MAE 3723 Matlab Exam Spring 2018

Type your name here: Diego Colón

Type your Student ID here:

***Instructions***

For each problem, create and run one or more Matlab programs to solve the problem.

To submit your solution, for each problem, paste the text of your programs into ***this*** Word document, and then paste into ***this*** Word document the Matlab results produced by your program (from the Workspace and/or Plots). Past your solution for each problem immediately after the problem statement.

You may also type in any comments you think are useful, into the word document (please type all such comments in RED).

After you have completed the exam, you must upload this ***single*** file containing your solutions for the entire exam, to the D2L drop box created for the exam. Note: you may upload this file multiple times to the D2L drop box (as a way to keep safe backup copies of your work). If you do upload multiple copies, only the ***LAST*** file uploaded will be graded. Earlier versions will be ignored!

As a safety precaution you may also e-mail your final solution file to yourself, and

maeonline@okstate.edu

PLEASE SAVE THIS FILE OFTEN (***and to different file names***) DURING THE EXAM, TO AVOID LOSING YOUR WORK BECAUSE OF AN ACCIDENT, OR A PROGRAM CRASH!

Rules for the exam:

You may use any Matlab or other text-like files contained on your computer hard drive, or contained on a “thumb drive” that you bring to the exam. You may NOT download Matlab files from the web, search the web for help, or use the web to communicate with anyone during the exam.

You may use a calculator, pen and paper if you find it helpful. You will not be able to turn-in your handwritten material.

There are 5 problems on this exam!

1. Paste your solution material for problem 1 after this line.

function SolveODEs()

clf %clear any existing plots

% State defined as X = [x,xdot, p1,p2]

[t,y1] = ode45( @deriv,[0,0.05],[0,0,1e5,1e5]);

plot(t,y1(:,2),'b');

title('Velocity of mass');

xlabel('Time - [s]');

ylabel('Velocity of piston- [m/s]');

pause

plot(t,y1(:,3),'r',t,y1(:,4),'b');

title('Pressure in chamber 1 and 2');

xlabel('Time - [s]');

ylabel('Pressure - [Pa]');

legend('p1','p2')

pause % hit enter to go to the next plot

function XDOT = deriv(t,X)

% System Parameters

A =4.909e-4; Cd = 0.6; rho= 850;

V = 1.473e-4; beta = 2e9; pa=1e5;

m = 30; ps =1.4e7; Kv = 2e-5;

% Rename states

x = X(1); xdot = X(2); p1 = X(3); p2 = X(4);

% Initiate forcing function

y = 0.002;

