslant\frac{x}{L}\leqslant 3.0$$

When x/L=1.6:

A/At =1.002223

When x/L=3

A/At =1.500175

Density:

$$\rho = \sqrt{\frac{P*M\_air}{R*T}}$$

Velocity of sound: $c=\sqrt{\frac{\kappa*R*T}{M_{air}}}$

κ=7/5 Hence c=347.140 m/s

Given Mach number = 1.2

Velocity of flow= (Mach number) *(velocity of sound)

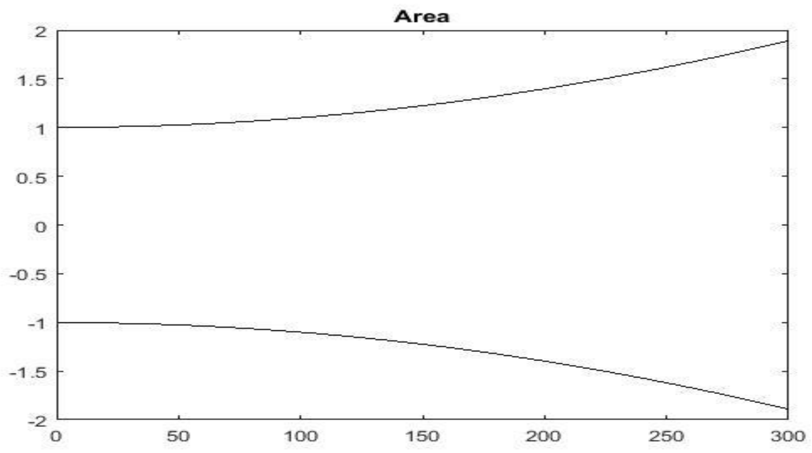
Velocity of flow=1.2*347.140=416.568m/s

MATLAB Code

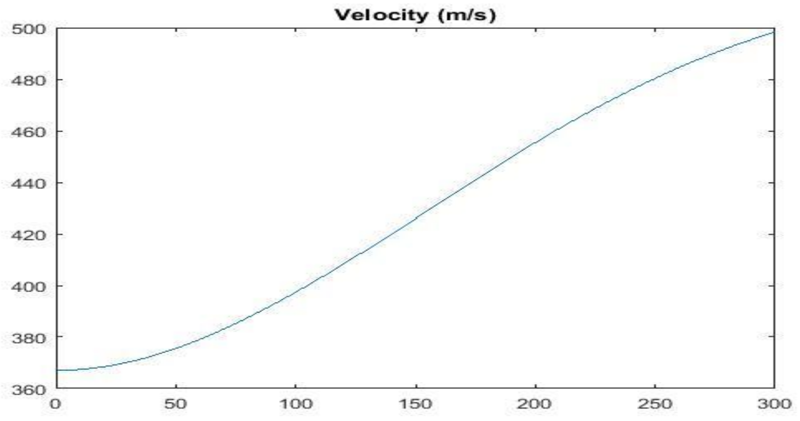
```
clear all
close all
M_a          =   28.976; %mass of air
Kappa        =   1.4;
Mach_no      =   1.2; %mach number
CFL          =   0.4;
P0           =   100000; %given pressure
T0           =   300;
T            =   0.77639751*T0;
R            =   8314.472; %universal gas constant
Im           =   300;
J            =   1:1:Im;
Tst          =   1000;
At           =   1.0022;
c_sound      =   sqrt(Kappa*R/M_a*T);
dx           =   2/Im;
dt           =   CFL*dx/c_sound;
A            =   ones(1,Im);

for i=1:Im;
 xdL         =   2*i/Im;
 A(i)        =   At*(1+0.2223*(xdL)^2);
 end
 p           =   ones(Tst,Im)*P0;
 rho         =   p/R/T*M_a;
 u           =  (ones(Tst,Im)*Mach_no*c_sound);
 e           =   ones(Tst,Im)*(p(1,1)/(Kappa-1)/rho(1,1)+0.5*(u(1,1))^2);
 for n=1:(Tst-1)
  for i=2:Im
 rho(n+1,i)      =    -dt/(dx*A(i))*((rho(n,i)*u(n,i)*A(i))-(rho(n,i-1)...,
                      *u(n,i-1)*A(i-1)))+rho(n,i);
 c               =    ((rho(n,i)*(u(n,i))^2 +p(n,i))*A(i))-((rho(n,i-1)...,
                      *(u(n,i-1))^2+p(n,i-1))*A(i-1));
 u(n+1,i)        =    1/rho(n+1,i)*(-dt/(dx*A(i))*(c-p(n,i)*(A(i)-A(i-1)))...,
                      +rho(n,i)*u(n,i));
 d               =    (e(n,i)+p(n,i))*u(n,i)*A(i)-(e(n,i-1)+p(n,i-1))*...,
                      u(n,i-1)*A(i-1);
 e(n+1,i)        =    (-dt/(dx*A(i))*d+rho(n,i)*e(n,i))/rho(n+1,i);
 p(n+1,i)        =    rho(n+1,i)*R*T/M_a;
 end
end

figure(1);plot(J,A,'k',J,-A,'k');title({'Area'});
figure(2);plot(u(Tst,:));title({'Velocity (m/s)'});
figure(3);plot(p(Tst,:));title({'Pressure (Pa)'});
figure(4);plot(e(Tst,:));title({'Energy (J)'});
figure(5);plot(rho(Tst,:));title({'Density(kg/m3'});
```

