./

Learning Report – Control Systems

Course Code: <CODE>



Version Number:3.0

Team Members :

Team No:

Module: Control System

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
| 1 | 15-04-2021 | ShivaKumar Naga Vankadhara |  | DR. Pagala Prithvi Sekhar | Added the scripts |
| 2 | 15-04-2021 | ShivaKumar Naga Vankadhara |  | DR. Pagala Prithvi Sekhar | Added the Comparision for Modellica and Simulink |
| 3 | 15-04-2021 | ShivaKumar Naga Vankadhara |  | DR. Pagala Prithvi Sekhar | Added the Comaprision for Octave Script and Simulink script |

**Document History**

# 

Content

Title:Control System-First Order System: Analysis by poles and parameters 5

This Document has equation for motion differential system 5

Math analysis 5

IVT 5

IVT 6

IVT 8

Poles plotting 10

Response analysis (SAS) 11

Title:Control System-First Order System: System analysis by changing gain 12

This Document has equation for motion differential system 12

Math analysis 13

Changing the gain of system 13

Analysis: 15

Change the control function 16

Analysis: 18

Title:Control System-First Order System: adding P,I,D controllers 19

This Document has equation for motion differential system 19

Math analysis 19

Negative feedback 19

Positive feedback 22

Title:Control System-Second Order System:open loop with different values 27

This Document has equation for DC Motor 27

Math analysis 27

IVT 27

Analysis 31

Title:Control System-Second Order System:varying zeta value open system 32

This Document has equation for Second Order System 32

Analysis based on zeta 41

Title:Control System-Second Order System: p,i,d OPEN 42

This Document has equation for DC Motor 42

Math analysis 42

Analysis 48

Title:Control System-Second Order System 49

This Document has equation for DC Motor 49

Math analysis 49

Negtaive Feedback 49

Positive Feedback 54

Analysis 60

Title:ControlSystemsecondorder:negative fb with different parameter values 61

This Document has equation for DC motor system 61

Math Analysis 61

Analysis: 68

normal 69

pi 71

PD 73

PID 75

Title:Control System-Individual System(Thermometer) 79

This Document has equation for First Order Thermometer Equation 79

Math analysis 79

Basic 79

With Gain 81

With PI 82

With PD 84

With PID 86

Analysis 94

With POsitive feedback 95

Analysis 105

With Negative feedback 106

Analysis 118

Comparison between Simulink & Modellica: 119

Comparison between Matlab Script and GNU Octave Script 121

## Title:Control System-First Order System: Analysis by poles and parameters

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.7

## This Document has equation for motion differential system

%Equation:mdv/dt+bv=u

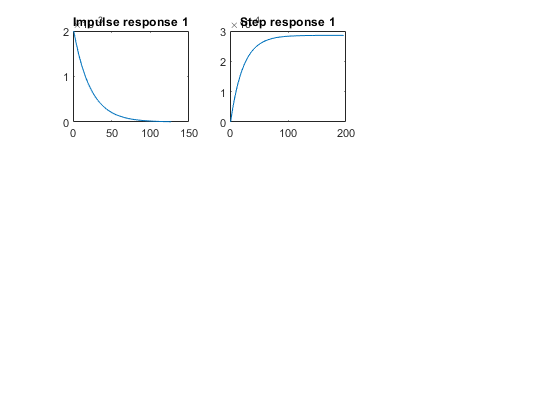
## Math analysis

%dependent variables:v  
%independent variables:t,u  
%constant:m,b  
%Root:-b/m

## IVT

%for impulse is 1/m=0.002  
%for step is 0  
%%FVT  
%for impulse is 0;  
%for step is 1/b=0.00028  
  
m1=500;  
b1=3500;  
Tau=m1/b1;  
TF=tf([0,1/b1],[Tau,1])  
T\_R=4\*Tau  
subplot(3,3,1),plot(impulse(TF))  
title("Impulse response 1")  
subplot(3,3,2),plot(step(TF))  
title("Step response 1")  
S = stepinfo(TF)

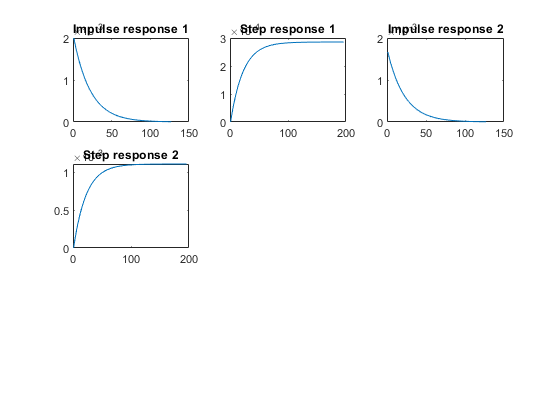
TF =  
   
 0.0002857  
 ------------  
 0.1429 s + 1  
   
Continuous-time transfer function.  
  
  
T\_R =  
  
 0.5714  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.3139  
 SettlingTime: 0.5589  
 SettlingMin: 2.5843e-04  
 SettlingMax: 2.8571e-04  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 2.8571e-04  
 PeakTime: 1.5065



## IVT

%for impulse is 1/m=0.00166  
%for step is 0  
%%FVT  
%for impulse is 0;  
%for step is 1/b=0.001111  
  
m2=600;  
b2=900;  
Tau=m2/b2;  
T\_R=4\*Tau  
TF=tf([0,1/b2],[Tau,1])  
subplot(3,3,3),plot(impulse(TF))  
title("Impulse response 2")  
subplot(3,3,4),plot(step(TF))  
title("Step response 2")  
S = stepinfo(TF)

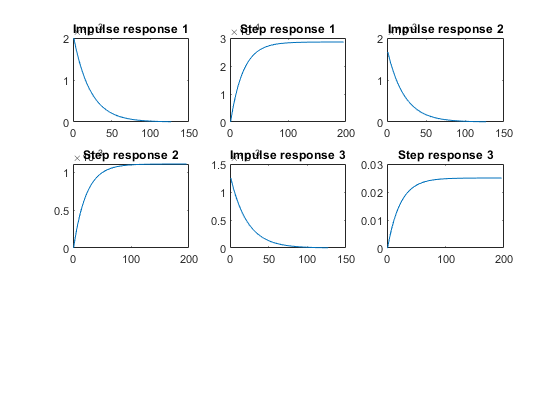
T\_R =  
  
 2.6667  
  
  
TF =  
   
 0.001111  
 ------------  
 0.6667 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.4647  
 SettlingTime: 2.6080  
 SettlingMin: 0.0010  
 SettlingMax: 0.0011  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0011  
 PeakTime: 7.0306



## IVT

%for impulse is 1/m=0.00125  
%for step is 0  
%%FVT  
%for impulse is 0;  
%for step is 1/b=0.025  
  
m3=800;  
b3=40;  
Tau=m3/b3;  
T\_R=4\*Tau  
TF=tf([0,1/b3],[Tau,1])  
subplot(3,3,5),plot(impulse(TF))  
title("Impulse response 3")  
subplot(3,3,6),plot(step(TF))  
title("Step response 3")  
S = stepinfo(TF)

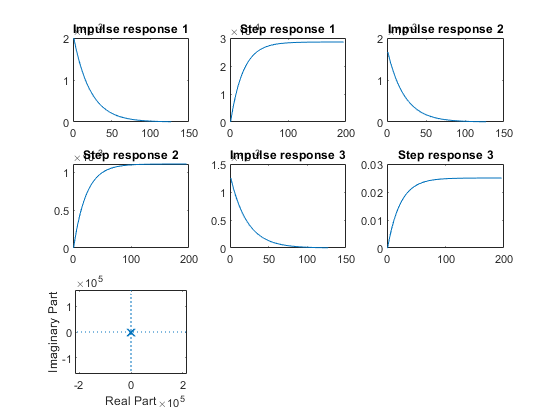
T\_R =  
  
 80  
  
  
TF =  
   
 0.025  
 --------  
 20 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 43.9401  
 SettlingTime: 78.2415  
 SettlingMin: 0.0226  
 SettlingMax: 0.0250  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0250  
 PeakTime: 210.9168



## Poles plotting

hold on  
  
subplot(3,3,7)  
[z1,p1,k1]= tf2zp([0,1/b1],[m1/b1,1])  
zplane(z1,p1)  
  
hold on  
  
subplot(3,3,7)  
[z2,p2,k2]= tf2zp([0,1/b2],[m2/b2,1])  
zplane(z2,p2)  
  
hold on  
subplot(3,3,7)  
[z3,p3,k3]= tf2zp([0,1/b3],[m3/b3,1])  
zplane(z3,p3)

z1 =  
  
 0×1 empty double column vector  
  
  
p1 =  
 -7  
k1 =  
 0.0020  
z2 =  
 0×1 empty double column vector  
p2 =  
 -1.5000  
k2 =  
 0.0017  
z3 =  
 0×1 empty double column vector  
p3 =  
 -0.0500  
k3 =  
 0.0013



## Response analysis (SAS)

Rise time

%T1=0.3139  
%T2=1.4647  
%T3=43.9401  
%System 1 has the least rise time so the speed of system is greatest  
%System 3 has the greatest rise time so the speed of system is least  
  
% Settling time  
%S1=0.5589  
%S2=2.6080  
%S3=78.2415  
%System 1 is taking least time to get settled so the system is accurate  
%System 3 is taking most time to get settled so the system is least accurate  
  
% Pole position  
%P1=-7.0  
%P2=-1.5000  
%P3=-0.0500  
% system 1 pole is farthest away from pole:best stabilty among 3  
% system 1 pole is farthest away from pole:worst stablity among 3

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

Title:Control System-First Order System: System analysis by changing gain 12

This Document has equation for motion differential system 12

Math analysis 13

Changing the gain of system 13

Analysis: 15

Change the control function 16

Analysis: 18

## Title:Control System-First Order System: System analysis by changing gain

%Author:Shivakumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.4

## This Document has equation for motion differential system

%Equation:mdv/dt+bv=u

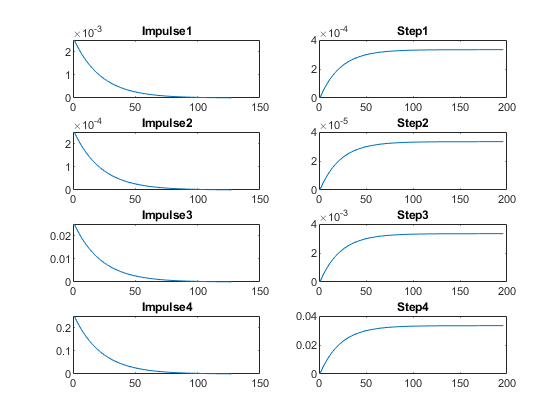
## Math analysis

%dependent variables:v  
%independent variables:t,u  
%constant:m,b  
%Root:-b/m

## Changing the gain of system

%gain is 1  
m1=400;  
b1=3000;  
Tau=m1/b1;  
TF1=tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(4,2,1),plot(impulse(TF1))  
title("Impulse1")  
subplot(4,2,2),plot(step(TF1))  
title("Step1")  
S = stepinfo(TF1)  
  
%gain is 0.1  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=0.1;  
TF2=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(4,2,3),plot(impulse(TF2))  
title("Impulse2")  
subplot(4,2,4),plot(step(TF2))  
title("Step2")  
S = stepinfo(TF2)  
  
%gain is 10  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=10;  
TF3=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(4,2,5),plot(impulse(TF3))  
title("Impulse3")  
subplot(4,2,6),plot(step(TF3))  
title("Step3")  
S = stepinfo(TF3)  
  
%gain is 100  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=100;  
TF4=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(4,2,7),plot(impulse(TF4))  
title("Impulse4")  
subplot(4,2,8),plot(step(TF4))  
title("Step4")  
S = stepinfo(TF4)

S =   
  
 struct with fields:  
  
 RiseTime: 0.2929  
 SettlingTime: 0.5216  
 SettlingMin: 3.0150e-04  
 SettlingMax: 3.3332e-04  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 3.3332e-04  
 PeakTime: 1.4061  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.2929  
 SettlingTime: 0.5216  
 SettlingMin: 3.0150e-05  
 SettlingMax: 3.3332e-05  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 3.3332e-05  
 PeakTime: 1.4061  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.2929  
 SettlingTime: 0.5216  
 SettlingMin: 0.0030  
 SettlingMax: 0.0033  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0033  
 PeakTime: 1.4061  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.2929  
 SettlingTime: 0.5216  
 SettlingMin: 0.0302  
 SettlingMax: 0.0333  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0333  
 PeakTime: 1.4061

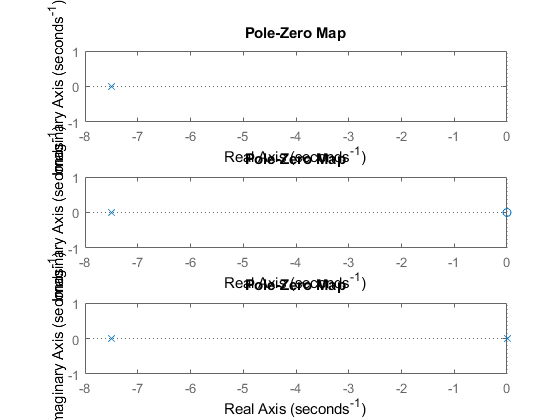
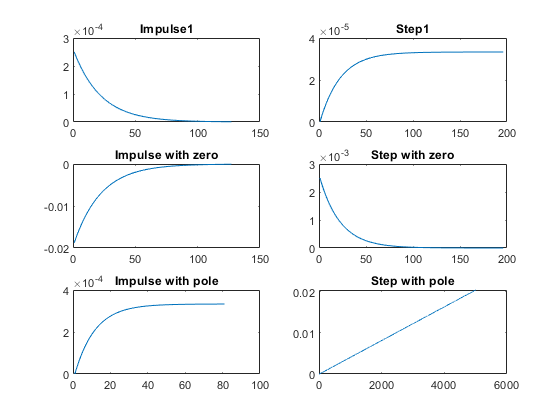


## Analysis:

%On changing the gain of the transfer function:  
%1. By changing gain we can see that only amplitude is getting changed.  
%2. Even after changing the gain settling time,rise time and peak time is  
%not getting changed  
%3. peak, settling min and settling max is varying by factor of gain  
%4.

## Change the control function

figure  
% system with proportion  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=0.1;  
TF5=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(3,2,1),plot(impulse(TF5))  
title("Impulse1")  
subplot(3,2,2),plot(step(TF5))  
title("Step1")  
S = stepinfo(TF5);  
  
% system with differentiator  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=tf([1,0],[1]);  
TF6=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(3,2,3),plot(impulse(TF6))  
title("Impulse with zero")  
subplot(3,2,4),plot(step(TF6))  
title("Step with zero")  
S = stepinfo(TF6);  
  
% system with integrator  
m1=400;  
b1=3000;  
Tau=m1/b1;  
CF=tf([0,1],[1,0]);  
TF7=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
subplot(3,2,5),plot(impulse(TF7))  
title("Impulse with pole")  
subplot(3,2,6),plot(step(TF7))  
title("Step with pole")  
S = stepinfo(TF7);  
  
%poles printing  
figure  
subplot(3,1,1)  
pzmap(TF5)  
subplot(3,1,2)  
pzmap(TF6)  
subplot(3,1,3)  
pzmap(TF7)



## Analysis:

%1. Proportional: 1 pole  
%2. By adding a Differentiator we are getting a zero added.  
%3. By adding an integrator a pole is getting added.  
%4. There is no affect on the poles in the first order only poles and  
%zeroes are geeting added.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

Title:Control System-First Order System: adding P,I,D controllers 19

This Document has equation for motion differential system 19

Math analysis 19

Negative feedback 19

Positive feedback 22

## Title:Control System-First Order System: adding P,I,D controllers

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.7

## This Document has equation for motion differential system

%Equation:mdv/dt+bv=u

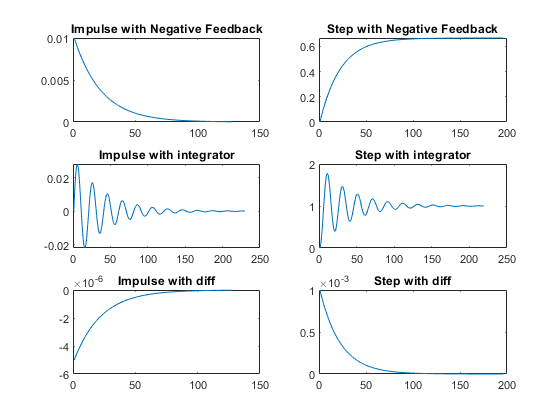
## Math analysis

%dependent variables:v  
%independent variables:t,u  
%constant:m,b  
%Root:-b/m

## Negative feedback

m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=10;  
TF=CF\*tf([0,1/b1],[Tau,1]);  
%S = stepinfo(TF)  
NCTF1=feedback(TF,1);  
subplot(3,2,1),plot(impulse(NCTF1))  
title("Impulse with Negative Feedback")  
subplot(3,2,2),plot(step(NCTF1))  
title("Step with Negative Feedback")  
S1 = stepinfo(NCTF1)  
p1=pole(NCTF1)  
  
m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=tf([0,1],[1,0]);  
TF=CF\*tf([0,1/b1],[Tau,1]);  
NCTF2=feedback(TF,1);  
subplot(3,2,3),plot(impulse(NCTF2))  
title("Impulse with integrator")  
subplot(3,2,4),plot(step(NCTF2))  
title("Step with integrator")  
S2 = stepinfo(NCTF2)  
p2=pole(NCTF2)  
z2=zero(NCTF2)  
  
m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=tf([1,0],[1]);  
TF=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
NCTF3=feedback(TF,1);  
T\_R=4\*Tau;  
subplot(3,2,5),plot(impulse(NCTF3))  
title("Impulse with diff")  
subplot(3,2,6),plot(step(NCTF3))  
title("Step with diff")  
p3=pole(NCTF3)  
S3 = stepinfo(NCTF3)  
  