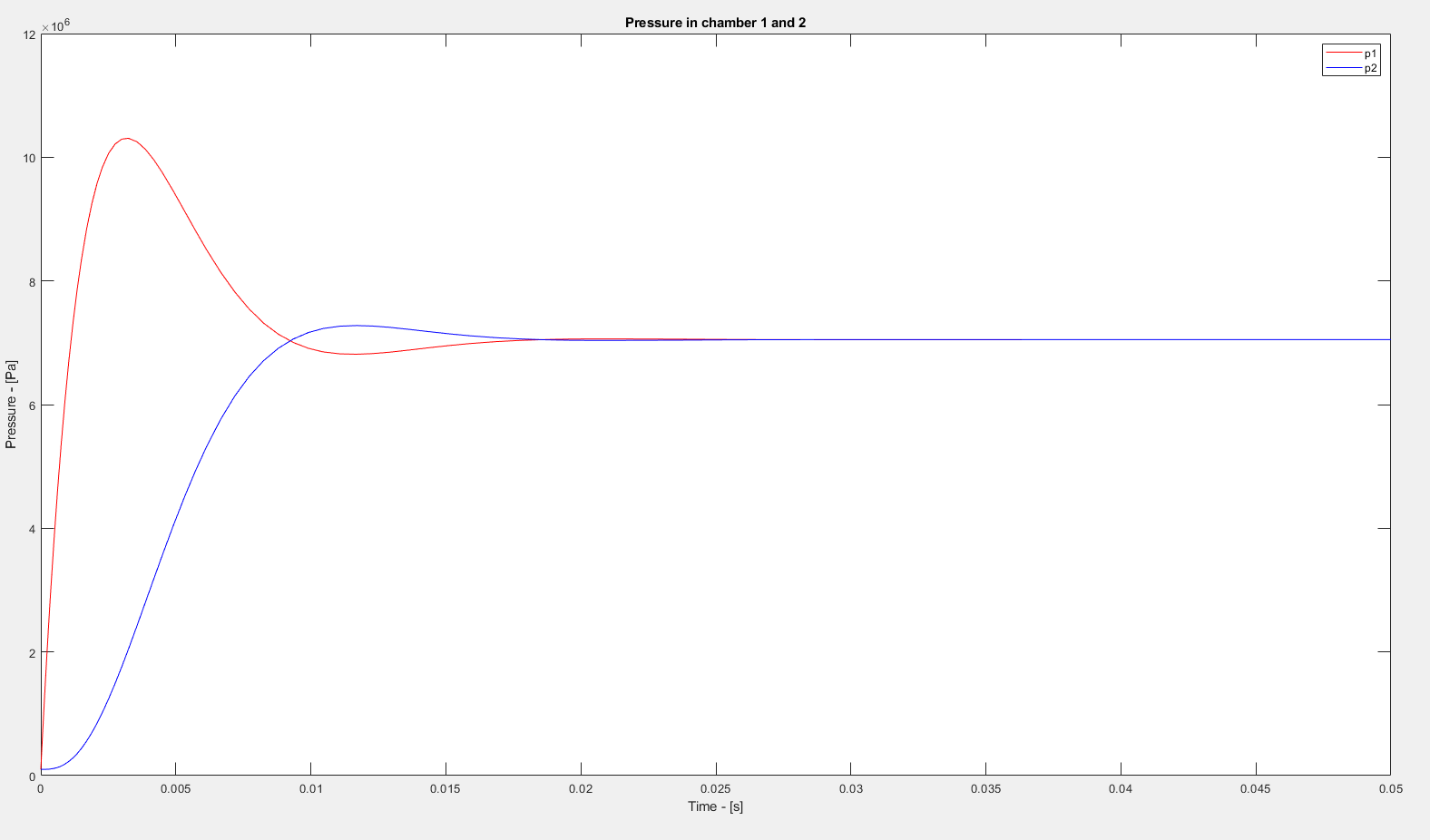
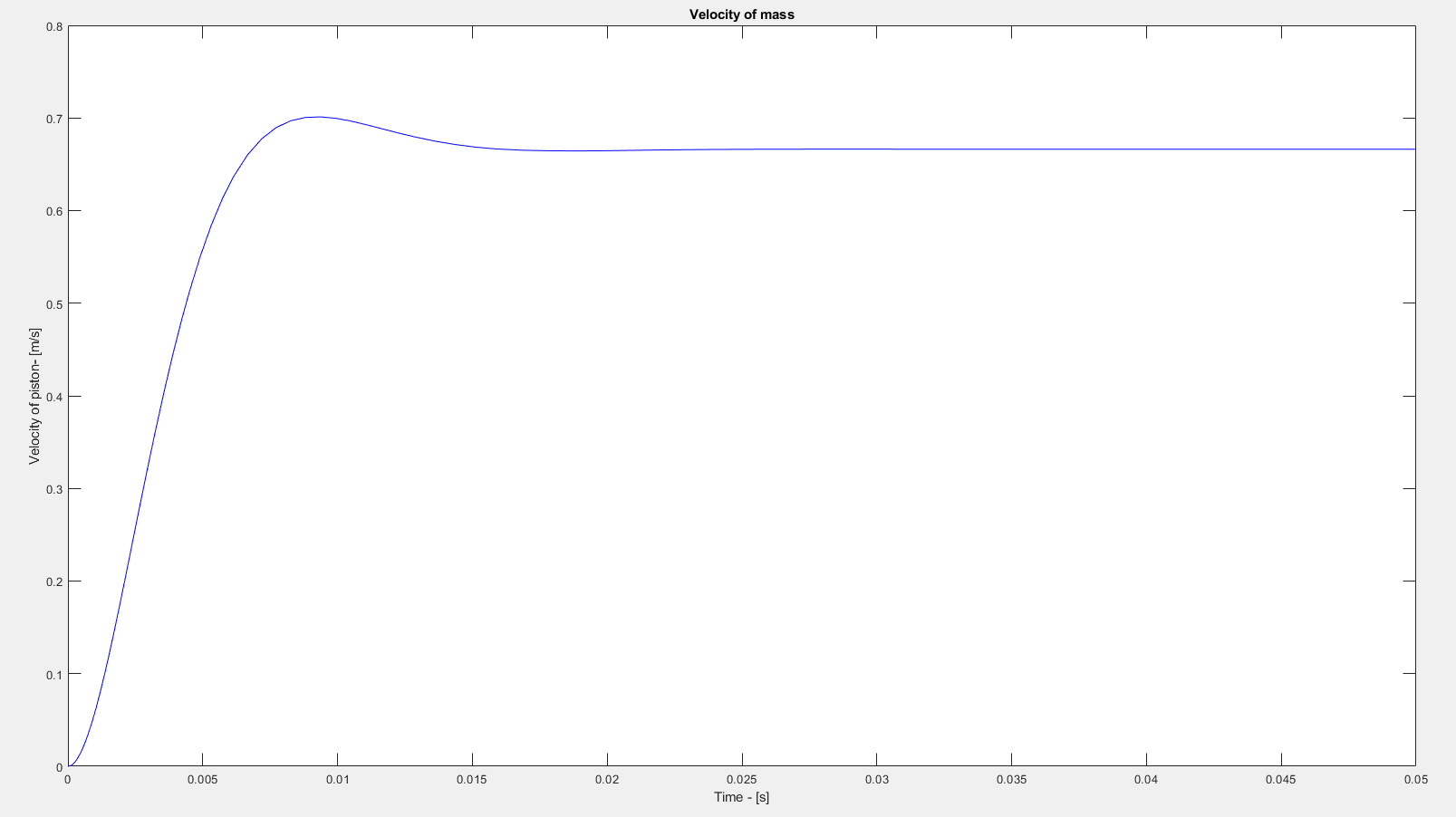
% write the non-trivial equations using nice names

