Replacing Eq (2) into Eq (1) yields:

$$m\ddot{y}(t) = k_P e(t) + k_I \int_0^t e(\tau)d\tau + k_D \dot{e}(t)$$

Replacing $e(t)$ from Eq. (3) results in:

$$m\ddot{y}(t) = k_P(r(t) - y(t)) + k_I \int_0^t (r(\tau) - y(\tau))d\tau + k_D(\dot{r}(t) - \dot{y}(t))$$

Sorting the variables leads to:

$$m\ddot{y}(t) + k_D\dot{y}(t) + k_P y(t) + k_I \int_0^t y(\tau)d\tau = k_D\dot{r}(t) + k_P r(t) + k_I \int_0^t r(\tau)d\tau$$

For creating the simulink model:

$$\ddot{y}(t) = \frac{1}{m}\left(-k_D\dot{y}(t) - k_P y(t) - k_I \int_0^t y(\tau)d\tau + k_D\dot{r}(t) + k_P r(t) + k_I \int_0^t r(\tau)d\tau\right)$$

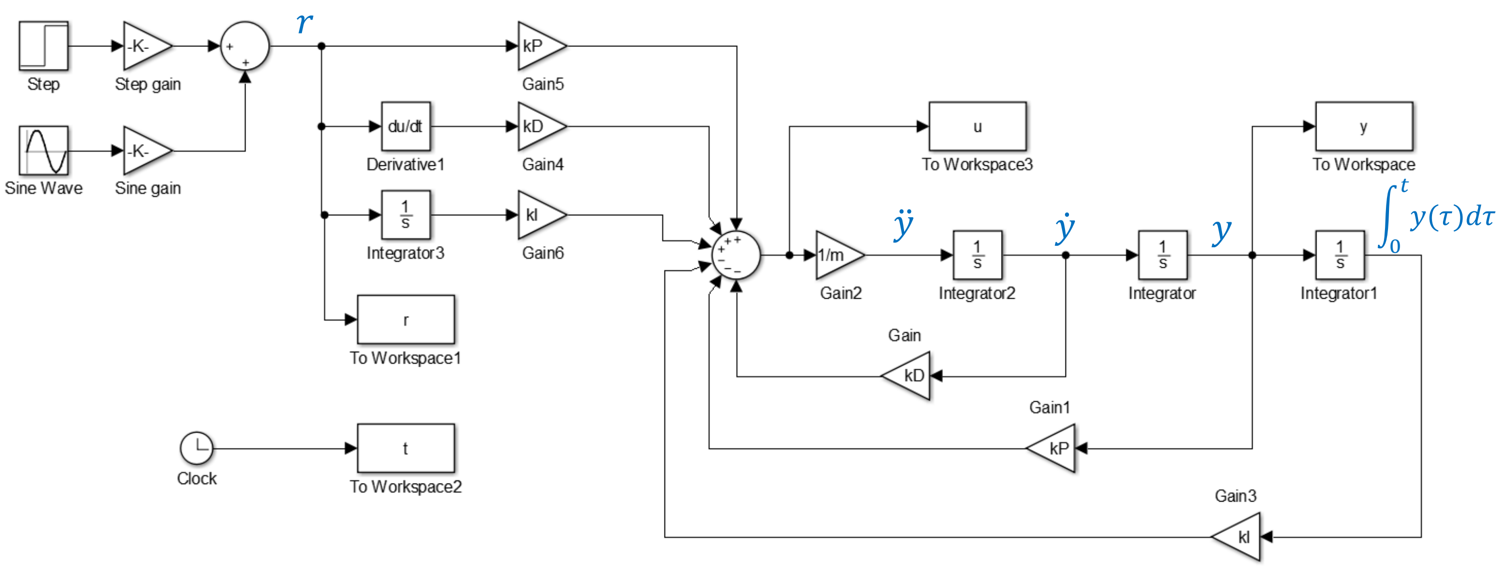


Table of Contents

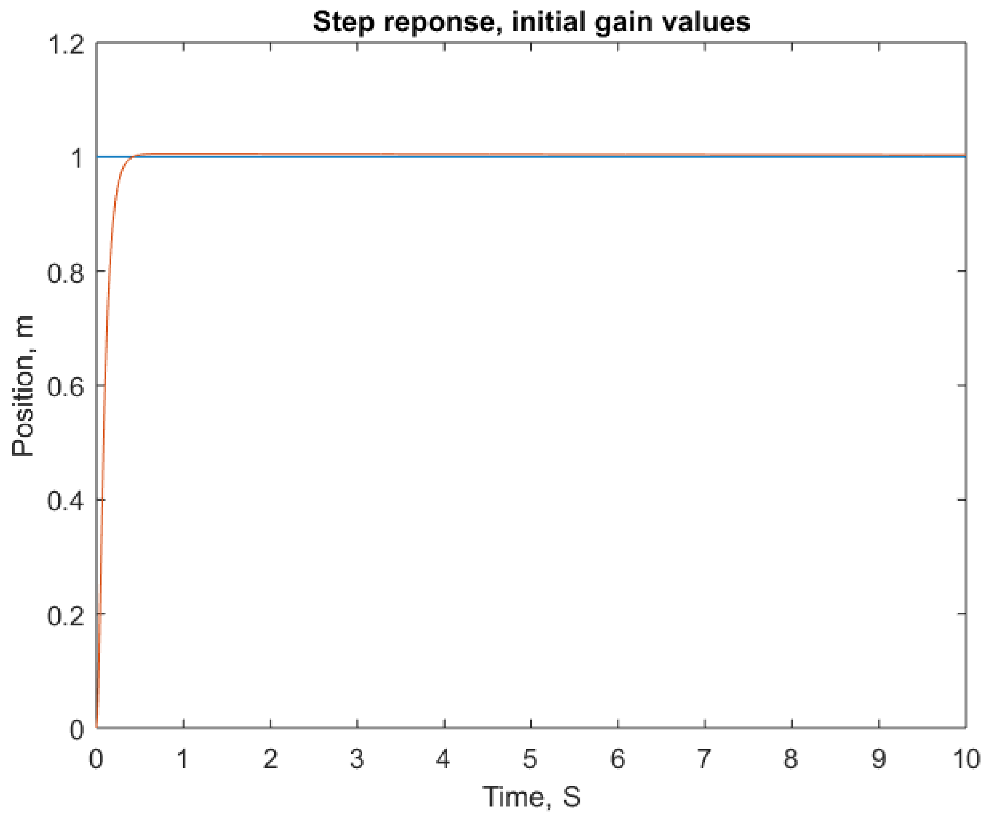
Simulating a PID Controller for a Free Mass System

```
clear; close all; clc

m = 0.2;

kP = 100;
kI = 5;
kD = 10;
```

Exercise 2: Step function response

There is less than 1% overshoot

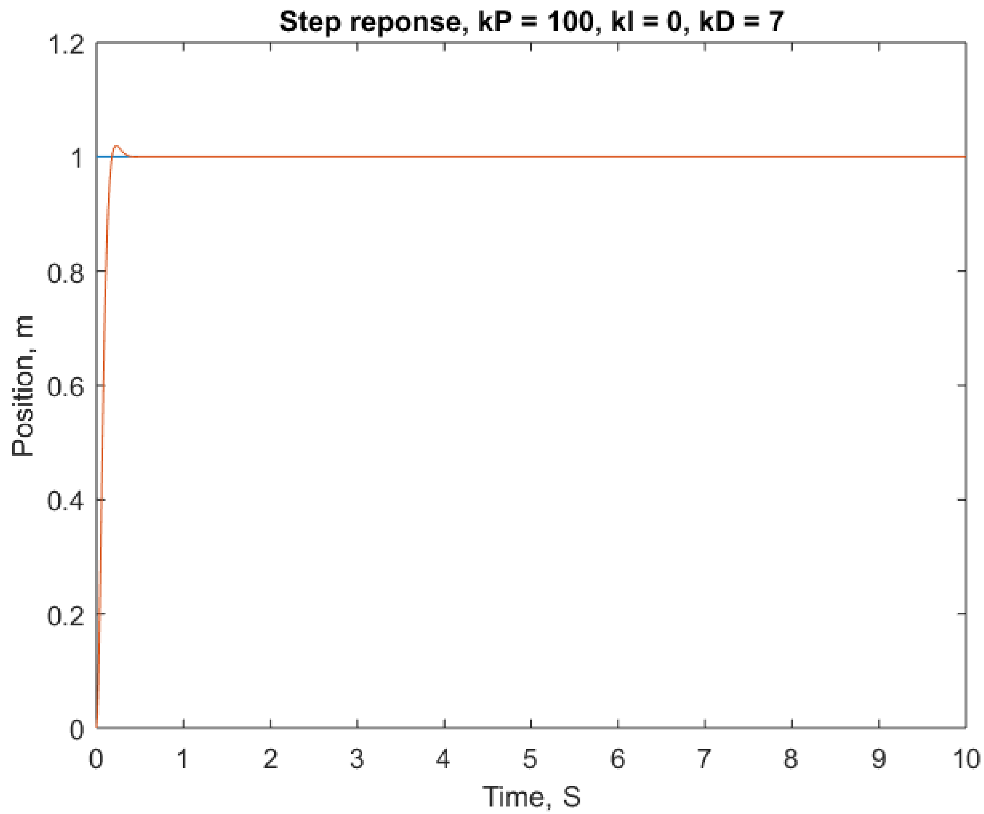
```
step_gain = 1;
sine_gain = 0;
sim('PID_Sim.slx')
figure; plot(t,r,t,y);
xlabel('Time, S'); ylabel('Position, m')
title('Step reponse, initial gain values')
```