%%Analysis:  
%1. Rise time of the system increases on adding the integartor.  
%2. Rise time of the system decreases on adding the diffrentiator.  
%3. settling time of the system increases on adding integrator system is  
%taking some time to settle and operate.  
%4. accuracy of system decreases on adding differentiator  
%5. overshoot increase is greater on adding differentiator than integrator  
%6. Peak increase is greater on adding integrator than differentiator  
%7. all the poles of negative feedback present in left side of plane

S1 =   
  
 struct with fields:  
  
 RiseTime: 146.4671  
 SettlingTime: 260.8050  
 SettlingMin: 0.6030  
 SettlingMax: 0.6666  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.6666  
 PeakTime: 703.0560  
  
  
p1 =  
  
 -0.0150  
  
  
S2 =   
  
 struct with fields:  
  
 RiseTime: 35.0513  
 SettlingTime: 1.5129e+03  
 SettlingMin: 0.3925  
 SettlingMax: 1.7794  
 Overshoot: 77.9429  
 Undershoot: 0  
 Peak: 1.7794  
 PeakTime: 99.3459  
  
  
p2 =  
  
 -0.0025 + 0.0315i  
 -0.0025 - 0.0315i  
  
  
z2 =  
  
 0×1 empty double column vector  
  
  
p3 =  
  
 -0.0050  
  
  
S3 =   
  
 struct with fields:  
  
 RiseTime: 439.8407  
 SettlingTime: 783.1973  
 SettlingMin: 2.6276e-08  
 SettlingMax: 9.5404e-05  
 Overshoot: 4.6071e+17  
 Undershoot: 0  
 Peak: 9.9900e-04  
 PeakTime: 0



## Positive feedback

figure  
m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=10;  
TF=CF\*tf([0,1/b1],[Tau,1]);  
%S = stepinfo(TF)  
PCTF1=feedback(TF,-1);  
subplot(3,2,1),plot(impulse(PCTF1))  
title("Impulse with Positive feedback")  
subplot(3,2,2),plot(step(PCTF1))  
title("Step with Positive feedback")  
S = stepinfo(PCTF1)  
p4=pole(PCTF1)  
  
m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=tf([0,1],[1,0]);  
TF=CF\*tf([0,1/b1],[Tau,1]);  
PCTF2=feedback(TF,-1);  
subplot(3,2,3),plot(impulse(PCTF2))  
title("Impulse with integrator")  
subplot(3,2,4),plot(step(PCTF2))  
title("Step with integrator")  
p5=pole(PCTF2)  
S = stepinfo(PCTF2)  
  
m1=1000;  
b1=5;  
Tau=m1/b1;  
CF=tf([1,0],[1]);  
TF=CF\*tf([0,1/b1],[Tau,1]);  
T\_R=4\*Tau;  
PCTF3=feedback(TF,-1);  
T\_R=4\*Tau;  
subplot(3,2,5),plot(impulse(PCTF3))  
title("Impulse with diff")  
subplot(3,2,6),plot(step(PCTF3))  
title("Step with diff")  
p6=pole(PCTF3)  
z2=zero(PCTF3)  
S = stepinfo(PCTF3)  
  
%%Analysis:  
%1. on adding differentiator to positive feedback system, system is  
% becoming stable and poles got shifted to left side  
%2. The system is unstable in case of positive feedback with gain  
% and integrator  
%3. As the system is unstable in case of gain and integrator we are not  
% getting parameters, also the peak is infinite  
%4. Parameters can be obtained in differentiator as differentiator making  
% the system stable  
%5. positive feedback unstable system poles lies in right side of plane

S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p4 =  
  
 0.0050  
  
  
p5 =  
  
 -0.0342  
 0.0292  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p6 =  
  
 -0.0050  
  
  
z2 =  
  
 0  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 438.9619  
 SettlingTime: 781.6325  
 SettlingMin: 2.6329e-08  
 SettlingMax: 9.5595e-05  
 Overshoot: Inf  
 Undershoot: 0  
 Peak: 0.0010  
 PeakTime: 0

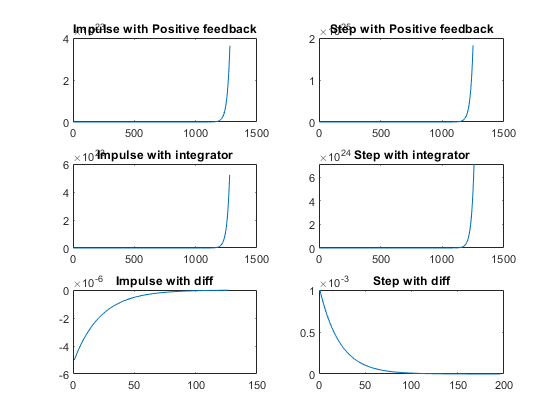
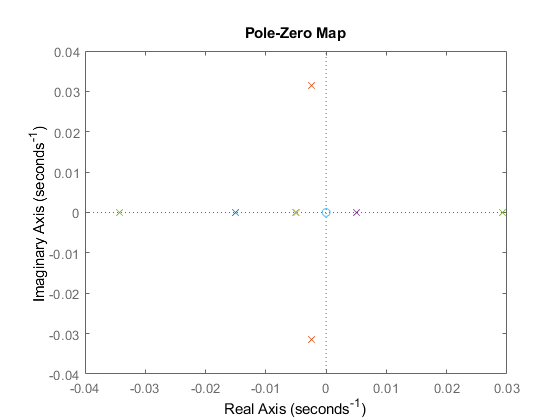


figure  
hold on  
pzmap(NCTF1)  
pzmap(NCTF2)  
pzmap(NCTF3)  
pzmap(PCTF1)  
pzmap(PCTF2)  
pzmap(PCTF3)



[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

Title:Control System-Second Order System:open loop with different values 27

This Document has equation for DC Motor 27

Math analysis 27

IVT 27

Analysis 31

## Title:Control System-Second Order System:open loop with different values

%Author:ShivaKumar Naga VAnkadhara  
%PS No:99003727  
%Date:11/04/2021  
%Version:1.7

## This Document has equation for DC Motor

%Equation:Ldi/dt+Ri+Kw=V  
% Jdw/dt+bw=Ki  
%T(s)=(K/LJ)/(s^2+((b/J)+(R/L)s+(R\*b)/(L\*J)+(K\*K)/(L\*J)

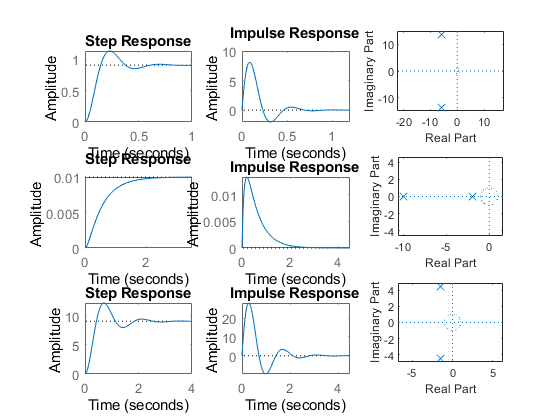
## Math analysis

%dependent variables:w  
%independent variables:t  
%constant:K,R,L,J,b  
%Roots:0.5\*(-(b/J)-(R/L))+sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))  
% 0.5\*(-(b/J)-(R/L))-sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))

## IVT

%for impulse is 0  
%for step is 0  
%%FVT  
%for impulse is K/((b\*L)+(R\*J))=0.1667  
%for step is K/((R\*b)+(K\*K))=0.0999001  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
%TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
sys = tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
subplot(3,3,1)  
step(sys)  
subplot(3,3,2)  
impulse(sys)  
subplot(3,3,3)  
%S = stepinfo(sys)  
[z,p,k]= tf2zp([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
zplane(z,p)  
S = stepinfo(sys)  
  
J = 0.1;  
b = 1;  
K = 0.1;  
R = 10;  
L = 5;  
%TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
sys = tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
subplot(3,3,4)  
step(sys)  
subplot(3,3,5)  
impulse(sys)  
subplot(3,3,6)  
%S = stepinfo(sys)  
[z2,p2,k2]= tf2zp([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
zplane(z2,p2)  
S = stepinfo(sys)  
  
J = 0.01;  
b = 0.01;  
K = 0.1;  
R = 0.1;  
L = 0.05;  
%TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
sys = tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
subplot(3,3,7)  
step(sys)  
subplot(3,3,8)  
impulse(sys)  
subplot(3,3,9)  
%S = stepinfo(sys)  
[z1,p1,k1]= tf2zp([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))])  
zplane(z1,p1)  
S = stepinfo(sys)

sys =  
   
 200  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 -6.0000 +13.5647i  
 -6.0000 -13.5647i  
  
  
k =  
  
 200  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0993  
 SettlingTime: 0.5669  
 SettlingMin: 0.8527  
 SettlingMax: 1.1356  
 Overshoot: 24.9123  
 Undershoot: 0  
 Peak: 1.1356  
 PeakTime: 0.2303  
  
  
sys =  
   
 0.2  
 ------------------  
 s^2 + 12 s + 20.02  
   
Continuous-time transfer function.  
  
  
z2 =  
  
 0×1 empty double column vector  
  
  
p2 =  
  
 -9.9975  
 -2.0025  
  
  
k2 =  
  
 0.2000  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.1351  
 SettlingTime: 2.0652  
 SettlingMin: 0.0090  
 SettlingMax: 0.0100  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0100  
 PeakTime: 3.6758  
  
  
sys =  
   
 200  
 --------------  
 s^2 + 3 s + 22  
   
Continuous-time transfer function.  
  
  
z1 =  
  
 0×1 empty double column vector  
  
  
p1 =  
  
 -1.5000 + 4.4441i  
 -1.5000 - 4.4441i  
  
  
k1 =  
  
 200  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.2882  
 SettlingTime: 2.3810  
 SettlingMin: 8.0006  
 SettlingMax: 12.2393  
 Overshoot: 34.6325  
 Undershoot: 0  
 Peak: 12.2393  
 PeakTime: 0.7061



## Analysis

1.If rise time is less the system is not much stable and its speed 2.If the rise time is high the system may behave more stable its not speed in nature. 3.If the Over shoot is less the system is kind of stable. 4.If the Over shoot is more the system may behave less stable. 5.If settling time is less accuracy is high. 6.If the settling time is high accuracy is less. 7.In the above systems system 2 is more stable because overshoot is 0. 8.Peak time is inversly proportional to overshoot. so if peak time is more system is stable. 9.when we add proportional to the open loop no parameters get changed only peak time and overshoot changes.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

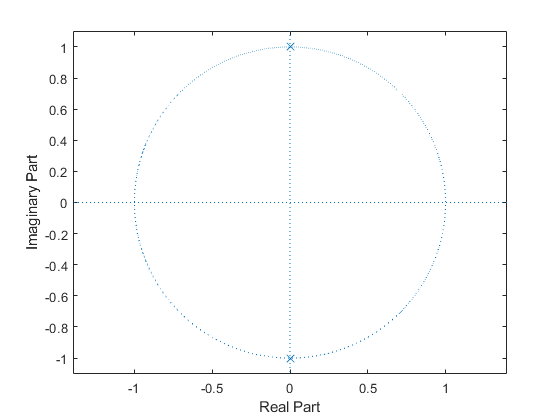
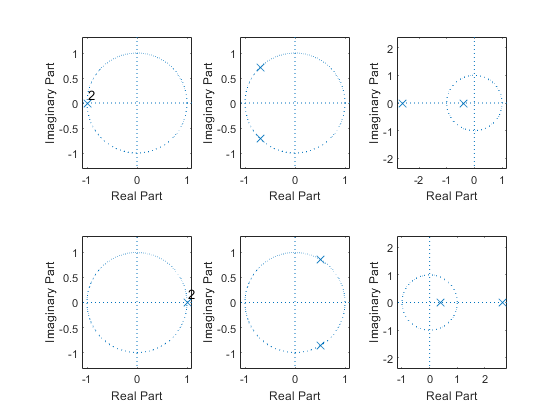
## Title:Control System-Second Order System:varying zeta value open system

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.0

## This Document has equation for Second Order System

%w=1  
  
jeta=1;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
figure  
subplot(2,3,1)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
jeta=0.7;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
%hold on  
subplot(2,3,2)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
jeta=1.5;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
subplot(2,3,3)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
jeta=-1;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
subplot(2,3,4)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
  
jeta=-0.5;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
subplot(2,3,5)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
jeta=-1.5;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
subplot(2,3,6)  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)  
  
figure  
jeta=0;  
TF=tf([1],[1,(2\*jeta),1])  
sys = tf([1],[1,(2\*jeta),1])  
S = stepinfo(sys)  
[z,p,k]= tf2zp([1],[1,(2\*jeta),1])  
zplane(z,p)

TF =  
   
 1  
 -------------  
 s^2 + 2 s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -------------  
 s^2 + 2 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 3.3579  
 SettlingTime: 5.8339  
 SettlingMin: 0.9000  
 SettlingMax: 0.9994  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9994  
 PeakTime: 9.7900  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 -1  
 -1  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 ---------------  
 s^2 + 1.4 s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 ---------------  
 s^2 + 1.4 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.1268  
 SettlingTime: 5.9789  
 SettlingMin: 0.9001  
 SettlingMax: 1.0460  
 Overshoot: 4.5986  
 Undershoot: 0  
 Peak: 1.0460  
 PeakTime: 4.4078  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 -0.7000 + 0.7141i  
 -0.7000 - 0.7141i  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 -------------  
 s^2 + 3 s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -------------  
 s^2 + 3 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 5.8584  
 SettlingTime: 10.6547  
 SettlingMin: 0.9012  
 SettlingMax: 0.9999  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9999  
 PeakTime: 25.9983  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 -2.6180  
 -0.3820  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 -------------  
 s^2 - 2 s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -------------  
 s^2 - 2 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 1  
 1  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 -----------  
 s^2 - s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -----------  
 s^2 - s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 0.5000 + 0.8660i  
 0.5000 - 0.8660i  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 -------------  
 s^2 - 3 s + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -------------  
 s^2 - 3 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 2.6180  
 0.3820  
  
  
k =  
  
 1  
  
  
TF =  
   
 1  
 -------  
 s^2 + 1  
   
Continuous-time transfer function.  
  
  
sys =  
   
 1  
 -------  
 s^2 + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
z =  
  
 0×1 empty double column vector  
  
  
p =  
  
 0.0000 + 1.0000i  
 0.0000 - 1.0000i  
  
  
k =  
  
 1



## Analysis based on zeta

1. If zeta>0 we may get the roots on the left side of the imaginary axis. 2. If zeta<0 we may get the roots on the right side of the imaginary axis. 3. If zeta lies in the range of [0-1] we get complex conjugate roots. 4. If zeta ranges greater than 1 we get real roots and distinct. 5. If zeta is equal to 1 we get real roots. 6. If zeta is zero poles lies on the imaginary axis like complex conjugate roots system is undamped.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

## Title:Control System-Second Order System: p,i,d OPEN

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.7

## This Document has equation for DC Motor

%Equation:Ldi/dt+Ri+Kw=V  
% Jdw/dt+bw=Ki  
%T(s)=(K/LJ)/(s^2+((b/J)+(R/L)s+(R\*b)/(L\*J)+(K\*K)/(L\*J)

