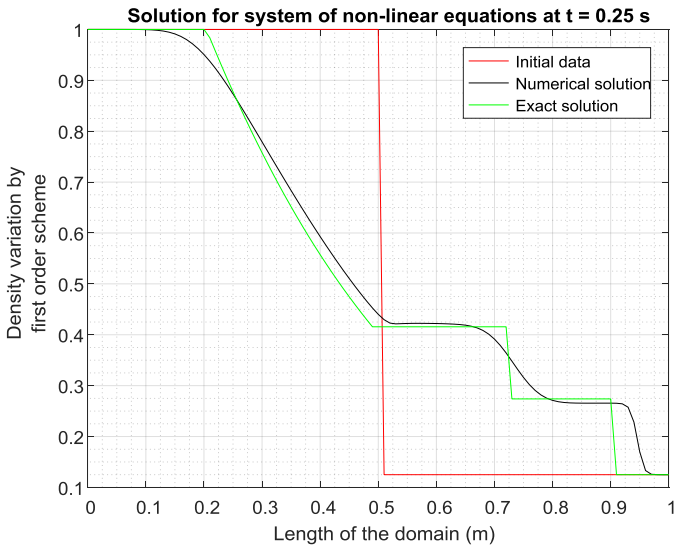
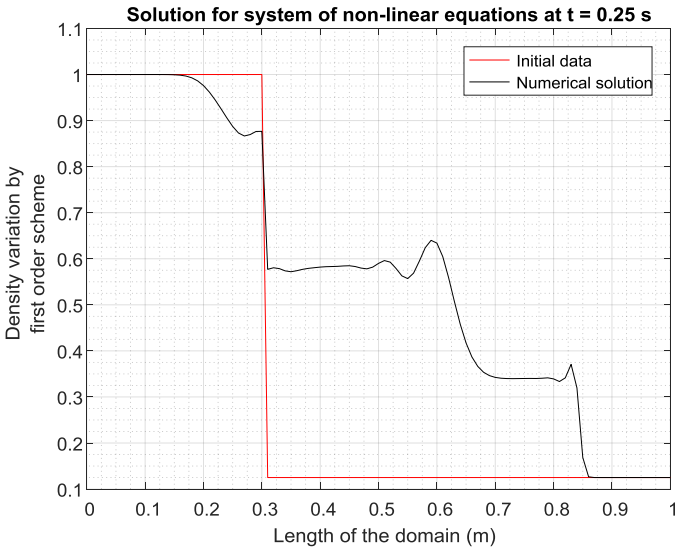
10/27/2018

