EEE 342 Lab 1 Preliminary Work

Emir Arda Bayer

Department of Electrical and Electronics Engineering, Bilkent University, 06800 Ankara, Turkey

1. Introduction

In the preliminary work students were asked to find a method to dictate the transfer function (first order) between input voltage and output angular velocity. Using the suggested method the transfer function for a DC motor will be revealed. The input and output will be compared and analyzed. Furthermore the topics of Proportional and Proportional Integral controllers will be discussed.

2. Laboratory Content

2.1. Question 1

Using Simulink, a low pass filter will be applied to the given data in order to remove noise, as in Fig. 1.

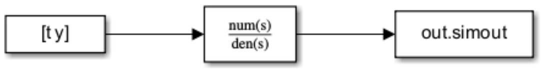


Figure 1: Simulink Block Diagram of LPF

The plots of data when subjected to the filter and when it was not are portrayed as in Figures 2 and 3.

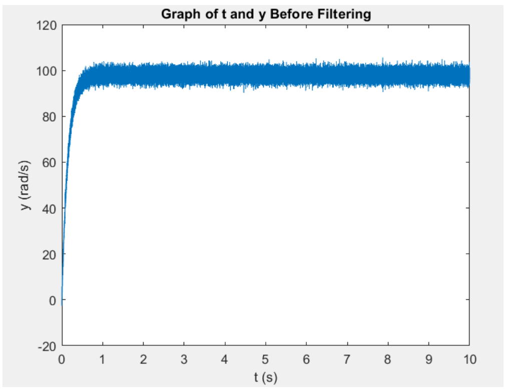


Figure 2: Before Filtering

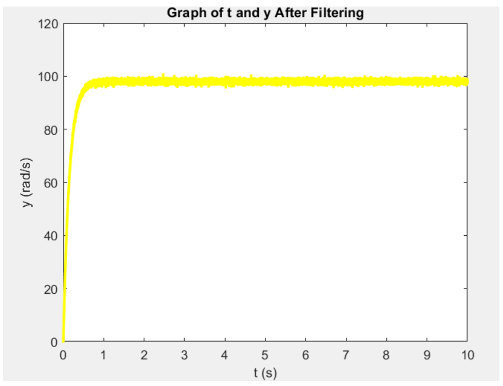


Figure 3: After Filtering (Decrease in Noise)

The general form of a first order transfer function n Laplace domain is as H(s) (as below), and the output r(t) can be expressed in the same domain as Y(s).

$$H(s) = \frac{K}{1+s\tau}$$

$$\mathrm{Y}(s) = \frac{K}{1+s\tau}\frac{6}{s}$$

To solve the Laplace transform:

$$\mathrm{Y}(s) = \frac{c_1}{1+s\tau} + \frac{c_2}{s}$$

$$6\mathrm{K} = c_1 s + c_2(1+s\tau)$$

So, c_2 is 6K whereas c_1 is -6Kτ.

$$Y(s) = \frac{-6K\tau}{1+s\tau} + \frac{6K}{s}$$

$$y(t) = 6K(1 - e^{\frac{-t}{\tau}})u(t)$$

As τ goes to infinity, y(t) equals 6K, meaning examining the array y(t) in MATLAB in the interval of 2 to 10 seconds will reveal K. The found value is as K=16.333.

Later, in order to find the time constant another instance in the y(t) array is selected, being the 388'th instance. The value for y(t) at that point is 0.0387.

$$22.06 = 6 * 16.333 * (1 - e^{\frac{-0.0387}{\tau}})$$

Meaning the time constant is 0.15174. This implies that the transfer function can be written as:

$$Y(s) = \frac{16.333}{1+0.15s} = \frac{98}{6+0.9s}$$

The output of the system can be portrayed as in Fig. 4.