xddot = A\*(p1-p2)/m;

p1dot = (y\*Kv\*(ps-p1)-rho\*A\*xdot)\*((beta)/(V\*rho));

p2dot = (y\*Kv\*(p2-pa)-rho\*A\*xdot)\*((-beta)/(V\*rho));

XDOT = [ xdot; xddot; p1dot; p2dot] ; %return the derivative values

**

1. Paste your solution material for problem 2 after this line.

function SolveODEs()

clf %clear any existing plots

global Kv

% State defined as X = [x, xdot, p1, p2]

Kv = 1e-5;

[t1,y1] = ode45(@deriv,[0,0.1],[0,0,1e5,1e5]); % derivative, time range, initial conditions

Kv = 2e-5;

[t2,y2] = ode45(@deriv,[0,0.1],[0,0,1e5,1e5]);

Kv = 10e-5;

[t3,y3] = ode45(@deriv,[0,0.1],[0,0,1e5,1e5]);

Kv = 20e-5;

[t4,y4] = ode45(@deriv,[0,0.1],[0,0,1e5,1e5]);

plot(t1,y1(:,3),'r',t2,y2(:,3),'g',t3,y3(:,3),'b',t4,y4(:,3),'y');

title('Pressure in chamber 1');

xlabel('Time - [s]');

ylabel('Pressure 1 - [Pa]');

legend('Kvalve = 1e-5','Kvalve = 2e-5','Kvalve = 10e-5','Kvalve = 20e-5')

pause

function XDOT = deriv(t,X)

% System Parameters

A =4.909e-4; Cd = 0.6; rho= 850;

V = 1.473e-4; beta = 2e9; pa=1e5;

m = 30; ps =1.4e7; global Kv

% Rename states

x = X(1); xdot = X(2); p1 = X(3); p2 = X(4);

