Lab 1 Report

Introduction to LTI Systems in Matlab and Simulink

ECE311

Lab Group: 17

Name of Students: Alexander Mark, Kyu Bum Kim, Jack Khoueiri

Date: October 6, 2021

Output 1

Given the motor parameter in the lab, we constructed the transfer functions of both the simplified and full motor model in Matlab:

```
G_motor =

          833.3
  --------------------
  s^2 + 150.2 s + 33.33

Continuous-time transfer function.


G_motor_simplified =

    5.556
  ----------
  s + 0.2222

Continuous-time transfer function.
```

Figure 1: Transfer functions for G_motor and G_motor_simplified

We then created the zpk form of the original transfer function for G_motor to identify the poles in the motor model's transfer function:

```
zpk_motor =

          833.33
  --------------------
  (s+0.2223) (s+149.9)

Continuous-time zero/pole/gain model.


poles =

   -0.2223
 -149.9444
```

Figure 2: ZPK form of G_motor and its poles

We observe that the poles are located at -0.2223 and -149.9444. They sit on the negative s-axis and are far apart from each other. The magnitude of the -0.2 pole is a lot larger than the -149.9 pole so the shape of the output will be mostly dictated by the -0.2 pole. Since both poles are on the negative s-axis and the input is bounded (unit step), the motor output should converge to a value rather than exponentially increase, meaning the system is stable. The shape of the output should look like an exponential decay reflected on the t-axis as the poles are negative real numbers and the rate of convergence to steady state would be gradual because -0.2 is not very far into the negative s-axis.

We then printed the numerator and denominator arrays for both motor models' transfer functions to be used in the next section:

```
num =

         0         0  833.3333


den =

    1.0000  150.1667   33.3333


num1 =

         0    5.5556


den1 =

    1.0000    0.2222
```

Figure 3: Numerator and Denominator arrays of G_motor and G_motor_simplified respectively

Output 2

In *Figure 4*, we can see the step response of the motor step and its simplified version. The last plot of *Figure 4* represents the difference between the two plots.

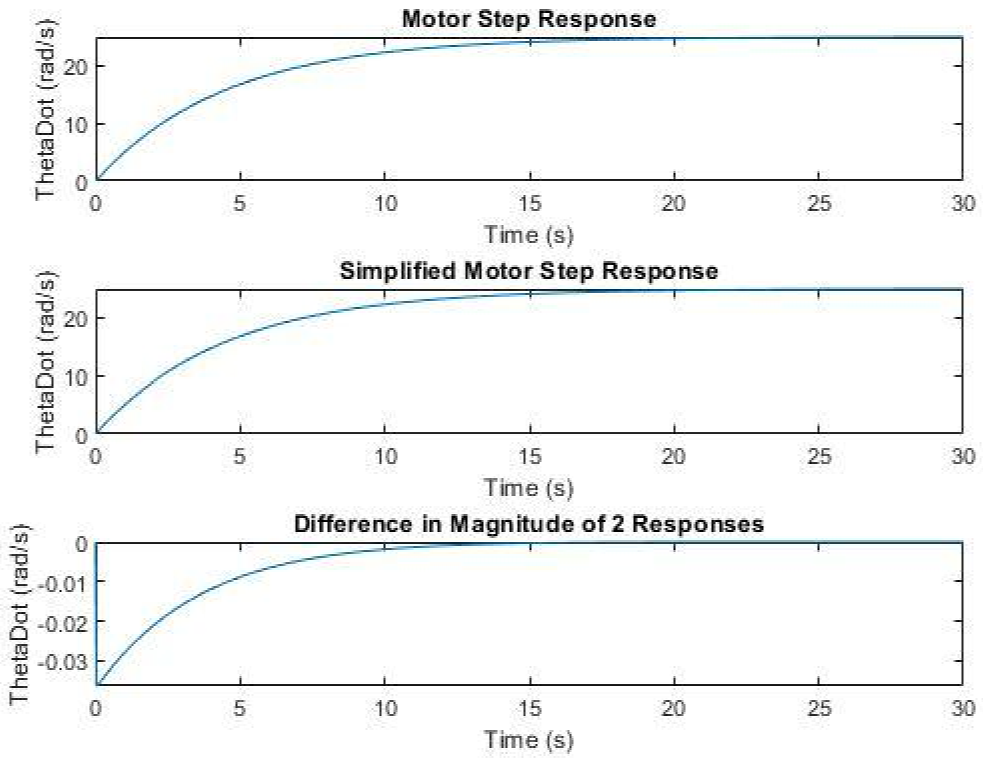

Figure 4: Motor Step Response of Original and Simplified, and their Difference

We discover that the maximum error magnitude of the simplified version is approximately 0.04 rad/s at the beginning of the step response, shown at t = 0 on the last plot of *Figure 4.* As it approaches the steady state of the step function, the error asymptotically approaches 0. Therefore, we can say that the simplified version is a good approximation of the original step response as the error is small and soon becomes negligible.

```
approxAsympMotorSpeed =

   24.9682


theoreticalAsympMotorSpeed =

   25.0000
```

