ME 190 HW 2 John Phung 9/19/2017

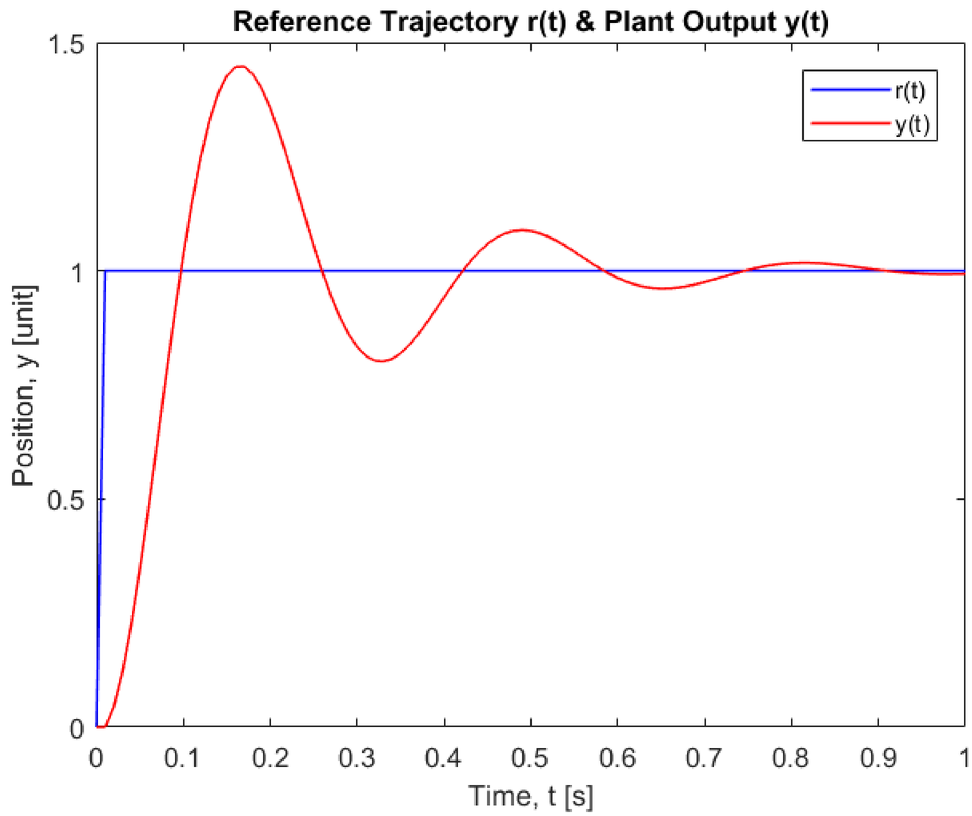
Table of Contents

Exercise 1: Combine Equation (1)-(3) into Single Differential Equation

```
%{





%}
```

Exercise 2: Develop a Simulink Model for Differntial Equation

```
m = 0.5; k_p = 200; k_i = 1; k_d = 5;

sim ('HW2_Exercise2_Step.slx');

figure (2);
plot (tout, r_t, 'b', 'linewidth', 1)
title ('Reference Trajectory r(t) & Plant Output y(t)')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')

hold on
plot (tout, y_t, 'r', 'linewidth', 1)
legend ('r(t)', 'y(t)')
hold off
```