% Initiate forcing function

y = 0.002;

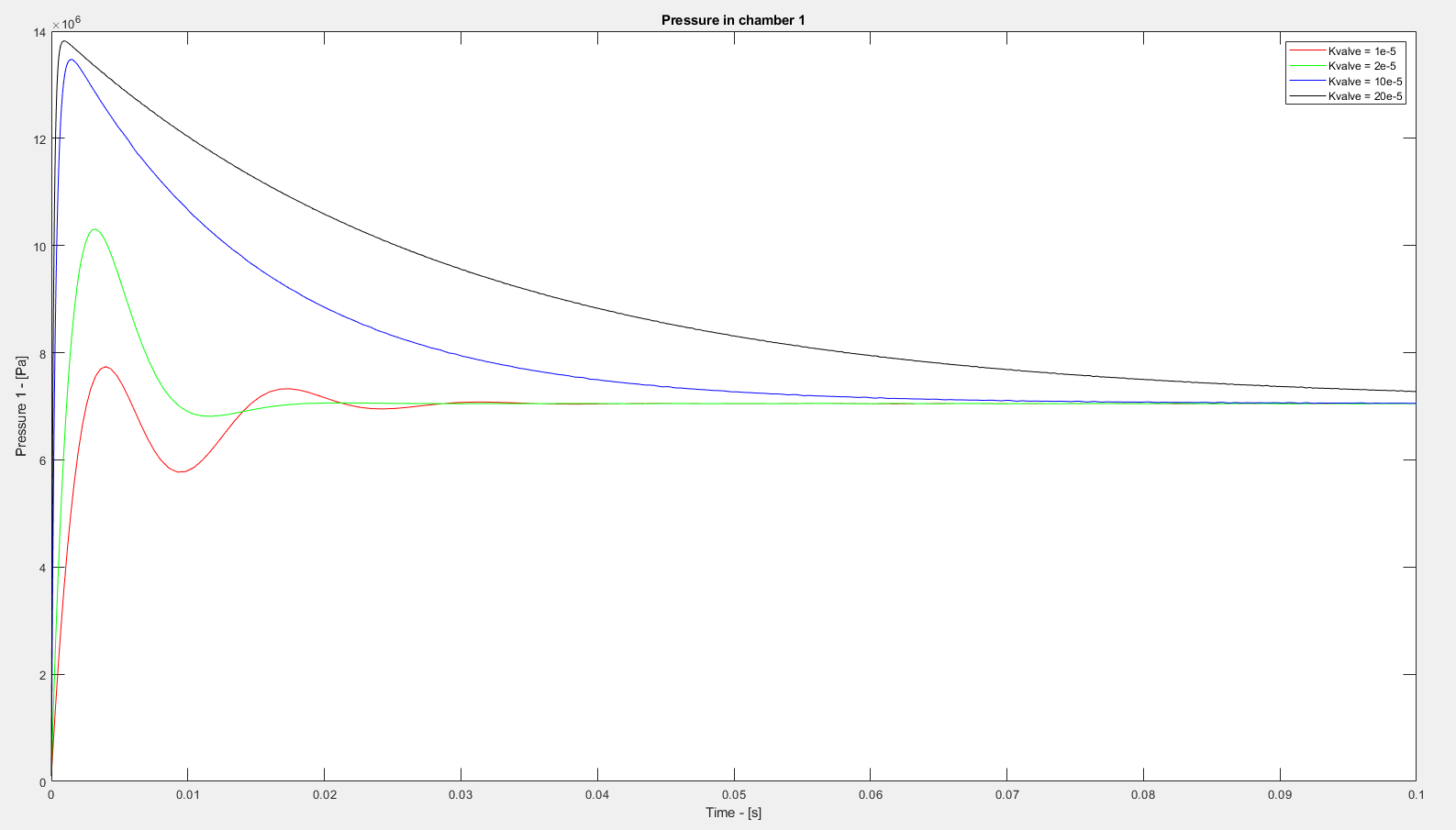
% write the non-trivial equations using nice names

xddot = A\*(p1-p2)/m;

p1dot = (y\*Kv\*(ps-p1)-rho\*A\*xdot)\*((beta)/(V\*rho));

p2dot = (y\*Kv\*(p2-pa)-rho\*A\*xdot)\*((-beta)/(V\*rho));