ASSIGNMENT 3

NUMERICAL METHODS FOR CONSERVATION LAWS

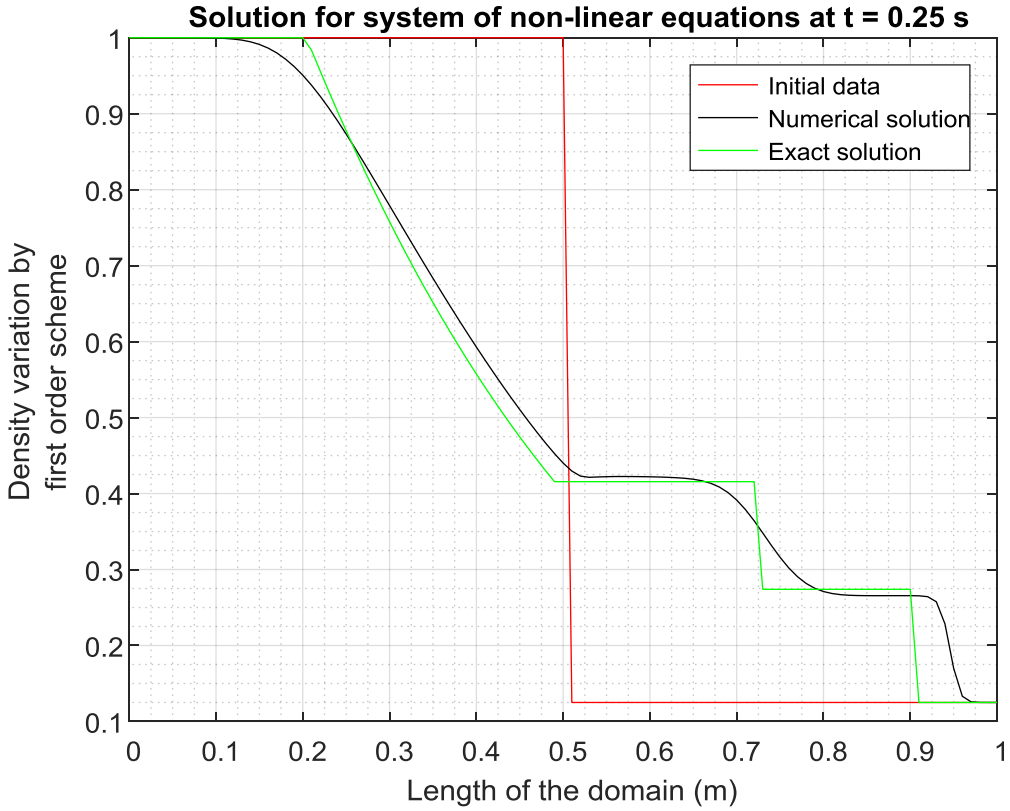


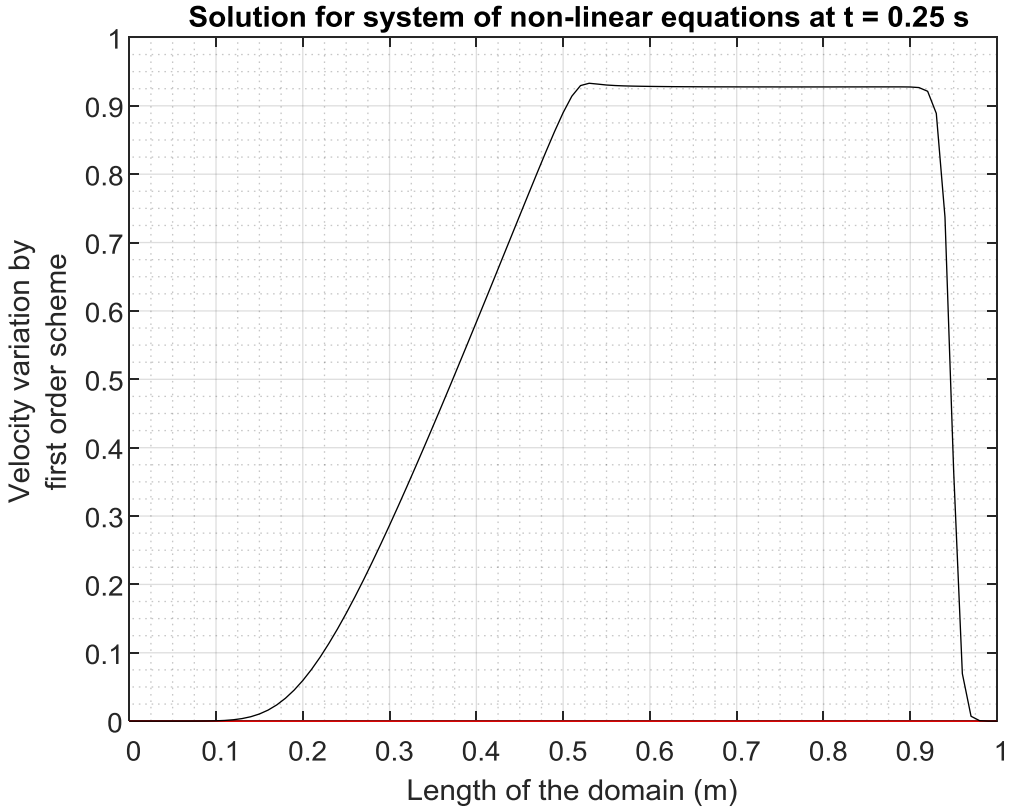


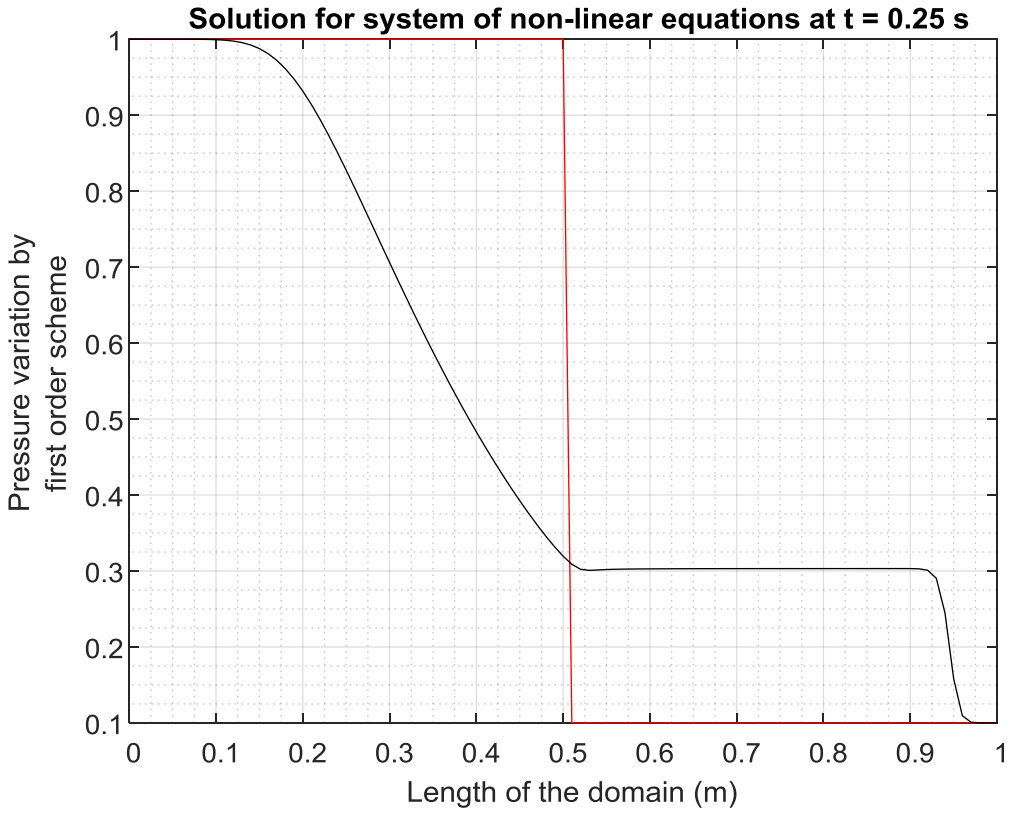
173109003 | Sanit Prashant Bhatkar

Problem 1