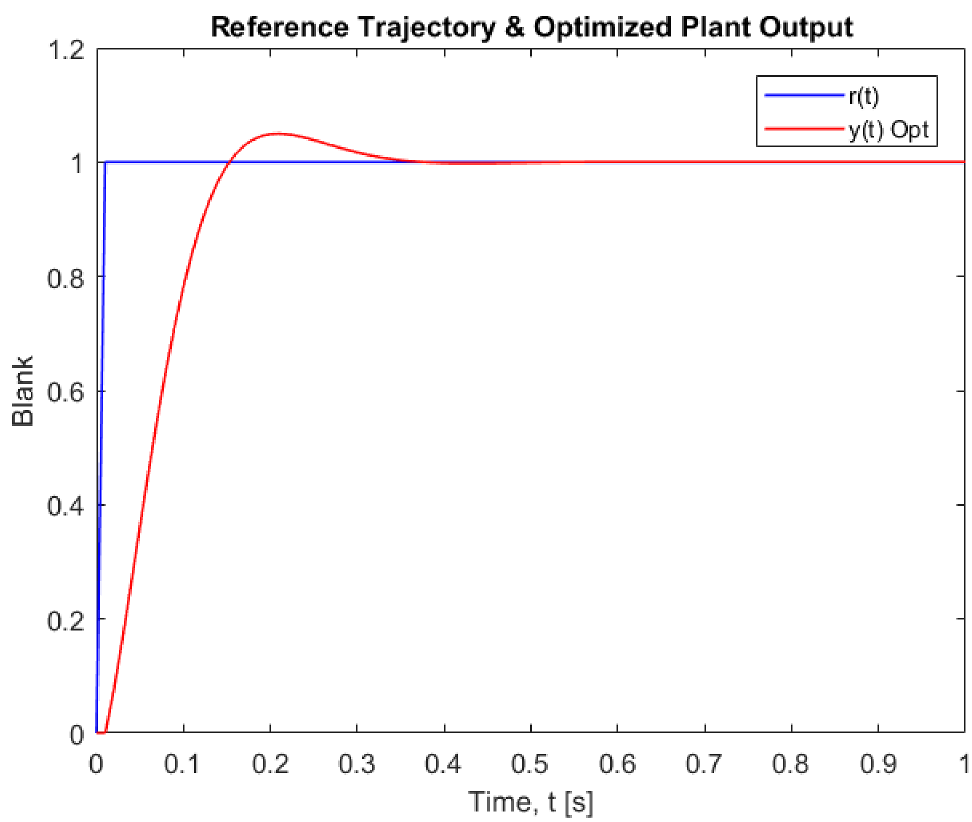


Exercise 3: Keeping k_i & k_p at their initial values. Find integer value of k_P (below 5%)

```
clc; clear all; close all;

m = 0.5; k_p = 200; k_i = 1; k_d_guess = [0:1:25];

%For Loop runs guess values from 0 to 25, intervals of 1.
for i = 1:length (k_d_guess)
 k_d = k_d_guess (i);
 sim ('HW2_Exercise2_Step.slx');
 y_sweep (i) = max (y_t);        %Question: In workspace y_t is based
 on k_d = 25
end

y_opt = find (y_sweep < 1.05);
kd_opt = min(k_d_guess(y_opt));

disp (['Optimal k_d value for exercise 3 is ' num2str(kd_opt)]);

%Update k_d to plot
k_d = kd_opt

sim('HW2_Exercise2_Step.slx');
```

```
figure (3);
plot (tout, r_t, 'b', 'linewidth', 1)
title ('Reference Trajectory & Optimized Plant Output')
xlabel ('Time, t [s]')
ylabel ('Blank')

hold on
plot (tout, y_t, 'r', 'linewidth', 1)
legend ('r(t)', 'y(t) Opt')
hold off
```

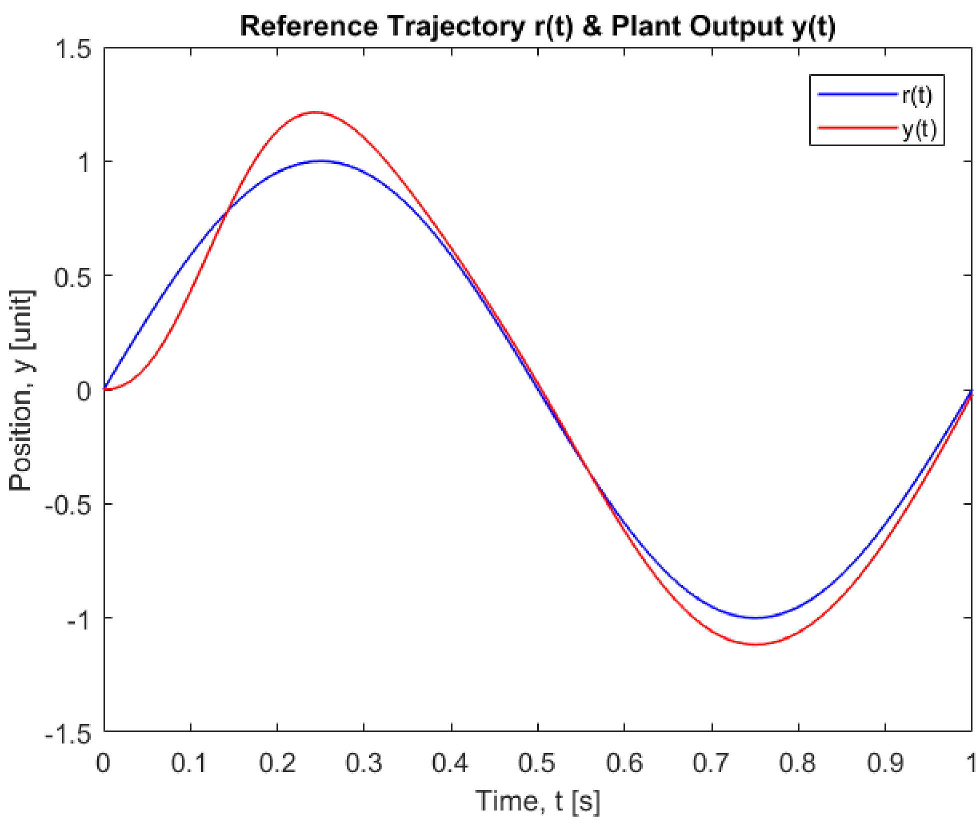
```
Optimal k_d value for exercise 3 is 14

k_d =

    14
```