Matlab Results:

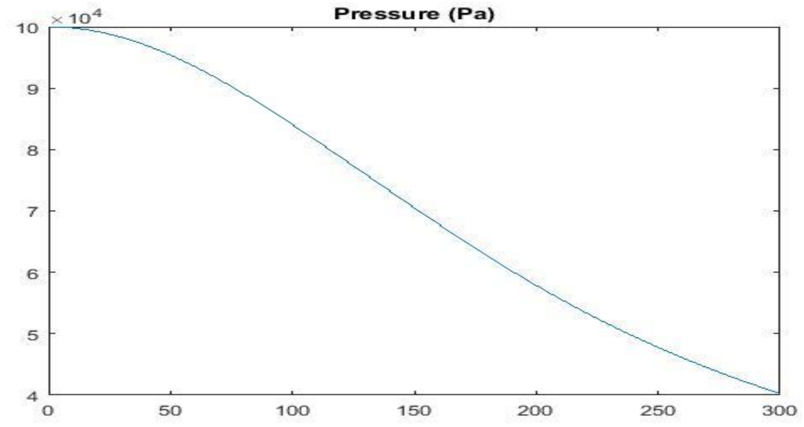
Area:



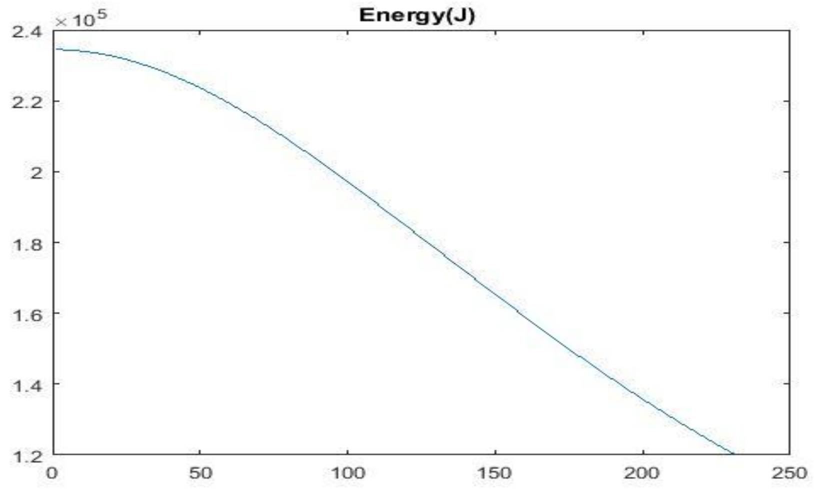
Velocity:



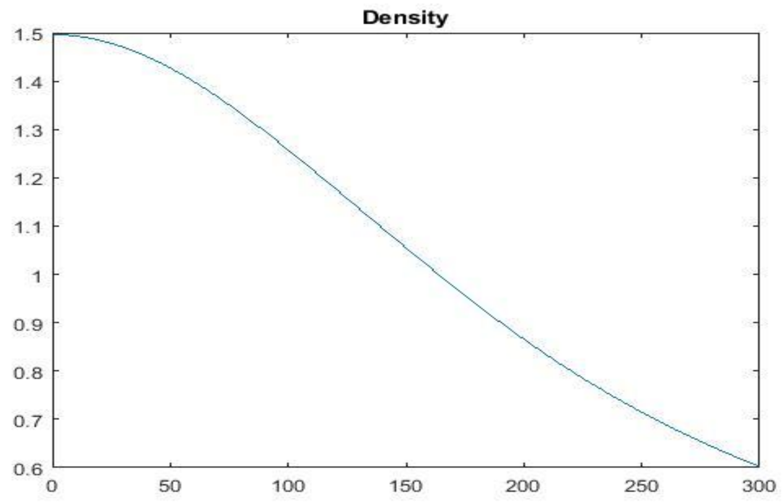
Pressure:



Energy:



Density:


Density