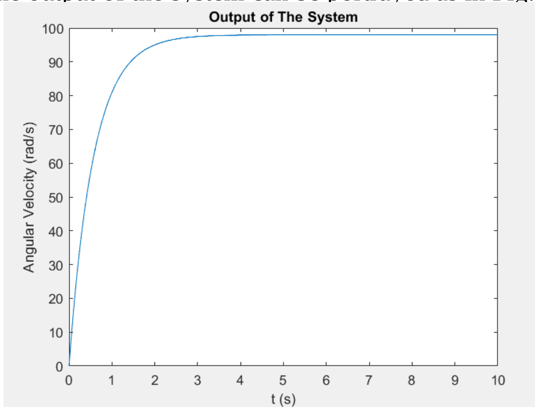


Figure 4: System Output

This implies that the transfer function which is in first order matches the plot in Figure 4. The given requirement between poles of the transfer function and the LPF below is also met.

$$p_{TF} < \frac{p_{LPF}}{10}$$

$$6.666 = \frac{1}{0.15} < \frac{1}{10 * 0.001} = 100$$

2.2. Question 2

The outputs for a proportional controller and a proportional integral controller are as:

$$u = K_p e \qquad \text{(P)}$$

$$u = K_I \int edt + K_p e \qquad \text{(PI)}$$

Where K_p and K_I are proportional gains. P controllers are reasons for overshooting, whereas PI are reducers. The integral in the output of PI controllers imply better removal of steady state errors meaning they reduce settling time, demonstrating that PI controllers are more powerful in eliminating steady state errors.

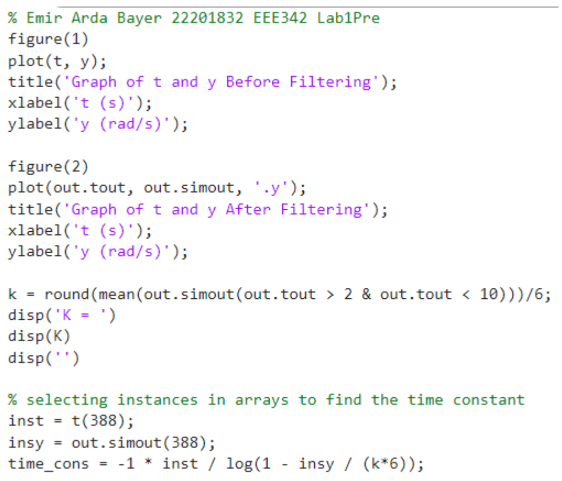
3. Conclusion

In order to dictate the approximate transfer function, in this case between angular velocity and input voltage, a method was found, the system's response was analyzed using Laplace domain transfer function. Additionally, proportional and proportional integral controllers were compared and their effects on response were studied.

REFERENCES

1. "Proportional Integral (PI) Control, Dynamics and Control" https://apmonitor.com/pdc/index.php/Main/ProportionalIntegralControl (accessed 14.02.2025).

MATLAB CODE

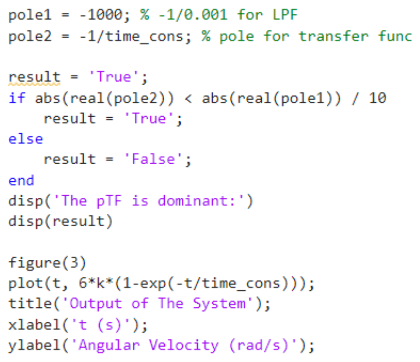


```
% Emir Arda Bayer 22201832 EEE342 Lab1Pre
figure(1)
plot(t, y);
title('Graph of t and y Before Filtering');
xlabel('t (s)');
ylabel('y (rad/s)');

figure(2)
plot(out.tout, out.simout, '.y');
title('Graph of t and y After Filtering');
xlabel('t (s)');
ylabel('y (rad/s)');

k = round(mean(out.simout(out.tout > 2 & out.tout < 10)))/6;
disp('K = ')
disp(K)
disp('')

% selecting instances in arrays to find the time constant
inst = t(388);
insy = out.simout(388);
time_cons = -1 * inst / log(1 - insy / (k*6));
```



```
pole1 = -1000; % -1/0.001 for LPF
pole2 = -1/time_cons; % pole for transfer func

result = 'True';
if abs(real(pole2)) < abs(real(pole1)) / 10
    result = 'True';
else
    result = 'False';
end
disp('The pTF is dominant:')
disp(result)

figure(3)
plot(t, 6*k*(1-exp(-t/time_cons)));
title('Output of The System');
xlabel('t (s)');
ylabel('Angular Velocity (rad/s)');
```