ENB301: Practical Report

Lachlan Nicholson (n8866864) Declan Gilmour (n8871566)

25th May, 2015

1 Executive Summary:

This report provides the documentation and analysis of three closely associated practical experiments; the open loop system, the closed loop motor control system, and the effects of varying the systems gain and input signal frequency. The methods of analysis conducted within this document include thorough mathematical estimations and computational simulations. The results of the previously mentioned experiments, beyond showing clearly the benefits and application of closed loop control systems, conclusively show that the input frequency does not effect the overshoot or settling time of the closed loop system, but the gain is proportional these properties. Moreover, the difference in theoretical calculations and practical applications due to un-modeled system interactions was shown in depth.

Still not really done
subject matter
methods
findings
conclusions
recommendations
limitations

2 Aim:

1-2 sentences on the report purpose
The purpose of this report is to demonstrate sound knowledge and understanding of basic control systems; more specifically, closed loop feedback systems. This report entails (beyond other things) the procedures followed during the practical lab, the associated results or observational data, and the answers to all questions posed during the lab.

3 Introduction:

0.5-1 page overview of the lab

4 Procedure:

brief summary of prac procedure as described in this document (0.5-1page for each part (bcd))
This section consists of individual summaries of the procedures required in each section, accomplished during the practical labs. Briefly cover what each part of prac is to accomplish.

4.1 Experiment B

Experiment B utilizes experimentally measured time response data to calculate approximate values for K_m and α. The pre-lab preparation for this section of the practical required familiarization with all aspects of the oscilloscopes and their functions (triggers, scales, saving data); moreover, it was suggest that the user should also be familiar with the construction process outlined within the coming procedures. In an attempt to better show the procedures of this particular experiment, the required tasks will be separated into two categories; the setup, and the analysis. Both of the aforementioned sections will be displayed in numbered point formation to show clearly the extent of each step.

SETUP:

1. The dual power supply was setup in independent mode, with 5.0 V and 2.0 V respectively. (measured using a multimeter to ensure accuracy)
2. Next, the 5V supply was connected across the potentiometer (outer wires), whilst using a multimeter to measure the wiper voltage (middle wire). Moreover, turning the flywheel clockwise increased the wiper voltage; however, if the inverse was true, the 5V and 0V wires would have been swapped.
3. After having adjusted the flywheel to the center position, and after connecting the 0V rail to one side of the motor; the circuit was briefly connected using the 2V supply. The polarity of the two connections were then adjusted to ensure the motor moves in a clockwise direction when voltage is applied.
4. The flywheel was returned to the central location whilst the wiper voltage of the potentiometer and the positive terminal of the motor were measured using the digital CRO. The trigger was set to 0.5 V and a USB inserted.
5. After briefly completing the motor circuit, the CRO was triggered and recorded the response of the potentiometer and input voltage. The channel gain was inspected to ensure it had been set correctly and all important regions of the response can be seen, then the data was saved to the USB.
6. The previous step was repeated multiple times to ensure viable data had been collected.

ANALYSIS:

1. After the experimental data had been saved to a USB and transfered to a computer, it was then plotted in matlab.
2. The experimental data was also plotted against the systems estimated transfer function $y_1(t)$.
3. Using a robust self constructed function to estimate the system parameters according the experimental data, approximate values for K_m and α were obtained.
4. The system transfer function $y_1(t)$ was then adjusted, and another diagram constructed to compare the experimental and calculated systems response.
5. Lastly, using the second order model for the motor and the previously calculated α value; K_m and β were approximated.

4.2 Experiment C

Section C utilized the open loop response of the motor that had been measured and approximated in the previous procedures to achieve a closed loop control system for the position of the motor (voltage across the potentiometer). The pre-lab component of experiment C required the extensive analysis of a provided circuit diagram; breaking the system down into functional elements of the control system, calculating individual transfer functions for these elements, and constructing an overall transfer function for the system ($G_c(s)$). As mentioned previously, the procedures followed in this experiment will also be separated into the setup or preparation, and the analysis.

The pre-lab aspect of this experiment required the operator to be familiar with typical breadboard design strategies, the pin-out of the op-amp used within the experiment, common resistor code colours, and the use of noise mitigation capacitors.

Refer to figure 19 for the complete closed loop motor control system schematic used in the following procedures.

SETUP:

1. The function generator was setup to output a 0.1 Hz square wave an amplitude of 0.5 V.
2. The aforementioned circuit was constructed, but the motor was only connected after taking multiple measurements to ensure the circuit was operating as expected.
3. With the motor connected, the response of the system was captured and saved by the digital CRO; this step was repeated multiple times to ensure accuracy.
4. The gain was then adjusted to produce a 5% overshoot, the resistor values used and the systems response were recorded.

ANALYSIS:

1. The collected data was imported into matlab, and compared against the predicted model derived previously in **part A and B**.
2. After which, the experimentally found gain required to produce an overshoot of 5% was compared against the predicted gain value.
3. Lastly, a discussion took place surrounding the use of alternate methods to derive the open and/or closed loop response for the system using the same equipment.

4.3 Experiment D

The final experiment, using the same circuit constructed in the previous experiment, examined the response of the system to different input frequencies, the impact of gain on the systems response, and the implementation of a new method to approximate the open and/or closed loop response.

ANALYSIS:

1. The closed loop response of the system was estimated **using the method outlined in the previous procedures**
2. The K_m and α values were then compared to **the method outlined in worksheet**
3. Using the same circuit constructed in the previous procedure (system gain set at the experimentally found gain required to produce a 5% OS) the overshoot was measured and recorded, for input frequencies of 0.5Hz, 0.75Hz, 1Hz, 1.25Hz and 1.5Hz.
4. Additionally, the systems response for each frequency was also recorded and saved in-case of future need.
5. The impact of altering the gain was then examined, as the response of the system was recorded twice more; using a new gain value each time.
6. The closed loop system was then simulated in Simulink, using the model parameters determined in the previous experiment. A group discussion was raised over the quality of the model, the results of which can be found in section 7.2.3.
7. Another group discussion was began, as a proposal for a **PI/PD/PID** controller that would achieve a faster response was formed and simulated. Refer to the previously mentioned section for the in depth proposal.

5 Results:

summary of what you observed in parts B,C,D (less than 2 pages per part), noting detailed answers are to be provided in the appendix.

5.1 Part B

calculate predicted K_m and α values. correct polarity of the motor above B5

5.2 Part C

As mentioned previously, the goal of experiment C was to construct a closed loop control system (feedback) for the position of the motor (voltage across the potentiometer); making use of the estimated open loop response variables found in part B (K_m and α).

The pre-experimental aspect of this section required the breakdown and analysis of the circuit diagram captured in figure 19, the full method and calculations has been provided in section 7.2.2; the in depth analysis resulted in an estimated overall transfer function for the system.

$$G_c(s) = \frac{\frac{R_f}{R_1}K_m}{s^2 + s\alpha + \frac{R_f}{R_1}K_m}$$

$$G_c(s) = \frac{1075.8}{s^2 + 38.61s + 1075.8}$$

After constructing the circuit shown in figure 19 and asserting the correct polarity of the motor, the response of the system was captured and saved by the digital CRO. The closed loop response of this system was measured 3 times, and the graphical representation of this data has been included below.

FIGURES

WHAT WE CAN SEE FROM THIS (AS VOLTAGE GOES UP, THIS GOES UP), and here is the closed loop response with experimentally found gain required to produce an overshoot of 5%.

FIGURES

WHAT WE CAN SEE FROM THIS

compare models
compare gains
new method

closed loop control system for the position of the motor overall TF estimated gain for 5% OS practical gain for 5% OS comparisons C8

5.3 Part D

Experiment D explored the effect of input frequency on the closed loop system response, as well as examining the effect of increasing or decreasing the systems gain. and bode

D1/C8

After measuring and recording the systems response and percentage overshoot for five different input frequencies; the following table was constructed (refer to section 7.2.3 for calculations responsible for the construction of this table).

Input Frequency (Hz)	Overshoot (%)
0.5	6.1
0.75	5.7
1	6.1
1.25	6.5
1.5	5.8

Both the above table, and the systems response (figure 28) show clearly that increasing the frequency has no effect on overshoot, settling time, or steady state error. However, after the frequency exceeds a certain limit ($1/f < Ts$) the system does not have enough time to reach a steady state value. Moreover, if the frequency of the input where to continue increasing, eventually the operation amplifier will begin attenuating the output voltage.

The reason the frequency has no effect on these values, as explained in section 7.2.3, is that the circuit contains no energy storing devices (such as capacitors or inductors), and as such- **frequency should not affect any circuit properties.** Additionally, the approximated transfer function of this system is independent of the input frequency, or more simply;

$$G_c(s) = \frac{\frac{R_f}{R_1}K_m}{s^2 + s\alpha + \frac{R_f}{R_1}K_m}$$

and recall,

$$W_n = \sqrt{b} = \sqrt{K_m * \frac{R_f}{R_1}}$$
$$\zeta = \frac{a}{2b} = \frac{a}{2 * W_n} = \frac{\alpha}{2 * \sqrt{K_m * \frac{R_f}{R_1}}}$$

and

$$\%OS = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}} * 100$$
$$T_s = \frac{4}{\zeta W_n}$$

Thus, as neither W_n or ζ are dependent on frequency, both overshoot and settling time are also independent of frequency. This explains why besides the reduced time available for the signal to settle, no visible changes to these properties were observed as the frequency was increased.

The impact of the systems gain however, was much more noticeable. From figure 29, it can be seen that increasing the gain resulted in an increased overshoot and a decreased settling time. This has been proven more in depth in the associated answers section, but a summary of the findings can be found below.
From the previous equations, it can be seen that both ζ and W_n rely on the gain ($K = R_f/R_1$).

Furthermore, this means both the overshoot and settling time rely on the gain factor. **SUB IN TO FIND OS and Ts AND SHOW THEY RELY ON GAIN, PROP/invPROP**
WHY IS THIS USEFUL - prop controller?
D4/D5

6 Discussion/Recommendations:

0.5-1page

7 Appendices

7.1 Pre-lab:

7.1.1 Lachlan Nicholson

1. Below is the requested functional diagram for the complete servo motor control system.

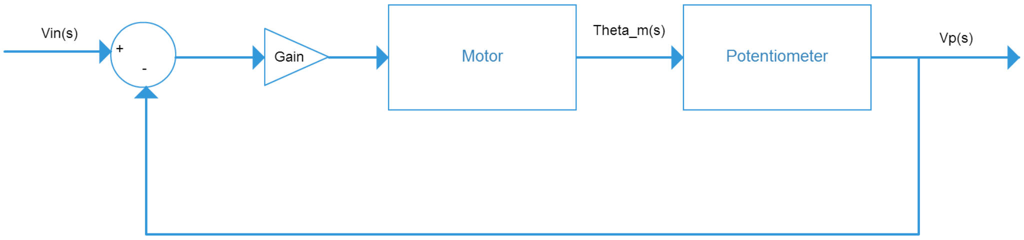


Figure 1: Functional Diagram of the control system

2. The updated functional diagram has been included below.

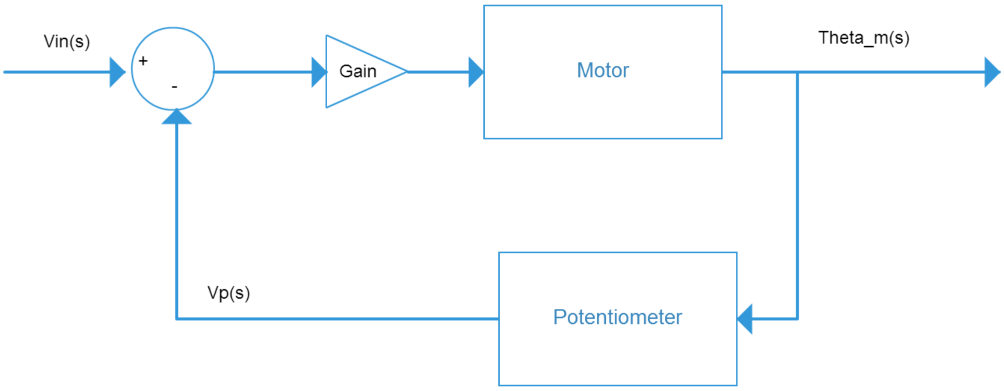


Figure 2: Updated Functional Diagram

3. The output of the requested matlab script has been included below, after which, the code itself has been provided.

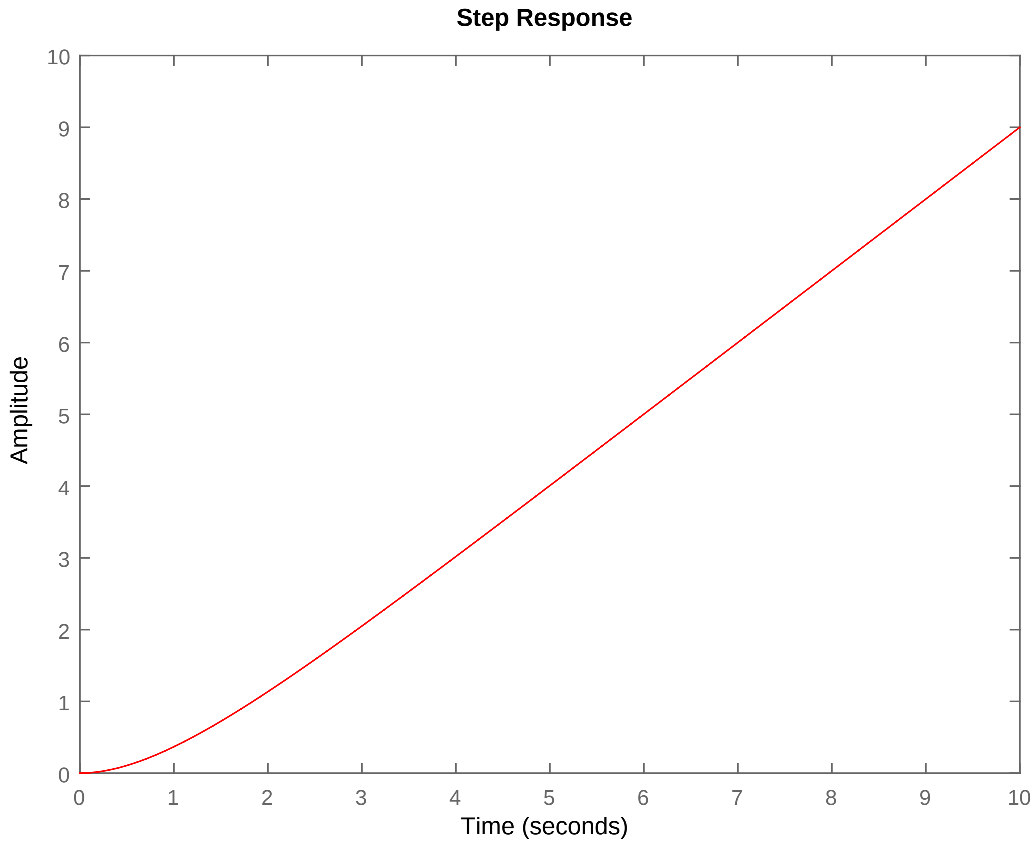


Figure 3: Simulated Response

```
%% A3
Alpha = 1;
K_m = 1;

t = linspace(0,10,100);
G_0 = tf([K_m],[1 Alpha 0]);
figure();
step(G_0, t, 'r')
print('-depsc','a3')
```

4. The given data describing the step response of an ideal servo model has been compared to the previously estimated system response.