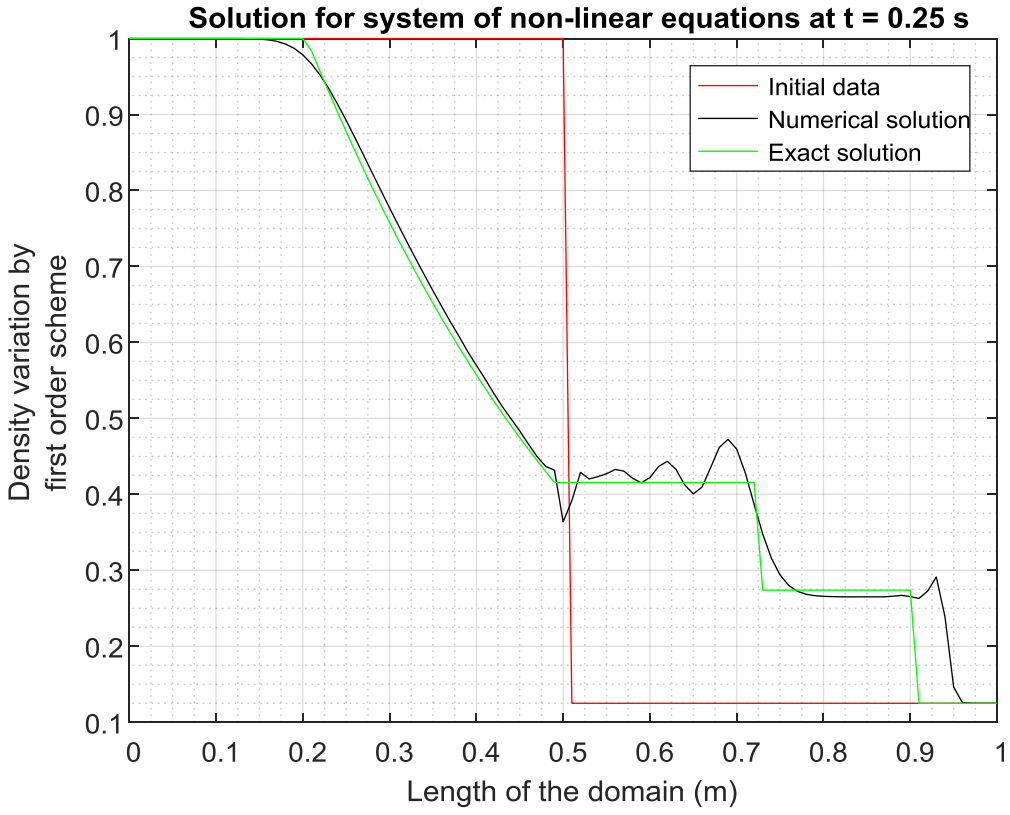
A. First order method with 101 points

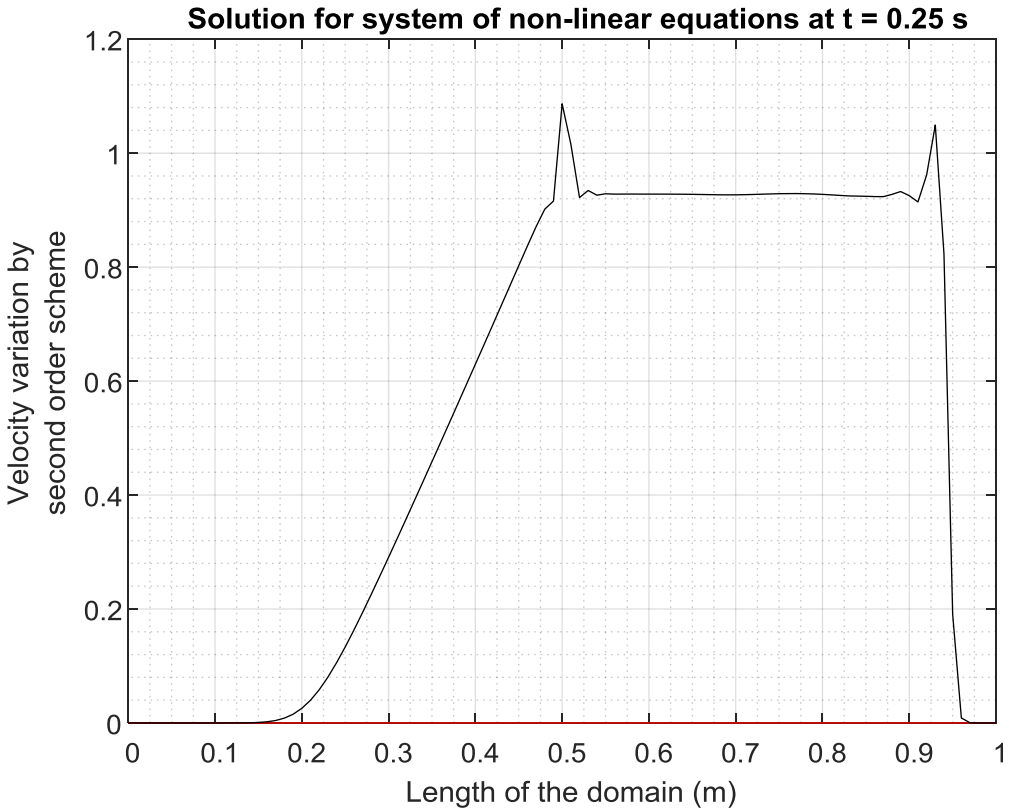


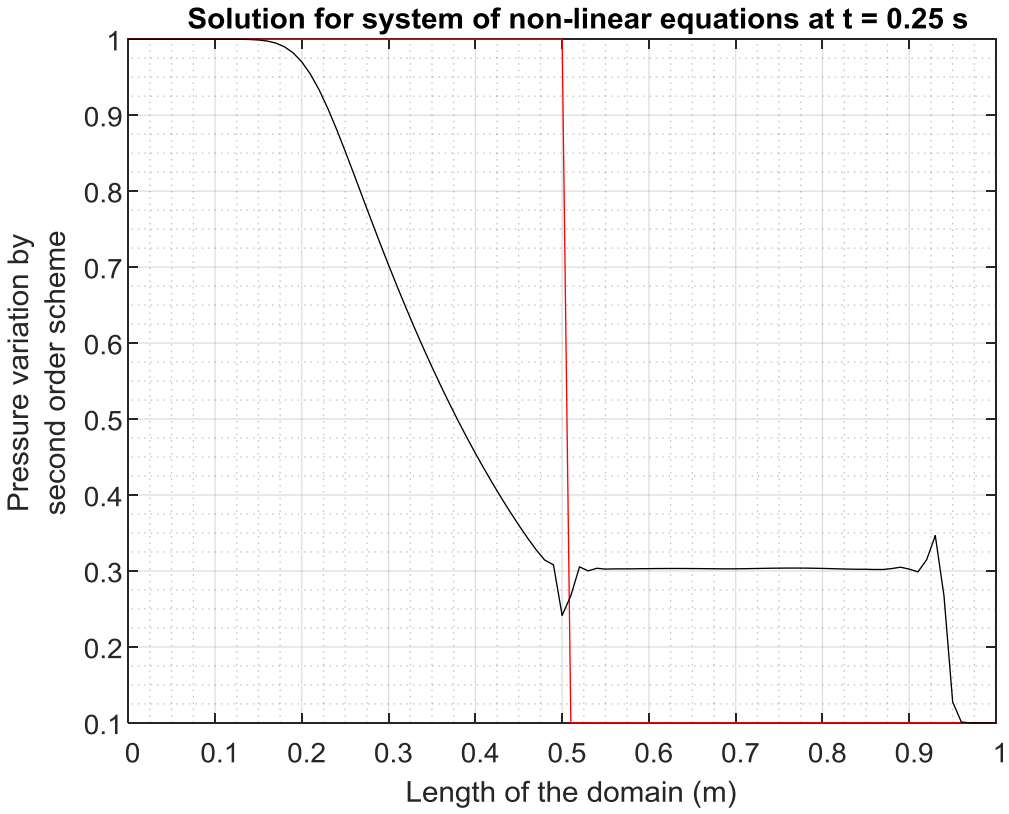




B. Second order method with 101 points