## Math analysis

%dependent variables:w  
%independent variables:t  
%constant:K,R,L,J,b  
%Roots:0.5\*(-(b/J)-(R/L))+sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))  
% 0.5\*(-(b/J)-(R/L))-sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=1;  
sys1 = CF\*TF;  
subplot(4,2,1)  
step(sys1)  
title("Step ")  
subplot(4,2,2)  
impulse(sys1)  
title("Impulse")  
S = stepinfo(sys1);  
[wn,zeta]=damp(sys1)  
p1=pole(sys1)  
z1=zero(sys1)  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10;  
sys2 = CF\*TF;  
subplot(4,2,3)  
step(sys2)  
title("Step with gain")  
subplot(4,2,4)  
impulse(sys2)  
title("impulse with gain")  
S = stepinfo(sys2)  
[wn,zeta]=damp(sys2)  
p2=pole(sys2)  
z2=zero(sys2)  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1,0],[1]);  
sys3 = CF\*TF;  
subplot(4,2,5)  
step(sys3)  
title("Step with zero ")  
subplot(4,2,6)  
impulse(sys3)  
title("impulse with zero ")  
S = stepinfo(sys3)  
[wn,zeta]=damp(sys3)  
p3=pole(sys3)  
z3=zero(sys3)  
  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1],[1,0]);  
sys4 = CF\*TF;  
subplot(4,2,7)  
step(sys4)  
title("Step with pole ")  
subplot(4,2,8)  
impulse(sys4)  
title("impulse with pole ")  
S = stepinfo(sys4)  
[wn,zeta]=damp(sys4)  
p4=pole(sys4)  
z4=zero(sys4)

wn =  
  
 14.8324  
 14.8324  
  
  
zeta =  
  
 0.4045  
 0.4045  
  
  
p1 =  
  
 -6.0000 +13.5647i  
 -6.0000 -13.5647i  
  
  
z1 =  
  
 0×1 empty double column vector  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0993  
 SettlingTime: 0.5669  
 SettlingMin: 8.5269  
 SettlingMax: 11.3557  
 Overshoot: 24.9123  
 Undershoot: 0  
 Peak: 11.3557  
 PeakTime: 0.2303  
  
  
wn =  
  
 14.8324  
 14.8324  
  
  
zeta =  
  
 0.4045  
 0.4045  
  
  
p2 =  
  
 -6.0000 +13.5647i  
 -6.0000 -13.5647i  
  
  
z2 =  
  
 0×1 empty double column vector  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0  
 SettlingTime: 0.6520  
 SettlingMin: -2.0155  
 SettlingMax: 8.0919  
 Overshoot: Inf  
 Undershoot: Inf  
 Peak: 8.0919  
 PeakTime: 0.0844  
  
  
wn =  
  
 14.8324  
 14.8324  
  
  
zeta =  
  
 0.4045  
 0.4045  
  
  
p3 =  
  
 -6.0000 +13.5647i  
 -6.0000 -13.5647i  
  
  
z3 =  
  
 0  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
wn =  
  
 0  
 14.8324  
 14.8324  
  
  
zeta =  
  
 -1.0000  
 0.4045  
 0.4045  
  
  
p4 =  
  
 0.0000 + 0.0000i  
 -6.0000 +13.5647i  
 -6.0000 -13.5647i  
  
  
z4 =  
  
 0×1 empty double column vector

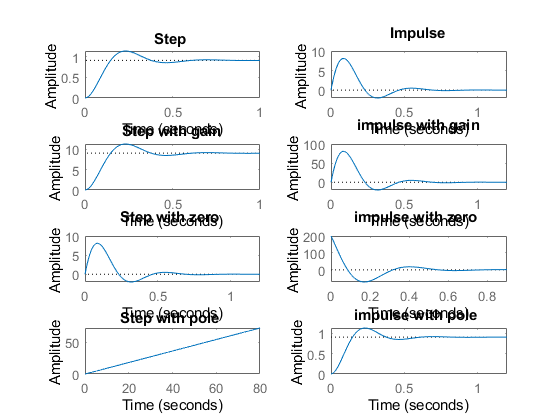
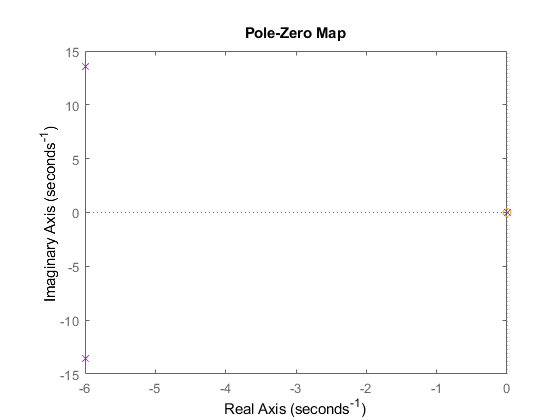


figure  
hold on  
pzmap(sys1)  
pzmap(sys2)  
pzmap(sys3)  
pzmap(sys4)



## Analysis

%1.There is no change in the poles when we add differentiator, integrator  
% and differentiator.  
%2. When we add a differentiator the system becomes more stable because a  
%zero is getting added to it.  
%3. Adding a differentiator IVT got shifted from zero, Fvt will remain same  
% for impulse response.  
%4. FVT of integrator of impulse got shifted to zero.  
%5. By adding integrator step response doesn't settle.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

## Title:Control System-Second Order System

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.4

## This Document has equation for DC Motor

%Equation:Ldi/dt+Ri+Kw=V  
% Jdw/dt+bw=Ki  
%T(s)=(K/LJ)/(s^2+((b/J)+(R/L)s+(R\*b)/(L\*J)+(K\*K)/(L\*J)

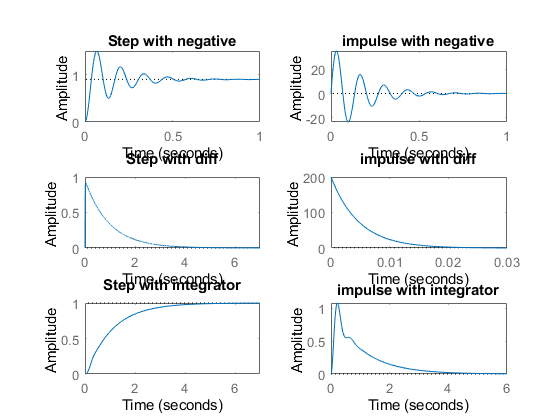
## Math analysis

%dependent variables:w  
%independent variables:t  
%constant:K,R,L,J,b  
%Roots:0.5\*(-(b/J)-(R/L))+sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))  
% 0.5\*(-(b/J)-(R/L))-sqrt((((b\*b)/(J\*J))+((R\*R)/(L\*L))-((2\*R\*b)/(L\*J))-((4\*K\*K)/(L\*J)))

## Negtaive Feedback

J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
NCTF1=feedback(sys,1)  
subplot(3,2,1)  
step(NCTF1)  
title("Step with negative")  
subplot(3,2,2)  
impulse(NCTF1)  
title("impulse with negative")  
S = stepinfo(NCTF1)  
[wn,zeta]=damp(NCTF1)  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1,0],[1])  
sys = CF\*TF  
NCTF2=feedback(sys,1)  
subplot(3,2,3)  
step(NCTF2)  
title("Step with diff")  
subplot(3,2,4)  
impulse(NCTF2)  
title("impulse with diff")  
S = stepinfo(NCTF2)  
[wn,zeta]=damp(NCTF2)  
  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1],[1,0])  
sys = CF\*TF  
NCTF3=feedback(sys,1)  
subplot(3,2,5)  
step(NCTF3)  
title("Step with integrator")  
subplot(3,2,6)  
impulse(NCTF3)  
title("impulse with integrator")  
S = stepinfo(NCTF3)  
[wn,zeta]=damp(NCTF3)

CF =  
  
 10  
  
  
sys =  
   
 2000  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
NCTF1 =  
   
 2000  
 -----------------  
 s^2 + 12 s + 2220  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0245  
 SettlingTime: 0.6206  
 SettlingMin: 0.4993  
 SettlingMax: 1.5026  
 Overshoot: 66.7860  
 Undershoot: 0  
 Peak: 1.5026  
 PeakTime: 0.0667  
  
  
wn =  
  
 47.1169  
 47.1169  
  
  
zeta =  
  
 0.1273  
 0.1273  
  
  
CF =  
   
 s  
   
Continuous-time transfer function.  
  
  
sys =  
   
 200 s  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
NCTF2 =  
   
 200 s  
 -----------------  
 s^2 + 212 s + 220  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0  
 SettlingTime: 3.7813  
 SettlingMin: 6.5963e-04  
 SettlingMax: 0.9234  
 Overshoot: Inf  
 Undershoot: 0  
 Peak: 0.9234  
 PeakTime: 0.0253  
  
  
wn =  
  
 1.0429  
 210.9571  
  
  
zeta =  
  
 1  
 1  
  
  
CF =  
   
 1  
 -  
 s  
   
Continuous-time transfer function.  
  
  
sys =  
   
 200  
 --------------------  
 s^3 + 12 s^2 + 220 s  
   
Continuous-time transfer function.  
  
  
NCTF3 =  
   
 200  
 --------------------------  
 s^3 + 12 s^2 + 220 s + 200  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.2719  
 SettlingTime: 4.1463  
 SettlingMin: 0.9044  
 SettlingMax: 0.9993  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9993  
 PeakTime: 7.6683  
  
  
wn =  
  
 0.9549  
 14.4725  
 14.4725  
  
  
zeta =  
  
 1.0000  
 0.3816  
 0.3816



## Positive Feedback

figure  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
PCTF1=feedback(sys,-1)  
subplot(3,2,1)  
step(PCTF1)  
title("Step with positive")  
subplot(3,2,2)  
impulse(PCTF1)  
title("impulse with positive")  
S = stepinfo(PCTF1)  
[wn,zeta]=damp(PCTF1)  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1,0],[1])  
sys = CF\*TF  
PCTF2=feedback(sys,-1)  
subplot(3,2,3)  
step(PCTF2)  
title("Step with diff")  
subplot(3,2,4)  
impulse(PCTF2)  
title("impulse with diff")  
S = stepinfo(PCTF2)  
[wn,zeta]=damp(PCTF2)  
  
  
  
  
J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=tf([1],[1,0])  
sys = CF\*TF  
PCTF3=feedback(sys,-1)  
subplot(3,2,5)  
step(PCTF3)  
title("Step with integrator")  
subplot(3,2,6)  
impulse(PCTF3)  
title("impulse with integrator")  
S = stepinfo(PCTF3)  
[wn,zeta]=damp(PCTF3)

CF =  
  
 10  
  
  
sys =  
   
 2000  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
PCTF1 =  
   
 2000  
 -----------------  
 s^2 + 12 s - 1780  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
wn =  
  
 36.6146  
 48.6146  
  
  
zeta =  
  
 -1  
 1  
  
  
CF =  
   
 s  
   
Continuous-time transfer function.  
  
  
sys =  
   
 200 s  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
PCTF2 =  
   
 200 s  
 -----------------  
 s^2 - 188 s + 220  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
wn =  
  
 1.1776  
 186.8224  
  
  
zeta =  
  
 -1  
 -1  
  
  
CF =  
   
 1  
 -  
 s  
   
Continuous-time transfer function.  
  
  
sys =  
   
 200  
 --------------------  
 s^3 + 12 s^2 + 220 s  
   
Continuous-time transfer function.  
  
  
PCTF3 =  
   
 200  
 --------------------------  
 s^3 + 12 s^2 + 220 s - 200  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
wn =  
  
 0.8653  
 15.2030  
 15.2030  
  
  
zeta =  
  
 -1.0000  
 0.4231  
 0.4231

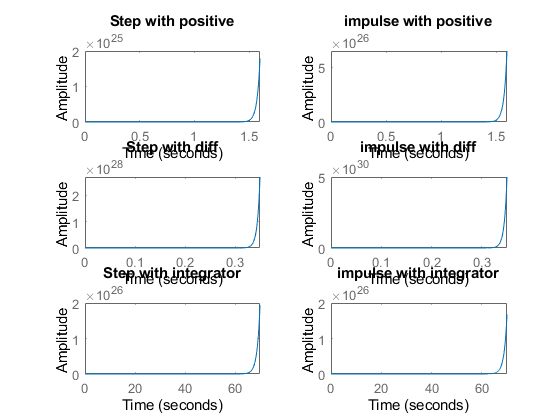
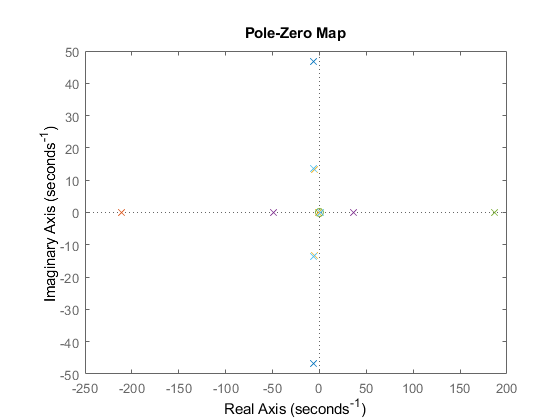


figure  
hold on  
pzmap(NCTF1)  
pzmap(NCTF2)  
pzmap(NCTF3)  
pzmap(PCTF1)  
pzmap(PCTF2)  
pzmap(PCTF3)



## Analysis

%1. Positive feedback system when P,I,D are added system becomes unstable.  
%2. Rise time will decrease when you add a differentiator because over  
%shoot increases, Ts also increases.  
%3. When we add an integrator to this system rise time bacame higher and  
%overshoot became zero this says that system is getting towards stable.  
%4. Adding the positive feed back makes the zeta value change.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

## Title:ControlSystemsecondorder:negative fb with different parameter values

%Author:Shivakumar Naga Vankadhara  
%PS No:99003727  
%Date:10/04/2021  
%Version:1.4

## This Document has equation for DC motor system

%Equation1:vi=IR+L(di/dt)+kw  
%Equation2:J(dw/dt)+bw=kI

## Math Analysis

Independent variables: T Dependent Variables:w,I Constants:L,K,R

%Roots:-(((RJ+bL)/JL)+-(2((R^2\*J^2+b^2\*L^2+2JbL)/J^2\*L^2)-4((bR+k^2)/JL))^1/2)/2

J = 0.01;  
b = 0.1;  
K = 1;  
R = 1;  
L = 0.5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
NCTF=feedback(sys,1)  
subplot(4,2,1)  
step(NCTF)  
title("Step 1")  
subplot(4,2,2)  
impulse(NCTF)  
title("impulse1")  
S = stepinfo(NCTF)  
[wn,zeta]=damp(NCTF)  
  
J = 0.1;  
b = 1;  
K = 0.1;  
R = 10;  
L = 5;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
NCTF1=feedback(sys,1)  
subplot(4,2,3)  
step(NCTF1)  
title("Step 2")  
subplot(4,2,4)  
impulse(NCTF1)  
title("impulse 2")  
S = stepinfo(NCTF1)  
[wn,zeta]=damp(NCTF1)  
  
J = 0.01;  
b = 0.01;  
K = 0.1;  
R = 0.1;  
L = 0.05;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
NCTF2=feedback(sys,1)  
subplot(4,2,5)  
step(NCTF2)  
title("Step 3")  
subplot(4,2,6)  
impulse(NCTF2)  
title("impulse 3")  
S = stepinfo(NCTF2)  
[wn,zeta]=damp(NCTF2)  
  
  
  
J = -0.01;  
b = -0.01;  
K = -0.1;  
R = -0.1;  
L = -0.05;  
TF=tf([K/(J\*L)],[1,((b/J)+(R/L)),(((K\*K)+(R\*b))/(L\*J))]);  
CF=10  
sys = CF\*TF  
NCTF3=feedback(sys,1)  
subplot(4,2,7)  
step(NCTF3)  
title("Step 3")  
subplot(4,2,8)  
impulse(NCTF3)  
title("impulse 3")  
S = stepinfo(NCTF3)  
[wn,zeta]=damp(NCTF3)

CF =  
  
 10  
  
  
sys =  
   
 2000  
 ----------------  
 s^2 + 12 s + 220  
   
Continuous-time transfer function.  
  
  
NCTF =  
   
 2000  
 -----------------  
 s^2 + 12 s + 2220  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0245  
 SettlingTime: 0.6206  
 SettlingMin: 0.4993  
 SettlingMax: 1.5026  
 Overshoot: 66.7860  
 Undershoot: 0  
 Peak: 1.5026  
 PeakTime: 0.0667  
  
  
wn =  
  
 47.1169  
 47.1169  
  
  
zeta =  
  
 0.1273  
 0.1273  
  
  
CF =  
  
 10  
  
  
sys =  
   
 2  
 ------------------  
 s^2 + 12 s + 20.02  
   
Continuous-time transfer function.  
  
  
NCTF1 =  
   
 2  
 ------------------  
 s^2 + 12 s + 22.02  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.0161  
 SettlingTime: 1.8471  
 SettlingMin: 0.0819  
 SettlingMax: 0.0907  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.0907  
 PeakTime: 3.0168  
  
  
wn =  
  
 2.2610  
 9.7390  
  
  
zeta =  
  
 1  
 1  
  
  
CF =  
  
 10  
  
  
sys =  
   
 2000  
 --------------  
 s^2 + 3 s + 22  
   
Continuous-time transfer function.  
  
  
NCTF2 =  
   
 2000  
 ----------------  
 s^2 + 3 s + 2022  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0238  
 SettlingTime: 2.5921  
 SettlingMin: 0.1871  
 SettlingMax: 1.8798  
 Overshoot: 90.0453  
 Undershoot: 0  
 Peak: 1.8798  
 PeakTime: 0.0699  
  
  
wn =  
  
 44.9667  
 44.9667  
  
  
zeta =  
  
 0.0334  
 0.0334  
  
  
CF =  
  
 10  
  
  
sys =  
   
 -2000  
 --------------  
 s^2 + 3 s + 22  
   
Continuous-time transfer function.  
  