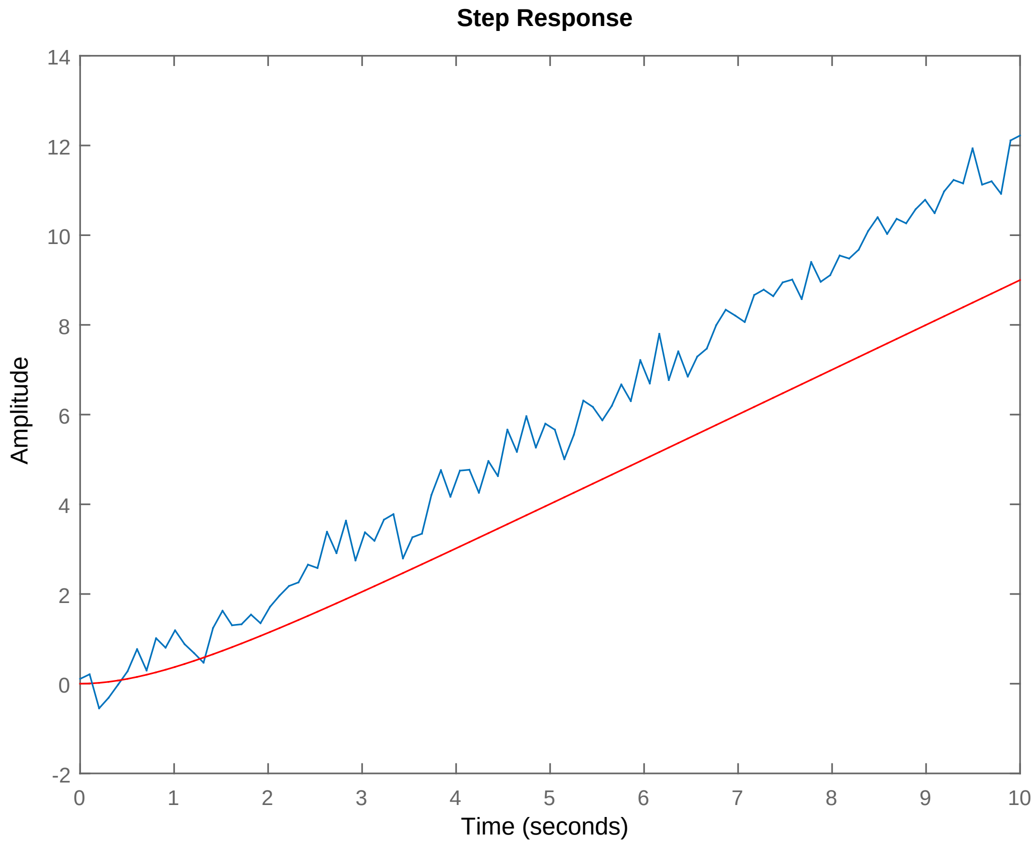


Figure 4: Given Data and Estimated Response

```
%% A4
Alpha = 1;
K_m = 1;
t = linspace(0,10,100);
G_0 = tf([K_m],[1 Alpha 0]);

load ENB301TestData_2015.mat
figure();
plot(t,y1);
hold;
step(G_0, t, 'r')
print('-depsc','a4')
```

5. **steady/transient Q**

6. After changing the values of K_m and α in the previous matlab script, it was found that values of $K_m = 1.3$ and $\alpha = 1$ resulted in an output that matched the test data. Furthermore, a script was created specifically to automatically find the best estimates for values of K_m and α within a given range. The results of both the manual and automatic estimations have been included below.

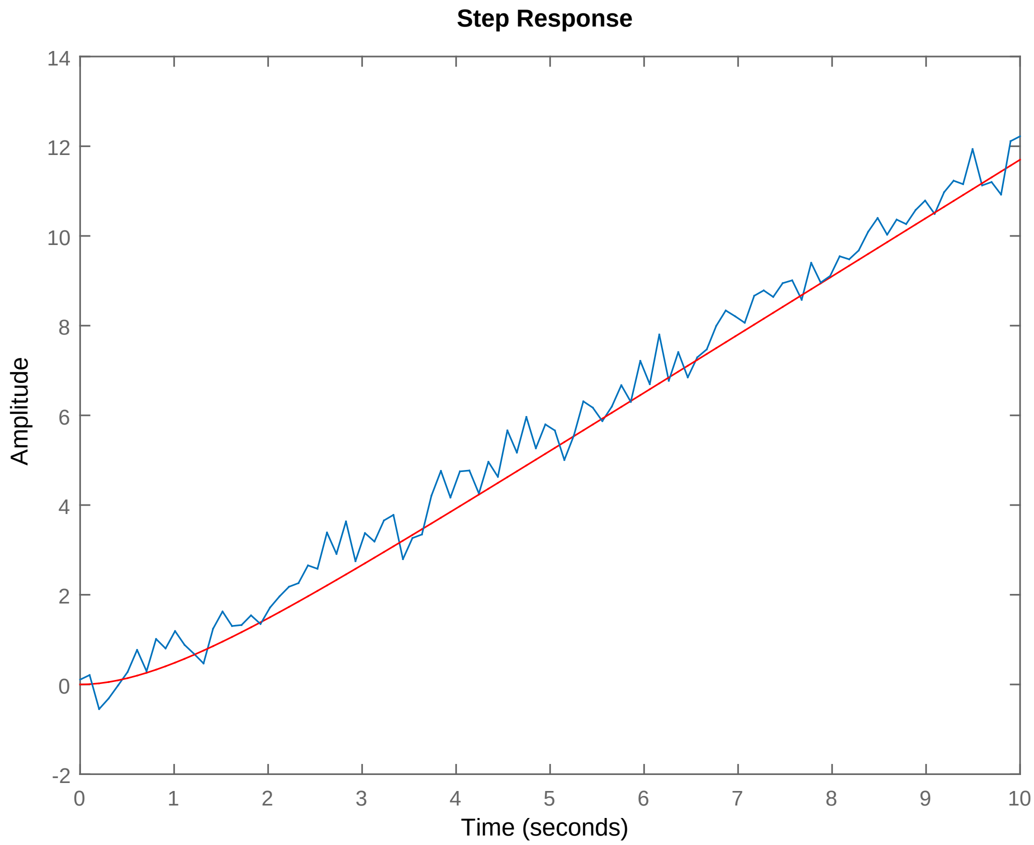


Figure 5: Manually Estimated System Values

```
%% A6
Alpha = 1; %0.5-1.5
K_m = 1.3; %1-2
G_0 = tf([K_m],[1 Alpha 0]);
figure();
plot(t,y1);
hold;
step(G_0, t, 'r');
print('-depsc','a6a')
```

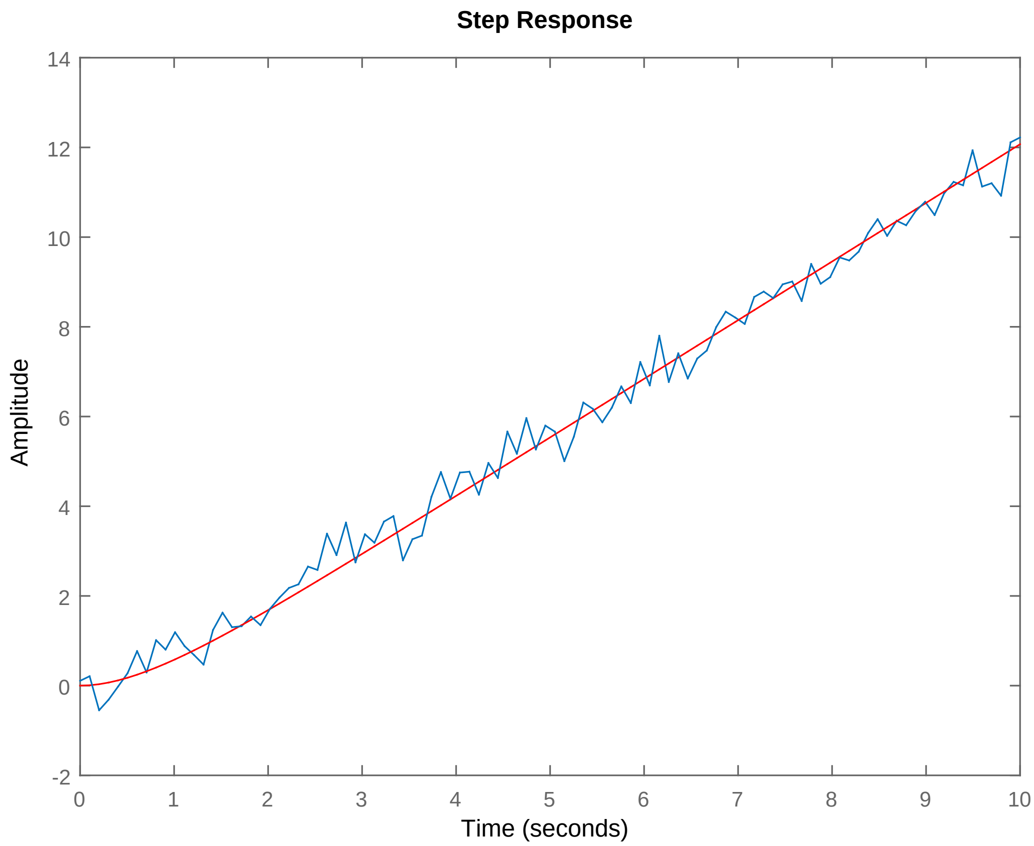


Figure 6: Inverting Amplifier System

```
%% Automated:
% loop values of alpha and k_m
% calculate RMS error each time
% pick values with the least error

y1_rms = rms(y1);
prevdiff = 10000;

for Alpha = 0.1:0.1:3
    for K_m = 0.1:0.1:3
        G_0 = tf([K_m],[1 Alpha 0]);
        G_t = step(G_0, t, 'r');
        Gt_rms = rms(G_t);
        diff = y1_rms - Gt_rms;
        if (abs(diff) < abs(prevdiff))
            prevdiff = diff;
            K_m_f = K_m;
            Alpha_f = Alpha;
        end
    end
end

G_0 = tf([K_m_f],[1 Alpha_f 0]);
figure();
plot(t,y1);
hold;
step(G_0, t, 'r');
print('-depsc','a6b')

% Final values:
% Alpha_f = 1.3
% K_m_f = 1.7
```

7. Using matlabs random number generator, uncertainty was added to the output $y_1(t)$, and both the ideal and noisy response were plotted. **QUESTIONS**

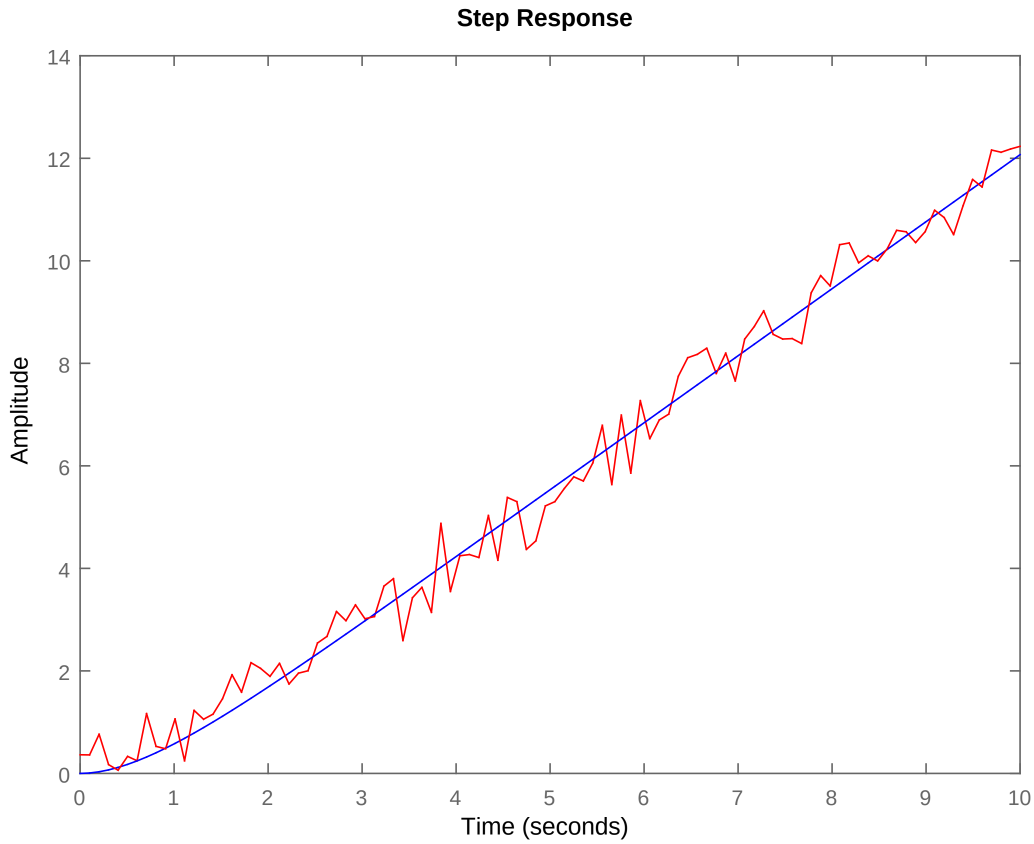


Figure 7: Estimated System Response + Uncertainty

```
%% A7
rand_power = 2;

y_1 = step(G_0, t, 'r');
y_n = y_1 + (rand_power/2)*rand(size(y_1,1),1) - ...
    (rand_power/2)*rand(size(y_1,1),1);

figure();
step(G_0, t, 'b');
hold;
plot(t,y_n,'r');
print('-depsc','a7')
```

7.1.2 Declan Gilmour

1. **Model the complete servo motor control system using a functional diagram. Label all signals and assumptions made including measurement units.**

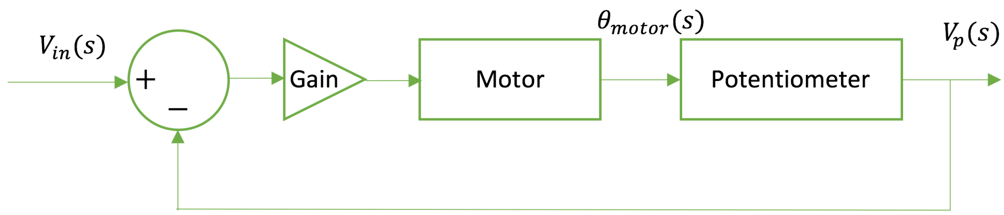


Figure 8: Servo Motor Functional Diagram

A voltage is fed into the motor with a magnitude gain. From here the voltage is converted into rotational movement $\theta_{motor}(s)$ by the motor. This rotational movement alters the displacement of the potentiometer wiper. The potentiometer acts as a sensor to indicate arm displacement in the form of output voltage $V_p(t)$.