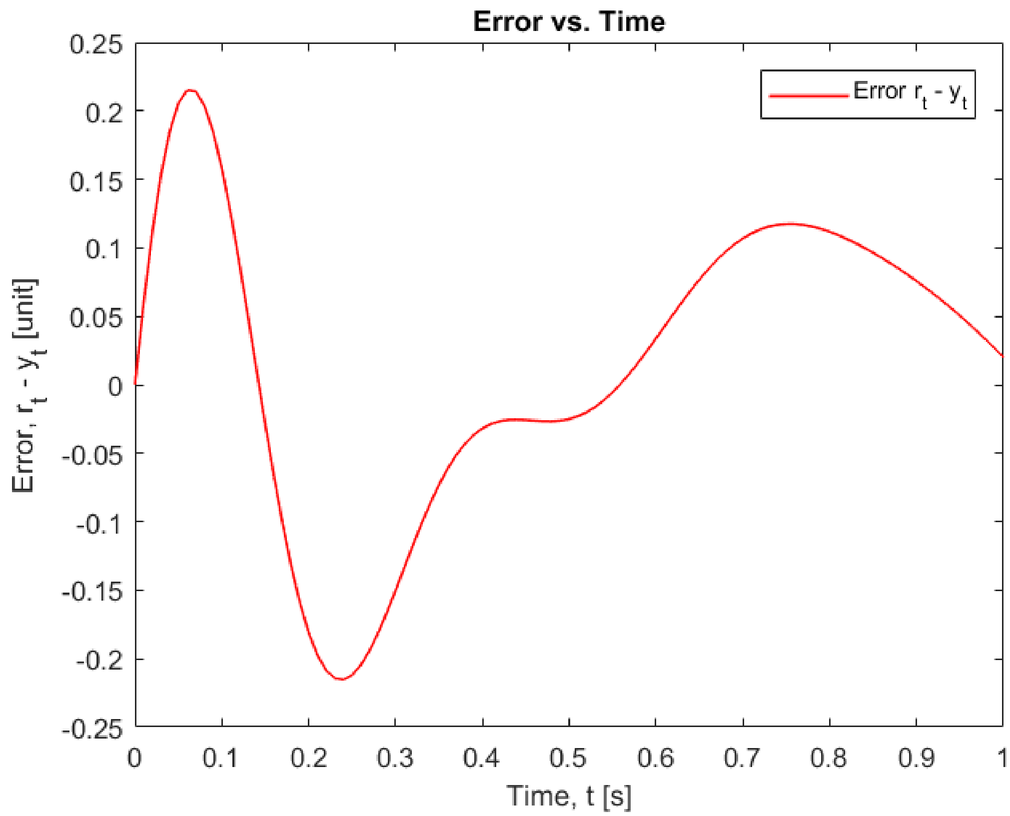


Exercise 4: Replace "step" Function with "sine" Function

```
clc; clear all; close all;

m = 0.5; k_p = 200; k_i = 1; k_d = 5;
```

```
sim ('HW2_Exercise4_Sine.slx');

figure (4);
plot (tout, r_t, 'b', 'linewidth', 1)
title ('Reference Trajectory r(t) & Plant Output y(t)')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')

hold on
plot (tout, y_t, 'r', 'linewidth', 1)
legend ('r(t)', 'y(t)')
hold off

%The system does not track the reference trajectory precisely at all.

%Error Plot
figure (5)
error = r_t - y_t;
plot (tout, error, 'r', 'linewidth', 1)
title ('Error vs. Time')
xlabel ('Time, t [s]')
ylabel ('Error, r_t - y_t [unit]')
legend ('Error r_t - y_t')
```