XDOT = [ xdot; xddot; p1dot; p2dot] ; %return the derivative values



1. Paste your solution material for problem 3 after this line.

function SolveODEs()

clf %clear any existing plots

% State defined as X = [x, xdot, p1,p2]

[t,y1] = ode45( @deriv,[0,0.05],[0,0,1e5,1e5]); % derivative, time range, initial conditions

p1\_analytic = (7-14.\*exp(-400.\*t).\*sin(800.\*t+1.55)).\*10^6;

plot(t,y1(:,3),'r',t,p1\_analytic,'b'); %tvals,yvals, color and style

title('Simulation and analytic solution comparison');

xlabel('Time - [s]');

ylabel('Pressure 1 - [Pa]');

legend('p1','analytic p1')

pause % hit enter to go to the next plot

function XDOT = deriv(t,X)

% System Parameters

A =4.909e-4; Cd = 0.6; rho= 850;

V = 1.473e-4; beta = 2e9; pa=1e5;

m = 30; ps =1.4e7; Kv = 2e-5;

% Rename states

x = X(1); xdot = X(2); p1 = X(3); p2 = X(4);

% Initiate forcing function

y = 0.002;

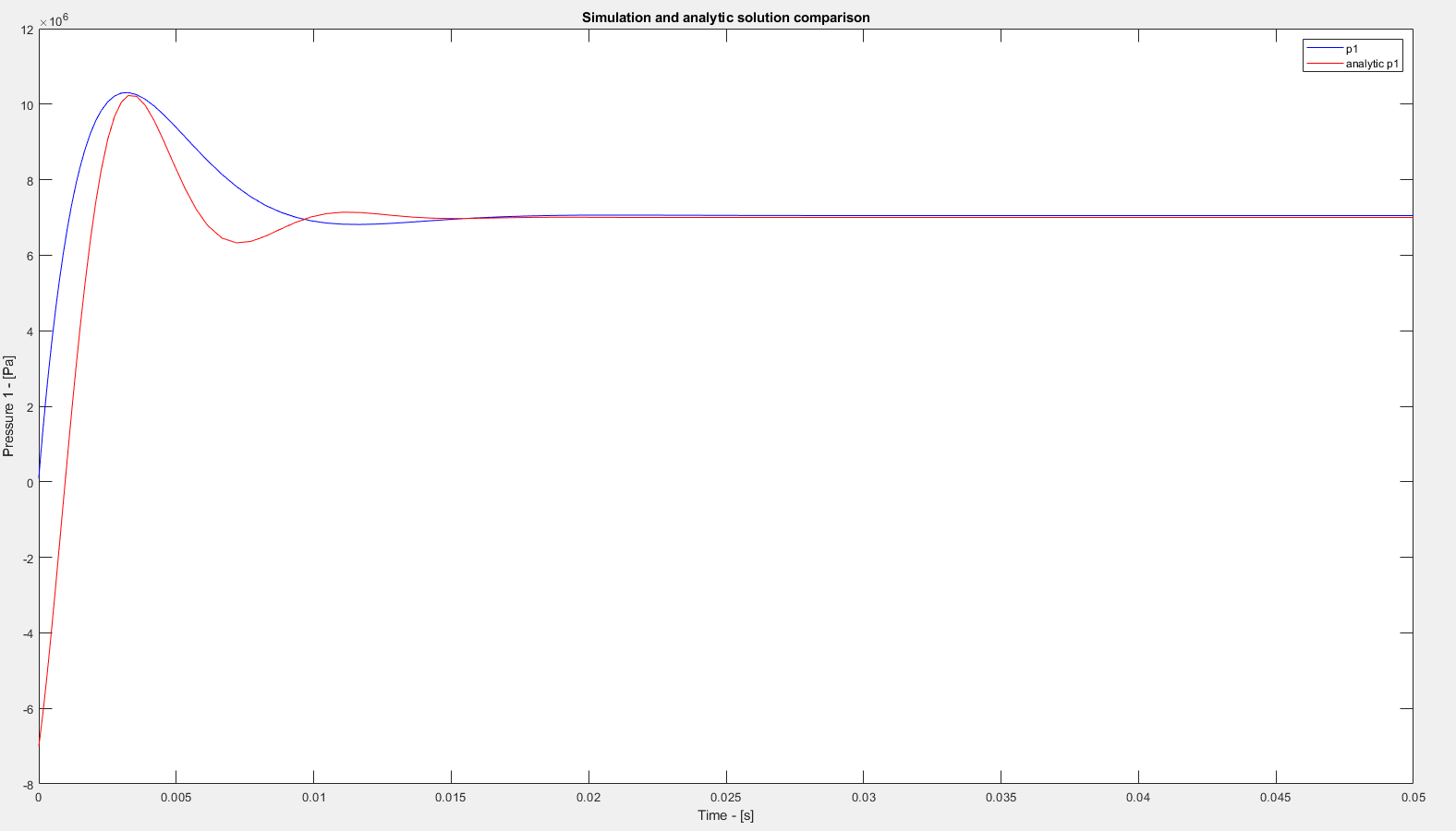
% write the non-trivial equations using nice names