2. **Update the servo motor functional diagram and make any necessary changes.**

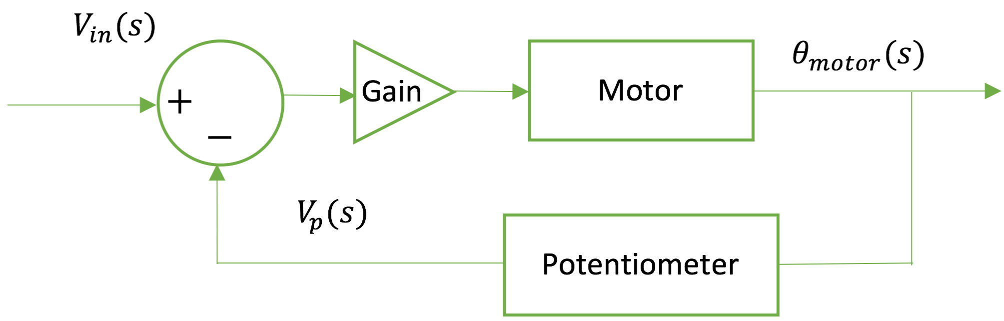


Figure 9: Updated Servo Motor Functional Diagram

To calculate the two servo motor system characteristics, α and K_m the system response would need to be recorded to a step input. This open loop time response is measured in terms of how the voltage applied to the motor affects the voltage seen at the potentiometer.
The selection of optimum values of α and K_m, wold require the potentiometer voltage to be measured with respect of the input voltage. The modification of the Servo Motor Functional Diagram is needed, incorperating a feedback loop as shown.

3. **Create code to simulate motor shaft angle for various values of K_m and α**
Bellow is the Matlab code used to simulate the motor shaft angle.

```
%% A3 - Simulate motor shaft angle for values of km and alpha
alpha = 1;
km = 1;
t = linspace(0,10,100);

% Build transfer function
G = tf(km, [1 alpha 0]);     % Set G(s) = km / (s + a)
G_0 = step(G,t);     % Set G_0(s) = km / (s * (s + a))

% Plot motor set response
figure
plot(t,G_0,'r')
title('Simulated Step Response')
xlabel('t (sec)')
ylabel('Amplitude')
print('-depsc','A3')
close
```

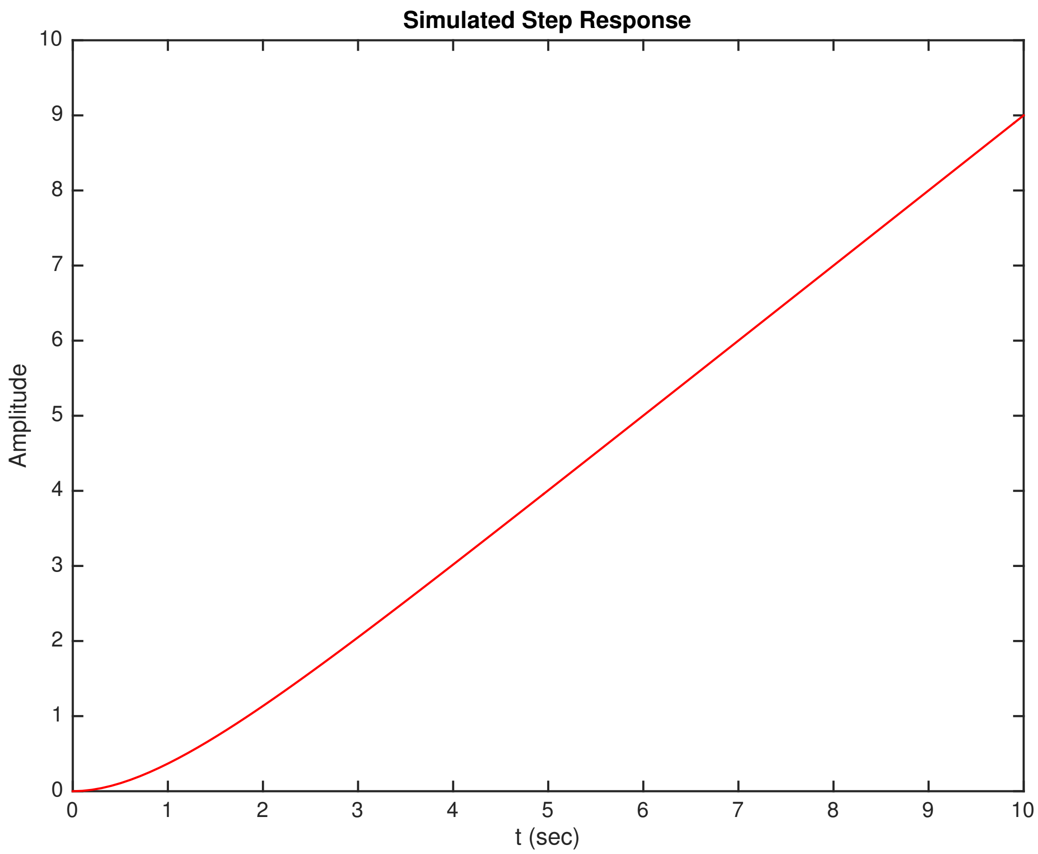


Figure 10: Updated Servo Motor Functional Diagram

4. **Plot $y_1(t)$ against the servo motor response in ENB301_TestData and compare.**
Bellow is the Matlab code used to generate the simulated step response against imported test data.

```
%% A4 - Plot motor unit step response data against simulated response
load ENB301TestData_2015.mat
alpha = 1;
km=1;
G = tf(km,[1 alpha 0]);
G_0 = step(G,t);

figure
hold on
plot(t,y1,'b')
plot(t,G_0,'r')
title('Simulated Step Response and Test Data')
xlabel('t (sec)')
ylabel('Amplitude')
legend('y1(t): Simulated Step Response','Test Data')
hold off
print('-depsc','A4')
close
```

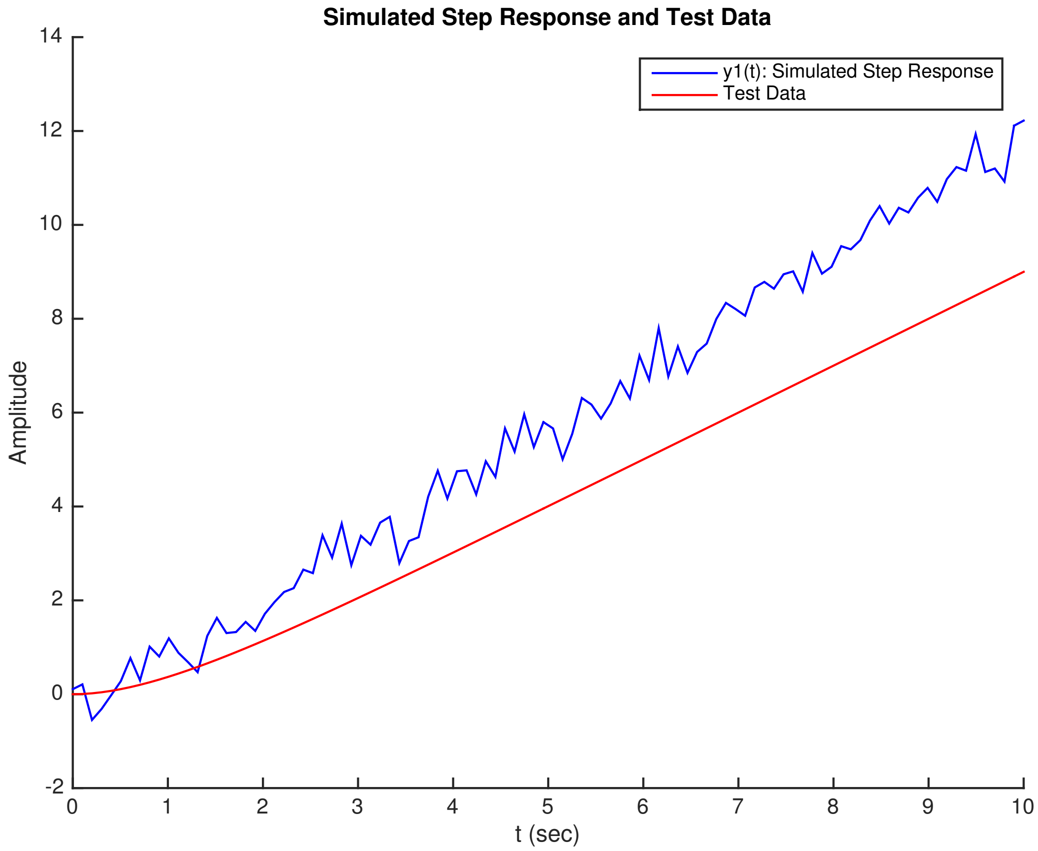


Figure 11: Updated Servo Motor Functional Diagram

5. **What is the steady state response and transient response predominantly determined by?**

6. **Change the values of K_m and α for to match the simulated response with the test data.**

   The estimated K_m value is 2.3838 and α 1.8990. Bellow is the Matlab Code to estimate km and alpha variables.

```
%% A6 - Estimate km and alpha variables
load ENB301TestData_2015
output = zeros(3,100*100);
error = 0;
ii = 1;
for km = linspace(0,4,100)
   for alpha = linspace(0,4,100)
       G = tf(km, [1 alpha 0]);
       G_0 = step(G,t);

       % Calculate mean square error
       for jj = 1 : length(t)
          error = error + (G_0(jj) - y1(jj))^2;
       end

       output(:,ii) = [km;alpha;error];
       ii = ii + 1;
       error = 0;
   end
end

[~,index] = min(output(3,:));
km = output(1,index); % Output variable
alpha = output(2,index);  % Output variable

% Build transfer function
G = tf(km, [1 alpha 0]);     % Set optimal G(s) = km / (s + a)
G_0 = step(G,t);     % Set optimal G_0(s) = km / (s * (s + a))

% Plot step function G_0(s)
figure
hold on
plot(t,G_0,'-b');
plot(t,y1,'-r');
title('Estimated Step Response of Test Data')
xlabel('t (sec)')
ylabel('Amplitude')
legend('Estimated Step Response', 'Test Data');
hold off;
print('-depsc','A6')
close

% Output km and alpha
disp(km)    %2.3838
disp(alpha) %1.8990
```

Figure 12: Estimated K_m and α values

7. **Plot the step response from original signal and a step response from a noisy signal.** Bellow is the Matlab Code to plot a step response from both original and noisy signals. $K_m = 2.4242$ and $\alpha = 1.9394$.

```
%load ENB301TestData_2015
% Generate noise
% Standard: sigma = 5, Decreased: signma = 0.2, Increased: sigma = 20
sigma = 5; %noise standard deviation
noise = sigma*randn(size(G_0)); %noise vector

% Create noisy signal
yn = y1 + noise;

output = zeros(3,100*100);
error = 0;
ii = 1;
for km = linspace(0,4,100)
   for alpha = linspace(0,4,100)
       G = tf(km, [1 alpha 0]);
       G_0 = step(G,t);

       % Calculate mean square error
       for jj = 1 : length(t)
          error = error + (G_0(jj) - yn(jj))^2;
       end

       output(:,ii) = [km;alpha;error];
       ii = ii + 1;
       error = 0;
   end
end

[~,index] = min(output(3,:));
km = output(1,index); % Output variable
alpha = output(2,index);  % Output variable

% Build transfer function
G_noisy = tf(km, [1 alpha 0]);    % Set optimal G(s) = km / (s + a)
G_0_noisy = step(G_noisy,t);    % Set optimal G_0(s) = km / (s * (s + a))

% Output km and alpha
disp(km)    %2.4242
disp(alpha) %1.9394

% Plot noisy signal vs noisy test data
figure(2)
hold on
plot(t,G_0_noisy, '-r');
plot(t,G_0,'-b');
title('Estimated Step Response of Noisy Test Data')
xlabel('t (sec)')
ylabel('Amplitude')
legend('Noisy Step Response', 'Original Step Response');
hold off
print('-depsc','A7')
close
```