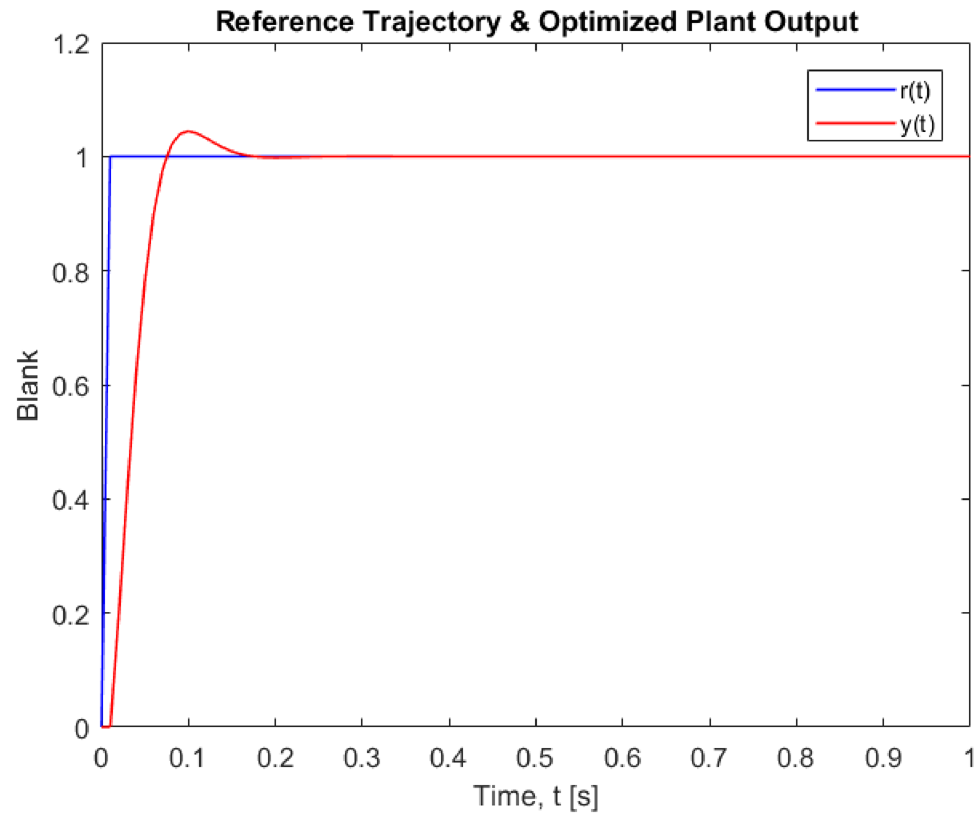


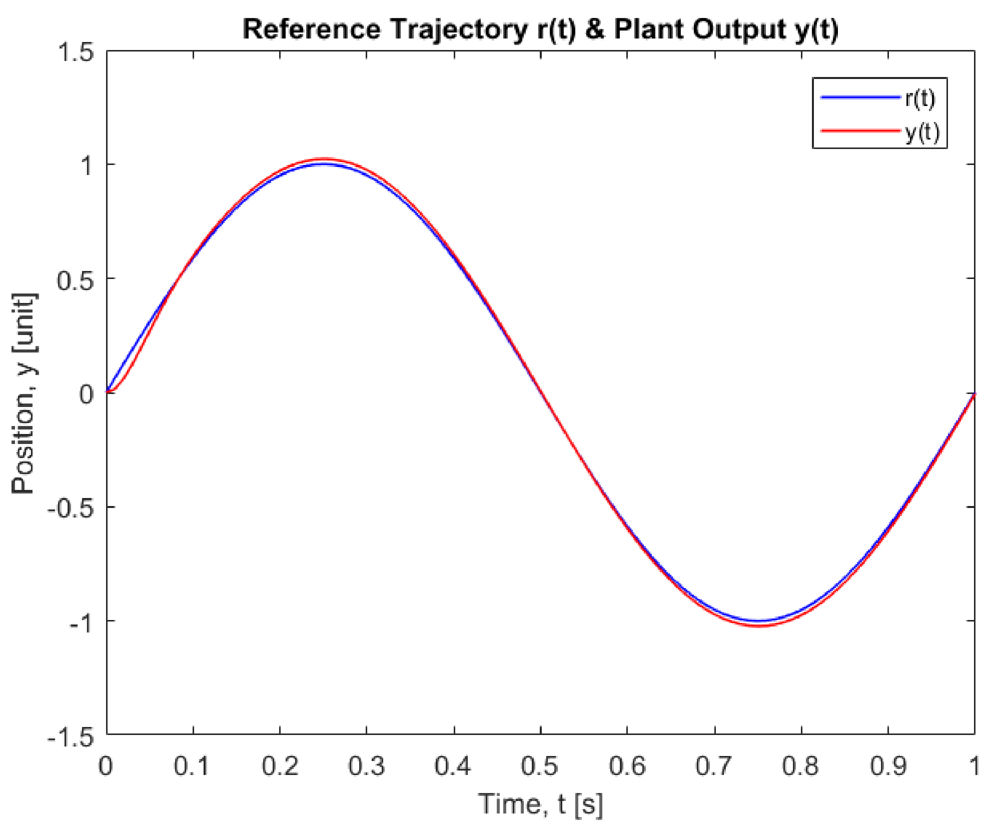
Exercise 5: Change Value of k_P to 1000 & find k_D, Check Tracking Error