  
NCTF3 =  
   
 -2000  
 ----------------  
 s^2 + 3 s - 1978  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
wn =  
  
 43  
 46  
  
  
zeta =  
  
 -1  
 1

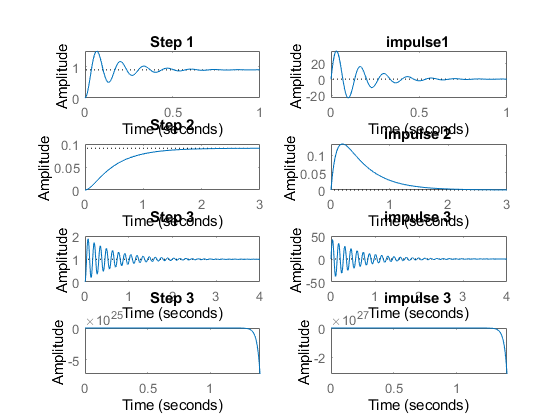
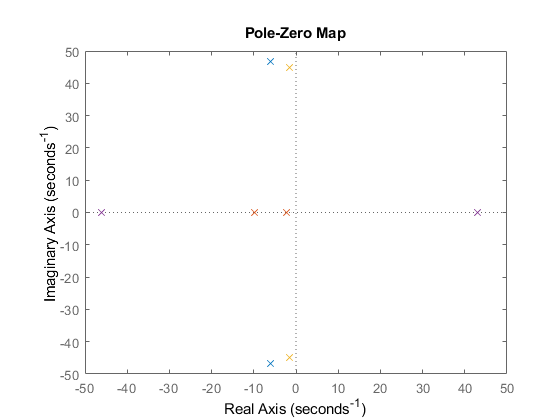


figure  
hold on  
pzmap(NCTF)  
pzmap(NCTF1)  
pzmap(NCTF2)  
pzmap(NCTF3)



## Analysis:

%1. For negative variables the root of a system becomes positive so the syste  
%m is unstable.  
%2. Rise time of negative feedback closed loop system is less when compared  
% to open loop system of the same second order.  
%3. Zeros & Poles locations got changed when we added a negative feed back.  
%4. System becomes under damped  
%5. Overshoot is high when compared to open loop system.  
%6. For the 3rd negative variables risetime, passtime every other parametrs  
%becomes inf.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

normal 69

pi 71

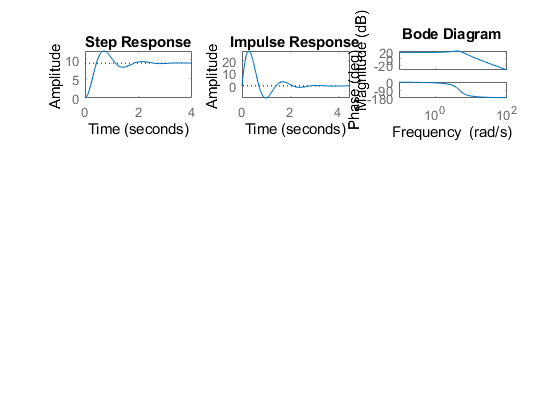
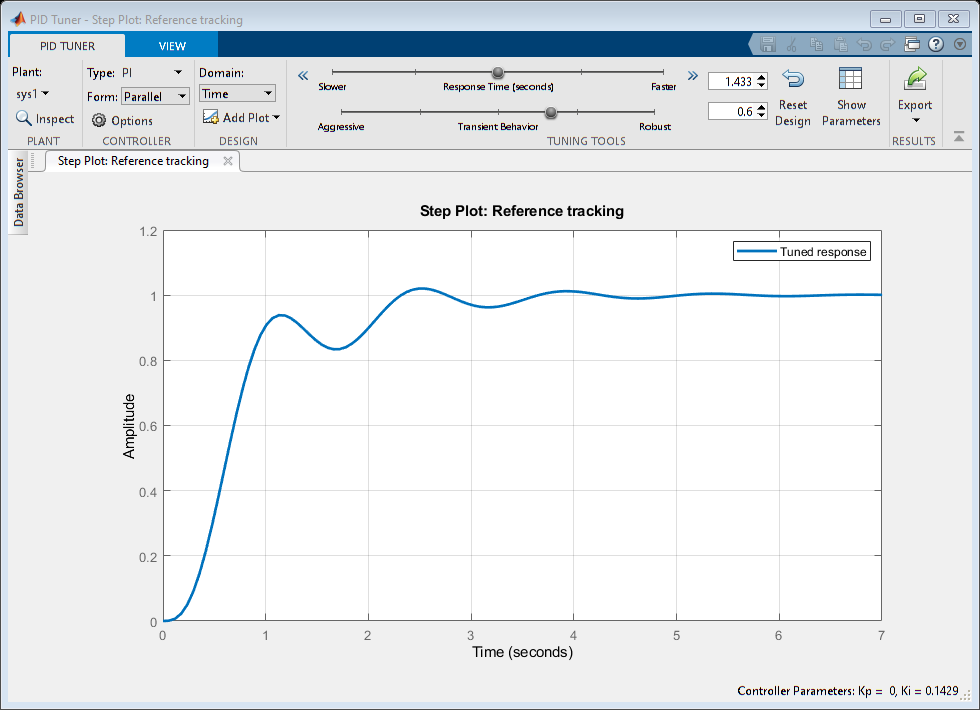
PD 73

PID 75

## normal

J1 = 0.01;  
b1 = 0.01;  
K1 = 0.1;  
R1 = 0.1;  
L1 = 0.05;  
sys1 = tf([K1/(J1\*L1)],[1,((b1/J1)+(R1/L1)),(((K1\*K1)+(R1\*b1))/(L1\*J1))])  
subplot(4,3,1)  
step(sys1)  
subplot(4,3,2)  
impulse(sys1)  
subplot(4,3,3)  
S = stepinfo(sys1)  
pzmap(sys1)  
 pidTuner(sys1)  
 bode(sys1)

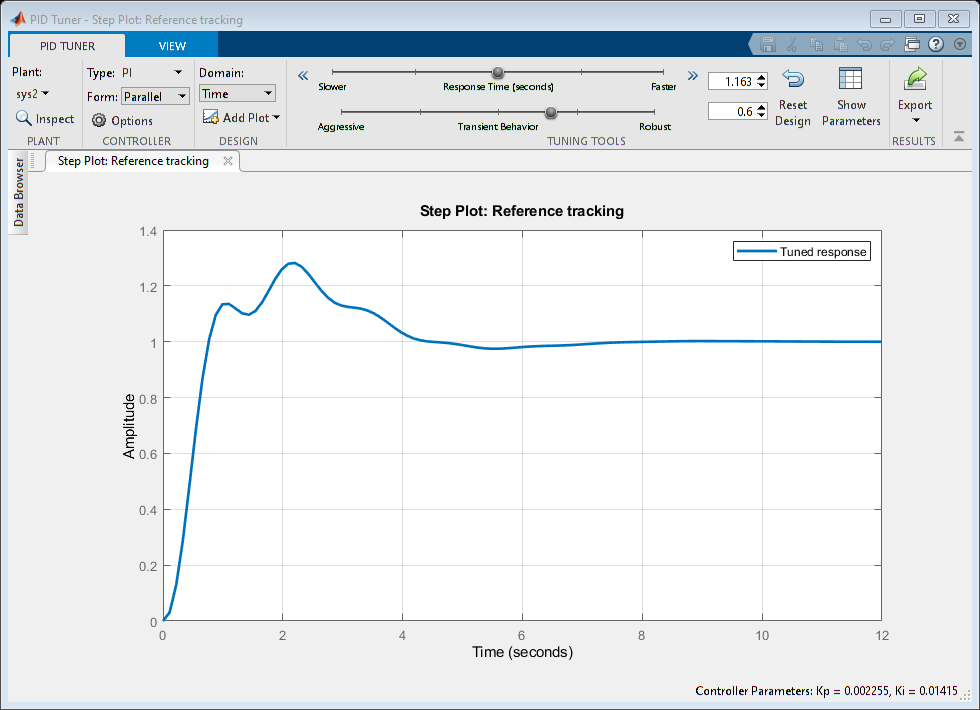
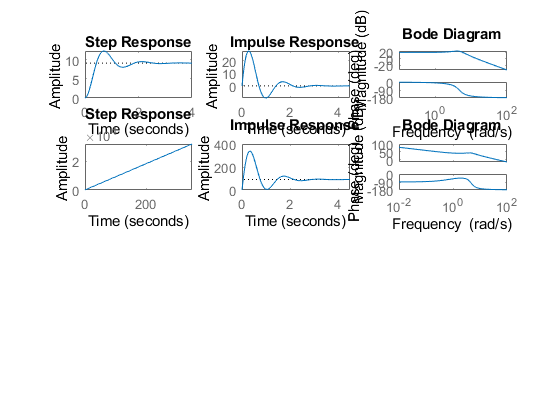
sys1 =  
   
 200  
 --------------  
 s^2 + 3 s + 22  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.2882  
 SettlingTime: 2.3810  
 SettlingMin: 8.0006  
 SettlingMax: 12.2393  
 Overshoot: 34.6325  
 Undershoot: 0  
 Peak: 12.2393  
 PeakTime: 0.7061



## pi

J2 = 0.01;  
b2 = 0.01;  
K2 = 0.1;  
R2 = 0.1;  
L2 = 0.05;  
Kp=10;  
I=tf([10],[1,0]); %Ki  
PI=Kp+I;  
sys2 = tf([K2/(J2\*L2)],[1,((b2/J2)+(R2/L2)),(((K2\*K2)+(R2\*b2))/(L2\*J2))])\*(PI)  
subplot(4,3,4)  
step(sys2)  
subplot(4,3,5)  
impulse(sys2)  
subplot(4,3,6)  
S = stepinfo(sys2)  
pzmap(sys2)  
 pidTuner(sys2)  
bode(sys2)

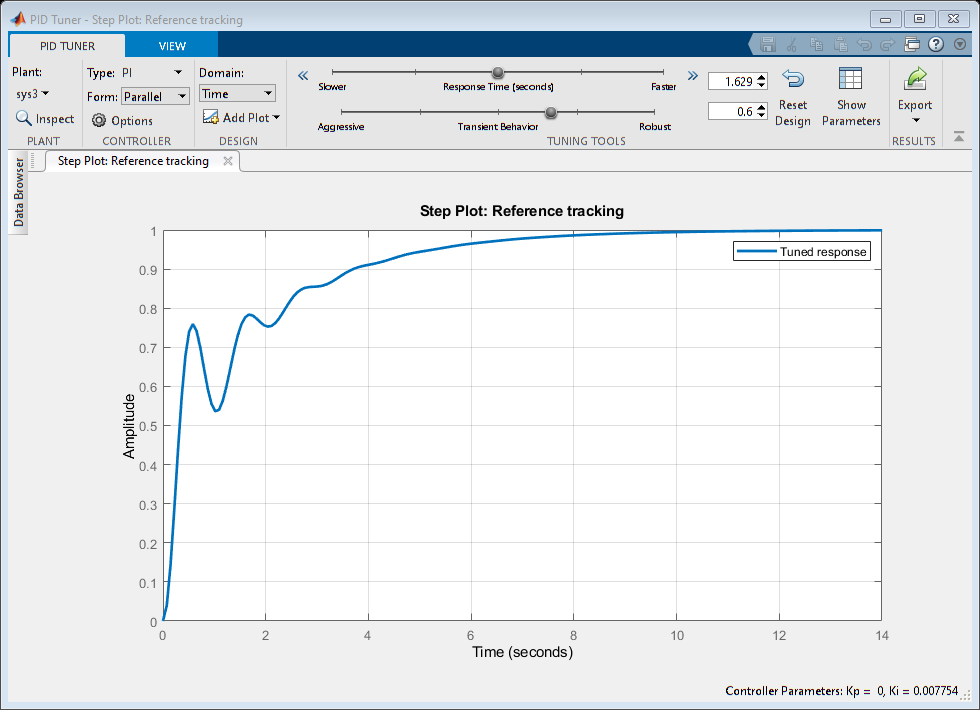
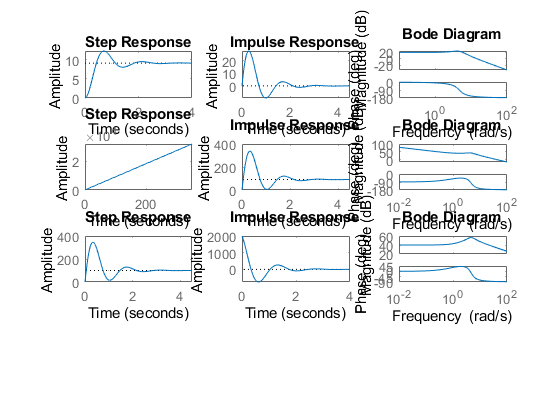
sys2 =  
   
 2000 s + 2000  
 ------------------  
 s^3 + 3 s^2 + 22 s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf



## PD

J3 = 0.01;  
b3 = 0.01;  
K3 = 0.1;  
R3 = 0.1;  
L3 = 0.05;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
PD=Kp+D;  
sys3 = tf([K3/(J3\*L3)],[1,((b3/J3)+(R3/L3)),(((K3\*K3)+(R3\*b3))/(L3\*J3))])\*(PD)  
subplot(4,3,7)  
step(sys3)  
subplot(4,3,8)  
impulse(sys3)  
subplot(4,3,9)  
S = stepinfo(sys3)  
pzmap(sys3)  
 pidTuner(sys3);  
 bode(sys3)

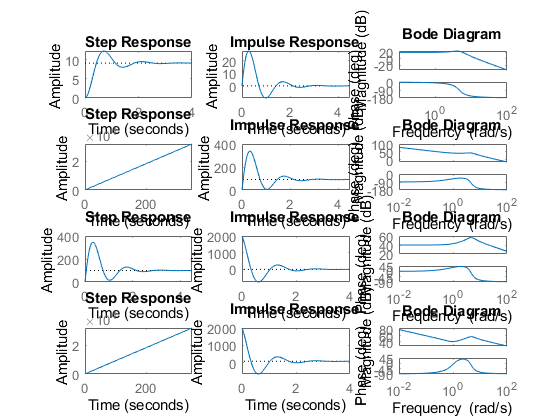
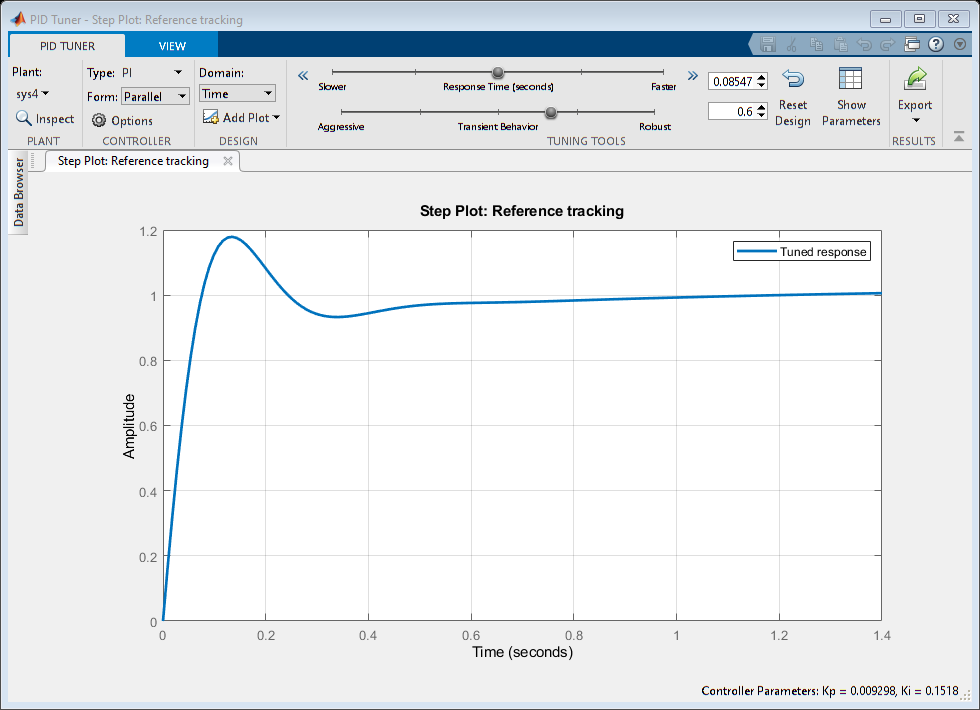
sys3 =  
   
 2000 s + 2200  
 --------------  
 s^2 + 3 s + 22  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.0426  
 SettlingTime: 2.7143  
 SettlingMin: 14.7945  
 SettlingMax: 346.0086  
 Overshoot: 246.0086  
 Undershoot: 0  
 Peak: 346.0086  
 PeakTime: 0.3377



## PID

J4 = 0.01;  
b4 = 0.01;  
K4 = 0.1;  
R4 = 0.1;  
L4 = 0.05;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
I=tf([10],[1,0]); %Ki  
PID=Kp+D+I;  
sys4 = tf([K4/(J4\*L4)],[1,((b4/J4)+(R4/L4)),(((K4\*K4)+(R4\*b4))/(L4\*J4))])\*(PID)  
subplot(4,3,10)  
step(sys4)  
subplot(4,3,11)  
impulse(sys4)  
subplot(4,3,12)  
S = stepinfo(sys4)  
pzmap(sys4)  
 pidTuner(sys4)  
 bode(sys4)

sys4 =  
   
 2000 s^2 + 2200 s + 2000  
 ------------------------  
 s^3 + 3 s^2 + 22 s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf



[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

## Title:Control System-Individual System(Thermometer)

%Author:ShivaKumar Naga Vankadhara  
%PS No:99003727  
%Date:12/04/2021  
%Version:1.0

## This Document has equation for First Order Thermometer Equation

%Equation:Tdm/dt+m=tem  
%T\_F=1/Ts+1

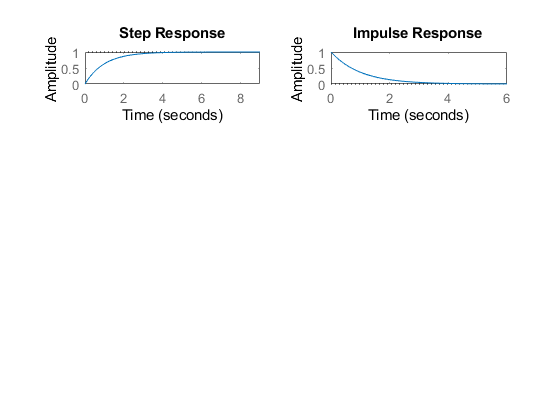
## Math analysis

%dependent variables:m,temp  
%independent variables:t  
%constant:T  
%Roots:-1/T