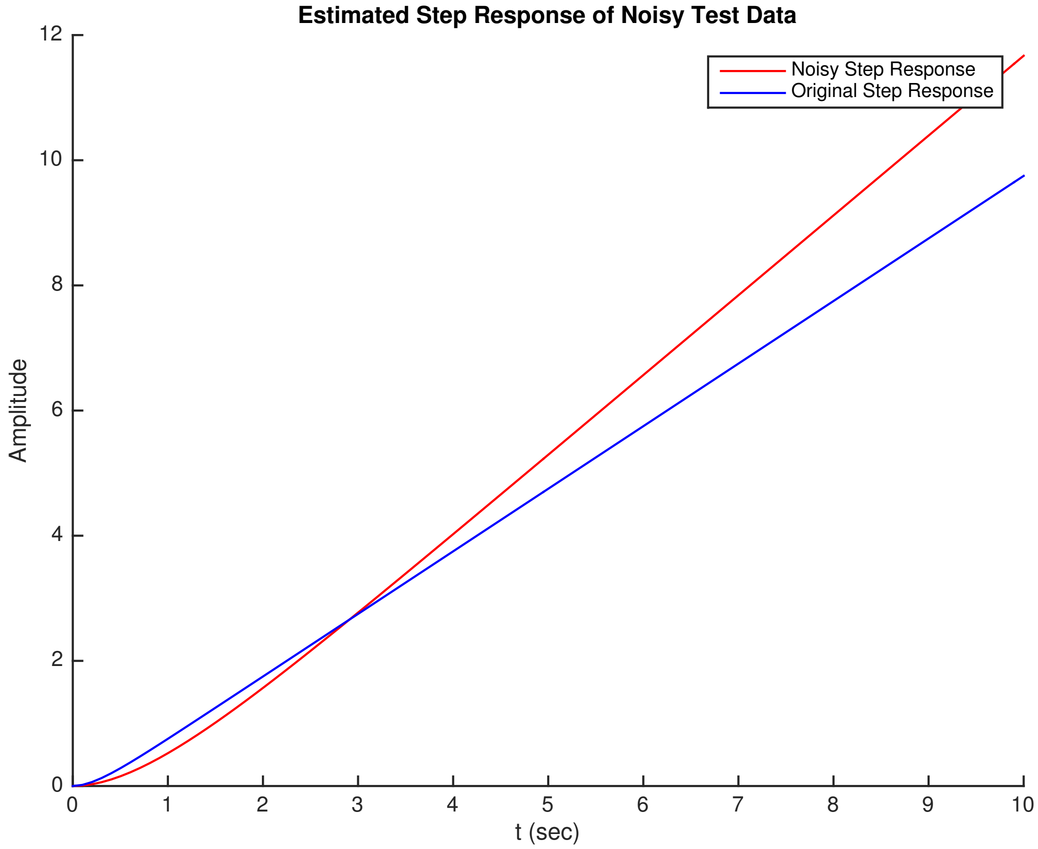
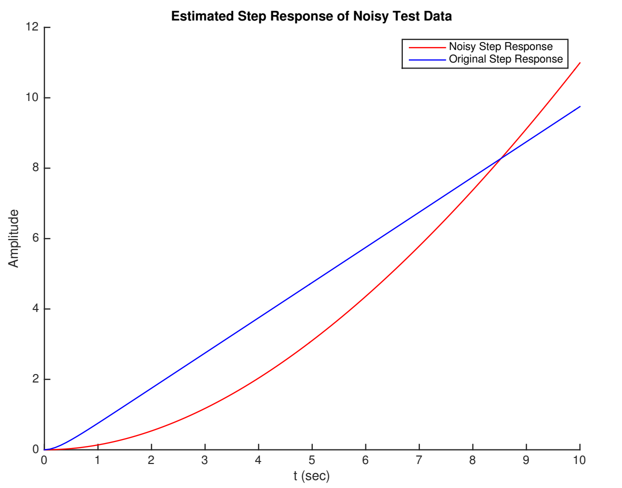


Figure 13: Original Step Response and Noisy Step Response

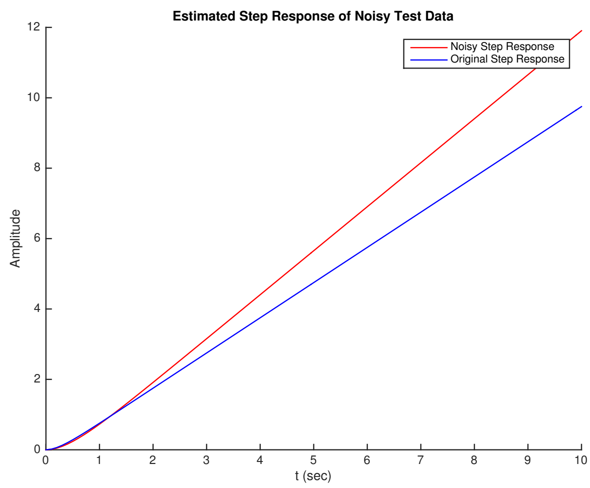
For this noisy step response, $K_m = 1.4949$ and $\alpha = 1.1717$. When the noise noise decreases, K_m and α decreases in magnitude. When the noise increases, K_m and α increases in magnitude. The transient response of the system appears to increase as the noise of the system increases.

Is this Realistic?

Whare are the source of uncertanty or noise in a real system?
Sources of uncertainty or noise in a system can include: control error, electrical noise, external factors such as weather and human error.



(a) Increased Noise: $K_m = 0.2828$ and $\alpha = 0.0808$



(b) Decreased Noise: $K_m = 2.6263$ and $\alpha = 2.1010$

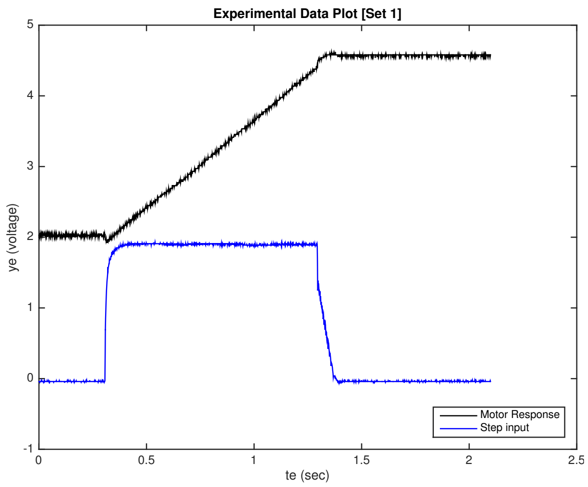
Figure 14: Increased and Decreased noisey Step Response

7.2 Answers:

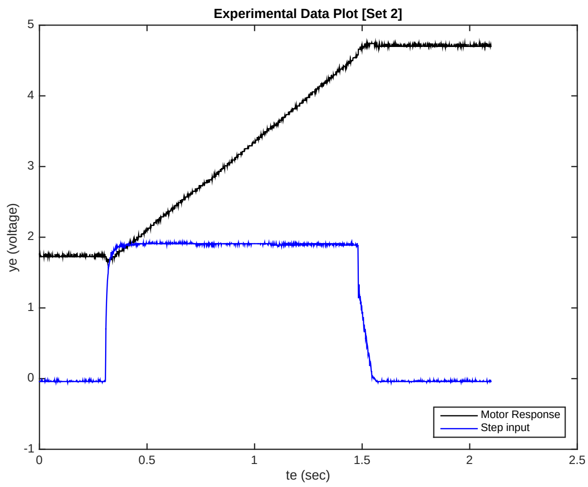
7.2.1 Part B

1. Read experimental data file(s) and store in vectors $y_e(t)$ and t_e. Plot the experimental results in black.

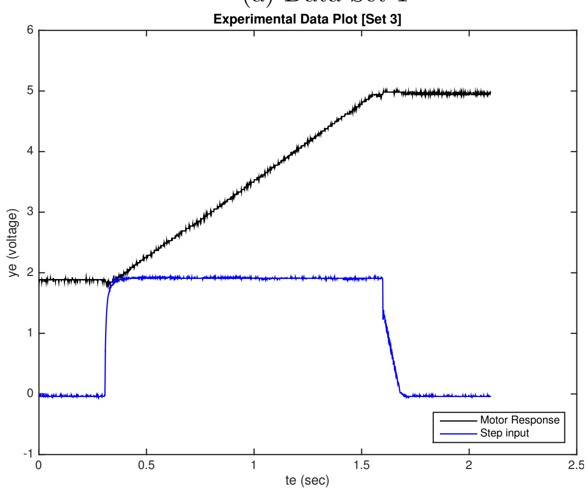
```matlab
%% B1
% Load experimental data and plot in black
data = csvread('PartB_Test1.csv',2,0);  % Read in test 1
te_1 = data(1:end,1);    % Store te variable
te_1 = te_1 + abs(te_1(1)); % Move te variable to start at zero
ye_1 = data(1:end,2);    % Store ye variable
ye_1step = data(1:end,3);    % Store ye step input variable

figure
plot(te_1,ye_1,'k',te_1,ye_1step,'b')   % Plot ye in black and ye in blue
title('Experimental Data Plot [Set 1]')
xlabel('te (sec)')
ylabel('ye (voltage)')
print('-depsc',strcat('figures',filesep,'B1_dataset1'));    % Store figure
close
```



(a) Data Set 1



(b) Data Set 2



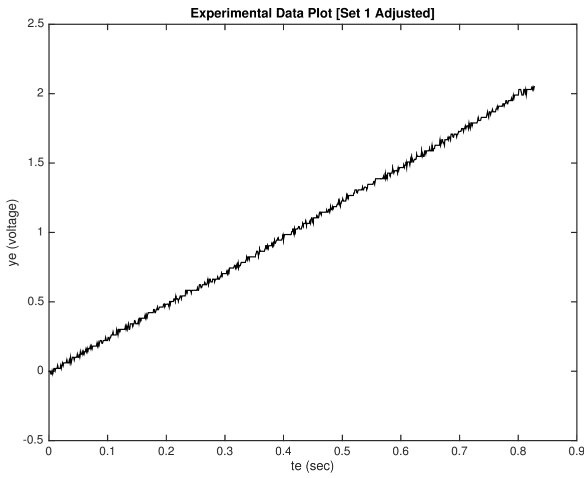
(c) Data Set 3

Figure 15: Open Loop response

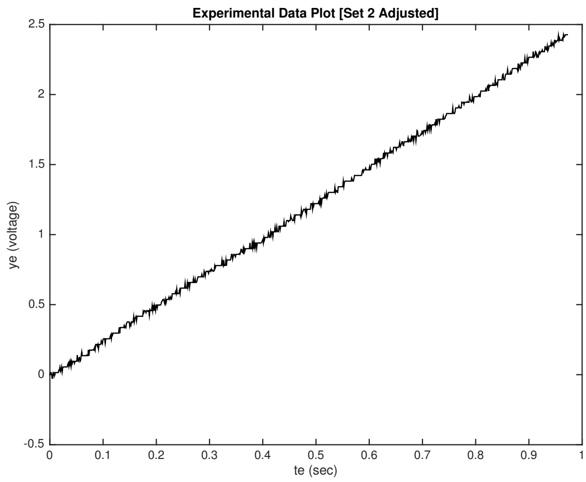
2. Modify the MATLAB script created in the pre-labs to simultaneously plot both the experimental data and $y_1(t)$. No real idea what to put here...

3. Derive a figure of merit for the estimation compared with experimental results.
   Mean Square error calculation: $error = (G_0 - y_e)^2$
   Root Mean Square error calculation: $error = \sqrt[2]{(G_0 - y_e))^2}$

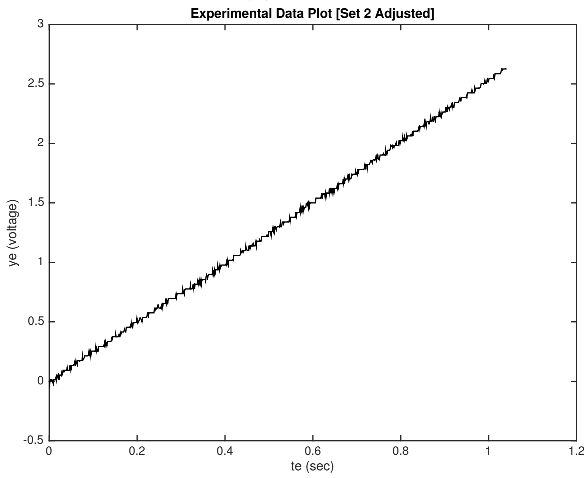
4. Improve estimates using the plots and figure of merit calculations. Derive the two parameters for the servo motor function $G(s) = \frac{K_m}{(S+\alpha)(S+\beta)}$.