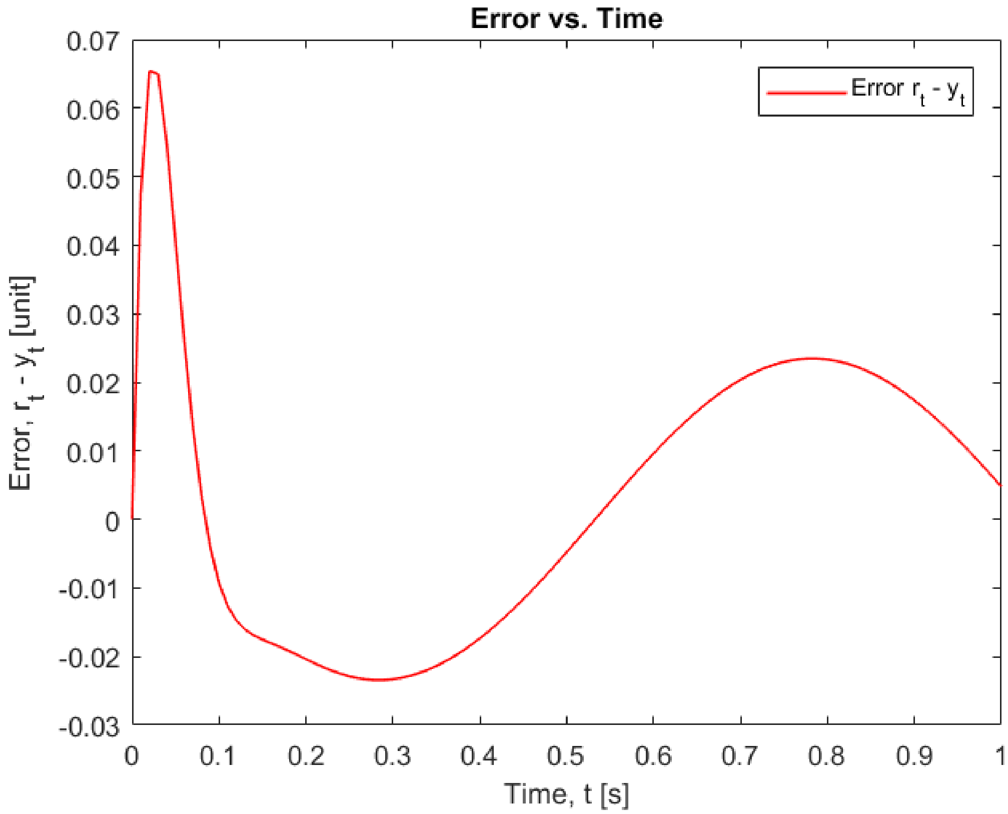
```
%Exercise 5a: Change Value of k_P to 1000 & find k_D

clc; clear all; close all;

m = 0.5; k_p = 1000; k_i = 1; k_d_guess = [0:1:50];

%For Loop runs guess values from 0 to 50, intervals of 1.
for i = 1:length (k_d_guess)
 k_d = k_d_guess (i);
 sim ('HW2_Exercise2_Step.slx');
 y_sweep (i) = max (y_t);        %Question: In workspace y_t is based
 on k_d = 25
end

y_opt = find (y_sweep < 1.05);
kd_opt = min(k_d_guess(y_opt));

disp (['Optimal k_d value for Exercise 5 is ' num2str(kd_opt)]);

%Update k_d to plot
k_d = kd_opt
```

```
sim('HW2_Exercise2_Step.slx');

figure (6);
plot (tout, r_t, 'b', 'linewidth', 1)
title ('Reference Trajectory & Optimized Plant Output')
xlabel ('Time, t [s]')
ylabel ('Blank')

hold on
plot (tout, y_t, 'r', 'linewidth', 1)
legend ('r(t)', 'y(t)')
hold off

%Exercise 5b: Repeat Exercise 4 & Check Tracking Error

sim ('HW2_Exercise4_Sine.slx');

figure (7);
plot (tout, r_t, 'b', 'linewidth', 1)
title ('Reference Trajectory r(t) & Plant Output y(t)')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')

hold on
plot (tout, y_t, 'r', 'linewidth', 1)
legend ('r(t)', 'y(t)')
hold off

%Error Plot
figure (8)
error = r_t - y_t;
plot (tout, error, 'r', 'linewidth', 1)
title ('Error vs. Time')
xlabel ('Time, t [s]')
ylabel ('Error, r_t - y_t [unit]')
legend ('Error r_t - y_t')

% In part 5b, tracking error is drastically reduced.
```

```
Optimal k_d value for Exercise 5 is 32

k_d =

    32
```


Reference Trajectory & Optimized Plant Output
r(t)
y(t)
Blank
Time, t [s]
1.2
1
0.8
0.6
0.4
0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