## Basic

T=1  
sys1 = tf([1],[T,1])  
subplot(5,2,1)  
step(sys1)  
subplot(5,2,2)  
impulse(sys1)  
S = stepinfo(sys1)  
p1=pole(sys1)  
z1=zero(sys1)

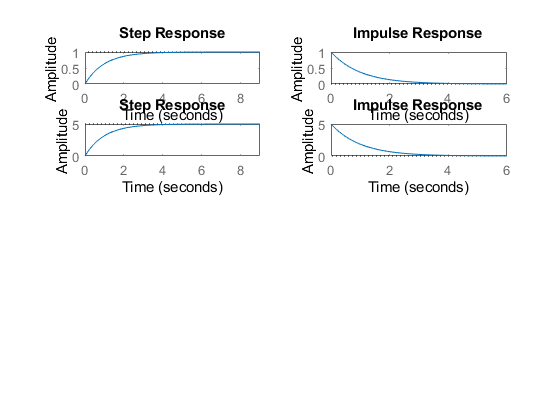
T =  
  
 1  
  
  
sys1 =  
   
 1  
 -----  
 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.1970  
 SettlingTime: 3.9121  
 SettlingMin: 0.9045  
 SettlingMax: 1.0000  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 1.0000  
 PeakTime: 10.5458  
  
  
p1 =  
  
 -1  
  
  
z1 =  
  
 0×1 empty double column vector



## With Gain

T=1;  
k=5;  
sys\_G = k\*tf([1],[T,1])  
subplot(5,2,3)  
step(sys\_G)  
subplot(5,2,4)  
impulse(sys\_G)  
S = stepinfo(sys\_G)  
p\_g=pole(sys\_G)  
z\_g=zero(sys\_G)

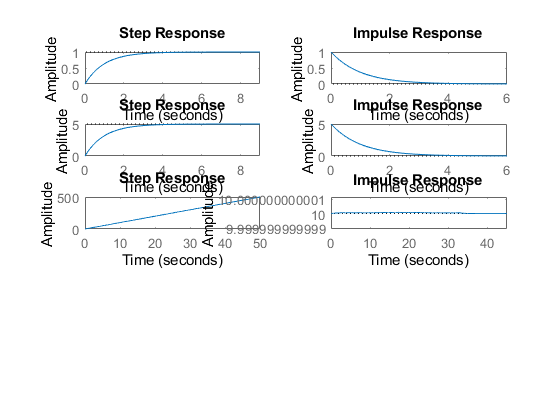
sys\_G =  
   
 5  
 -----  
 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.1970  
 SettlingTime: 3.9121  
 SettlingMin: 4.5225  
 SettlingMax: 4.9999  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 4.9999  
 PeakTime: 10.5458  
  
  
p\_g =  
  
 -1  
  
  
z\_g =  
  
 0×1 empty double column vector



## With PI

T=1;  
k=5;  
Kp=10;  
I=tf([10],[1,0]); %Ki  
PI=Kp+I;  
sys\_PI = PI\*tf([1],[T,1])  
subplot(5,2,5)  
step(sys\_PI)  
subplot(5,2,6)  
impulse(sys\_PI)  
S = stepinfo(sys\_PI)  
p\_pi=pole(sys\_PI)  
z\_pi=zero(sys\_PI)

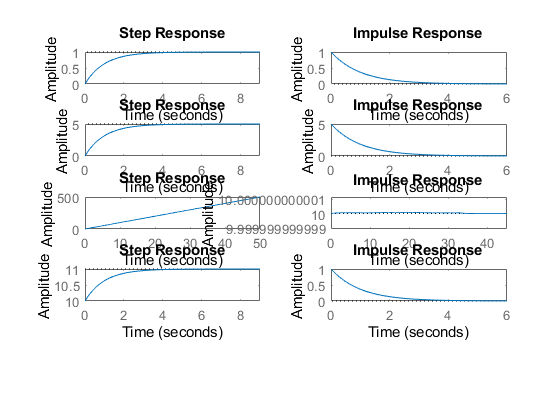
sys\_PI =  
   
 10 s + 10  
 ---------  
 s^2 + s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p\_pi =  
  
 0  
 -1  
  
  
z\_pi =  
  
 -1



## With PD

T=1;  
k=5;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
PD=Kp+D;  
sys\_PD = PD\*tf([1],[T,1])  
subplot(5,2,7)  
step(sys\_PD)  
subplot(5,2,8)  
impulse(sys\_PD)  
S = stepinfo(sys\_PD)  
p\_pd=pole(sys\_PD)  
z\_pd=zero(sys\_PD)

sys\_PD =  
   
 10 s + 11  
 ---------  
 s + 1  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.1970  
 SettlingTime: 3.9121  
 SettlingMin: 10.9045  
 SettlingMax: 11.0000  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 11.0000  
 PeakTime: 10.5458  
  
  
p\_pd =  
  
 -1  
  
  
z\_pd =  
  
 -1.1000



## With PID

T=1;  
k=5;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
I=tf([10],[1,0]); %Ki  
PID=Kp+D+I;  
sys\_PID = PID\*tf([1],[T,1])  
subplot(5,2,9)  
step(sys\_PID)  
subplot(5,2,10)  
impulse(sys\_PID)  
S = stepinfo(sys\_PID)  
p\_pid=pole(sys\_PID)  
z\_pid=zero(sys\_PID)

sys\_PID =  
   
 10 s^2 + 11 s + 10  
 ------------------  
 s^2 + s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p\_pid =  
  
 0  
 -1  
  
  
z\_pid =  
  
 -0.5500 + 0.8352i  
 -0.5500 - 0.8352i

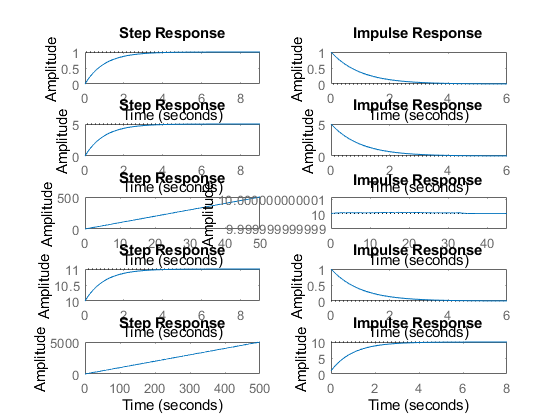
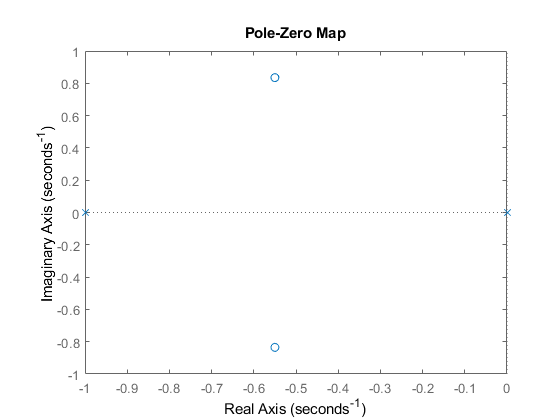
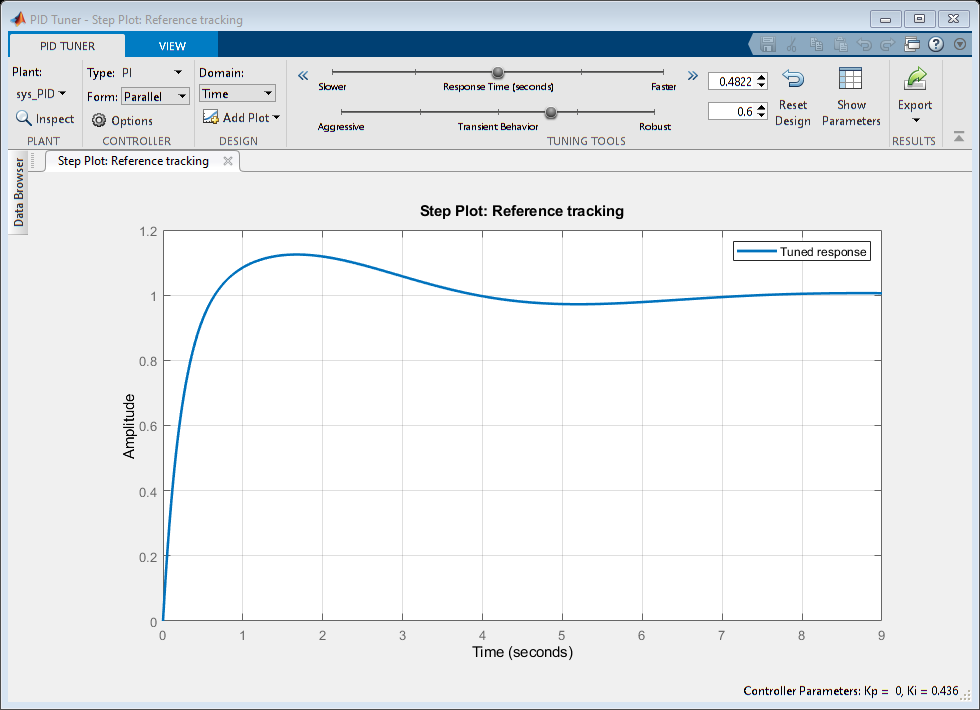
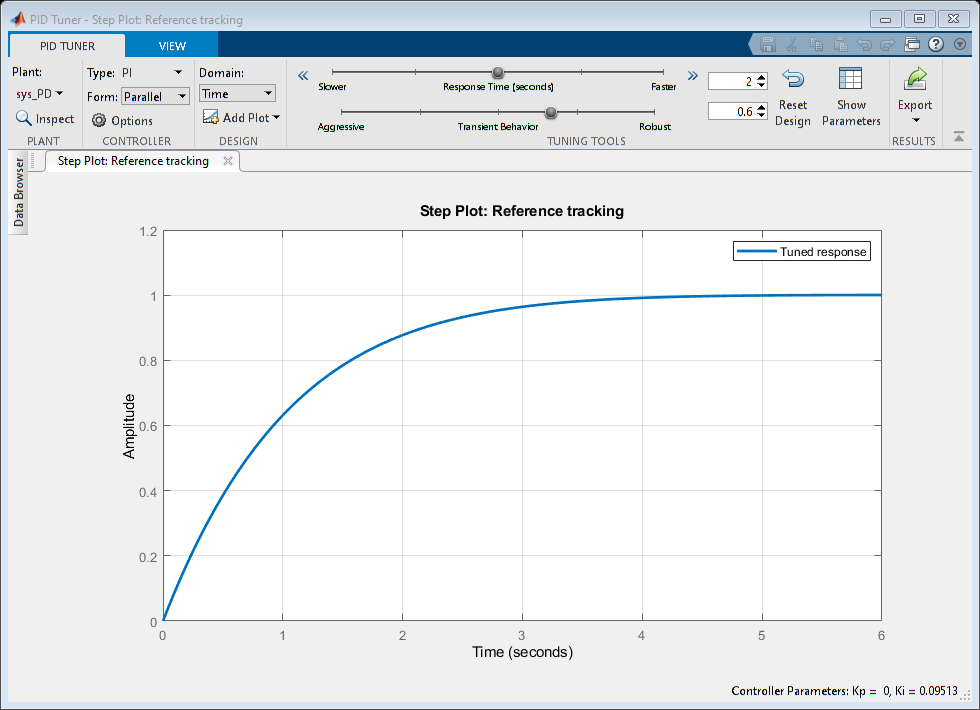
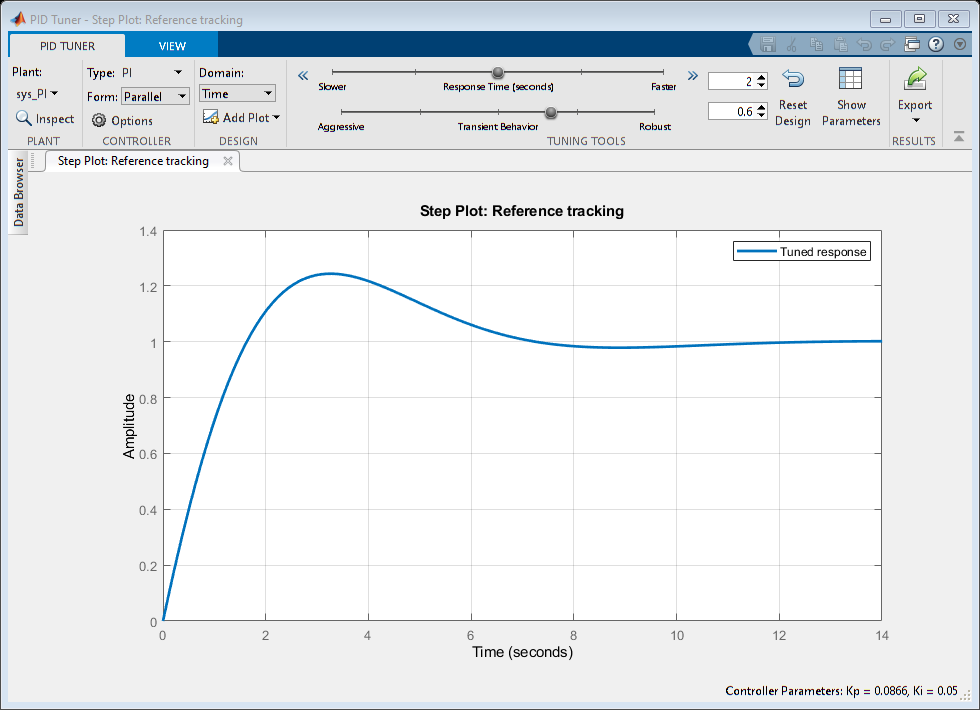
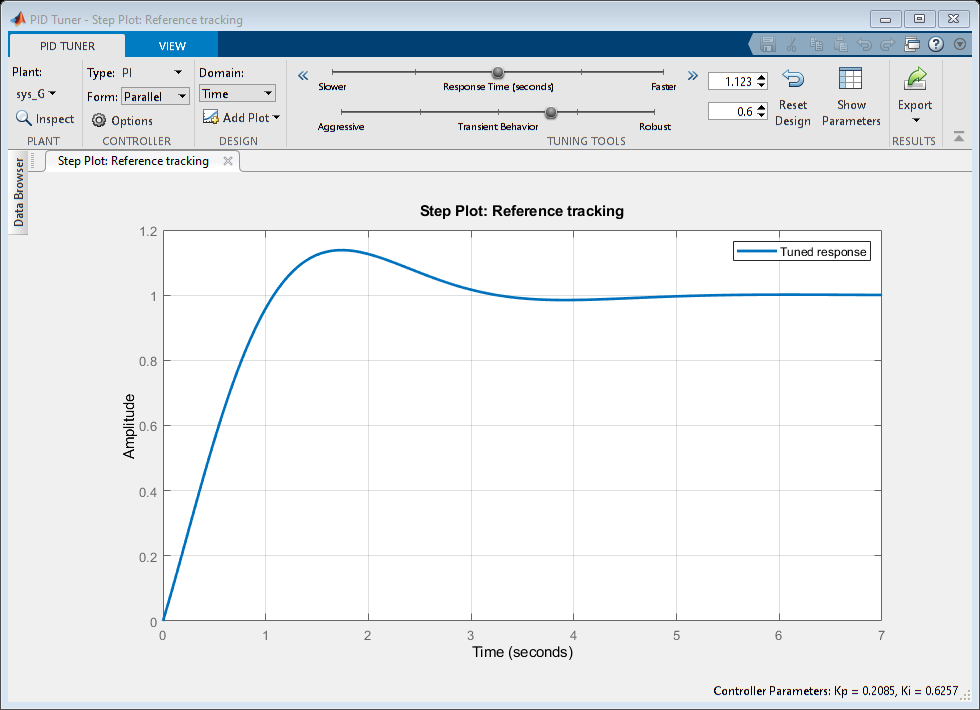
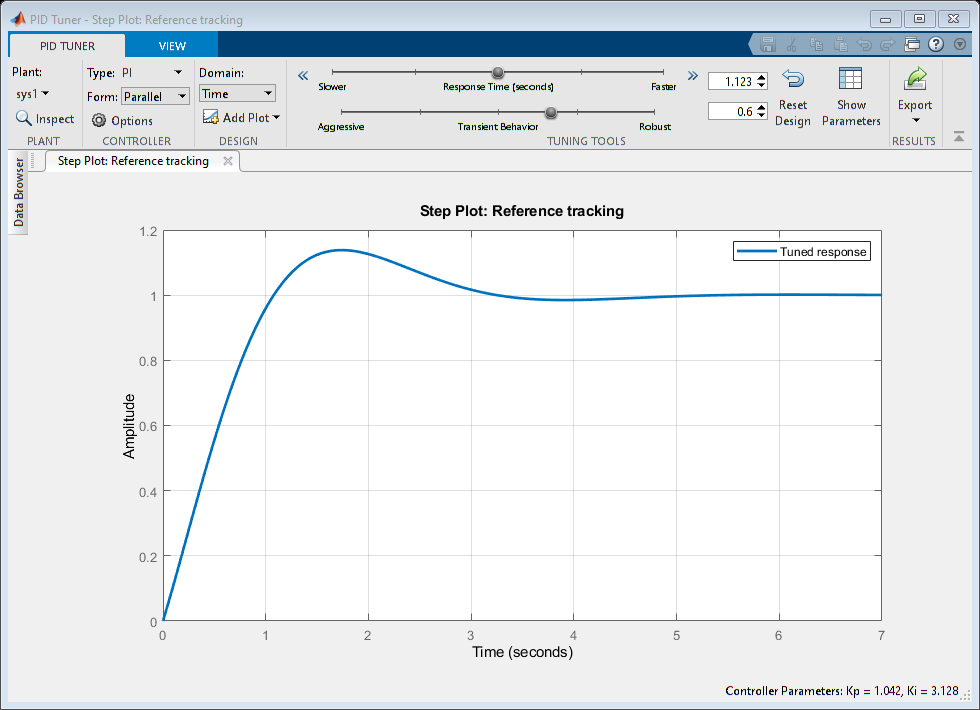


figure  
pzmap(sys1)  
pzmap(sys\_G)  
pzmap(sys\_PI)  
pzmap(sys\_PD)  
pzmap(sys\_PID)  
  
  
pidTuner(sys1)  
pidTuner(sys\_G)  
pidTuner(sys\_PI)  
pidTuner(sys\_PD)  
pidTuner(sys\_PID)



## Analysis

%1.For the Basic the root lies on the left side of the imaginary axis that  
% means the system is stable.  
%Rise time is : 2.1970  
%settling time is:3.9121 & Overshoot=0 for the basic  
%2. For the system with gain also the root lies on the left side of the  
%imaginary axis that means the system is stable.  
%Rise time is:2.1970, settling time:3.9121, overshoot=0 for the gain. poles  
%is also same only there is a change of amplitude.  
%3. For the system with PI we got 2 poles one pole is at p1=0, p2=-1 and  
%one zero is at z=-1 so we can say that 1 pole will nullify the effect of  
%zero and we will be remained with 1 pole left on the left side so we can  
%say that system is stable.  
%4. For the system with PD we got 1 pole at -1 and 1 zero at -1.10000 on  
%the left side of imaginary axis the settling time is 2.1970, R\_t is 3.9121  
%5. For the system with PID controller we got 2 poles and 2 zeroes p1=0,  
%p1=-1 and z1=-0.5500+0.8352i,z2=-0.5500-0.8352i the poles and zeores le on  
%the left side of the imaginary axis again the system is stable again here  
%also.