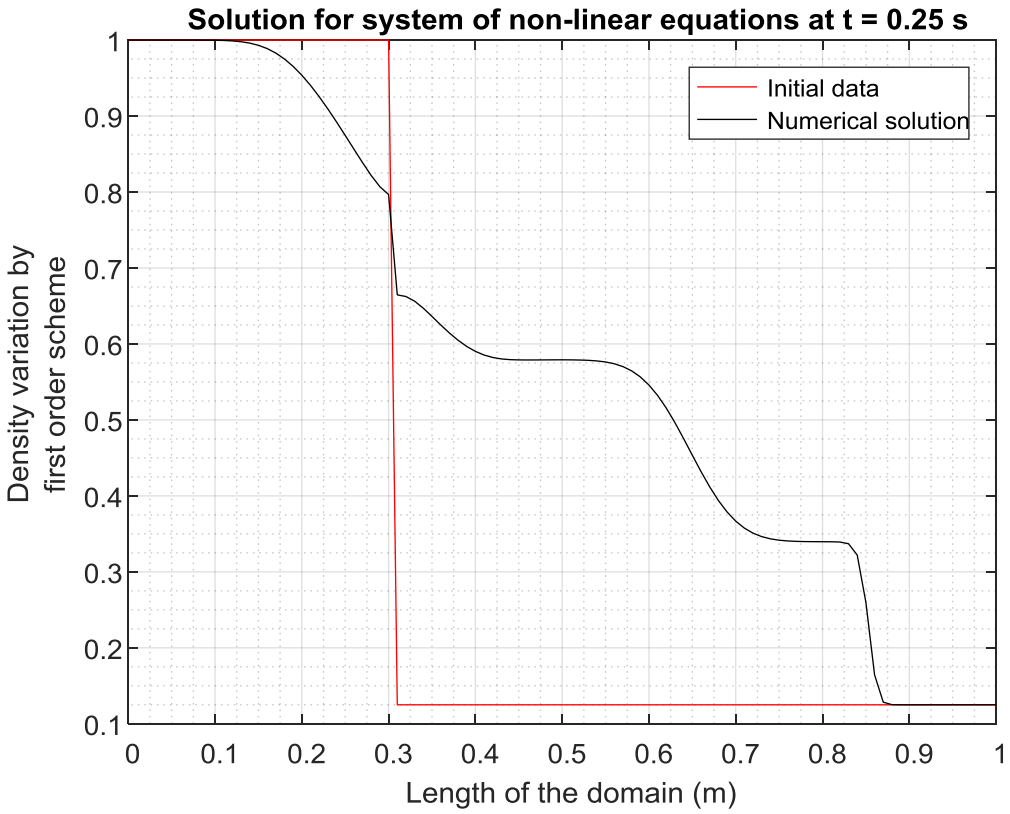


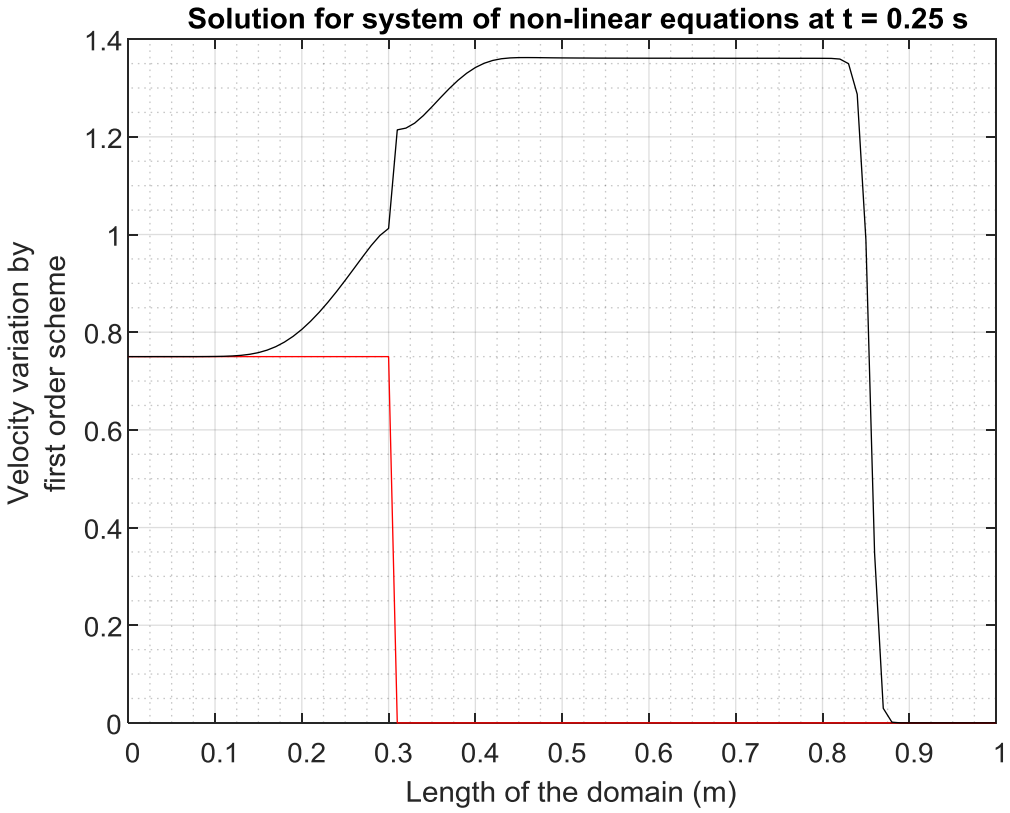

Solution for system of non-linear equations at t = 0.25 s
Velocity variation by second order scheme
Length of the domain (m)
0
0.2
0.4
0.6
0.8
1
1.2
0.1
0.3
0.5