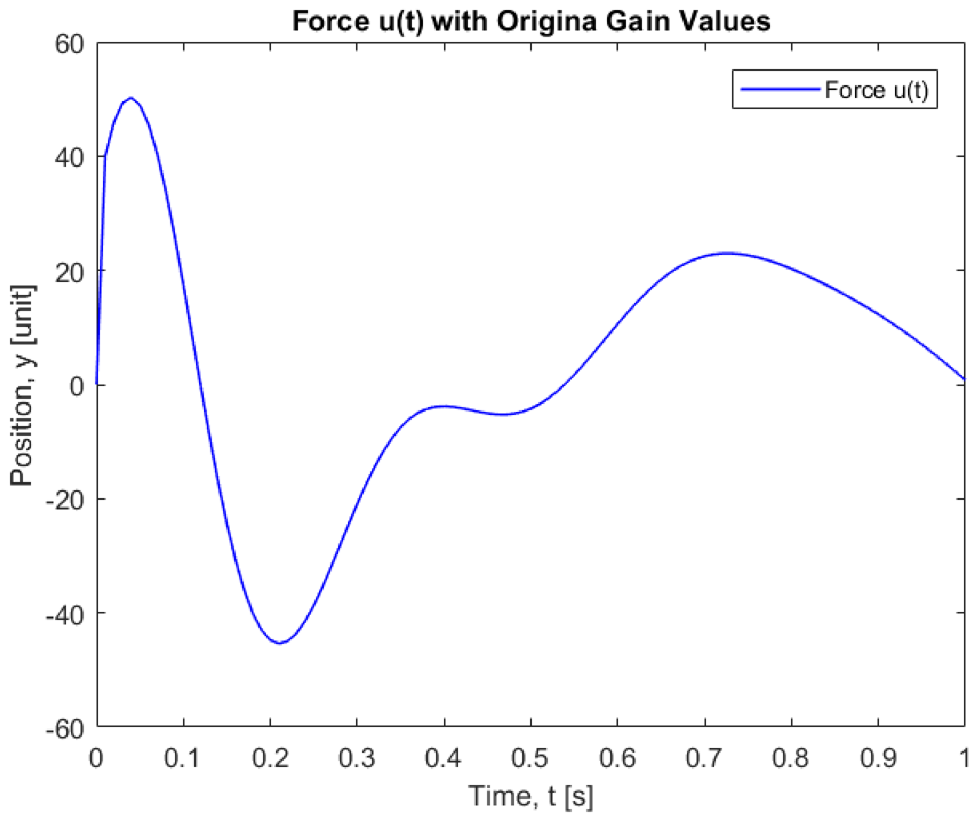

Reference Trajectory r(t) & Plant Output y(t)
r(t)
y(t)
Position, y [unit]
Time, t [s]
1.5
1
0.5
0
-0.5
-1
-1.5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