## With POsitive feedback

figure  
T=1  
sys = tf([1],[T,1])  
sys\_P=feedback(sys,-1)  
subplot(5,2,1)  
step(sys\_P)  
subplot(5,2,2)  
impulse(sys\_P)  
S = stepinfo(sys\_P)  
p1=pole(sys\_P)  
z1=zero(sys\_P)  
  
T=1;  
CF=10;  
sys = CF\*tf([1],[T,1]);  
sys\_G\_P=feedback(sys,-1);  
subplot(5,2,3)  
step(sys\_G\_P)  
subplot(5,2,4)  
impulse(sys\_G\_P)  
S = stepinfo(sys\_G\_P)  
p\_g=pole(sys\_G\_P)  
z\_g=zero(sys\_G\_P)  
  
T=1;  
Kp=10;  
I=tf([10],[1,0]); %Ki  
PI=Kp+I;  
sys = PI\*tf([1],[T,1]);  
sys\_PI\_P=feedback(sys,-1);  
subplot(5,2,5)  
step(sys\_PI\_P)  
subplot(5,2,6)  
impulse(sys\_PI\_P)  
S = stepinfo(sys\_PI\_P)  
p\_pi=pole(sys\_PI\_P)  
z\_pi=zero(sys\_PI\_P)  
  
T=1;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
PD=Kp+D;  
sys = PD\*tf([1],[T,1]);  
sys\_PD\_P=feedback(sys,-1);  
subplot(5,2,7)  
step(sys\_PD\_P)  
subplot(5,2,8)  
impulse(sys\_PD\_P)  
S = stepinfo(sys\_PD\_P)  
p\_pd=pole(sys\_PD\_P)  
z\_pd=zero(sys\_PD\_P)  
  
T=1  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
I=tf([10],[1,0]); %Ki  
PID=Kp+D+I;  
sys = PID\*tf([1],[T,1]);  
sys\_PID\_P=feedback(sys,-1);  
subplot(5,2,9)  
step(sys\_PID\_P)  
subplot(5,2,10)  
impulse(sys\_PID\_P)  
S = stepinfo(sys\_PID\_P)  
p\_pid=pole(sys\_PID\_P)  
z\_pid=zero(sys\_PID\_P)

T =  
  
 1  
  
  
sys =  
   
 1  
 -----  
 s + 1  
   
Continuous-time transfer function.  
  
  
sys\_P =  
   
 1  
 -  
 s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p1 =  
  
 0  
  
  
z1 =  
  
 0×1 empty double column vector  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p\_g =  
  
 9  
  
  
z\_g =  
  
 0×1 empty double column vector  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: NaN  
 SettlingTime: NaN  
 SettlingMin: NaN  
 SettlingMax: NaN  
 Overshoot: NaN  
 Undershoot: NaN  
 Peak: Inf  
 PeakTime: Inf  
  
  
p\_pi =  
  
 10  
 -1  
  
  
z\_pi =  
  
 -1  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.9773  
 SettlingTime: 3.5209  
 SettlingMin: -1.1011  
 SettlingMax: -1.1000  
 Overshoot: 1.0101  
 Undershoot: 0  
 Peak: 1.1111  
 PeakTime: 0  
  
  
p\_pd =  
  
 -1.1111  
  
  
z\_pd =  
  
 -1.1000  
  
  
T =  
  
 1  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.5943  
 SettlingTime: 7.1081  
 SettlingMin: -1.0101  
 SettlingMax: -0.9841  
 Overshoot: 11.1111  
 Undershoot: 0  
 Peak: 1.1111  
 PeakTime: 0  
  
  
p\_pid =  
  
 -0.5556 + 0.8958i  
 -0.5556 - 0.8958i  
  
  
z\_pid =  
  
 -0.5500 + 0.8352i  
 -0.5500 - 0.8352i

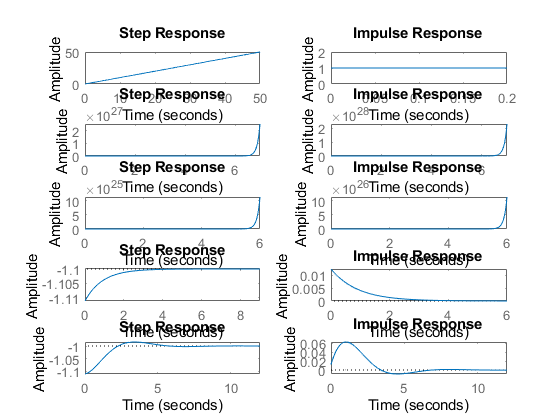
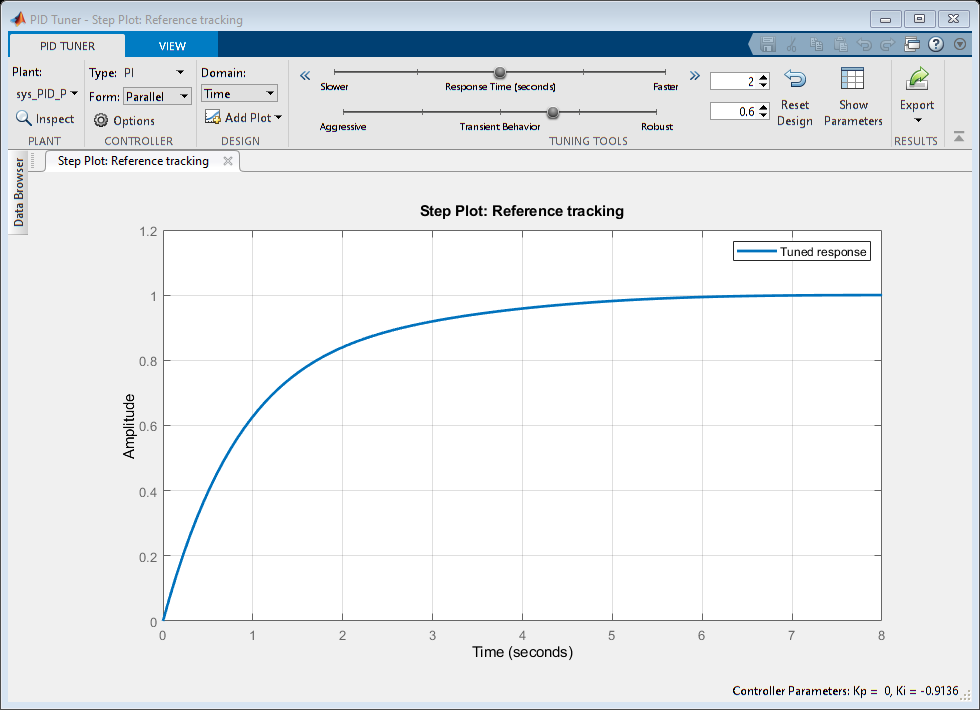
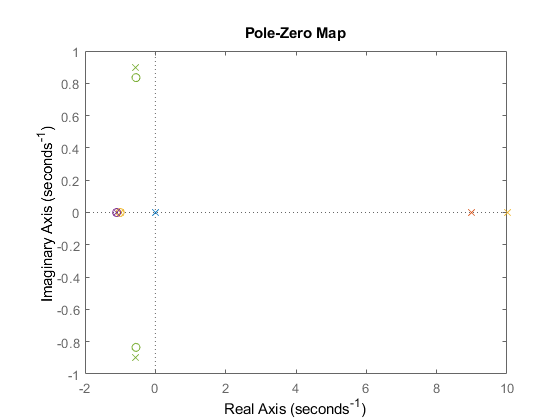
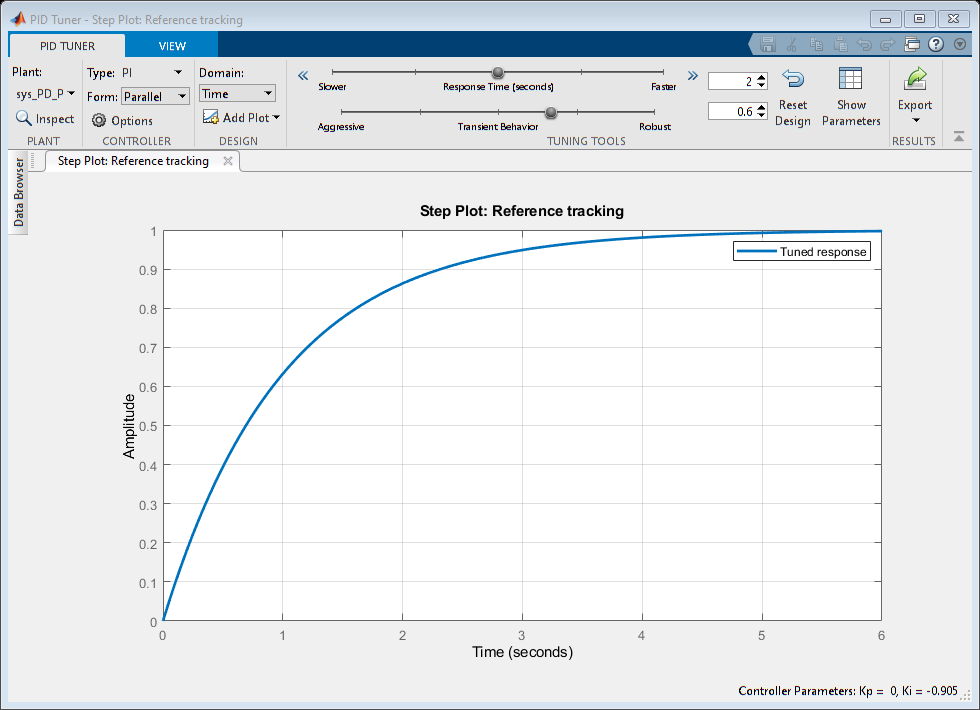
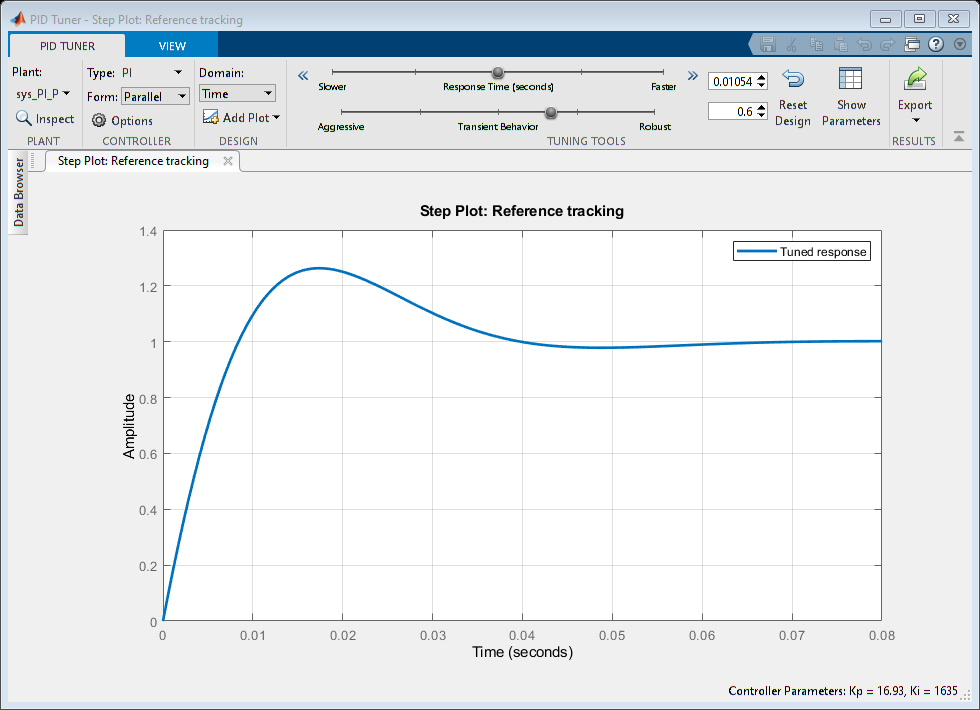
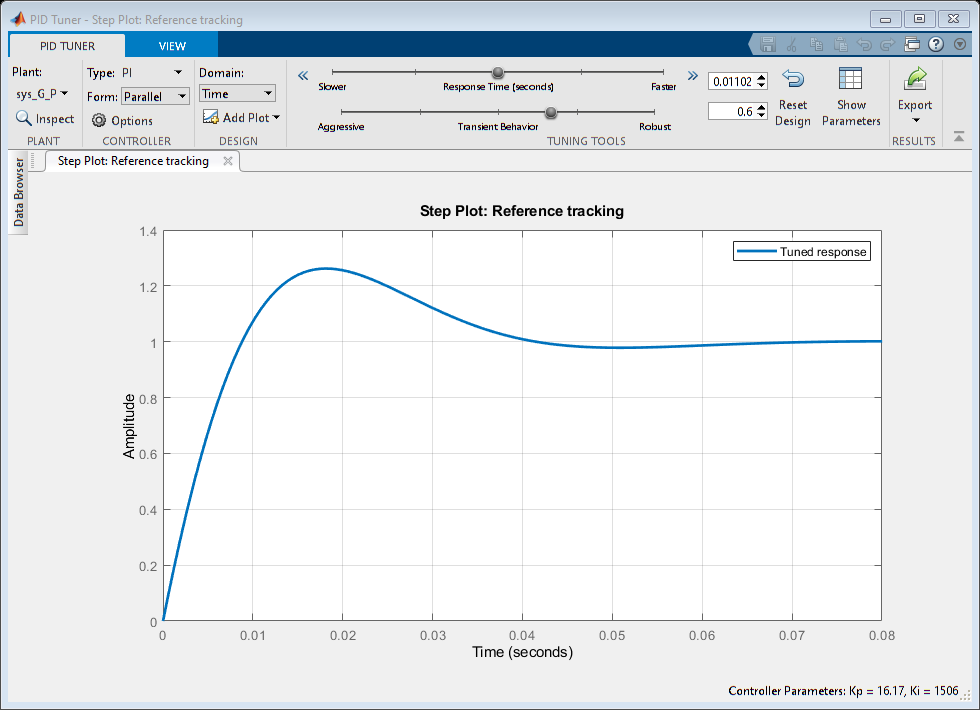
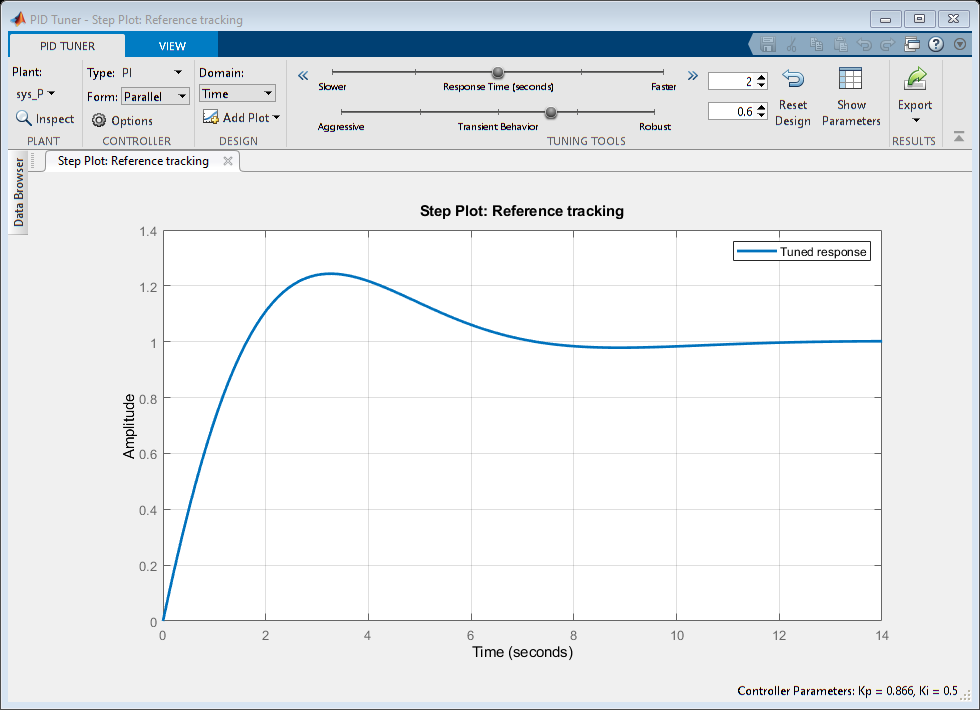


figure  
hold on  
pzmap(sys\_P)  
pzmap(sys\_G\_P)  
pzmap(sys\_PI\_P)  
pzmap(sys\_PD\_P)  
pzmap(sys\_PID\_P)  
  
  
pidTuner(sys\_P)  
pidTuner(sys\_G\_P)  
pidTuner(sys\_PI\_P)  
pidTuner(sys\_PD\_P)  
pidTuner(sys\_PID\_P)