Exercise 3: Step function response for 5% overshoot

The value of kD = 7 brings the overshoot down to 5%.

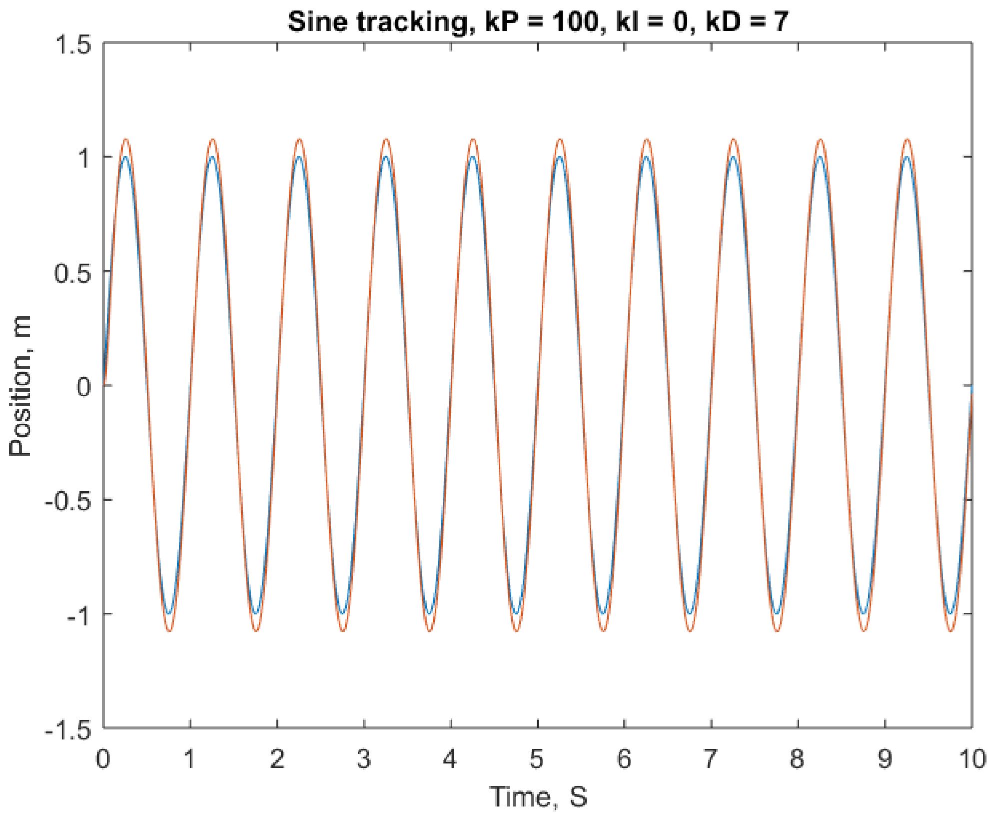
```
kI = 0;
kD = 7;
sim('PID_Sim.slx')
figure; plot(t,r,t,y);
xlabel('Time, S'); ylabel('Position, m')
title('Step reponse, kP = 100, kI = 0, kD = 7')
```



Exercise 4: Sine wave tracking

The tracking is good, but there is some error.