(a) Data Set 1



(b) Data Set 2



(c) Data Set 3

Figure 16: Servo Motor Response Modified: k_m and α

```
% Plot experimental data ye against systems estimated tf y1
[te_1new,ye_1new] = timing_fix(te_1,ye_1);
[te_2new,ye_2new] = timing_fix(te_2,ye_2);
[te_3new, ye_3new] = timing_fix(te_3,ye_3);

figure
plot(te_1new,ye_1new,'k')
title('Experimental Data Plot [Set 1 Adjusted]')
xlabel('te (sec)')
ylabel('ye (voltage)')
print('-depsc',strcat('figures',filesep,'B2_dataset1'));
close

figure
plot(te_1new,ye_1new,'k')
```

```
title('Experimental Data Plot [Set 1 Adjusted]')
xlabel('te (sec)')
ylabel('ye (voltage)')
print('-depsc',strcat('figures',filesep,'B2_dataset1'));
close
```

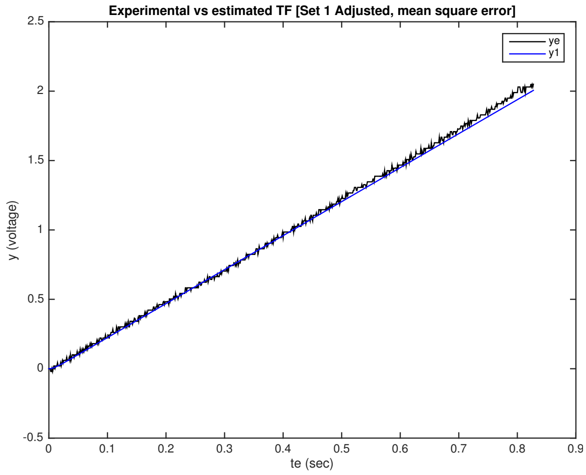
```
% Pepare experimental data for alpha and km parameter calculations
[te_1trim, ye_1trim] = trimForCalculation(te_1new,ye_1new,mf1);
[te_2trim, ye_2trim] = trimForCalculation(te_2new,ye_2new,mf1);
[te_3trim, ye_3trim] = trimForCalculation(te_3new,ye_3new,mf1);

km_num = 500;    % number of km values used
km_max = 500;   % max km value used
alpha_num = 500;    % number of alpha values used
alpha_max = 500;    % max alpha value used
```

```
% Preallocate size for speed
output_ms = zeros(3,km_num*alpha_num);
output_rms = zeros(3,km_num*alpha_num);
km_ms = zeros(1,3);
alpha_ms = zeros(1,3);
km_rms = zeros(1,3);
alpha_rms = zeros(1,3);

for iteration = 1 : 3

    % Set te and ye based on iteration
    if (iteration == 1)
        te = te_1trim;
        ye = ye_1trim;
    elseif (iteration == 2)
        te = te_2trim;
        ye = ye_2trim;
    else
        te = te_3trim;
        ye = ye_3trim;
    end

    % Set cycle variables
    error_ms = 0;
    error_rms = 0;
    ii = 1;
    count = 0;

    for km = linspace(0,km_max,km_num)   % Cycle km values
       for alpha = linspace(0,alpha_max,alpha_num)  % Cycle alpha values
           G = tf(km, [1 alpha 0]);
           G_0 = step(G,te);

           % Calculate error
           for jj = 1 : length(te)
              % Calculate mean square error
              error_ms = error_ms + (G_0(jj) - ye(jj))^2;

              % Calculate root mean square error
              error_rms = error_rms + rms(G_0(jj) - ye(jj));
           end

           % Store km, alpha and the error taken to calculate
           output_ms(:,ii) = [km;alpha;error_ms];
           output_rms(:,ii) = [km;alpha;error_rms];

           % Reset cycle variables
           ii = ii + 1;
```

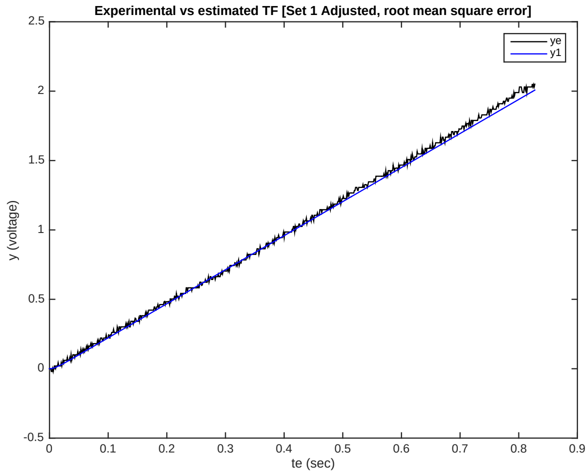
```
            error_ms = 0;
            error_rms = 0;
        end

        % Output km iterations
        %count = count + 1;
        %fprintf('%d %s %d\n',count,'/',km_num);
    end

    % Calculate km and alpha values for mean square error calculation
    [~,index] = min(output_ms(3,:));
    km_ms(iteration) = output_ms(1,index); % Output variable
    alpha_ms(iteration) = output_ms(2,index);  % Output variable

    % Calculate km and alpha values for root mean square error calculation
    [~,index] = min(output_rms(3,:));
    km_rms(iteration) = output_rms(1,index); % Output variable
    alpha_rms(iteration) = output_rms(2,index);  % Output variable
end
km_mean = (mean(km_ms) + mean(km_rms)) / 2;
alpha_mean = (mean(alpha_ms) + mean(alpha_rms)) / 2;
```

```
% Plot y1 against ye
% Data Set 1
G = tf(km_ms(1), [1 alpha_ms(1) 0]);
y1_ms = step(G,te_1new);
figure
plot(te_1new,medfilt1(ye_1new,1),'k',te_1new,medfilt1(y1_ms,1),'b')
title('Experimental vs estimated TF [Set 1 Adjusted, mean square error]')
xlabel('te (sec)')
ylabel('y (voltage)')
legend('ye','y1')
print('-depsc',strcat('figures',filesep,'y1_dataset1_ms'));
close
```

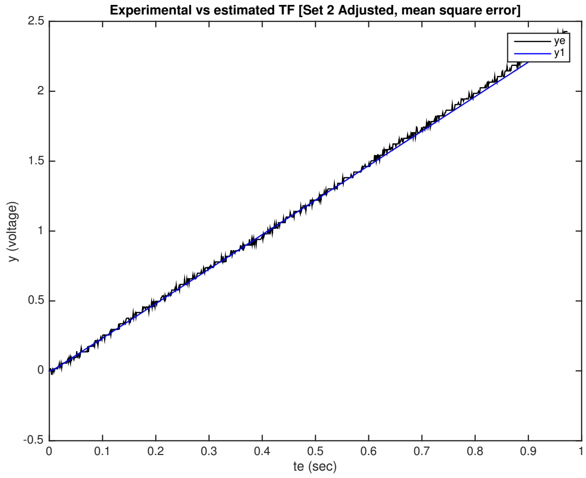
The average constants found are as follows: $\alpha = 170.3407$ $k_m = 422.0107$



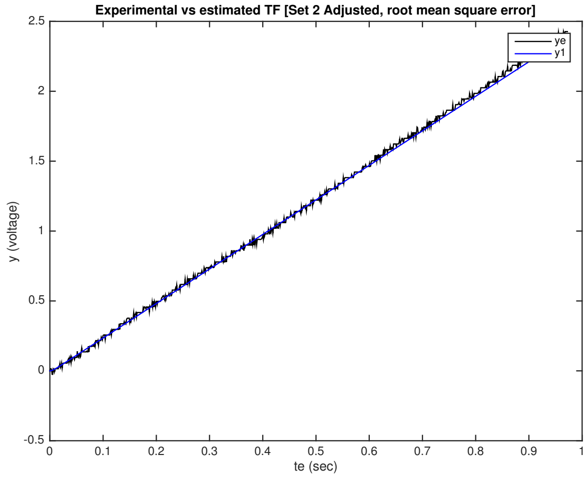
(a) Data Set 1 (ms)



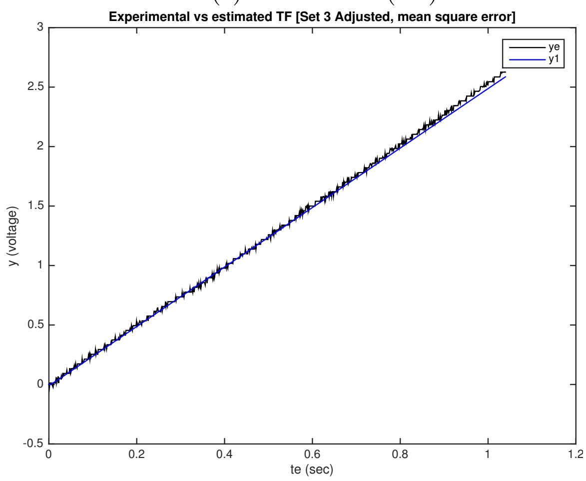
(b) Data Set 1 (rms)



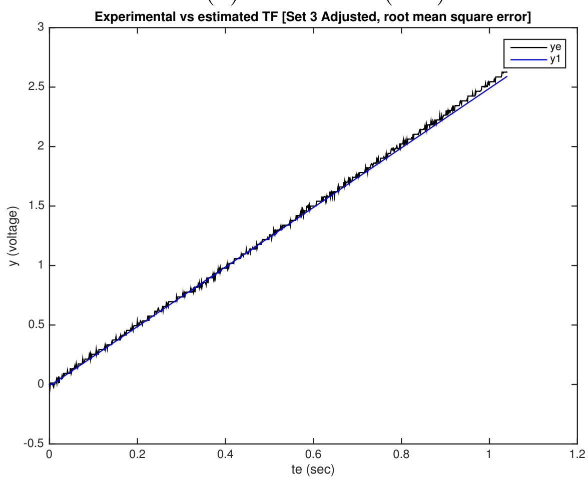
(c) Data Set 2 (ms)



(d) Data Set 2 (rms)



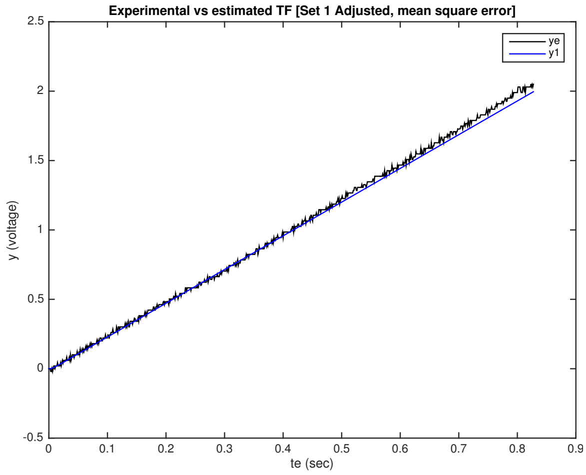
(e) Data Set 3 (ms)



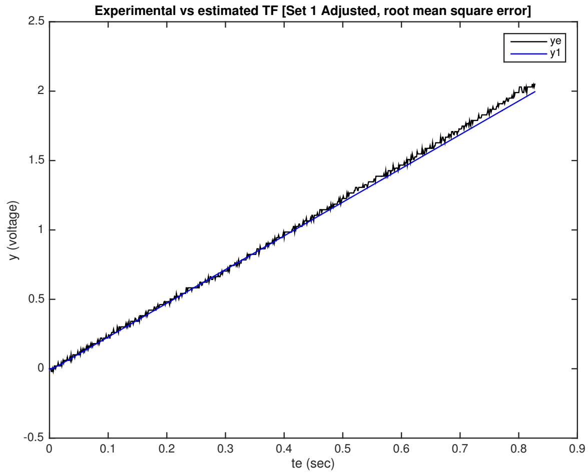
(f) Data Set 3 (rms)

Figure 17: Servo Motor Response Modeled

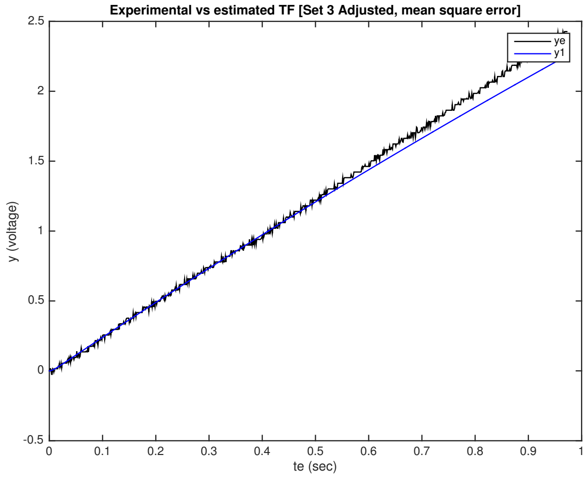
5. The mechanical constant α was found to be $\alpha = alpha_{mean}$. Using the second order equation $G(s) = \frac{K_m}{(S+\alpha)}$ new values of gain constant k_m and electrical constant β. Values were found for $y_1(t) \approx y_2(t)$. The average constants found are as follows: $\beta = 0.0646$ $k_m = 426.6867$



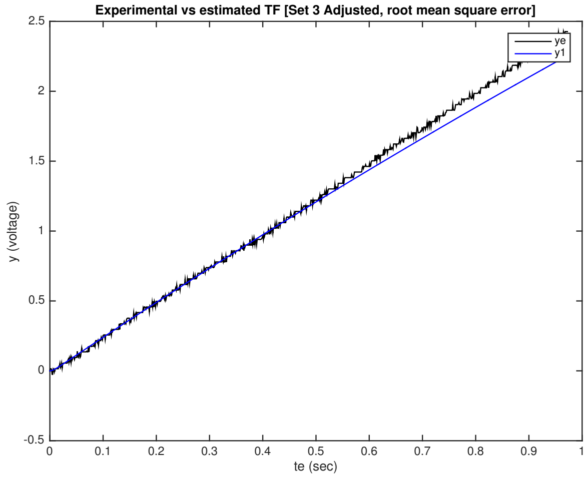
(a) Data Set 1 (ms)



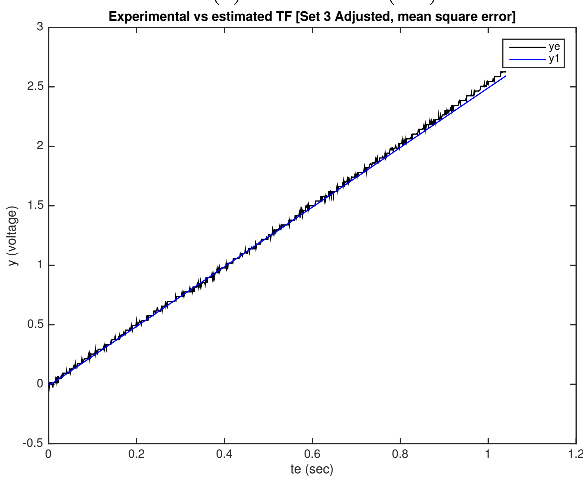
(b) Data Set 1 (rms)



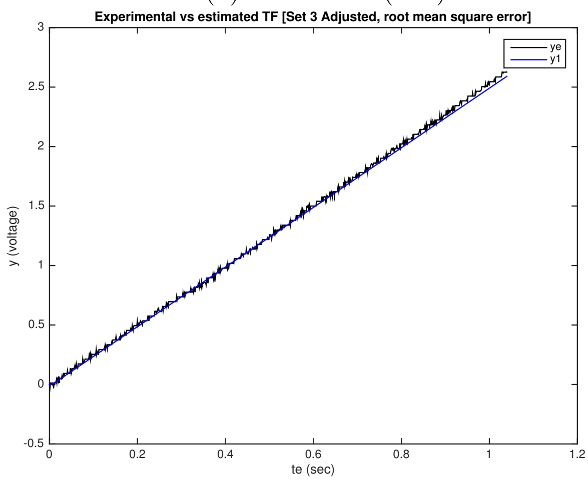
(c) Data Set 2 (ms)



(d) Data Set 2 (rms)



(e) Data Set 3 (ms)



(f) Data Set 3 (rms)