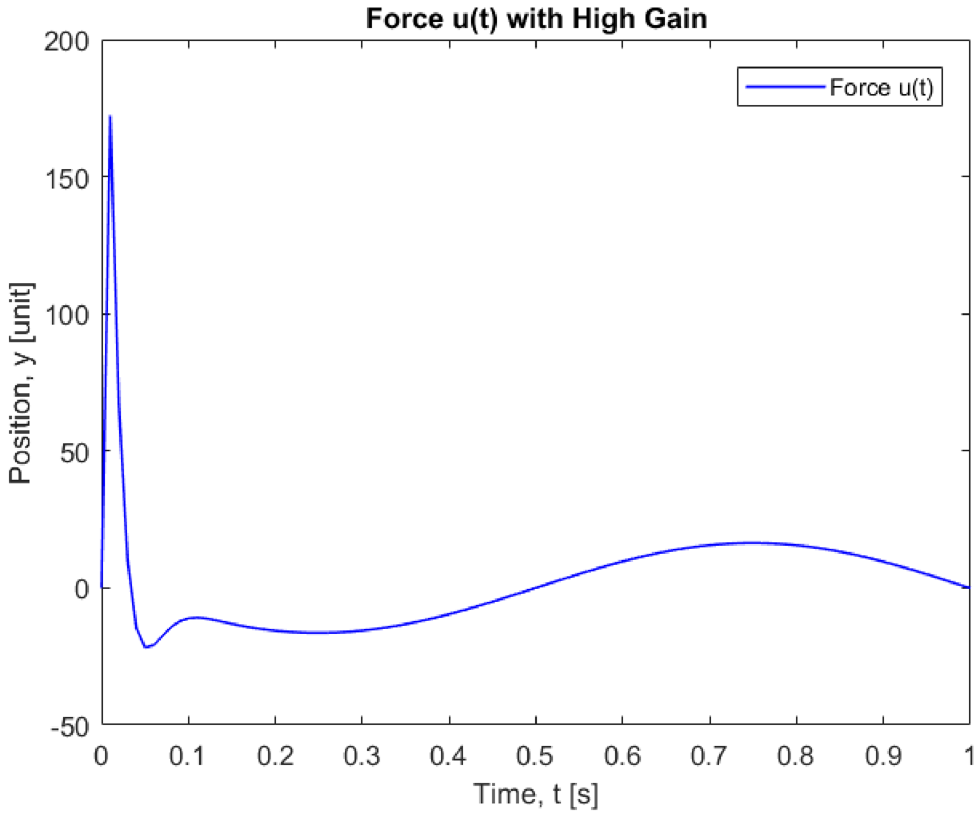


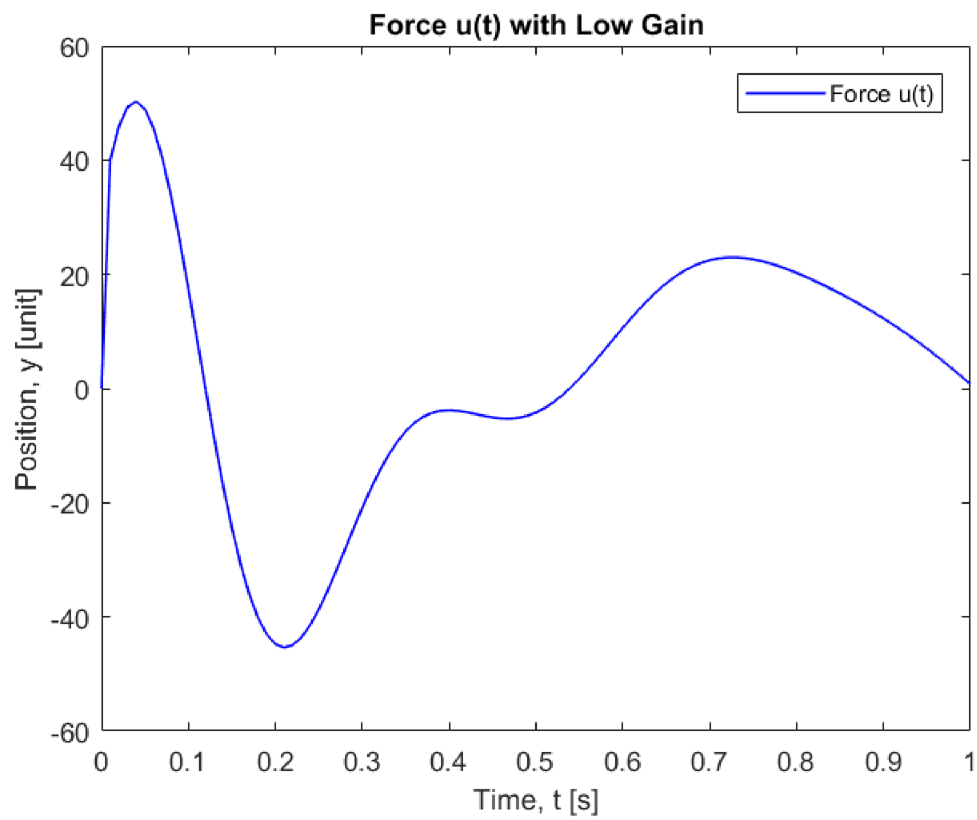
Exercise 6: Large Control Gain

```
%Control Gains: Do we adjust all k values up and down?
%Exercise 6a: High Gain
clc; clear all; close all;

m = 0.5; k_p = 200; k_i = 1; k_d = 5;

sim ('HW2_Exercise6_Sine.slx');

figure (9);
plot (tout, u_t, 'b', 'linewidth', 1)
title ('Force u(t) with Origina Gain Values')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')
legend ('Force u(t)')


%Arbitrary High Values multiplied by magnitude of 10
m = 0.5; k_p = 2000; k_i = 10; k_d = 50;

sim ('HW2_Exercise6_Sine.slx');

figure (10);
plot (tout, u_t, 'b', 'linewidth', 1)
```

```
title ('Force u(t) with High Gain')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')
legend ('Force u(t)')

%Exercise 6b: Low Gain

%Arbitrary Low Values multiplied by magnitude of .10
m = 0.5; k_p = 200; k_i = 1; k_d = 5;

sim ('HW2_Exercise6_Sine.slx');

figure (11);
plot (tout, u_t, 'b', 'linewidth', 1)
title ('Force u(t) with Low Gain')
xlabel ('Time, t [s]')
ylabel ('Position, y [unit]')
legend ('Force u(t)')

% In exercise 6 there was almost no oscillations in u(t), force
 applied to mass with high gain but some oscillation with low gain.
 Low gain modeled the original gain values
% a lot closer where high gain was completely off therefore low gain
 would be more practical if there was a limit on u(t).
```



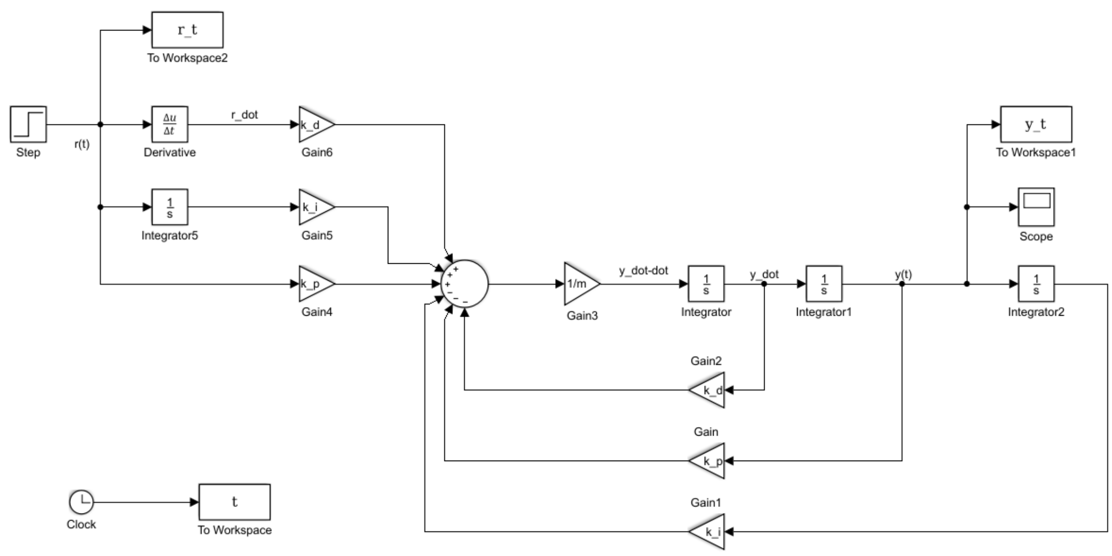

Force u(t) with High Gain
Force u(t)
Position, y [unit]
Time, t [s]
200
150
100
50
0
-50
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Force u(t) with Low Gain
Force u(t)
Position, y [unit]
Time, t [s]
60
40
20
0
-20
-40
-60
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Appendix