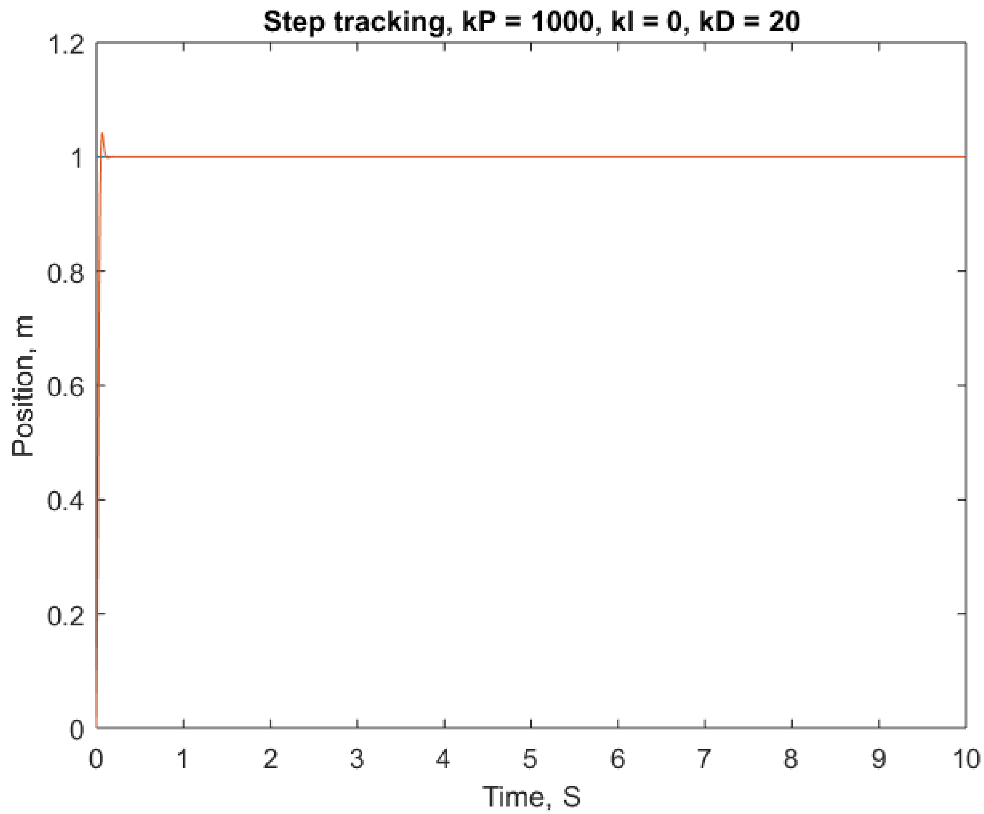
```
step_gain = 0;
sine_gain = 1;
sim('PID_Sim.slx')
figure; plot(t,r,t,y);
xlabel('Time, S'); ylabel('Position, m')
title('Sine tracking, kP = 100, kI = 0, kD = 7')
```