Figure 5: Approximate and Theoretical Asymptotic Motor Speed

As shown in *Figure 5*, the approximate asymptotic motor speed as it approaches steady state from the step response is 24.9682 rads/s. Meanwhile the theoretical motor speed is 25 rads/s. We can conclude that the two are very close in value; a 0.13% margin of error.

```
approximateAmpOfOscillation =

    5.4326


theoreticalAmpOfOscillation =

    5.4251
```

Figure 6: Approximate and Theoretical Amplitude of Oscillation of the System's Response to a Sinusoidal Input

The approximate amplitude of oscillation in steady state of the motor speed in response to a sinusoidal input is around 5.4326. The theoretical value is 5.4251. We can note that the values are very close in value and almost negligible; 0.14% error. Both of these instances show that the simplified motor is an accurate representation of the real motor.

Output 3

Using Simulink, we analysed the output for both the full motor and simplified motor TF models using 2 input signals: the unit step and the sine wave.

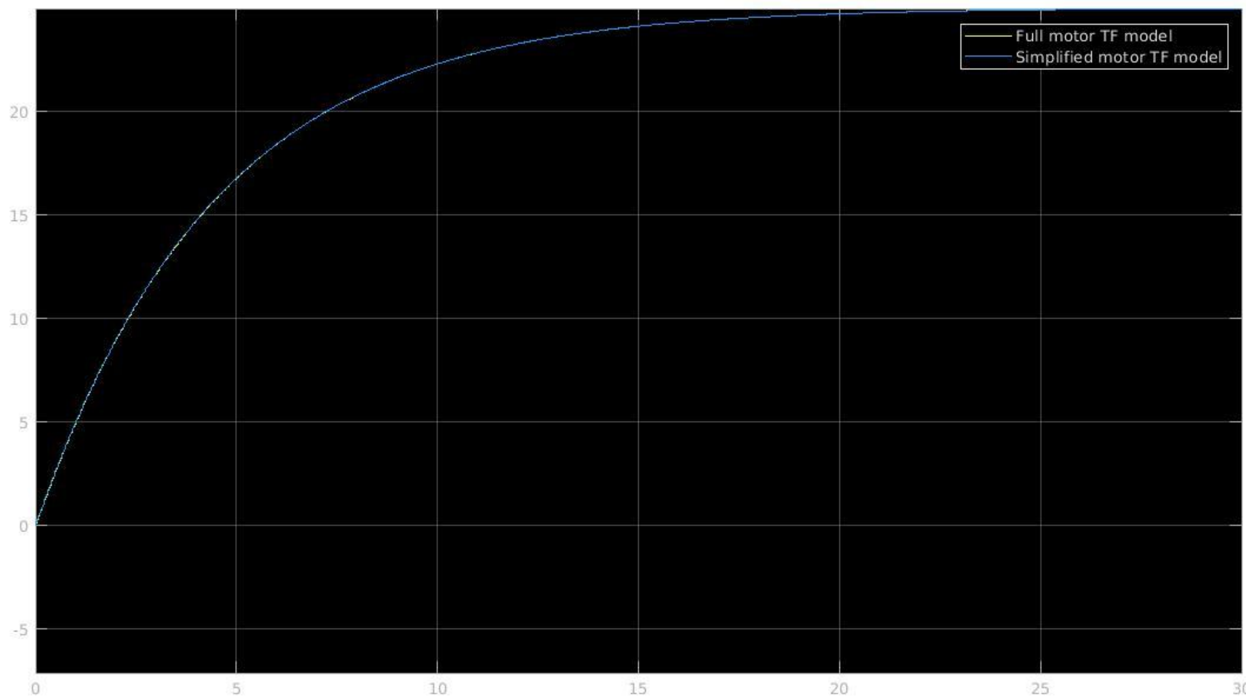

Figure 7: DC Motor Output with Unit Step Input

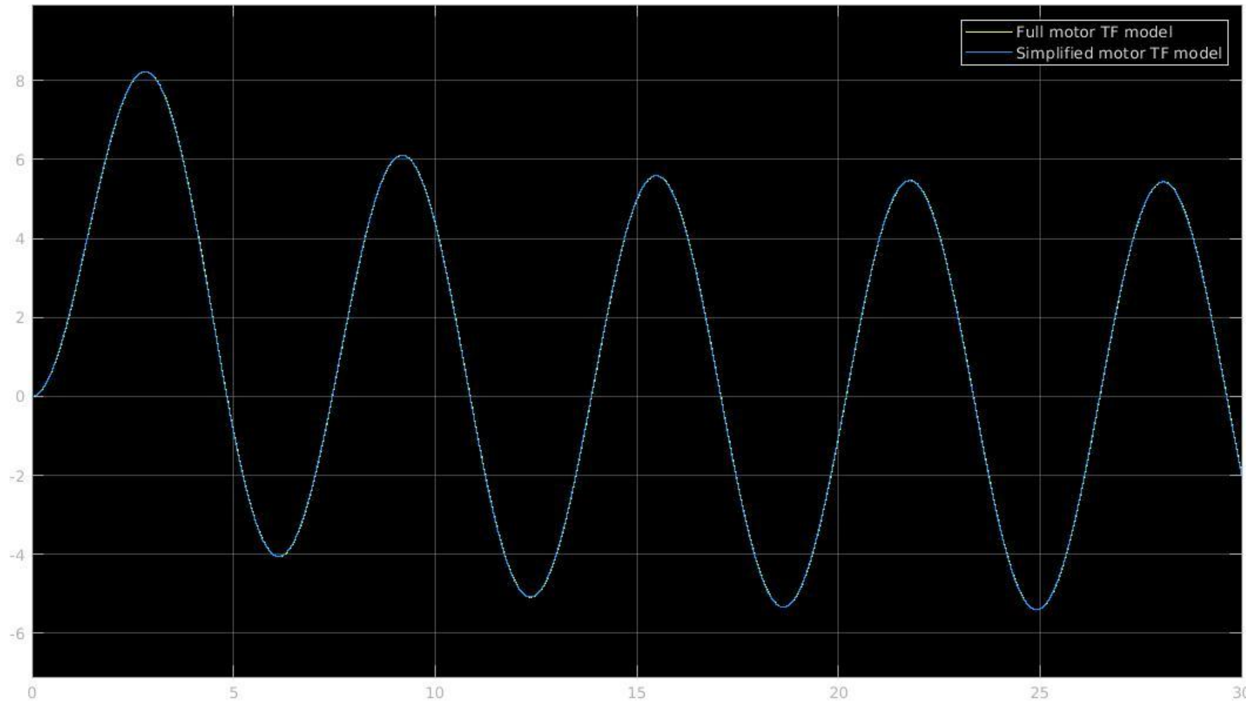

Figure 8: DC Motor Output with sin(t) Input

From the two plots, we can conclude that the simplified transfer function model approximated the full transfer function model very well as the lines are practically the same on both graphs.

Output 4

Below are the 2 figures that were generated from the proportional controller for K = 0.1 and 1. They contain the graph for the reference signal and the motor speed.