Figure 18: Servo Motor Response Modeled: k_m and β

```
km_num = 500;     % number of km values used
km_max = 500;    % max km value used
beta_num = 50;     % number of alpha values used
beta_max = 1;     % max alpha value used

% Preallocate size for speed
output_ms = zeros(3,km_num*beta_num);
output_rms = zeros(3,km_num*beta_num);
km2_ms = zeros(1,3);
beta_ms = zeros(1,3);
km2_rms = zeros(1,3);
beta_rms = zeros(1,3);

for iteration = 1:3
    if(iteration == 1)
        te = te_1trim;
        ye = ye_1trim;
    elseif(iteration==2)
        te = te_2trim;
        ye = ye_2trim;
    else
        te = te_3trim;
        ye = ye_3trim;
    end

    % Set cycle variables
    error_ms = 0;
    error_rms = 0;
    ii = 1;
    count = 0;

    for km = linspace(0,km_max,km_num)    % Cycle km values
       for beta = linspace(0,beta_max,beta_num)  % Cycle alpha values
           G = tf(km, [1 (alpha_mean+beta) (alpha_mean*beta)]);
           G_0 = step(G,te);

           % Calculate error
           for jj = 1 : length(te)
              % Calculate mean square error
              error_ms = error_ms + (G_0(jj) - ye(jj))^2;

              % Calculate root mean square error
              error_rms = error_rms + rms(G_0(jj) - ye(jj));
           end

           % Store km, alpha and the error taken to calculate
           output_ms(:,ii) = [km;beta;error_ms];
           output_rms(:,ii) = [km;beta;error_rms];

           % Reset cycle variables
           ii = ii + 1;
           error_ms = 0;
           error_rms = 0;
       end

        % Output km iterations
       %count = count + 1;
       %fprintf('%d %s %d\n',count,'/',km_num);
    end

    % Calculate km and alpha values for mean square error calculation
    [~,index] = min(output_ms(3,:));
    km2_ms(iteration) = output_ms(1,index); % Output variable
    beta_ms(iteration) = output_ms(2,index);  % Output variable

    % Calculate km and alpha values for root mean square error calculation
```

```
    [~,index] = min(output_rms(3,:));
    km2_rms(iteration) = output_rms(1,index); % Output variable
    beta_rms(iteration) = output_rms(2,index);  % Output variable
end

km_mean2 = (mean(km_ms) + mean(km2_rms)) / 2;
beta_mean = (mean(beta_ms) + mean(beta_rms)) / 2;
```

```
% Plot y2 against ye
% Data Set 1
G = tf(km2_ms(1), [1 (alpha_mean+beta_ms(1)) (alpha_mean*beta_ms(1))]);
y2_ms = step(G,te_1new);
figure
plot(te_1new,medfilt1(ye_1new,1),'k',te_1new,medfilt1(y2_ms,1),'b')
title('Experimental vs estimated TF [Set 1 Adjusted, mean square error]')
xlabel('te (sec)')
ylabel('y (voltage)')
legend('ye','y1')
print('-depsc',strcat('figures',filesep,'y2_dataset1_ms'));
close
```

7.2.2 Part C

1. **Re-draw figure 19 and place boxes around the set of components that correspond to each functional element of the control system.**

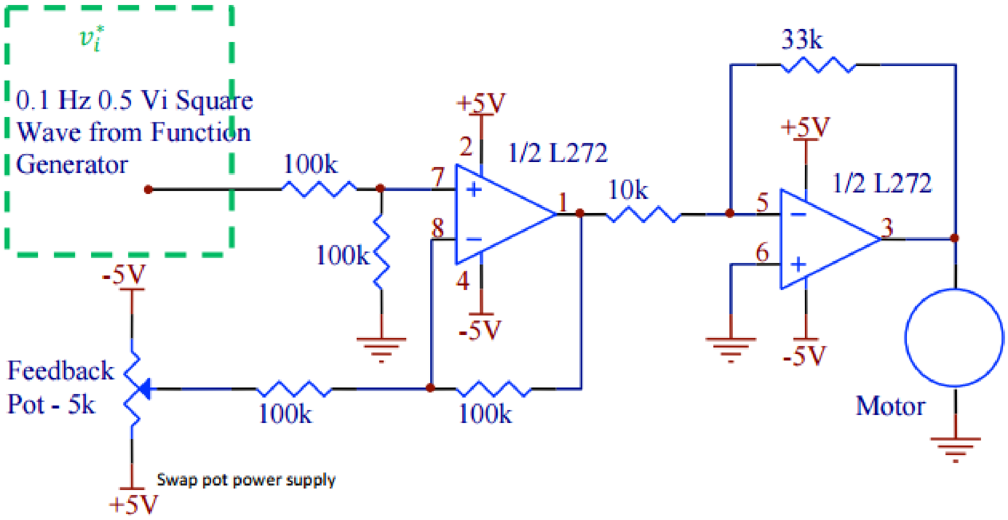


Figure 19: Closed Loop Motor Control Schematic

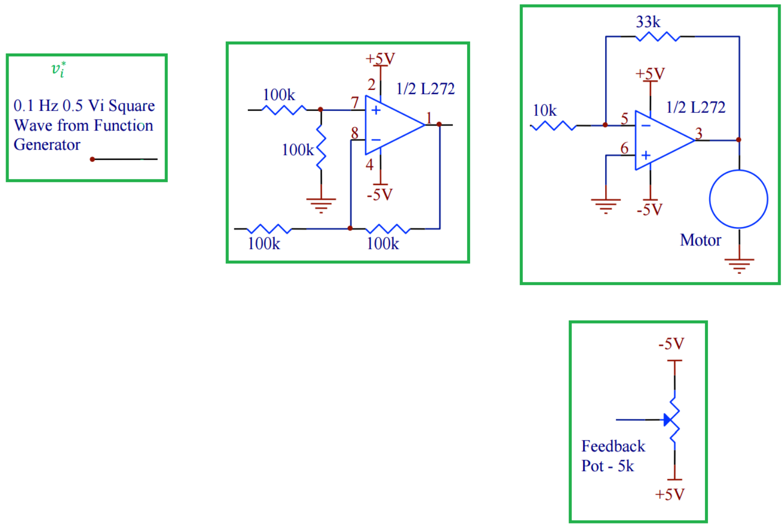
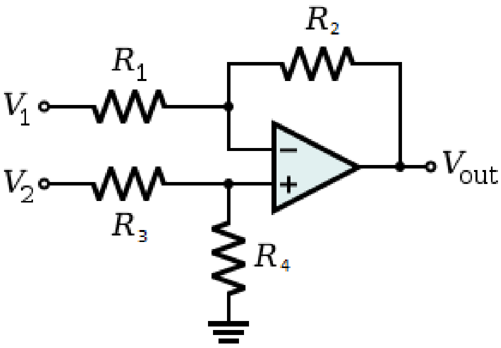


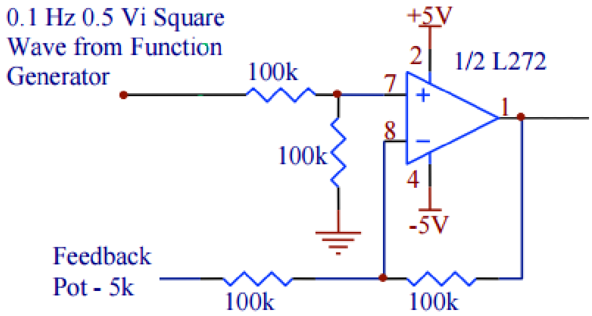
Figure 20: Closed Loop Motor Control System

2. **Based on the results from Parts A and B, the component values given in figure 19 and your research in parts C1, calculate all of the transfer functions in your functional diagram (figure 20). Update the functional diagram, labeling all components and interfaces.**

Difference Op-amp:



(a) Standard Difference Op Amp



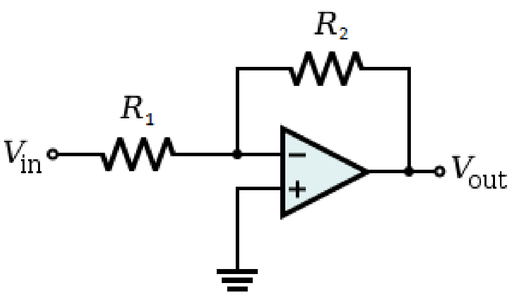
(b) System Difference Op Amp

Figure 21: Difference Op-amp System

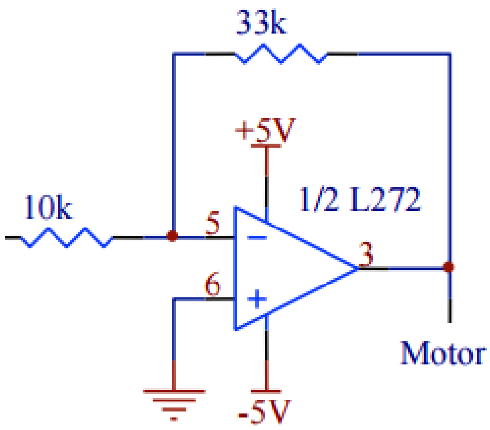
$$\frac{V_o(s)}{V_{in}(s)} = \frac{1 + \frac{R_2}{R_1}}{1 + \frac{R_3}{R_4}} V_2 - \frac{R_2}{R_1} V_1$$

$$\frac{V_o(s)}{V_{in}(s)} = V_o(s) - V_p(s)$$

Inverting Op-amp:



(a) Standard Difference Op Amp



(b) Inverting Op-amp System

Figure 22: Inverting Op-amp System

$$\frac{V_i(s)}{V_d(s)} = \frac{R_2}{R_1}$$

Note: The polarity on the inverter transfer function has been flipped.

Motor:

$$\frac{V_p(s)}{V_i(s)} = \frac{K_m}{s(s+\alpha)}$$

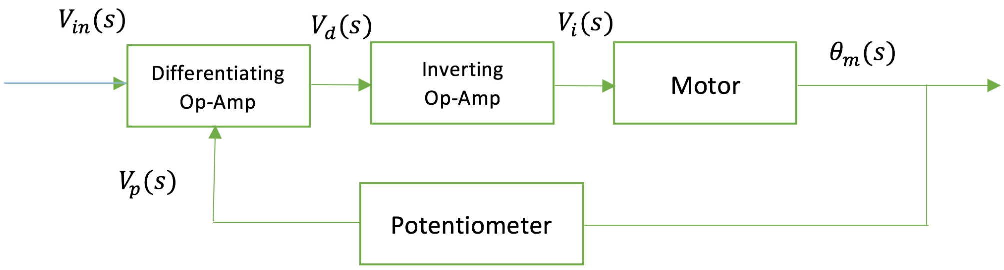


Figure 23: Updated Functional Diagram

3. Calculate the complete system transfer function $G_c(s)$.

System Transfer Function:

$$\begin{aligned}
G_c(s) &= \frac{V_p(s)}{V_{in}(s)} \\
&= \frac{V_p(s)}{V_i(s)} \times \frac{V_i(s)}{V_d(s)} \times \frac{V_d(s)}{V_{in}(s))} \\
&= \frac{K_m}{s(s+\alpha)} \times \frac{R_2}{R_1} \times (V_i(s) - V_p(s)) \\
&= \frac{-K_m R_2}{R_1} \left(\frac{V_i(s) - V_p(s)}{s(s+\alpha)} \right)
\end{aligned}$$

From Inverting OP amp: $V_m(s) = \frac{R_2}{R_2} \times V_o(s)$

From Motor: $V_m(s) = \frac{s(s+\alpha)V_p(s)}{K_m}$

$$\begin{aligned}
\therefore \frac{s(s+\alpha)V_p(s)}{K_m} &= \frac{R_2}{R_2} \times V_o(s) \\
V_o(s) &= \frac{R_1 s(s+\alpha)}{K_m R_2} V_p(s)
\end{aligned}$$

From Difference Op Amp: $V_o(s) = V_i(s) - V_p(s)$

$$\begin{aligned}
\therefore \frac{R_1 s(s+\alpha)}{K_m R_2} V_p(s) &= V_i(s) - V_p(s) \\
V_p(s) \left[1 - \frac{R_1 s(s+\alpha)}{K_m R_2} \right] &= V_i(s) \\
\frac{V_p(s)}{V_i(s)} &= \frac{1}{1 + \frac{R_1 s(s+\alpha)}{K_m R_2}} \\
&= \frac{1}{1 + \frac{R_1}{K_m R_2}(s^2 + \alpha s)} \\
&= \frac{1}{(\frac{R_1}{K_m R_2})s^2 (\frac{\alpha R_1}{K_m R_2})s + 1} \\
&= \frac{\frac{K_m R_2}{R_1}}{s^2 + (\frac{\alpha R_1 K_m R_2}{K_m R_2 R_1})s + \frac{K_m R_2}{R_1}} \\
&= \frac{\frac{K_m R_2}{R_1}}{s^2 + \alpha s + \frac{K_m R_2}{R_1}}
\end{aligned}$$

Ideal System Transfer Function:

$$G_c(s) = \frac{KG(S)}{1 + KG(s)H(s)}$$

$$G(s) = \frac{K_m}{s(s+\alpha)}$$
$$H(s) = 1$$
$$K = \frac{R_f}{R_1}$$

$$G_c(s) = \frac{\frac{R_f}{R_1}G(s)}{1 + \frac{R_f}{R_1}G(s)}$$
$$G_c(s) = \frac{\frac{R_f}{R_1}\frac{K_m}{s(s+\alpha)}}{1 + \frac{R_f}{R_1}\frac{K_m}{s(s+\alpha)}}$$
$$G_c(s) = \frac{\frac{\frac{R_f}{R_1}K_m}{s(s+\alpha)}}{1 + \frac{\frac{R_f}{R_1}K_m}{s(s+\alpha)}}$$
$$G_c(s) = \frac{\frac{R_f}{R_1}K_m}{s(s+\alpha) + \frac{R_f}{R_1}K_m}$$
$$G_c(s) = \frac{\frac{R_f}{R_1}K_m}{s^2 + s\alpha + \frac{R_f}{R_1}K_m}$$

As can be seen, both the of the above methods produce the same result, the first requiring more complex algebra and all of the calculated transfer functions; the second method understands that the summing amplifier and inverting amplifier can be modeled as a summing node and gain block respectively (refer to figure 2).