0.7
0.9

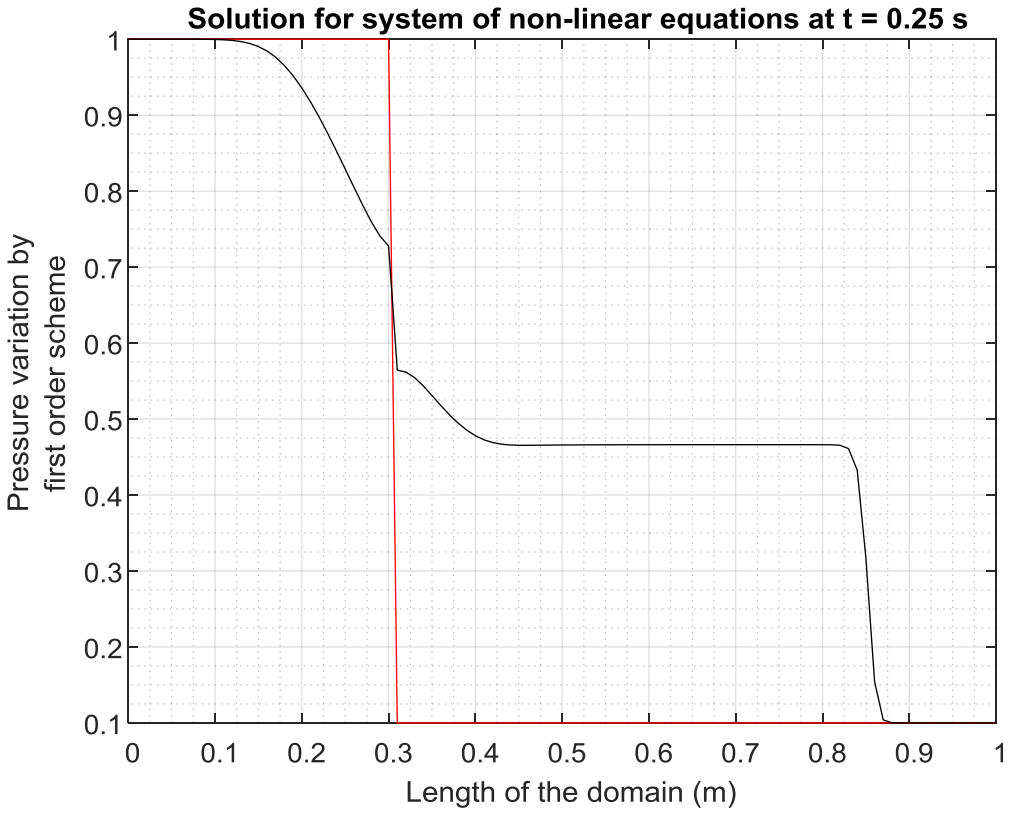

Solution for system of non-linear equations at t = 0.25 s
Pressure variation by second order scheme
Length of the domain (m)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