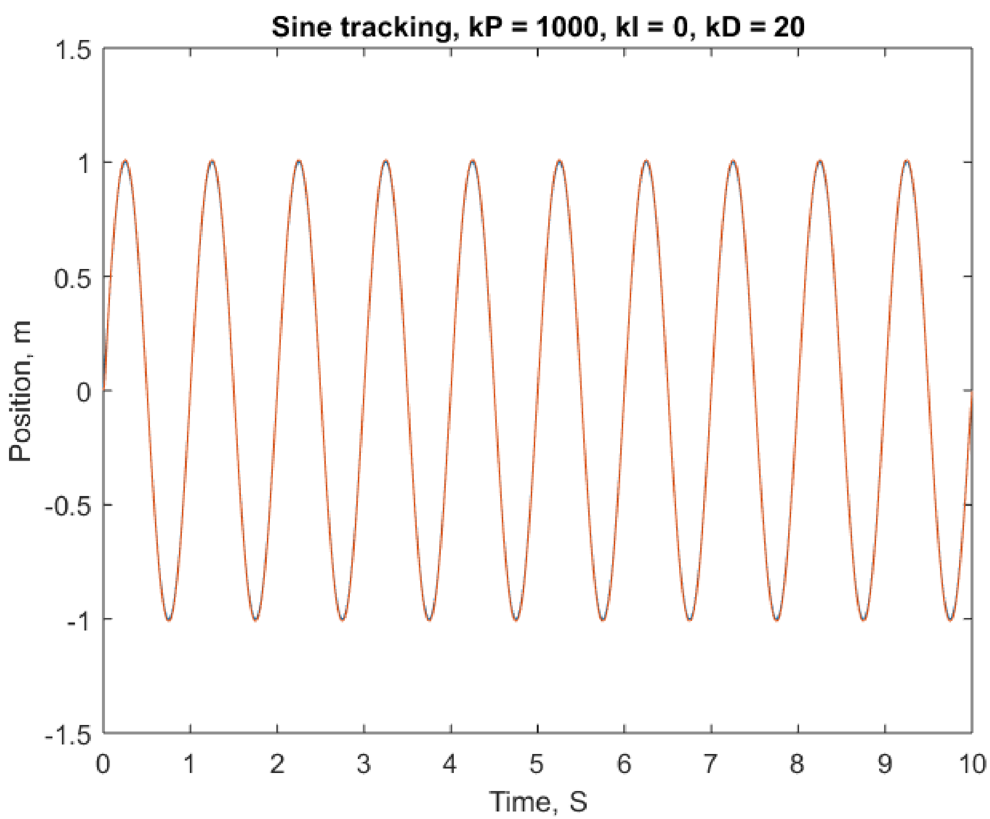


Exercise 5: Step for kP = 1000

The value of kD = 20 brings the overshoot down to 5%. The tracking is better.

```
kP = 1000;
kI = 0;
kD = 20;
step_gain = 1;
sine_gain = 0;
sim('PID_Sim.slx')
figure; plot(t,r,t,y);
xlabel('Time, S'); ylabel('Position, m')
title('Step tracking, kP = 1000, kI = 0, kD = 20')


step_gain = 0;
sine_gain = 1;
sim('PID_Sim.slx')
figure; plot(t,r,t,y);
xlabel('Time, S'); ylabel('Position, m')
title('Sine tracking, kP = 1000, kI = 0, kD = 20')
```

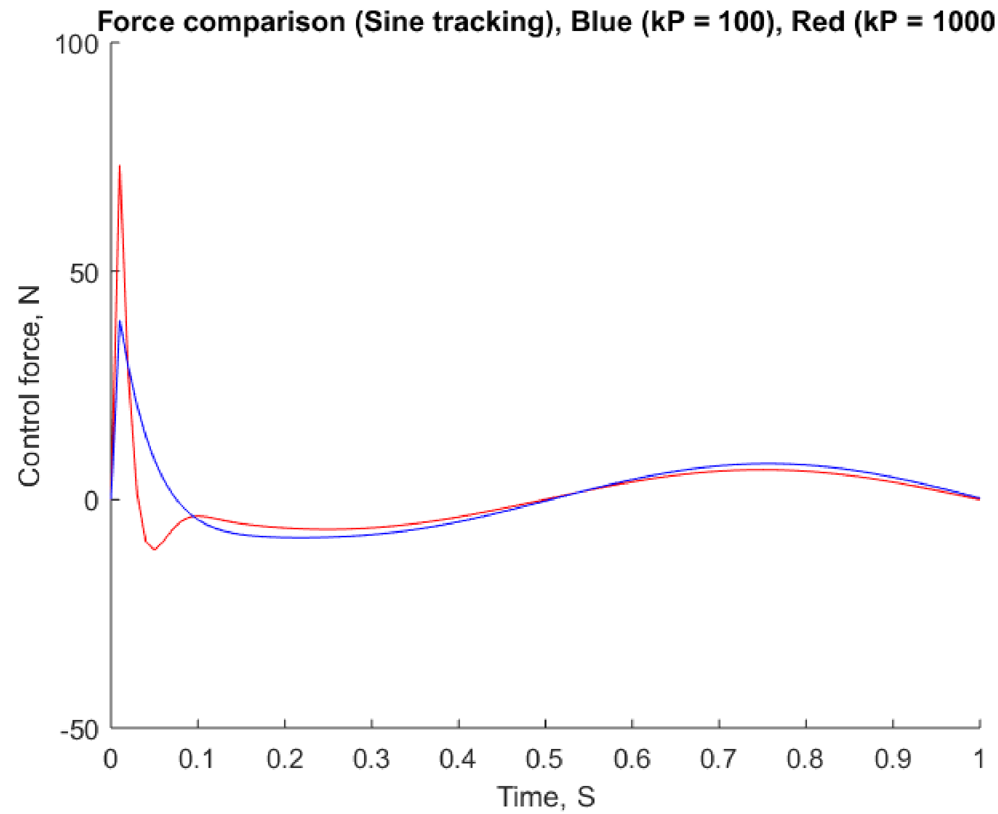

Step tracking, kP = 1000, kI = 0, kD = 20
Position, m
Time, S
0
0.2
0.4
0.6
0.8
1
1.2
1
2
3
4
5
6
7
8
9
10