xddot = A\*(p1-p2)/m;

p1dot = (y\*Kv\*(ps-p1)-rho\*A\*xdot)\*((beta)/(V\*rho));

p2dot = (y\*Kv\*(p2-pa)-rho\*A\*xdot)\*((-beta)/(V\*rho));

XDOT = [ xdot; xddot; p1dot; p2dot] ; %return the derivative values

**

1. Paste your solution material for problem 4 after this line.

function SolveODEs()

clf %clear any existing plots

% State defined as X = [x, xdot, p1,p2]

global lin; global Kv;

lin = 1; Kv = 2e-5;

[t\_lin,y\_lin] = ode45( @deriv,[0,0.1],[0,0,1e5,1e5]); % derivative, time range, initial conditions

lin = 0; Kv = 0.074;

[t\_nonlin,y\_nonlin] = ode45( @deriv,[0,0.1],[0,0,1e5,1e5]);

plot(t\_lin,y\_lin(:,3),'r',t\_nonlin,y\_nonlin(:,3),'b');

title('Linear and Nonlinear model comparison');

xlabel('Time - [s]');

ylabel('Pressure 1 - [Pa]');

legend('Linear solution','Non-linear solution')

pause

function XDOT = deriv(t,X)

% System Parameters

A =4.909e-4; Cd = 0.6; rho= 850;

V = 1.473e-4; beta = 2e9; pa=1e5;

m = 30; ps =1.4e7; global Kv

global lin

% Rename states

x = X(1); xdot = X(2); p1 = X(3); p2 = X(4);

% Initiate forcing function

y = 0.002;

% write the non-trivial equations using nice names

if lin == 1

xddot = A\*(p1-p2)/m;

p1dot = (y\*Kv\*(ps-p1)-rho\*A\*xdot)\*((beta)/(V\*rho));

p2dot = (y\*Kv\*(p2-pa)-rho\*A\*xdot)\*((-beta)/(V\*rho));