Problem 2

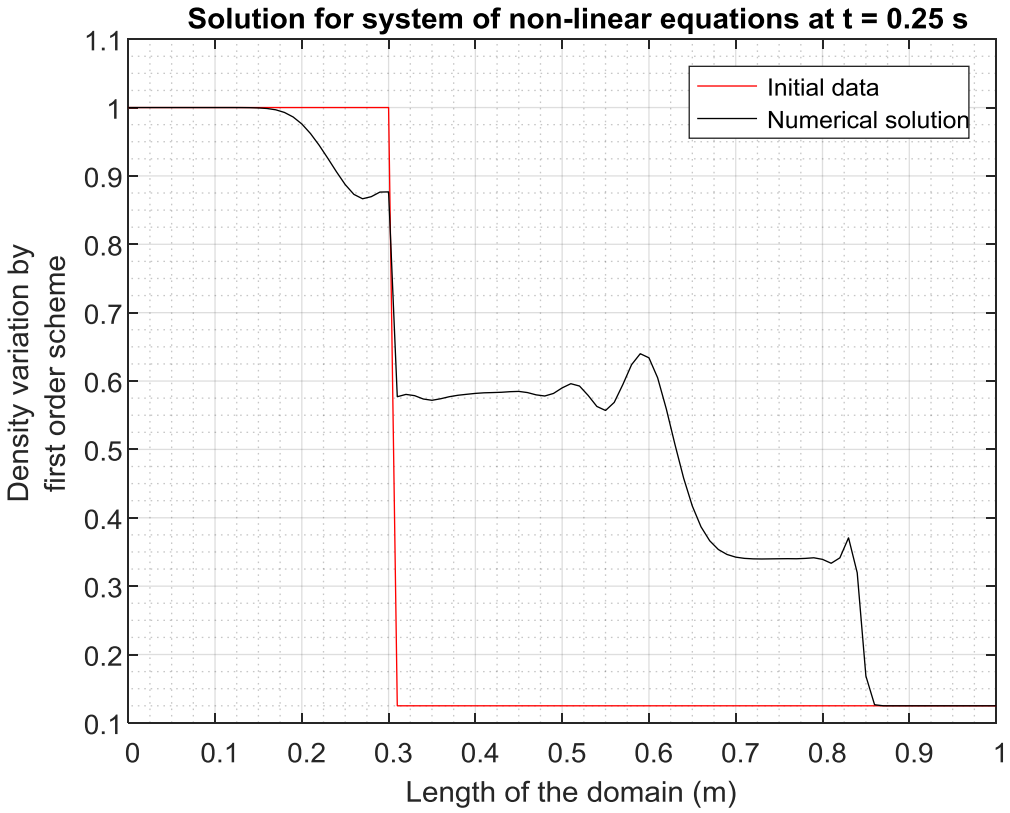
A. First order method with 101 points

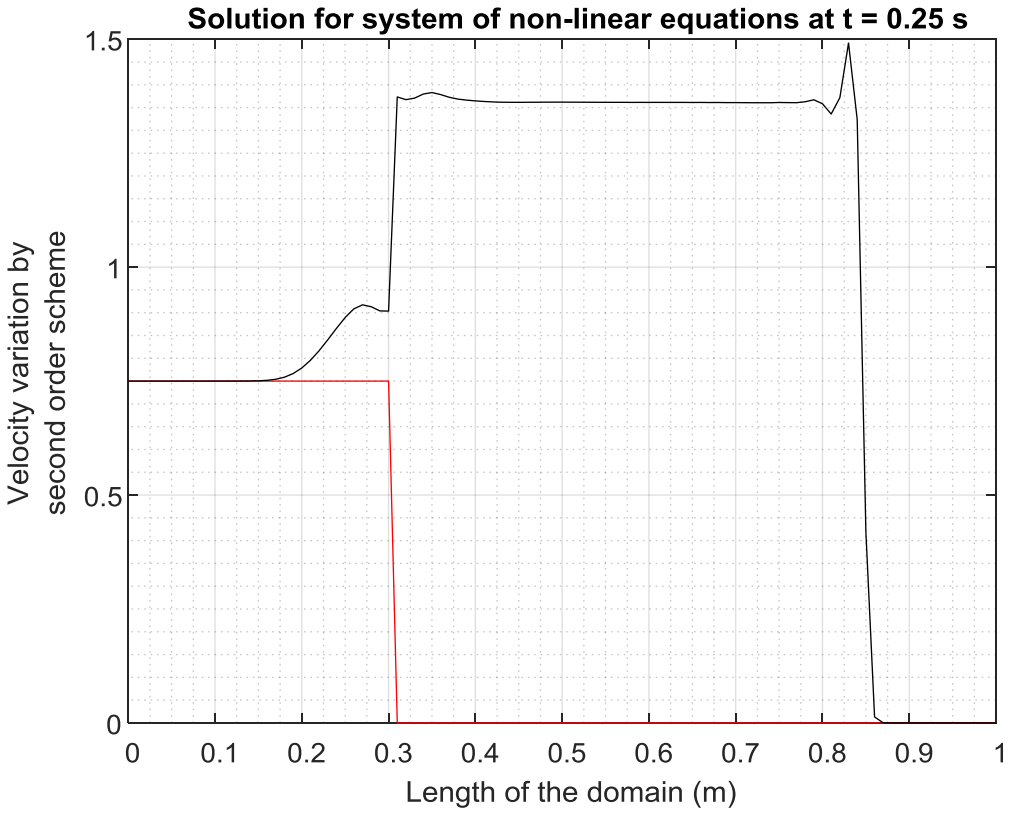


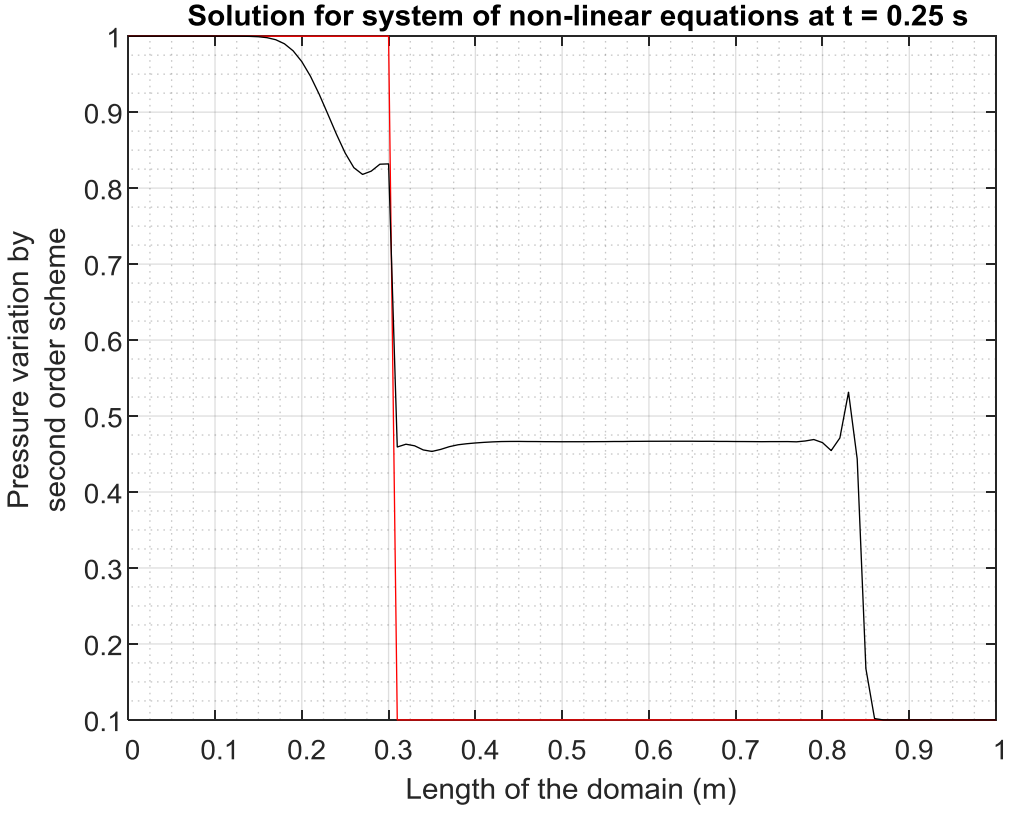




B. Second order method with 101 points




Solution for system of non-linear equations at t = 0.25 s
Velocity variation by second order scheme
Length of the domain (m)
0
0.5
1
1.5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9


Solution for system of non-linear equations at t = 0.25 s
Pressure variation by second order scheme
Length of the domain (m)
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

Safety factor taken = 0.8

$$\Delta t = \frac{\Delta x \times safety}{\left|\hat{\lambda}_p\right|_{j,\max}}$$

$$L_1 = dx \times \sum \left| exact - numerical \right|$$

$$L_2^2 = dx \times \sum \left| exact - numerical \right|^2$$

Table 1: Norm calculation for problem 1 by 1st order method

Number of points	L1 norm	L2 norm squared	L2 norm
101	0.0217	0.0012	0.0349
1001	0.0128	7.8472e-04	0.0280
10001	0.0114	8.1054e-04	0.0285

Table 2: Norm calculation for problem 1 by 2nd order method

Number of points	L1 norm	L2 norm squared	L2 norm
101	0.0174	0.0011	0.0330
1001	0.0130	9.4748e-04	0.0308
10001	0.0126	9.3919e-04	0.0306