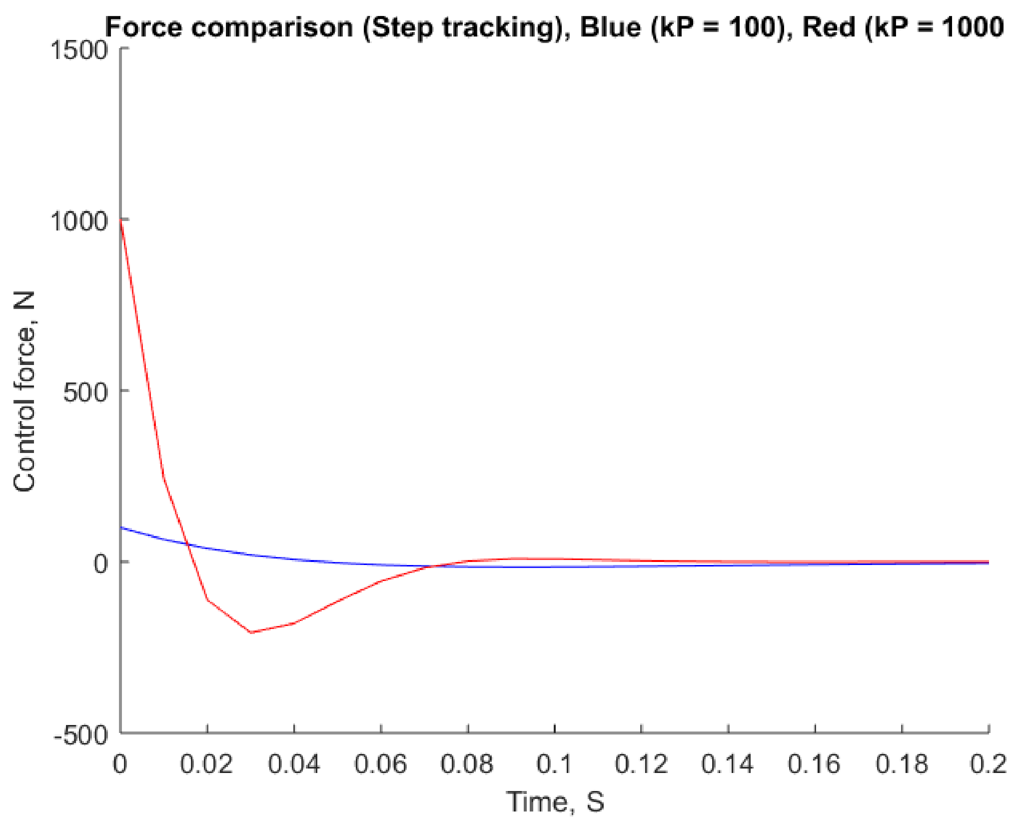

Sine tracking, kP = 1000, kI = 0, kD = 20
Position, m
Time, S
-1.5
-1
-0.5
0
0.5
1
1.5
1
2
3
4
5
6
7
8
9
10

Exercise 6: Control Force Comparison

The more accuracte controller requires higher initial force and might saturate in practice

```
figure; hold on;
plot(t,u,'r')

kP = 100;
kI = 0;
kD = 8;
sim('PID_Sim.slx')

plot(t,u,'b');
xlabel('Time, S'); ylabel('Control force, N')
title('Force comparison (Sine tracking), Blue (kP = 100), Red (kP =
 1000')
axis([0 1 -50 100])

step_gain = 1;
sine_gain = 0;

figure; hold on;
sim('PID_Sim.slx')
plot(t,u,'b');

kP = 1000;
kI = 0;
kD = 20;
sim('PID_Sim.slx')

plot(t,u,'r');
xlabel('Time, S'); ylabel('Control force, N')
title('Force comparison (Step tracking), Blue (kP = 100), Red (kP =
 1000')
axis([0 0.2 -500 1500])
```


Force comparison (Sine tracking), Blue (kP = 100), Red (kP = 1000
Control force, N
Time, S
100
50
0
-50
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Force comparison (Step tracking), Blue (kP = 100), Red (kP = 1000
Control force, N
Time, S
1500
1000
500
0
-500
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2

Published with MATLAB® R2016a