## Analysis

1.With the positive feed back system by giving the gain as 10 we got a

%pole at p=9 that says that system is unstable.  
% 2.with the Positive feed back system by givng the PI controller we got 2  
% poles 1 at p1=10,p2=-1 and 1 zero at z1=-1 so the pole and one zero  
% nullify each other and left a pole on the left side of imaginary axis  
% making the system stable.  
% 3.With the Pd controller we can see that 1 zero is getting added, and 1  
% pole is getting fixated at -1.1111 and a zero at -1.10000 as pole is  
% located at the left side of the imaginary axis the system is stable with  
% a rise time 1.9773, and settling time of 3.5209 with a overshoot of 1.010  
% 4.With the PID controller we can see that w eare getting complex  
% conjugate poles and pair. p1=-0.5556+0.8958i,p2=-0.5556-0.8958i and  
% zeroes arwe z1=-0.5500+0.8352i, z2=-0.5500-0.8352i anfd the s\_t=7.1081,  
% R\_t=1.5943  
% 5.So By observing the above mentioned settling time and rise time of the  
% different controllers we are getting a stable system with PID controller.

## With Negative feedback

figure  
T=1;  
sys = tf([1],[T,1])  
sys\_N=feedback(sys,1)  
subplot(5,2,1)  
step(sys\_N)  
subplot(5,2,2)  
impulse(sys\_N)  
S = stepinfo(sys\_N)  
p\_n=pole(sys\_N)  
z\_n=zero(sys\_N)  
  
T=1;  
CF=10;  
sys = CF\*tf([1],[T,1])  
sys\_G\_N=feedback(sys,1)  
subplot(5,2,3)  
step(sys\_G\_N)  
subplot(5,2,4)  
impulse(sys\_G\_N)  
S = stepinfo(sys\_G\_N)  
p\_gn=pole(sys\_G\_N)  
z\_gn=zero(sys\_G\_N)  
  
T=1;  
Kp=10;  
I=tf([10,0],[1,0]); %Ki  
PI=Kp+I;  
sys = PI\*tf([1],[T,1])  
sys\_PI\_N=feedback(sys,1)  
subplot(5,2,5)  
step(sys\_PI\_N)  
subplot(5,2,6)  
impulse(sys\_PI\_N)  
S = stepinfo(sys\_PI\_N)  
p\_npi=pole(sys\_PI\_N)  
z\_npi=zero(sys\_PI\_N)  
  
T=1;  
Kp=10;  
D=tf([10,1],[0,1]); %Kd  
PD=Kp+D;  
sys = PD\*tf([1],[T,1])  
sys\_PD\_N=feedback(sys,1)  
subplot(5,2,7)  
step(sys\_PD\_N)  
subplot(5,2,8)  
impulse(sys\_PD\_N)  
S = stepinfo(sys\_PD\_N)  
p\_npd=pole(sys\_PD\_N)  
z\_npd=zero(sys\_PD\_N)  
  
T=1;  
Kp=10;  
D=tf([10,1],[0,1]) %Kd  
I=tf([10],[1,0]) %Ki  
PID=Kp+D+I  
sys = PID\*tf([1],[T,1])  
sys\_PID\_N=feedback(sys,1)  
subplot(5,2,9)  
step(sys\_PID\_N)  
subplot(5,2,10)  
impulse(sys\_PID\_N)  
S = stepinfo(sys\_PID\_N)  
p\_npid=pole(sys\_PID\_N)  
z\_npid=zero(sys\_PID\_N)

sys =  
   
 1  
 -----  
 s + 1  
   
Continuous-time transfer function.  
  
  
sys\_N =  
   
 1  
 -----  
 s + 2  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.0985  
 SettlingTime: 1.9560  
 SettlingMin: 0.4523  
 SettlingMax: 0.5000  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.5000  
 PeakTime: 5.2729  
  
  
p\_n =  
  
 -2  
  
  
z\_n =  
  
 0×1 empty double column vector  
  
  
sys =  
   
 10  
 -----  
 s + 1  
   
Continuous-time transfer function.  
  
  
sys\_G\_N =  
   
 10  
 ------  
 s + 11  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.1997  
 SettlingTime: 0.3556  
 SettlingMin: 0.8223  
 SettlingMax: 0.9091  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9091  
 PeakTime: 0.9587  
  
  
p\_gn =  
  
 -11  
  
  
z\_gn =  
  
 0×1 empty double column vector  
  
  
sys =  
   
 20 s  
 -------  
 s^2 + s  
   
Continuous-time transfer function.  
  
  
sys\_PI\_N =  
   
 20 s  
 ----------  
 s^2 + 21 s  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 0.1046  
 SettlingTime: 0.1863  
 SettlingMin: 0.8614  
 SettlingMax: 0.9524  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9524  
 PeakTime: 0.5022  
  
  
p\_npi =  
  
 0  
 -21  
  
  
z\_npi =  
  
 0  
  
  
sys =  
   
 10 s + 11  
 ---------  
 s + 1  
   
Continuous-time transfer function.  
  
  
sys\_PD\_N =  
   
 10 s + 11  
 ---------  
 11 s + 12  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 2.0139  
 SettlingTime: 3.5861  
 SettlingMin: 0.9159  
 SettlingMax: 0.9167  
 Overshoot: 0  
 Undershoot: 0  
 Peak: 0.9167  
 PeakTime: 9.6670  
  
  
p\_npd =  
  
 -1.0909  
  
  
z\_npd =  
  
 -1.1000  
  
  
D =  
   
 10 s + 1  
   
Continuous-time transfer function.  
  
  
I =  
   
 10  
 --  
 s  
   
Continuous-time transfer function.  
  
  
PID =  
   
 10 s^2 + 11 s + 10  
 ------------------  
 s  
   
Continuous-time transfer function.  
  
  
sys =  
   
 10 s^2 + 11 s + 10  
 ------------------  
 s^2 + s  
   
Continuous-time transfer function.  
  
  
sys\_PID\_N =  
   
 10 s^2 + 11 s + 10  
 ------------------  
 11 s^2 + 12 s + 10  
   
Continuous-time transfer function.  
  
  
S =   
  
 struct with fields:  
  
 RiseTime: 1.8654  
 SettlingTime: 6.0686  
 SettlingMin: 0.9929  
 SettlingMax: 1.0102  
 Overshoot: 1.0208  
 Undershoot: 0  
 Peak: 1.0102  
 PeakTime: 3.8837  
  
  
p\_npid =  
  
 -0.5455 + 0.7820i  
 -0.5455 - 0.7820i  
  
  
z\_npid =  
  
 -0.5500 + 0.8352i  
 -0.5500 - 0.8352i

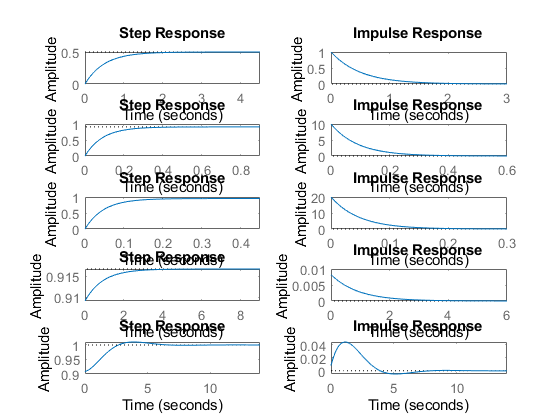
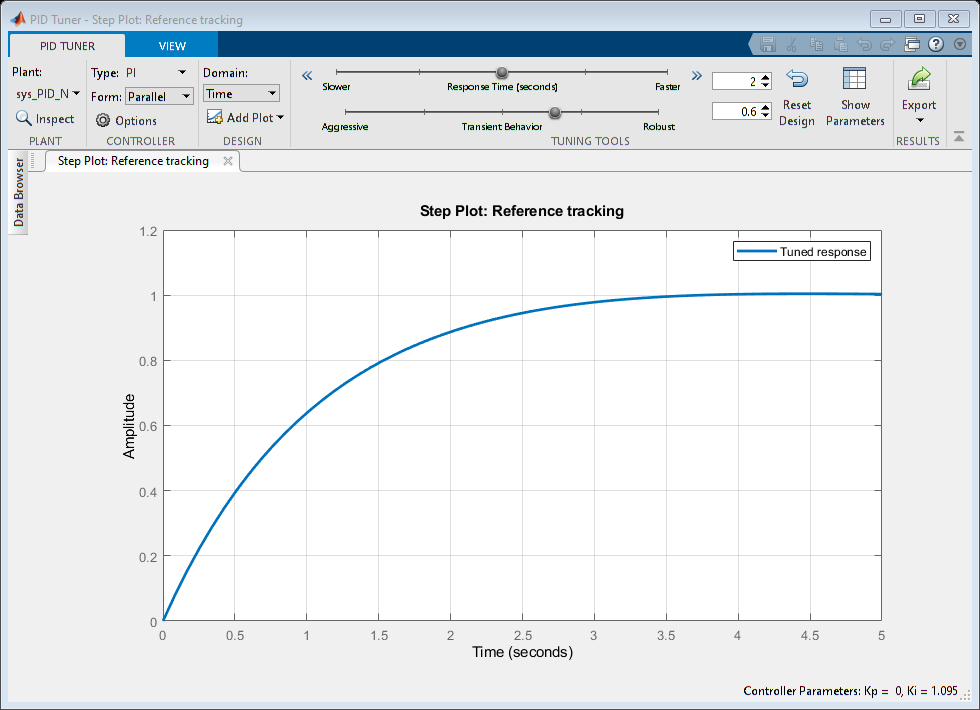
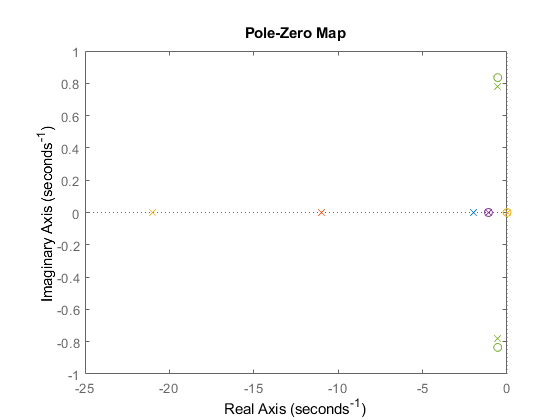
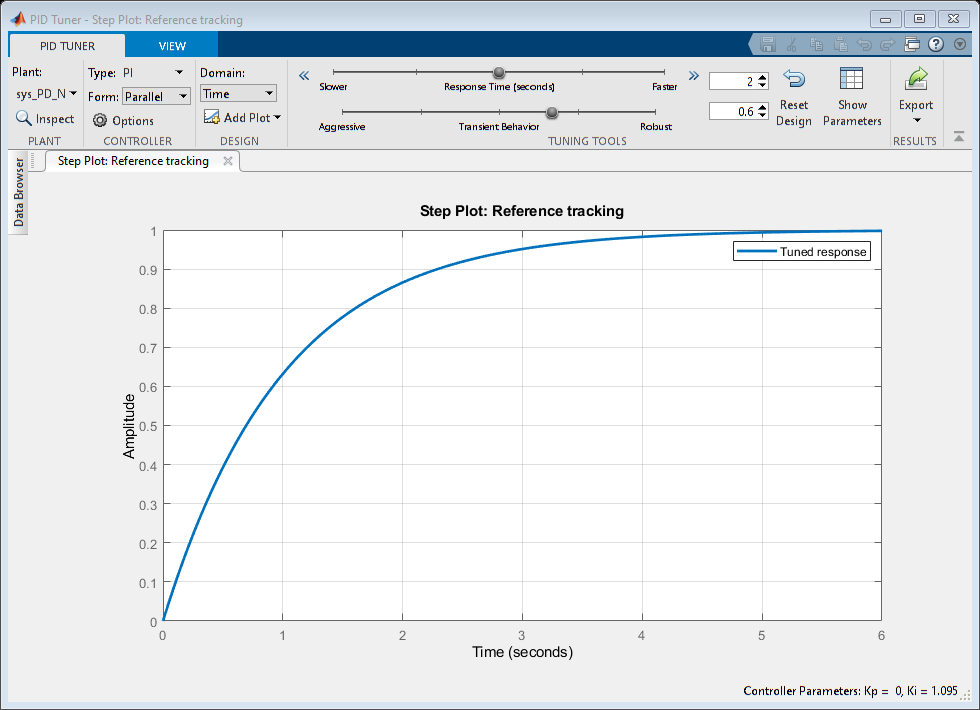
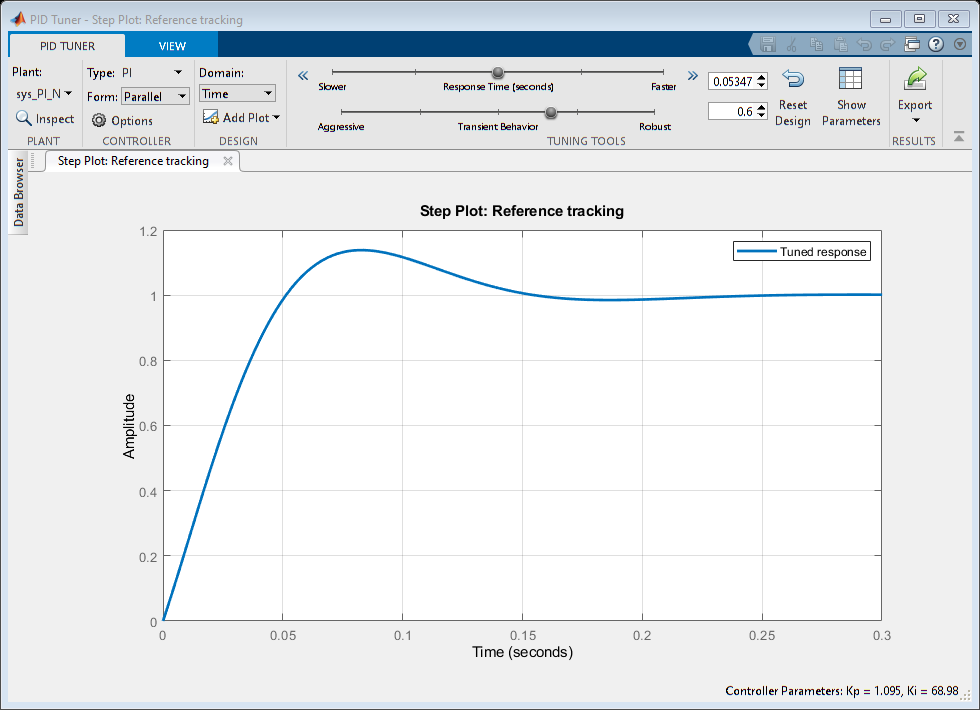
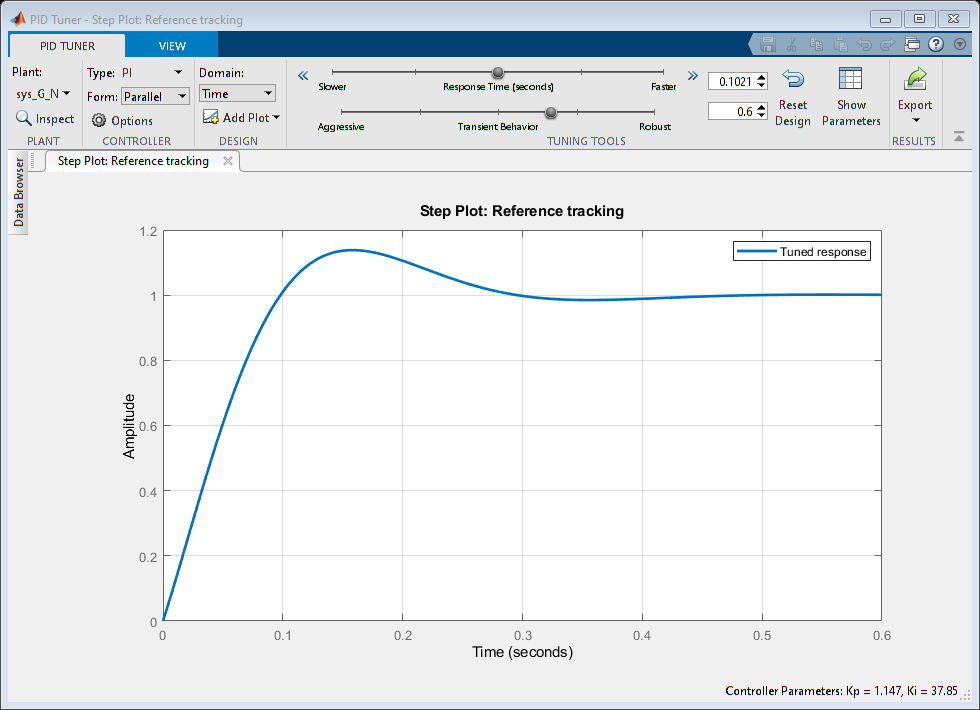
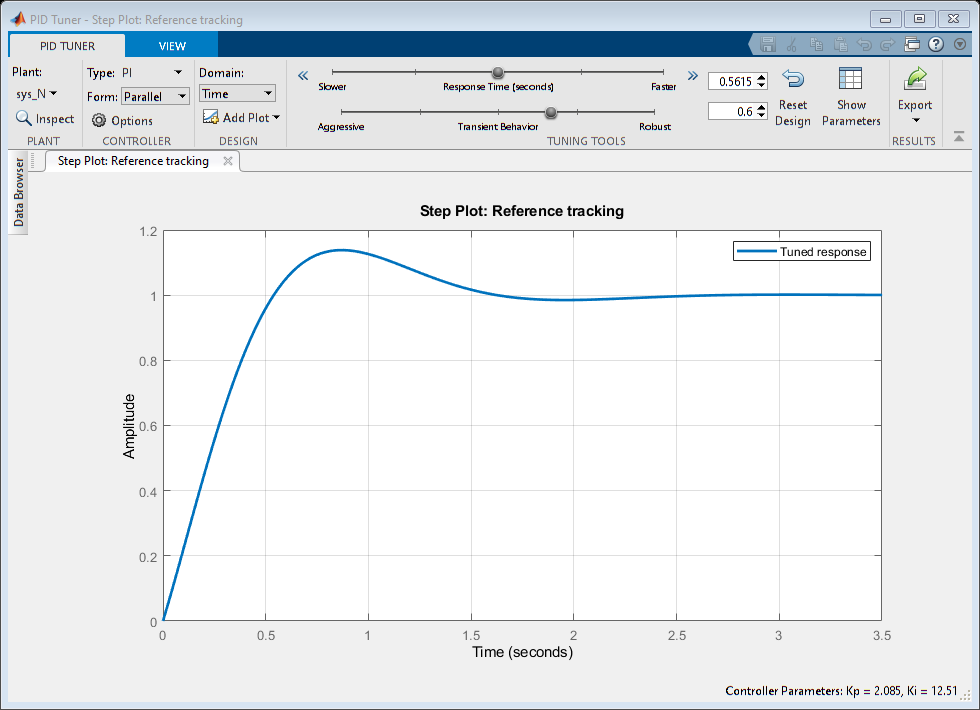