Code of the problem

```
%============================================================%
% Assignment 3 Numerical Methods for Conservation Laws AE 617
%
% Assignment Number 3. Inviscid Euler Equations
%
% AUTHOR:
% Sanit P. Bhatkar (173109003@iitb.ac.in)
% Roll No: 173109003
% Place: IIT BOMBAY.
%============================================================%

clc;
clearvars;
fprintf('\nInviscid Euler Equations');



%% Grid definition

prbno=input('\n\nProblem number: ');
xn=input('\n\nLocation of discontinuity: ');
n=input('Input odd number of points: ');
fprintf('\nChoose scheme\n1.First Order   2.Second Order');
scno=input('\n\nScheme Number: ');
safety=input('\nsafety:  ');
dx=1/(n-1);
x=0:dx:1;
gamma=1.4;
tf=0.25;

%% Data initialization

U =zeros(3,n);

%Eigen value
lamda1= zeros(1,n);
lamda2= zeros(1,n);
lamda3= zeros(1,n);

%Eigen vectors
R1= zeros(3,n);
R2= zeros(3,n);
R3= zeros(3,n);

% dU= ap*R
ap1= zeros(1,n-1);
ap2= zeros(1,n-1);
ap3= zeros(1,n-1);

t=0;


%% Initial Discontinuity data

%Location of discontinuity
dn=find(x==xn);

%Density matrix
```

```
rho=zeros(1,n);
rho(1,1:dn)=1;
rho(1,dn+1:end)=0.125;
rho_ini=rho;

%Velocity matrix
u=zeros(1,n);

if prbno==1
u(1,1:dn)=0;
else
u(1,1:dn)=0.75;
end

u(1,dn+1:end)=0;
u_ini=u;

%Pressure matrix
p=zeros(1,n);
p(1,1:dn)=1;
p(1,dn+1:end)=0.1;
p_ini=p;


%% Initial Energy and Enthalpy calculation

for i=1:n

    %Energy calculation
    E(1,i)=p(1,i)/(gamma-1)+(rho(1,i)*u(1,i)^2)/2;

    %Specific enthalpy calculation
    H(1,i)=(p(1,i)+E(1,i))/(rho(1,i));

    U(1,i) = rho(1,i);
    U(2,i) = rho(1,i)*u(1,i);
    U(3,i) = E(1,i);

end


%% Analytic solution

%to solve Sod's Shock Tube problem
%reference: "http://www.phys.lsu.edu/~tohline/PHYS7412/sod.html"
%   |       |   |     |          |
%   |       |   |     |          |
%   |       |   |     |          |
%___|_______|___|_____|__________|________________
%   x1      x2  x0    x3         x4
%
%input require: t (time)

% x0: Location of the initial membrane;
% x1: Head of the rarefaction wave that is moving to the left;
% x2: (To be discussed, below);
% x3: Location of the contact discontinuity that separates the "left" fluid
from the "right";
% x4: Location of the shock front that is moving to the right.
```

```
x0=xn;
pl=1;
pr=0.1;
rhol=1;
rhor=0.125;
ul=0;
ur=0;

cl=sqrt(gamma*pl/rhol);
cr=sqrt(gamma*pr/rhor);

rho_analytic=zeros(1,n);
rho_analytic(1,1:end)=rhor;

%% Calculated value

ppost=0.3190;
vpost=0.89095;
rhopost=0.27394;
vshock=1.6387;


%% Key positions and velocities

x1=x0-cl*tf;
if xn==0.5
x2=0.496;
else
x2=0.31;
end
x3=x0+vpost*tf;
x4=x0+vshock*tf;

%% Density analytic

dx=1/(n-1);
idx=1;

while x(idx)<=x1

    rho_analytic(1,idx)=rhol;
    idx=idx+1;
end

%% Expansion fan region

alpha=(gamma-1)/(gamma+1);

while x(idx)<=x2

    cs=alpha*abs(x(idx)-x0)/tf+(1-alpha)*cl;
    rho_analytic(1,idx)=rhol*(cs/cl)^(2/(gamma-1));
    idx=idx+1;
end

rhomiddle=rho_analytic(1,idx-1);

%% Middle contact discontinuity region
```

```
while x(idx)<=x3

    rho_analytic(1,idx)=rhomiddle;
    idx=idx+1;
end

%% Shock region

while x(idx)<=x4

    rho_analytic(1,idx)=rhopost;
    idx=idx+1;
end



%% First order upwind scheme

if scno == 1

while t <= tf

dum3 = 0;

for i=1:1:(n-1)

%% Roe linearization

uc = (sqrt(rho(1,i))*u(1,i) + sqrt(rho(1,i+1))*u(1,i+1)) / (sqrt(rho(1,i)) +
sqrt(rho(1,i+1)));
Hc = (sqrt(rho(1,i))*H(1,i) + sqrt(rho(1,i+1))*H(1,i+1)) / (sqrt(rho(1,i)) +
sqrt(rho(1,i+1)));
ac = sqrt( (gamma-1)*(Hc - 0.5*uc*uc) );

%% Lamda max calculation

lamda1(1,i) = uc - ac;
lamda2(1,i) = uc;
lamda3(1,i) = uc + ac;

dum1 = max(abs(lamda1(1,i)), abs(lamda2(1,i)));
dum2 = max(dum1, abs(lamda3(1,i)));
lmax = max(dum2, dum3);
dum3 = lmax;

 %% Eigen vector calculation

R1(1,i) = 1;
R1(2,i) = uc-ac;
R1(3,i) = Hc-uc*ac;

R2(1,i) = 1;
R2(2,i) = uc;
R2(3,i) = uc*uc*0.5;

R3(1,i) = 1;
R3(2,i) = uc+ac;
R3(3,i) = Hc+uc*ac;
```

```
%% Ap calculation from Leveque

ap2(1,i) =(gamma-1)*((Hc-uc*uc)*(U(1,i+1)-U(1,i))+uc*(U(2,i+1)-U(2,i))-
(U(3,i+1)-U(3,i)))/(ac*ac);
ap3(1,i) =((U(2,i+1)-U(2,i))+(ac-uc)*(U(1,i+1)-U(1,i))-ac*ap2(1,i))/(2*ac);
ap1(1,i) =((U(1,i+1)-U(1,i))-ap2(1,i)-ap3(1,i));

end

 %% Time step definition

dt = dx * safety/lmax;

 %% Inner node formulation

for i=2:1:(n-1)

% when lamb > 0
df1p = max(0,sign(lamda1(1,i-1)))*(lamda1(1,i-1)*ap1(1,i-1)*R1(1,i-1)) +
max(0,sign(lamda2(1,i-1)))*(lamda2(1,i-1)*ap2(1,i-1)*R2(1,i-1)) +
max(0,sign(lamda3(1,i-1)))*(lamda3(1,i-1)*ap3(1,i-1)*R3(1,i-1));
df2p = max(0,sign(lamda1(1,i-1)))*(lamda1(1,i-1)*ap1(1,i-1)*R1(2,i-1)) +
max(0,sign(lamda2(1,i-1)))*(lamda2(1,i-1)*ap2(1,i-1)*R2(2,i-1)) +
max(0,sign(lamda3(1,i-1)))*(lamda3(1,i-1)*ap3(1,i-1)*R3(2,i-1));
df3p = max(0,sign(lamda1(1,i-1)))*(lamda1(1,i-1)*ap1(1,i-1)*R1(3,i-1)) +
max(0,sign(lamda2(1,i-1)))*(lamda2(1,i-1)*ap2(1,i-1)*R2(3,i-1)) +
max(0,sign(lamda3(1,i-1)))*(lamda3(1,i-1)*ap3(1,i-1)*R3(3,i-1));

% when lamb < 0
df1m  = abs(min(0,sign(lamda1(1,i))))*(lamda1(1,i)*ap1(1,i)*R1(1,i)) +
abs(min(0,sign(lamda2(1,i))))*(lamda2(1,i)*ap2(1,i)*R2(1,i)) +
abs(min(0,sign(lamda3(1,i))))*(lamda3(1,i)*ap3(1,i)*R3(1,i));
df2m  = abs(min(0,sign(lamda1(1,i))))*(lamda1(1,i)*ap1(1,i)*R1(2,i)) +
abs(min(0,sign(lamda2(1,i))))*(lamda2(1,i)*ap2(1,i)*R2(2,i)) +
abs(min(0,sign(lamda3(1,i))))*(lamda3(1,i)*ap3(1,i)*R3(2,i));
df3m  = abs(min(0,sign(lamda1(1,i))))*(lamda1(1,i)*ap1(1,i)*R1(3,i)) +
abs(min(0,sign(lamda2(1,i))))*(lamda2(1,i)*ap2(1,i)*R2(3,i)) +
abs(min(0,sign(lamda3(1,i))))*(lamda3(1,i)*ap3(1,i)*R3(3,i));