end

% write the non-trivial equations using nice names

if lin == 0

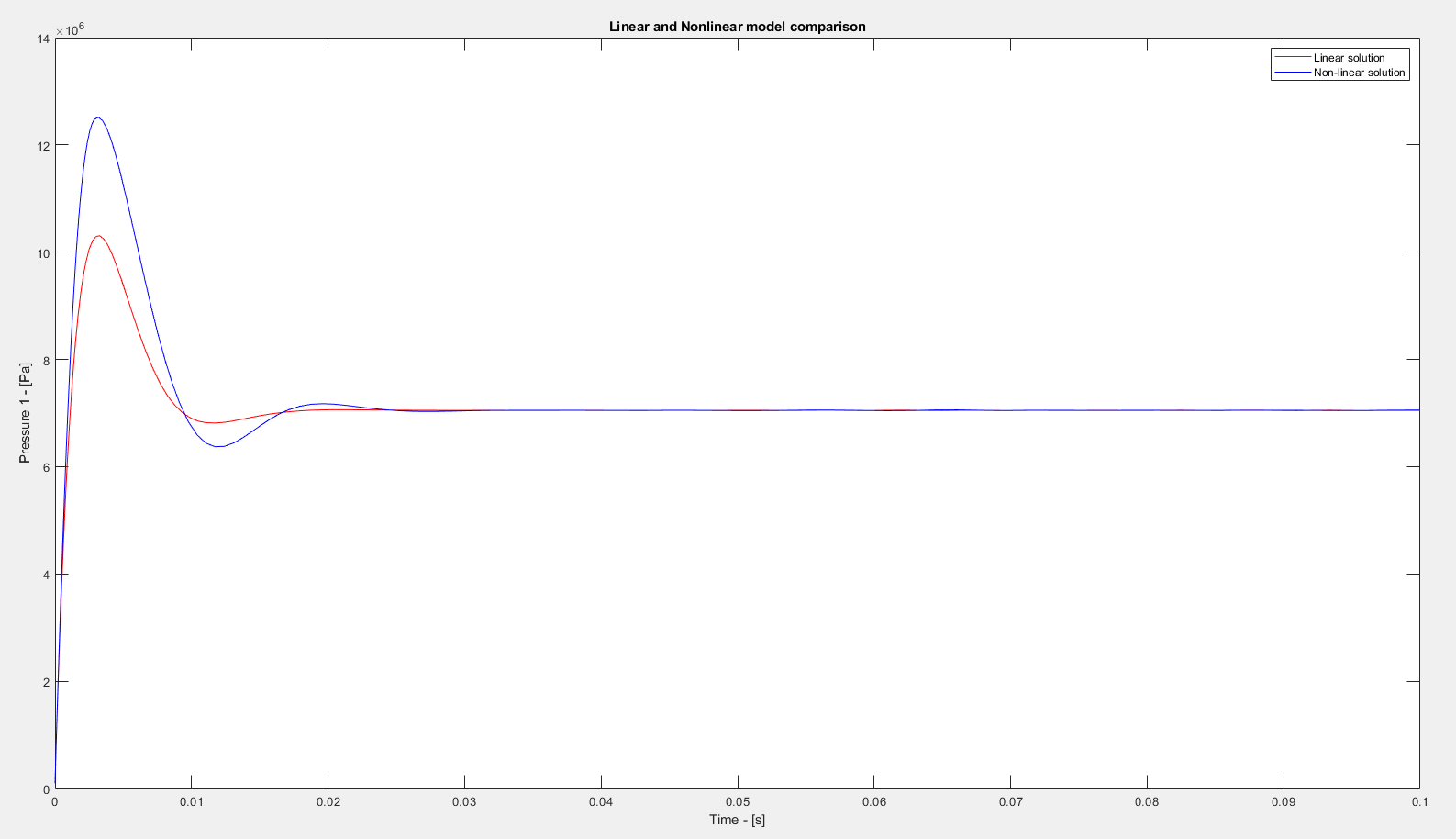
xddot = A\*(p1-p2)/m;

p1dot = (y\*Kv\*sign(ps-p1)\*sqrt(abs(ps-p1))-rho\*A\*xdot)\*((beta)/(V\*rho));

p2dot = (y\*Kv\*sign(p2-pa)\*sqrt(abs(p2-pa))-rho\*A\*xdot)\*((-beta)/(V\*rho));

end

XDOT = [xdot; xddot; p1dot; p2dot] ; %return the derivative values

**

1. Paste your solution material for problem 5 after this line.

% Define constants for RCL Circuit

R1 = 8000;

R2 = 8000;

L = 200;

C = 2.5\*10^-6;

% Initialize Transfer Function

num = [R2];

den = [R2\*C\*L, R1\*R2\*C+L, R1+R2];

T = tf(num,den);

% Step Response

opt = stepDataOptions('InputOffset',0,'StepAmplitude',12);

step(T,opt)

pause

