THE UNIVERSITY OF TEXAS AT ARLINGTON, TEXAS
DEPARTMENT OF ELECTRICAL ENGINEERING

EE 5322 - 002

INTELLIGENT CONTROL SYSTEMS

HW # 4
ASSIGNMENT

by

SOUTRIK PRASAD MAITI

1001569883

Presented to

Dr. Frank Lewis

Oct 31, 2017

EE 5322 Intelligent Control

Fall 2015

Homework Pledge of Honor

On all homeworks in this class - YOU MUST WORK ALONE.

Any cheating or collusion will be severely punished.

It is very easy to compare your software code and determine if you worked together

It does not matter if you change the variable names.

Please sign this form and include it as the first page of all of your submitted homeworks.

....

Typed Name: Soutrik Maiti

Pledge of honor:

"On my honor I have neither given nor received aid on this homework."

e-Signature: Soutrik Maiti

EE 5322 Homework 4

Problem 1:

MATLAB CODE –

```
close all;
clear all;


%Matrices A,B,C,D
a=[0 1;-25 -2];
b=[0;1];
c=[1 0];
d=0;

%Finding the poles and the transfer function
[n,d]=ss2tf(a,b,c,d);
sysTF=tf(n,d)           %Transfer Function
sysPoles=pole(sysTF)    %Poles

%Simulating system with unit step function for 10s
x0=[0;0];    %Initial conditions
t=0:0.01:10; %Total time of simulation
u=ones(length(t),1);    %Unit step function

plot(t,lsim(sysTF,u,t,x0))
```

System Transfer function and poles:

```
Command Window
>> P1a

sysTF =

        s
  --------------
  s^2 + 2 s + 25

Continuous-time transfer function.


sysPoles =

  -1.0000 + 4.8990i
  -1.0000 - 4.8990i

fx >>
```

Fig1: System Transfer function and poles

Unit step response of the system:

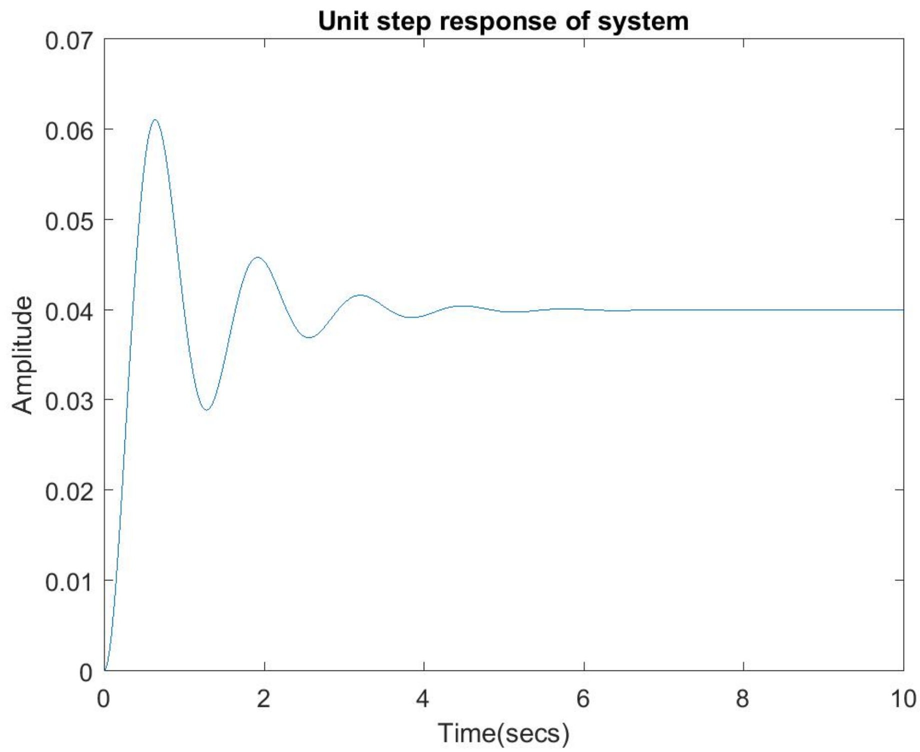

Fig2: Unit step response of the system

Problem 2:

MATLAB CODE –

```
function AC=AC(t,s)

%The states of the plant
y=s(1);
ydot=s(2);

%The states of adaptive Controller
a1=s(3);
a2=s(4);

%The plant system matrices
a=[0 1;-25 -2];
b=[0;1];

%The unit step input
yd=5;
e=yd-y; %Error
edot=-ydot;
lam=4;       %Lamda
r=edot+lam*e;   %sliding error
```

```
wT=[a1 a2];      %The unknown weights
phi=[y;ydot];    %The regression matrix
f=wT*phi;
F=50;
dw=F*phi*transpose(r);  %Adapted Parameters
k=60;
v=f+k*r;
u=v+lam*edot;        %Input to the plant
dx= (a*phi)+(b*u);    %Output of the plant

AC=[dx;dw];

end


clc;
clear all;
ini=[0;0;1;1];       %Initial Conditions
tin=0:0.1:10;        %Time of the simulation
[t,s]=ode45(@AC,tin,ini) %simulating the adaptive controller



plot(t,s(:,1))
```

It can be seen from the Adaptive controller dynamics that the controller makes the plant follow the input signal from Fig3.

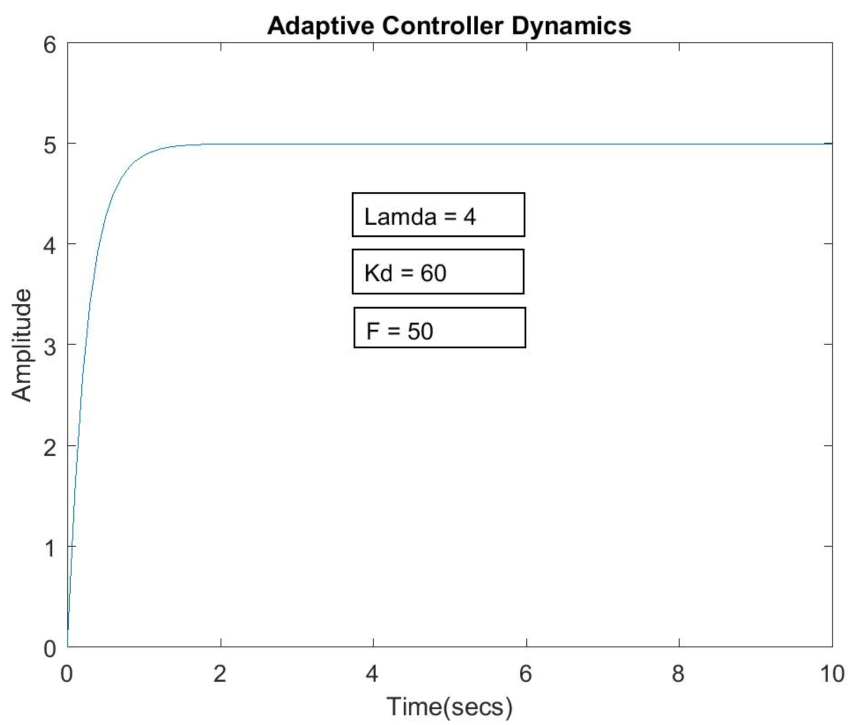

Fig 3 – Adaptive Controller dynamics

Problem 3:

MATLAB CODE –

```
function StateEstimate=robust(t,s)
%states for robust control
y=s(1);
ydot=s(2);
%The state matrices of the known plant
a=[0 1;-25 -2];
b=[0;1];
E=0.33; %Choosing value for epsilon

yd=5; %Unit step input
e=yd-y; %Error
edot=-ydot;
lam=8; %Lamda value
r=edot+lam*e; %The sliding error
wT=[1 20];     %The known weights
phi=[y;ydot];   %The regression matrix
f=wT*phi;       %Activation function
F=[1 50]*phi;

k=70;

%Under norm bound
if norm(r)<E
    v=-r*(F/E);
else
    v=-r*(F/norm(r));
end

tau=f+k*r-v;  %Plant Input
u=tau+lam*edot;
dx= (a*phi)+(b*u); %Plant Output

StateEstimate=[dx(1);dx(2)];

end
clc;
clear all;
ini=[0;0];  %Initial conditions for robust controller
[t,s]=ode45(@robust,[0:0.1:10],ini) %Simulating robust controller



plot(t,s(:,1))
```

It can be seen from the robust controller dynamics that the controller does not actually reach the input but stays close to it.(Fig 4)

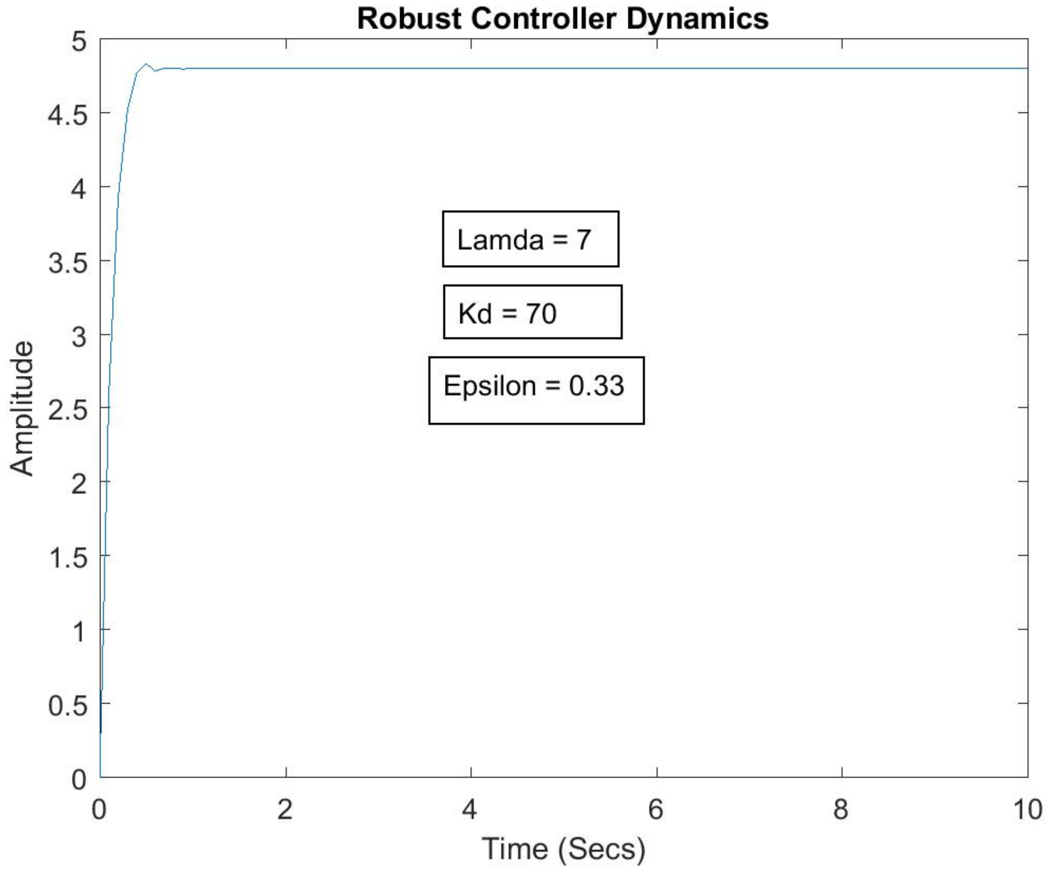

Fig 4- Robust Controller Dynamics