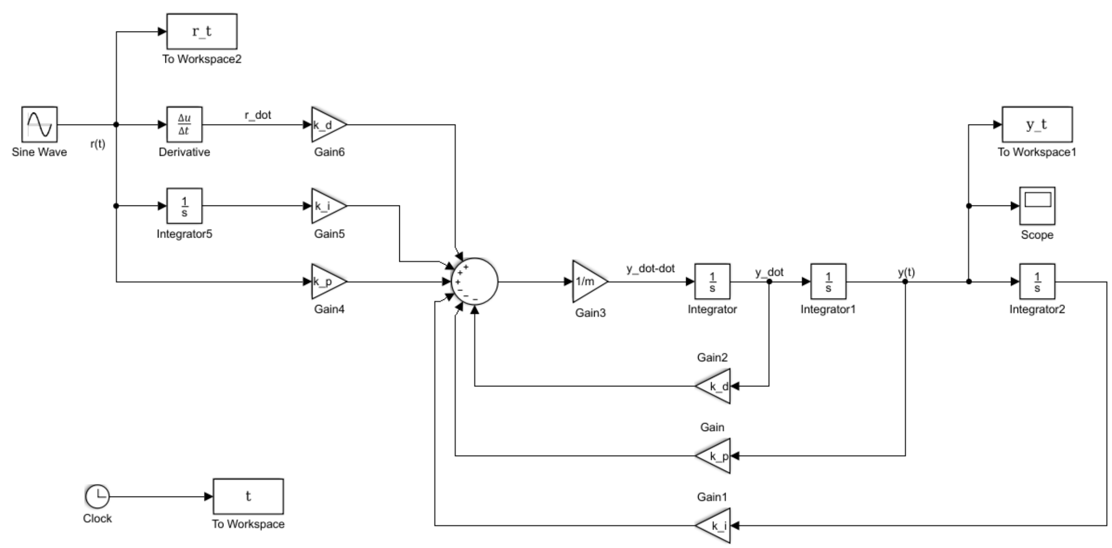
Exercise 2 & 3 Simulink

```
open_system ('HW2_Exercise2_Step.slx')
```



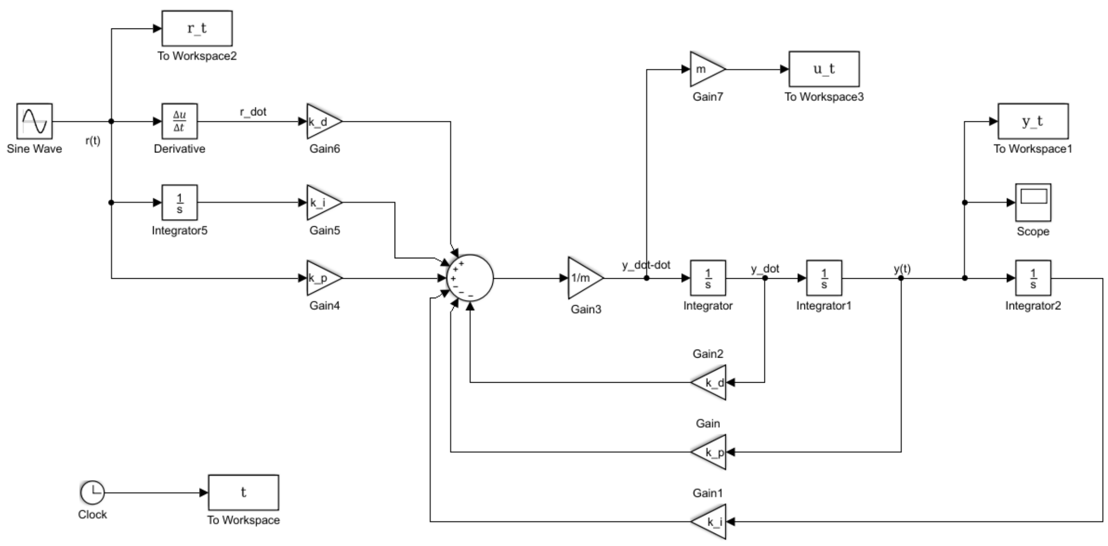
Exercise 4 Simulink

```
open_system ('HW2_Exercise4_Sine.slx')
```



Exercise 6 Simulink

```
open_system ('HW2_Exercise6_Sine.slx')
```



Conclusion

```
%{
The objective of this homework was to develop a simulink model of a
 PID controller for a free mass system. A PID controller consists of a
 "Plant" block where the system is to
be controlled, a "Controller" block is the control logic, r(t) is the
 reference trajectory, e(t) is the tracking error, u(t) is the plant
 input, and y(t) is the plant output.
In this homework assignment we generated a target function r(t) where
 a control system would be utilized to match or closely resemble the
 target function. We will refer to this
matched output as y(t) or th eplant output.
In order for this simulate and tune this PID controller we have to
 take into account three equations: plant,controller and tracking
 error. These three equations are combined into
for ease of use in Simulink. In Exercise 3 and 5 a FOR LOOP was used
 to sweep across a range of arbitrary values and a maximum was found
 to optimize overshoot below 5% or 1.05 m.
%}

% End of HW2
```

Published with MATLAB® R2017a