Where $K_m = 326$, $\alpha = 38.61$, $R_f = 33k$ and $R_1 = 10k$. Thus,

$$G_c(s) = \frac{\frac{33000}{10000}K_m}{s^2 + s\alpha + \frac{33000}{10000}K_m}$$
$$G_c(s) = \frac{3.3 * K_m}{s^2 + s\alpha + 3.3 * K_m}$$
$$G_c(s) = \frac{3.3 * 326}{s^2 + 38.61s + 3.3 * 326}$$
$$G_c(s) = \frac{1075.8}{s^2 + 38.61s + 1075.8}$$

4. **C4 - Matlab - Declan?**

5. **Calculate the gain required in the final stage to produce a 5% overshoot. Choose resistor values to match the required gain.**

   Recall, for 5% overshoot, $\zeta = \frac{-\ln(5/100)}{\sqrt{\pi^2+\ln^2(5/100)}} = 0.69$

   And, the systems estimated overall transfer function; $\frac{k_m \frac{R_f}{R_1}}{s^2+s\alpha+k_m \frac{R_f}{R_1}}$

   Whereas, the general second order transfer function; $\frac{W_n^2}{s^2+2\zeta W_n s+W_n^2}$

   Moreover,

$$
\begin{aligned}
\alpha &= 2\zeta W_n \\
W_n &= \frac{\alpha}{2\zeta}
\end{aligned}
$$

$$
\begin{aligned}
W_n^2 &= k_m \frac{R_f}{R_1} \\
\frac{R_f}{R_1} &= W_n^2/k_m \\
K &= (\frac{\alpha}{2\zeta})^2/k_m
\end{aligned}
$$

   From section B, we know $k_m = 326$ and $\alpha = 38.61$. We also calculated the required ζ previously, as 0.69.

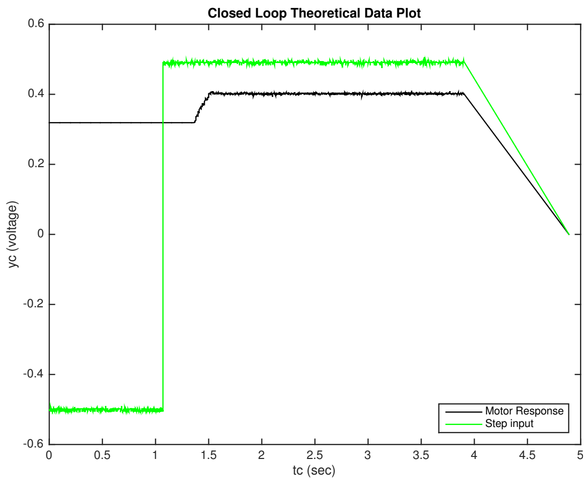
$$
\begin{aligned}
K &= (\frac{38.61}{2*0.69})^2/326 \\
K &= 2.4
\end{aligned}
$$

   Therefore, the gain required to achieve a 5% overshoot is as stated above; Moreover, to calculate the desired resistor values to achieve this game, we must make an initial assumption about either R_f or R_1.
   Assuming $R_1 = 10k$ (as to avoid changing both resistors);

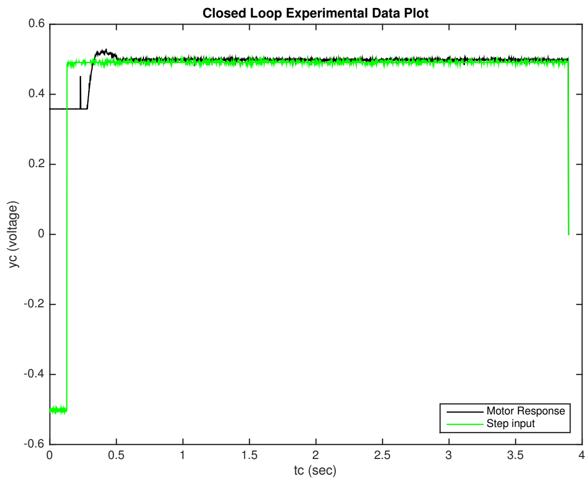
$$
\begin{aligned}
K &= 2.4 \\
\frac{R_f}{R_1} &= 2.4 \\
R_f &= 2.4 * R_1 \\
R_f &= 2.4 * 10000 \\
R_f &= 24000
\end{aligned}
$$

   Thus, theoretically, to achieve 5% overshoot a gain of $K = 2.4$ is required, to achieve this, $R_f = 24k \approx 22k + 1.8k + 220 = 24.2k$, and $R_1 = 10k$.

6. **Import experimental results into MATLAB. Compare your closed loop response data against your predicted model data $Y_c(t)$. Note any differences between the experimental result and the predicted result. What does this suggest about the model derived in part A and B?**

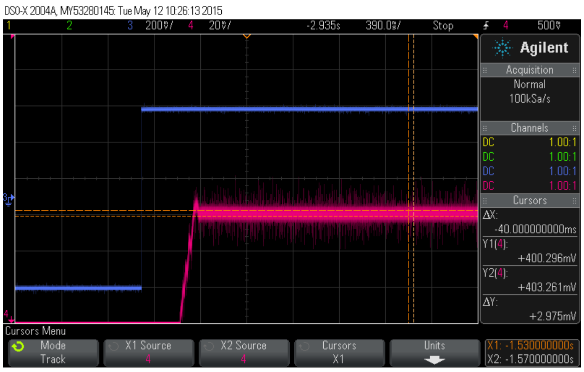


(a) Theoretical Closed Loop Response

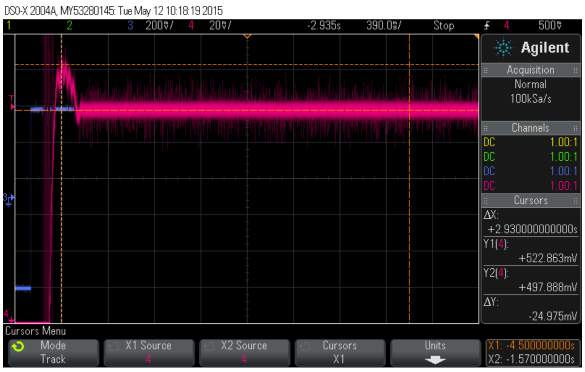


(b) Experimental Closed Loop Response

Figure 24: Servo Motor Closed Response



(a) Theoretical Closed Loop Response



(b) Experimental Closed Loop Response

Figure 25: Servo Motor Closed Response

$$\%OS = \frac{V_{max} - V_{avg}}{V_{avg}} \times 100$$

$$\begin{aligned}\%OS_{theoretical} &= \frac{403.261 - 400.296}{400.296} \times 100 \\ &= 0.7407\%\end{aligned}$$

$$\begin{aligned}\%OS_{experimental} &= \frac{522.863 - 497.888}{497.888} \times 100 \\ &= 5.016\%\end{aligned}$$

The motor response of the theoretical response was derived from $R_f = 25k$ ohms and $R_1 = 10k$ ohms with a calculated overshoot of 5%. As shown in the calculations above, the theoretical percentage overshoot is much smaller. To obtain a 5% overshoot,

7. **Compare your experimentally derived 5% overshot gain value against your predicted value. What is the percentage error? If your overshoot was too large for your derived gain value, could you use a controller other than the proportional controller to reduce overshoot?**

   If your steady state error was large, what other controller type could you use to minimize this error? What are the drawbacks of this type of controller? What control applications can you think of that require very low steady state error?

   From the procedure outlined in **experiment C**, an experimental gain of $K = 6.28$ resulted in an overshoot of 5%, and in **step 5** it was shown that the theoretical gain required to achieve this was $K = 2.4$.

   The percentage error between the calculated and experimentally found gain values has been calculated below.

$$\begin{aligned}
\%_{error} &= |\frac{K_{theoretical} - K_{experimental}}{K_{experimental}}| \times 100 \\
\%_{error} &= |\frac{6.28 - 2.4}{2.4}| \times 100 \\
\%_{error} &= 61.8\%
\end{aligned}$$

   A large % error value, as indicated above; provides strong evidence for model inaccuracies. The difference in gain between the theoretical and the practical values is likely caused by **WHAT**.

   The overshoot resulting from the derived gain value was under 5%.

   Steady state error was noticeable in the system, likely caused by mechanical losses (gearbox, heat, etc). To alleviate this issue, a PI compensator (integrator) could be used; this is because the PI controller adds up error over time and would remove the steady state error. Moreover, a PI controller retains a maximum overshoot and settling time similar to that of the proportional controller (adjusting the gain alone).

   However, the drawbacks for a PI controller include, but are not limited too; **WHAT**.

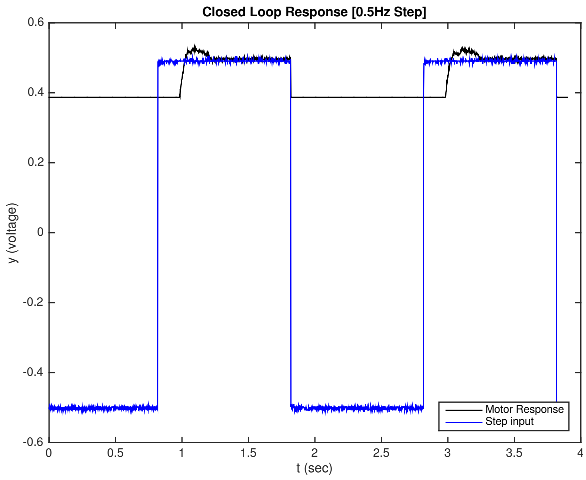
   A good example of a control system that required low steady state error is the cruise control on a car. This is largly

   Add the actual OS from prac/theo

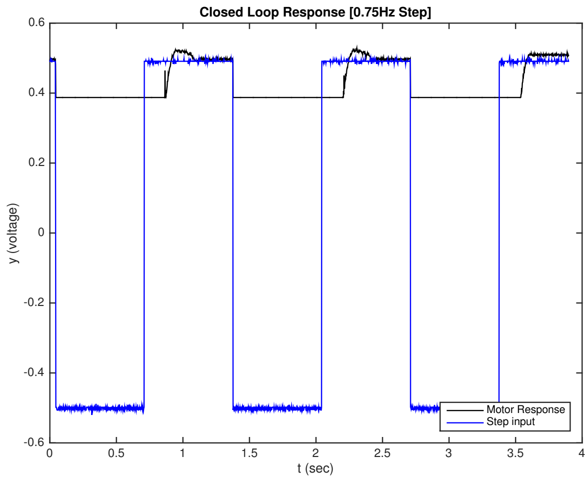
8. Another method of calculating TF? (bode)

7.2.3 Part D

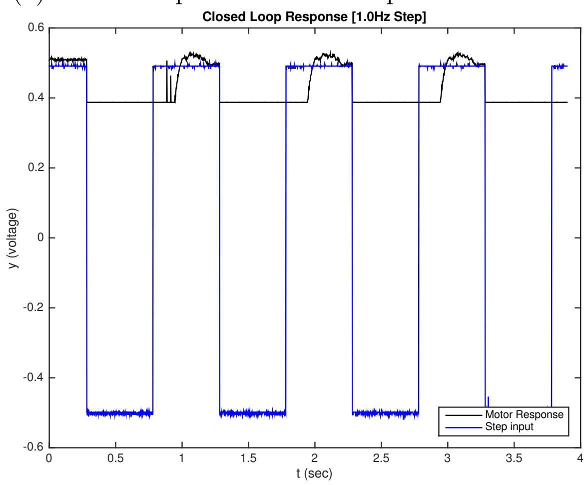
1. stuff
2. The overshoot was measured over five different input frequencies in order to determine the effect over varying the input signals frequency. The systems response to these signals was also recorded, and can be found as a set of graphical time domain plots below.



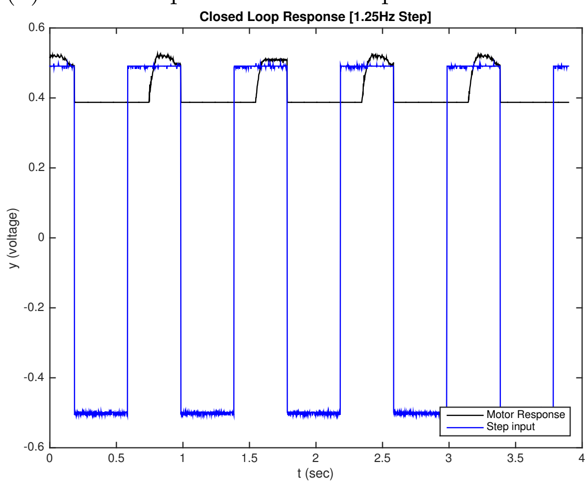
(a) Closed loop time domain response to 0.5Hz input.



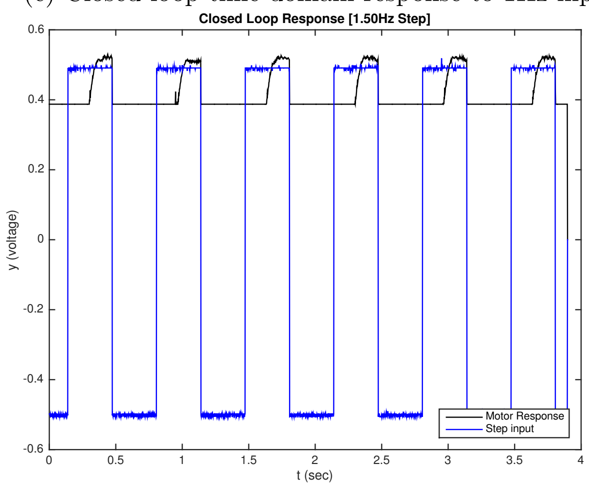
(b) Closed loop time domain response to 0.75Hz input.



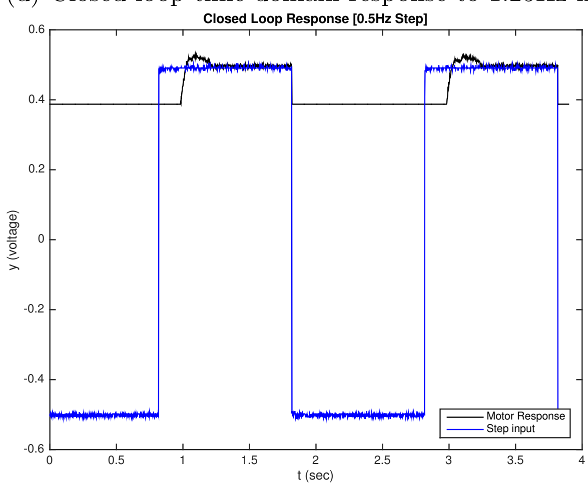
(c) Closed loop time domain response to 1Hz input.