U(1,i)= U(1,i)-(dt/dx)*(df1p+df1m );
U(2,i)= U(2,i)-(dt/dx)*(df2p+df2m );
U(3,i)= U(3,i)-(dt/dx)*(df3p+df3m );

end

 %% Value update

t=t+dt;

for i=2:1:(n-1)

%Density matrix
rho(1,i)=U(1,i);
%Velocity matrix
u(1,i)=U(2,i)/U(1,i);
%Energy matrix
E(1,i)= U(3,i);
%Pressure matrix
p(1,i)=(E(1,i)-0.5*rho(1,i)*u(1,i)*u(1,i))*(gamma-1);
```

```
%Enthaply matrix
H(1,i)=(E(1,i)+p(1,i))/rho(1,i);

end

end


%% Output 1o

%Density plot

figure (1)
plot(x,rho_ini,'-r');
hold on
plot(x,rho,'-k');
hold on
if prbno==1
plot(x,rho_analytic,'-g');
end
hold off
xlabel('Length of the domain (m)');
ylabel({'Density variation by';'first order scheme'});
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);
if prbno==1
legend('Initial data','Numerical solution','Exact solution');
else
legend('Initial data','Numerical solution');
end

set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on

%Velcocity plot

figure (2)
plot(x,u_ini,'-r');
hold on
plot(x,u,'-k');
hold off
xlabel('Length of the domain (m)');
ylabel({'Velocity variation by';'first order scheme'});
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);

set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on

%Pressure plot

figure (3)
plot(x,p_ini,'-r');
hold on
plot(x,p,'-k');
hold off
xlabel('Length of the domain (m)');
ylabel({'Pressure variation by';'first order scheme'});
```

```
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);

set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on

end



%% Second order Lax scheme

%Courant number calculation
v1 =zeros(1,n-1);
v2 =zeros(1,n-1);
v3 =zeros(1,n-1);


if scno == 2

 t = 0;

while t <= tf

dum3 = 0;
for i=1:1:(n-1)

uc = (sqrt(rho(1,i))*u(1,i) + sqrt(rho(1,i+1))*u(1,i+1)) / (sqrt(rho(1,i)) +
sqrt(rho(1,i+1)));
Hc = (sqrt(rho(1,i))*H(1,i) + sqrt(rho(1,i+1))*H(1,i+1)) / (sqrt(rho(1,i)) +
sqrt(rho(1,i+1)));
ac = sqrt((gamma-1)*(Hc-uc*uc*0.5));

%% Eigen value

lamda1(1,i) = uc-ac;
lamda2(1,i) = uc;
lamda3(1,i) = uc+ac;

%% max lamda calculation

dum1 = max(abs(lamda1(1,i)), abs(lamda2(1,i)));
dum2 = max(dum1, abs(lamda3(1,i)));
lmax = max(dum2, dum3);
dum3 = lmax;

%% Eigen vector calculation

R1(1,i) = 1;
R1(2,i) = uc-ac;
R1(3,i) = Hc-uc*ac;

R2(1,i) = 1;
R2(2,i) = uc;
R2(3,i) = uc*uc*0.5;

R3(1,i) = 1;
R3(2,i) = uc+ac;
R3(3,i) = Hc+uc*ac;
```

```
%% Ap calculation from Leveque

ap2(1,i) = (gamma-1)*((Hc-uc*uc)*(U(1,i+1)-U(1,i))+uc*(U(2,i+1)-U(2,i))-
(U(3,i+1)-U(3,i)))/(ac*ac);
ap3(1,i) = ((U(2,i+1)-U(2,i))+(ac-uc)*(U(1,i+1)-U(1,i))-ac*ap2(1,i))/(2*ac);
ap1(1,i) = ((U(1,i+1)-U(1,i))-ap2(1,i)-ap3(1,i));

end

% Time step calculation
dt = dx * safety/lmax;

for i=1:1:(n-1)
v1(1,i) = lamda1(1,i) * dt/dx;
v2(1,i) = lamda2(1,i) * dt/dx;
v3(1,i) = lamda3(1,i) * dt/dx;
end


for i=2:1:(n-1)

% when lamb > 0
df1p = (1+v1(1,i-1))*(lamda1(1,i-1)*ap1(1,i-1)*R1(1,i-1)) + (1+v2(1,i-
1))*(lamda2(1,i-1)*ap2(1,i-1)*R2(1,i-1)) + (1+v3(1,i-1))*(lamda3(1,i-
1)*ap3(1,i-1)*R3(1,i-1));
df2p = (1+v1(1,i-1))*(lamda1(1,i-1)*ap1(1,i-1)*R1(2,i-1)) + (1+v2(1,i-
1))*(lamda2(1,i-1)*ap2(1,i-1)*R2(2,i-1)) + (1+v3(1,i-1))*(lamda3(1,i-
1)*ap3(1,i-1)*R3(2,i-1));
df3p = (1+v1(1,i-1))*(lamda1(1,i-1)*ap1(1,i-1)*R1(3,i-1)) + (1+v2(1,i-
1))*(lamda2(1,i-1)*ap2(1,i-1)*R2(3,i-1)) + (1+v3(1,i-1))*(lamda3(1,i-
1)*ap3(1,i-1)*R3(3,i-1));

% when lamb < 0
df1m  = (1-v1(1,i))*(lamda1(1,i)*ap1(1,i)*R1(1,i)) + (1-
v2(1,i))*(lamda2(1,i)*ap2(1,i)*R2(1,i)) + (1-
v3(1,i))*(lamda3(1,i)*ap3(1,i)*R3(1,i));
df2m = (1-v1(1,i))*(lamda1(1,i)*ap1(1,i)*R1(2,i)) + (1-
v2(1,i))*(lamda2(1,i)*ap2(1,i)*R2(2,i)) + (1-
v3(1,i))*(lamda3(1,i)*ap3(1,i)*R3(2,i));
df3m = (1-v1(1,i))*(lamda1(1,i)*ap1(1,i)*R1(3,i)) + (1-
v2(1,i))*(lamda2(1,i)*ap2(1,i)*R2(3,i)) + (1-
v3(1,i))*(lamda3(1,i)*ap3(1,i)*R3(3,i));


U(1,i) = U(1,i) - 0.5*(dt/dx)* ( df1p + df1m );
U(2,i) = U(2,i) - 0.5*(dt/dx)* ( df2p + df2m );
U(3,i) = U(3,i) - 0.5*(dt/dx)* ( df3p + df3m );

end



t=t+dt;

for i=2:1:(n-1)

%Density matrix
rho(1,i)= U(1,i);
%Velocity matrix
```

```
u(1,i)= U(2,i)/U(1,i);
%Energy matrix
E(1,i)= U(3,i);
%Pressure matrix
p(1,i)= (E(1,i)- 0.5*rho(1,i)*u(1,i)*u(1,i))*(gamma-1);
%Enthaply matrix
H(1,i)= (E(1,i)+p(1,i) )/rho(1,i);

end

end



%% Output 2o

%Density plot

figure (1)
plot(x,rho_ini,'-r');
hold on
plot(x,rho,'-k');
hold on
if prbno==1
plot(x,rho_analytic,'-g');
end
hold off
xlabel('Length of the domain (m)');
ylabel({'Density variation by';'first order scheme'});
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);
if prbno==1
legend('Initial data','Numerical solution','Exact solution');
else
legend('Initial data','Numerical solution');
end

set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on

%Velcocity plot

figure (2)
plot(x,u_ini,'-r');
hold on
plot(x,u,'-k');
hold off
xlabel('Length of the domain (m)');
ylabel({'Velocity variation by';'second order scheme'});
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);


set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on

%Pressure plot
```

```
figure (3)
plot(x,p_ini,'-r');
hold on
plot(x,p,'-k');
hold off
xlabel('Length of the domain (m)');
ylabel({'Pressure variation by';'second order scheme'});
title(['Solution for system of non-linear equations at t = ',num2str(tf),'
s']);

set(gca,'XMinorGrid','on');
set(gca,'YMinorGrid','on');
grid on


end

if prbno==1
%% Norm calculation

fprintf('\n----Output---');

%L1 norm calculation
Norm_1=abs(rho_analytic-rho);
Norm_1=dx*sum(Norm_1)

%L2 norm calcultion
Norm_2=(rho_analytic-rho).^2;
Norm_2=dx*sum(Norm_2);
Norm_2_square=Norm_2
Norm_2=sqrt(Norm_2)
end
```