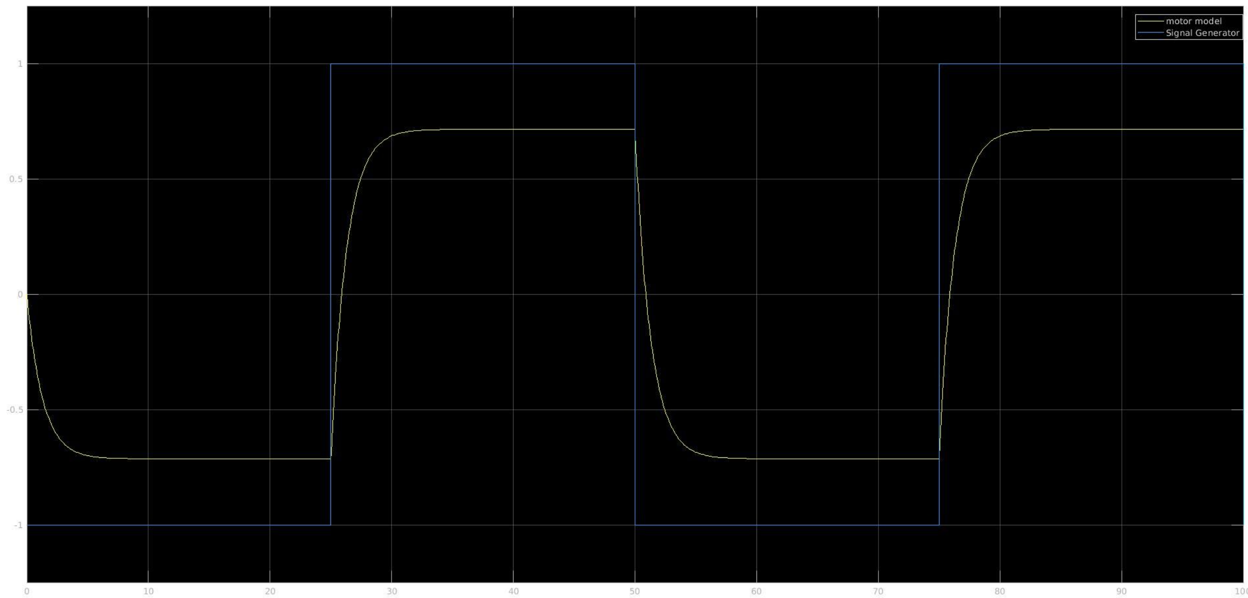

Figure 9: DC Motor Proportional Controller Output with K = 0.1

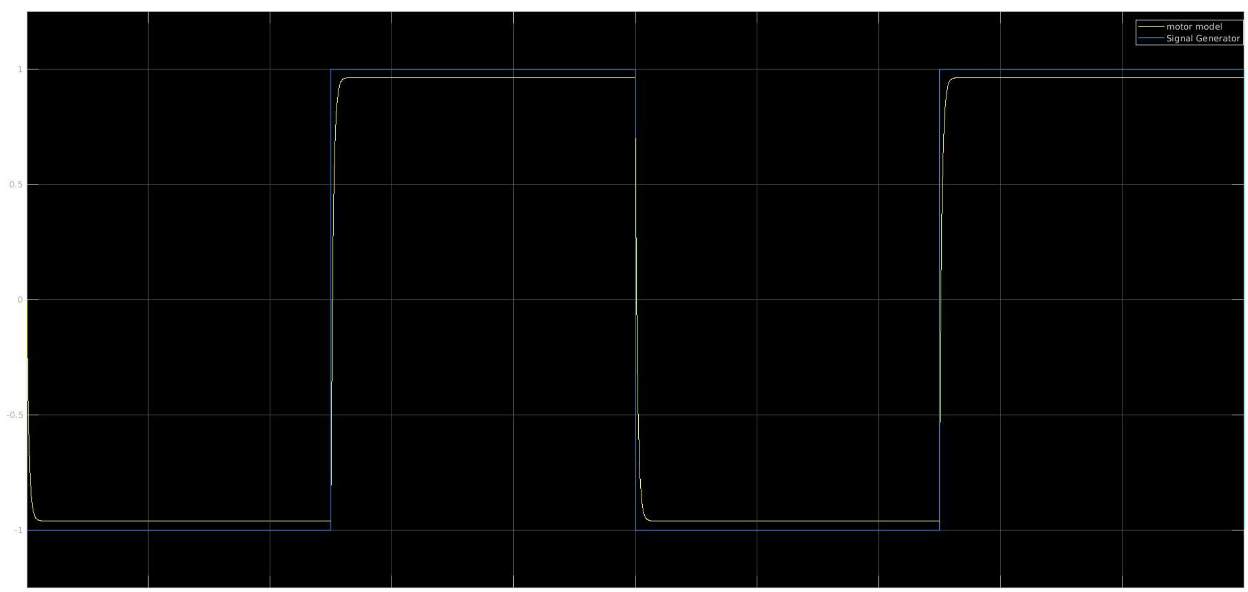

Figure 10: DC Motor Proportional Controller Output with K = 1

From the graphs, the steady state tracking error when the gain (K) was 0.1, is 0.2857 and the error was 0.03846 when the gain was 1. We observed that as K increases, the error decreases and the rate of convergence increases. This is because the controller is making more drastic corrections with a higher gain to get to the target speed. Since the input to the motor is proportional to K and the error, if we increase K, smaller error values will have a greater effect on the rate the motor changes speed and its actual output speed. We see this in the lower error value and higher rate of convergence to steady state when K = 1.

We do think that the proportional controller is sufficient to regulate the speed of the motor as the error is very close to 0 and the speed is near the target at steady state. In terms of improvements, the controller can look at the history of errors and make changes based off this information (integral controller) or use the difference between the last calculated error and current error to make adjustments (derivative controller). This can make it even more accurate at achieving the target speed.