(d) Closed loop time domain response to 1.25Hz input.



(e) Closed loop time domain response to 1.5Hz input.



(f) Closed loop time domain response to 1.5Hz input.

Figure 26: Systems response to various input frequencies.

The percentage overshoot was calculated using the cursors on the digital CRO machine to measure the peak overshoot and the steady state voltage (average).

$$\%OS = \frac{V_{peak} - V_{average}}{V_{average}}$$

$$\begin{aligned} OS(0.5Hz) &= \frac{525 - 495}{495} = 6.1 \\ OS(0.75Hz) &= \frac{523 - 495}{495} = 5.7 \\ OS(1Hz) &= \frac{525 - 495}{495} = 6.1 \\ OS(1.25Hz) &= \frac{525 - 495}{495} = 6.1 \\ OS(1.5Hz) &= \frac{524 - 495}{495} = 5.8 \end{aligned}$$

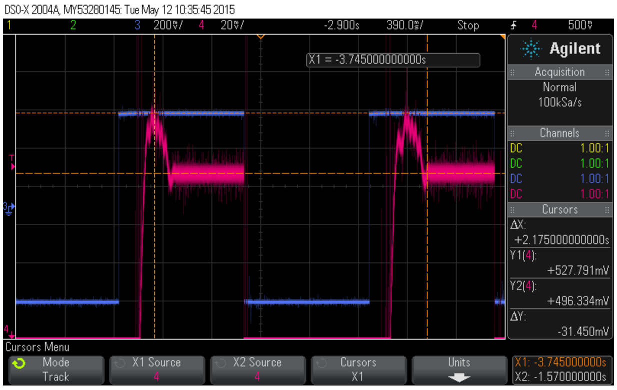
The results of this analysis can be found in the following table.

Input Frequency (Hz)	Overshoot (%)
0.5	6.1
0.75	5.7
1	6.1
1.25	6.5
1.5	5.8

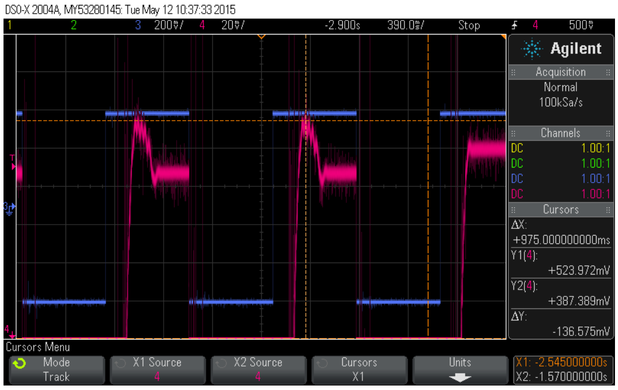
As can be seen almost immediately in the previous plots, as the frequency of the input signal is increased, the output has less time to reach a steady state value. Eventually, the increasing frequency limits the systems ability to reach a steady state value.

It is also evident from the previous figures and tables, that neither the overshoot, settling time or the steady state error are effected; until eventually the system does not have enough time to reach a steady state value. In addition to this, were the frequency to keep increasing, eventually the op-amps physical components would not be able to switch fast enough, and the output would be attenuated.

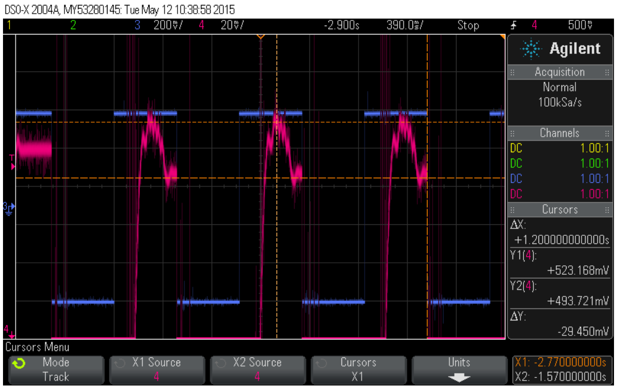
The frequency does not effect any of the aforementioned properties, because quite simply; the transfer function to which these properties can be derived, is independent of frequency. Which is to be expected, as there are no energy storing components in the system (capacitors, inductors).



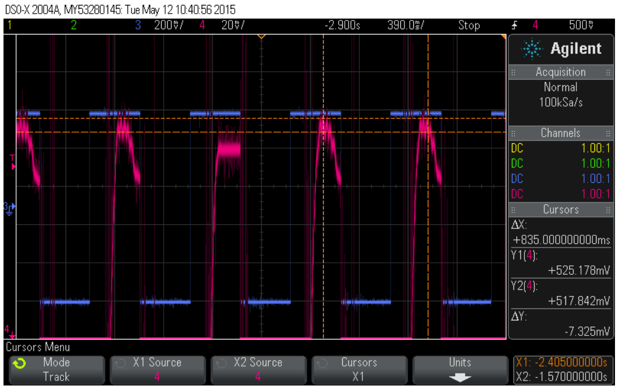
(a) Overshoot 0.5Hz



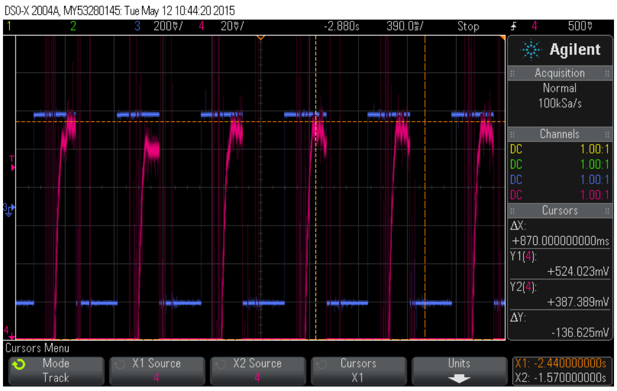
(b) Overshoot 0.75Hz



(c) Overshoot 1Hz

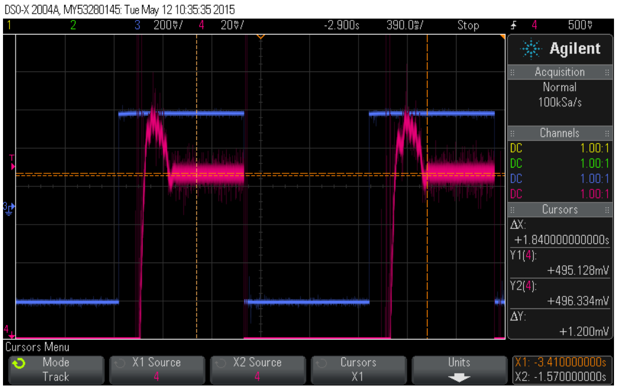


(d) Overshoot 1.25Hz

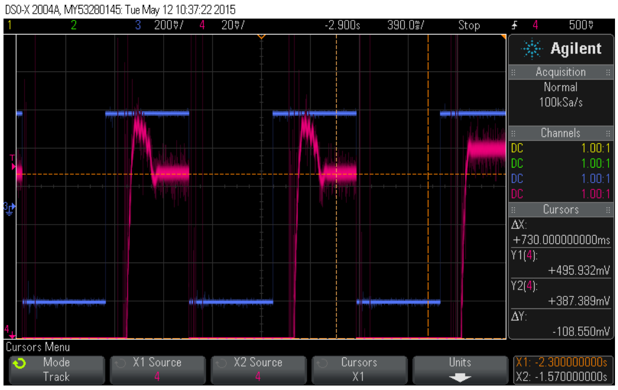


(e) Overshoot 1.5Hz

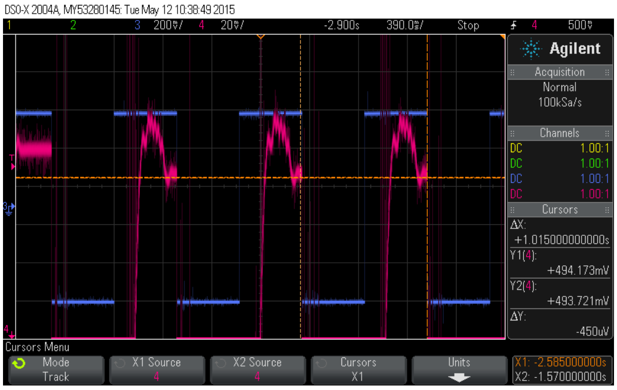
Figure 27: Ossilloscope SSE Measurements



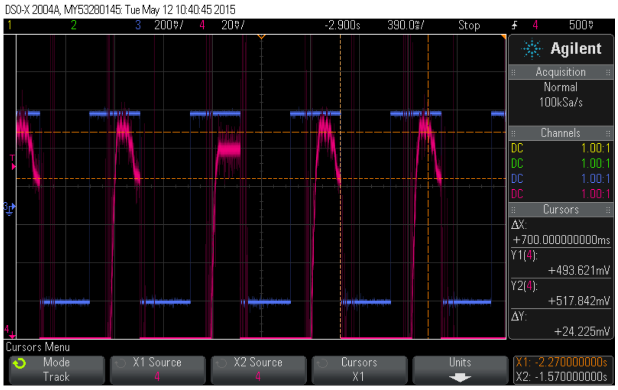
(a) SSE 0.5Hz



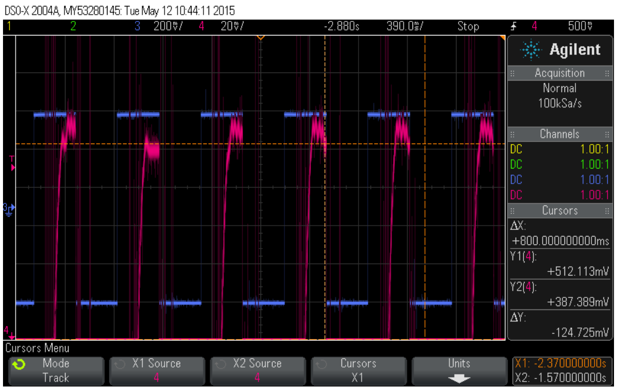
(b) SSE 0.75Hz



(c) SSE 1Hz



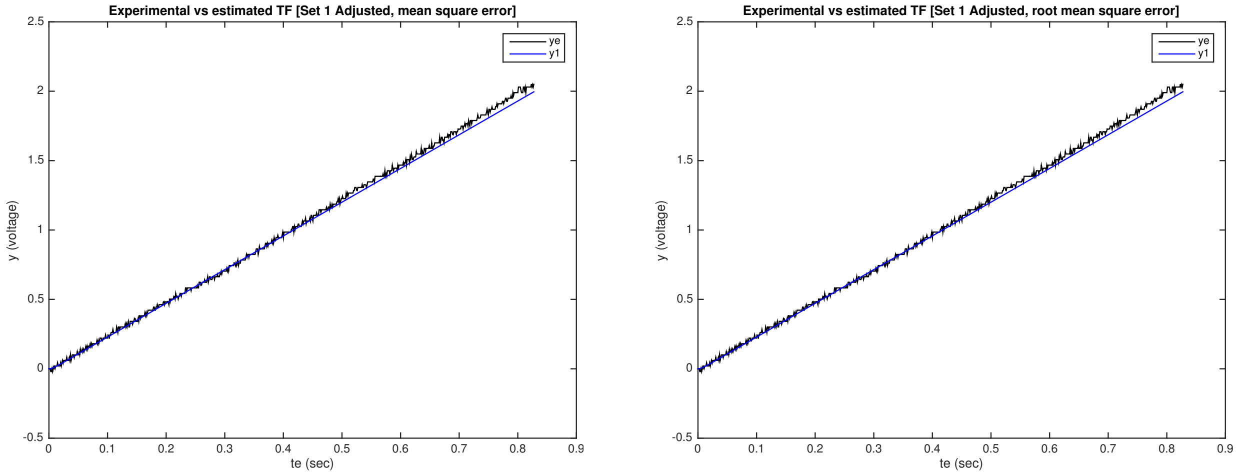
(d) SSE 1.25Hz



(e) SSE 1.5Hz

Figure 28: Ossilloscope SSE Measurements

3. To examine the impact of changing the gain value, the systems response data has been captured, recorded, and displayed below.



(a) Closed loop time domain response to 0.5Hz input. (b) Closed loop time domain response to 0.75Hz input.

Figure 29: Systems response to various gain values.

As you would expect, changing the gain value **increases OS?** and **decreases Ts?**. The above images show large evidence to support this comment; as the correlation between these values can be seen clearly. **WHY**
This is useful because **WHY**

4. **C4 - Matlab - Declan?** simulink

5. **C4 - Matlab - Declan?** PD/PI/PID controller

8 CHECKLIST

1. Executive Summary
2. Introduction
3. Procedure Intro
4. Procedure (D) - add new method
5. Results (B)
6. Results (C)
    (a) Is the transfer function as we'd expect?
    (b) Closed loop response figures (theoretical gain or given gain?)
    (c) What can be seen from the figures
    (d) Experimental gain figures
    (e) What can be seen from this
    (f) Compare models
    (g) Compare gains
    (h) **NEW METHOD for finding TF**
7. Results (D)
    (a) **DO NEW METHOD**
    (b) Check tables peak overshoot
    (c) Why is the effect of gain useful?
    (d) SIMULINK MODEL
    (e) PI/PD controller
8. **DISCUSSION and RECOMMENDATIONS**
9. Answers (B) - add more words?
10. Answers (C)
    (a) Fix figures
    (b) Add non-ideal method
    (c) C(4)
    (d) Finish C(7)
    (e) **NEW METHOD for finding TF**
11. Answers (D)
    (a) **DO NEW METHOD**
    (b) Does frequency affect anything?
    (c) Prove how gain affects shit
    (d) SIMULINK MODEL
    (e) PI/PD controller

NOTES: on first page, acknowledge assistance from students list group members / lab members