figure  
hold on  
pzmap(sys\_N)  
pzmap(sys\_G\_N)  
pzmap(sys\_PI\_N)  
pzmap(sys\_PD\_N)  
pzmap(sys\_PID\_N)  
  
  
pidTuner(sys\_N)  
pidTuner(sys\_G\_N)  
pidTuner(sys\_PI\_N)  
pidTuner(sys\_PD\_N)  
pidTuner(sys\_PID\_N)

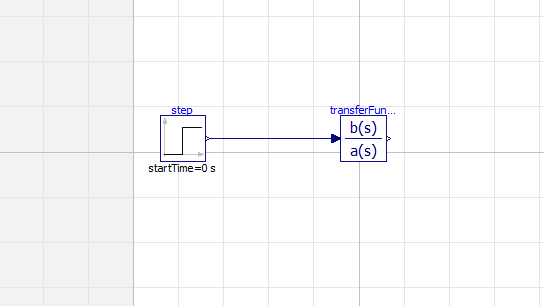
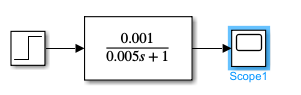


## Analysis

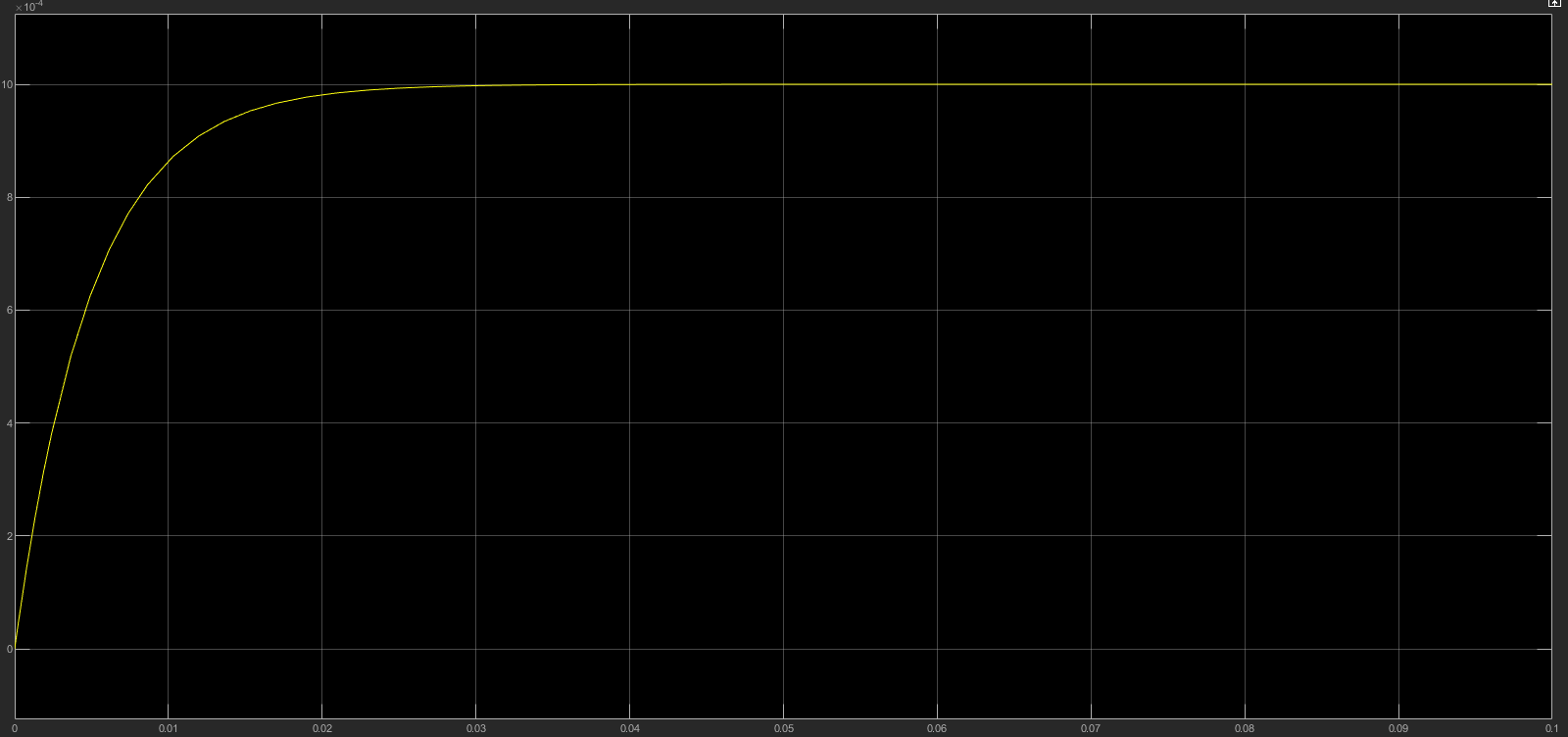
1.with negaitve feed back gain we get 1 pole at p1=-11 which has a rise time of 0.1997, settling time of 0.3556 the system is stable. 2.with negative feed back Pi controller we get 2 poles at p1=-10,p2=-1 and a zero at z=-1, because of integrator in PI controller we are getting an extra pole in it now Risetime=0.2197,settling time=0.3912 as the poles are on the left side of imaginary axis we can say that system is stable. 3.with a negative feed back PID controller we are getting complex conjugate poles and zeroes which are z1=-0.5500+0.8352i,z2=-0.5500-0.8352i,p1=-0.5455+0.7820i, p2=-0.5455+0.7820i the settling time is 1.8654 and the rise time is6.0686 so we can say that PID controller can not make the system more stable than PI and PD controllers did.

[*Published with MATLAB® R2021a*](https://www.mathworks.com/products/matlab)

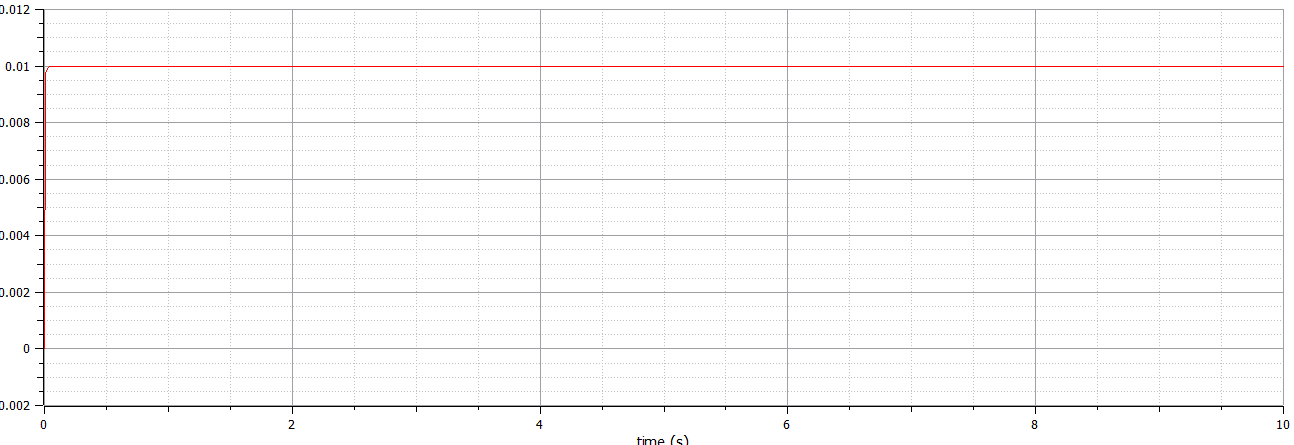
## Comparison between Simulink & Modellica:



Model designing in Simulink & Modellica



Simulink First Order Graph

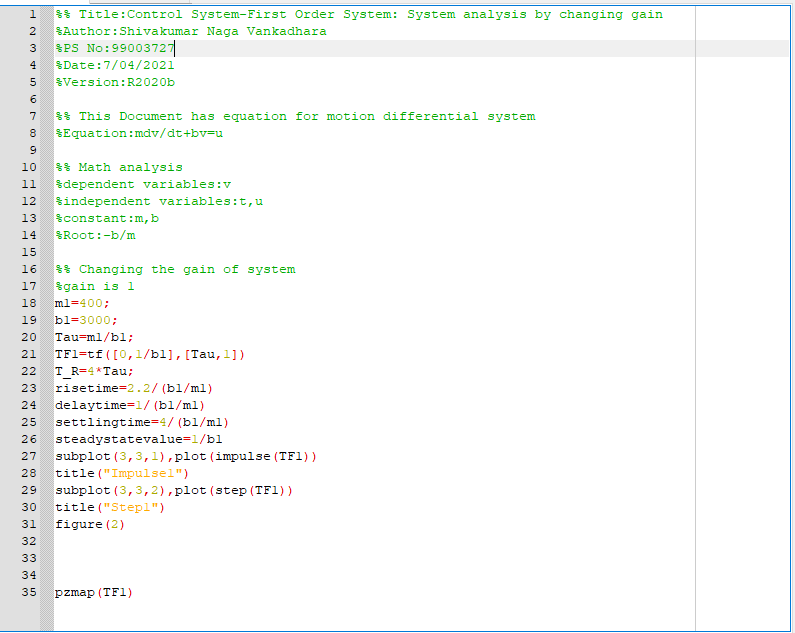


First Order Graph in modellica

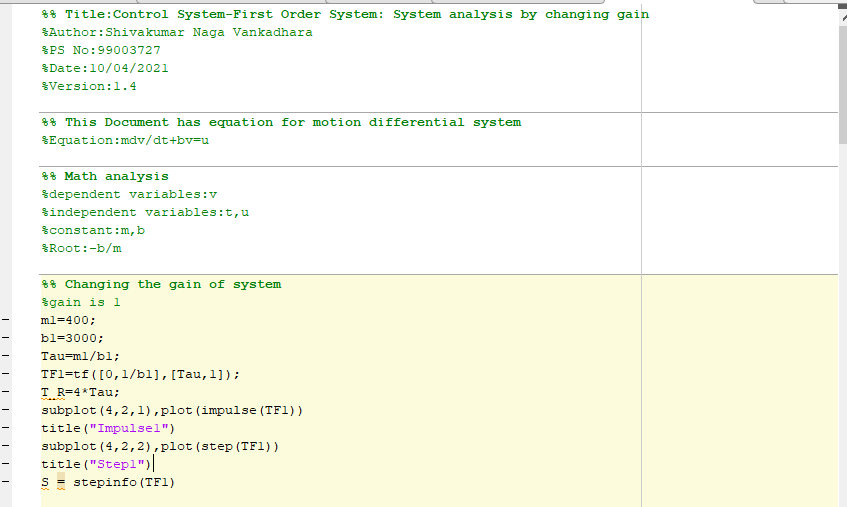
Analysis:

We took the same values in the transfer function and when we have done the model in Simulink and modellica so when we compare the step response we got the same values for both the models.

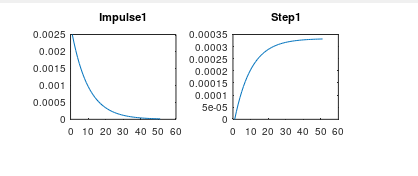
Comparison between Matlab Script and GNU Octave Script:



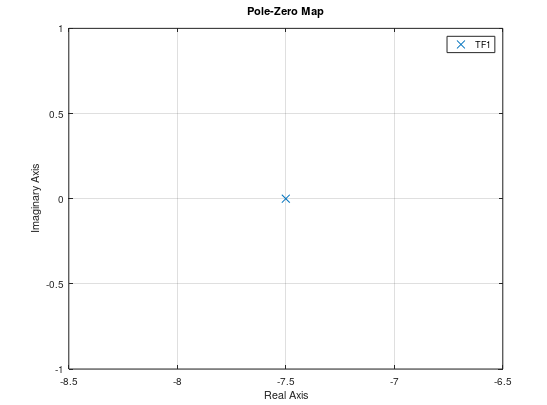
Octave First Order script



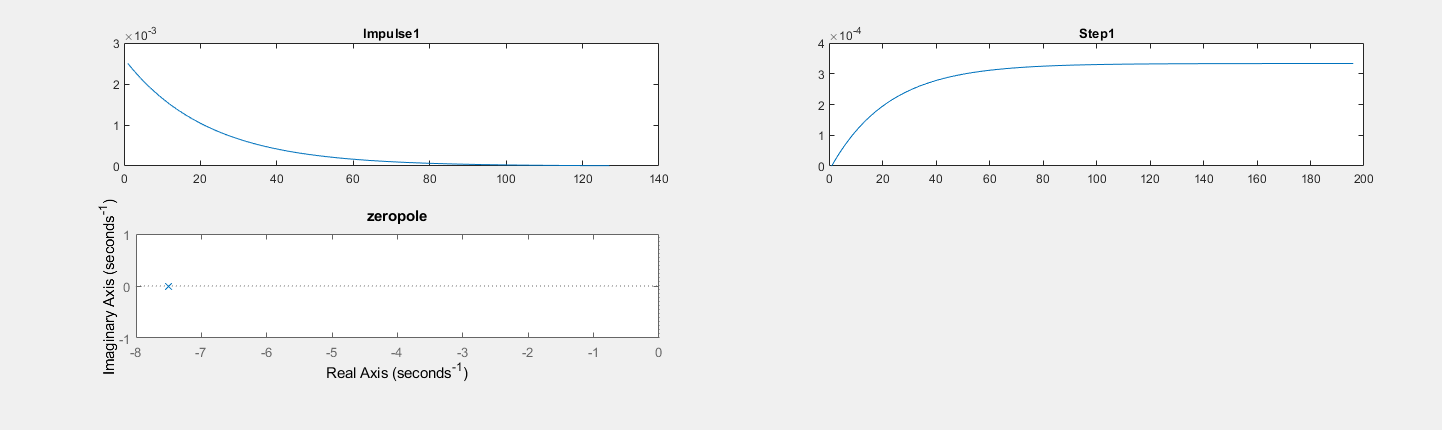
Matlab Script for First Order



Octave Script Step & Impulse Response



Octave Zero pole plot



Analysis:

For both the Octave scripting and Matlab scripting we got the same impulse response and step response and we also got the rise time, Settling time same in both the scripts.