Temple University

College of Engineering

Department of Electrical and Computer Engineering (ECE)

Student Report Cover Page

Course Number: ECE 3412

Lab 4: Experimental approach to DC motor system identification from step response

Student Name: Devin Trejo; Robert Irwin
TUid: 914924557; 914980083
Due Date: 2/24/2015

TA Name: Michael Korostelev

Grade: / 100

I. Introduction

In this experiment we will be introduced to open loop motor control. The point of this experiment is to use the tools at our disposal, i.e. MatLab and Simulink, to obtain the transfer function of the motors velocity using the data obtained experimentally.

This lab also serves as an introduction to the extremely useful system identification toolbox.

II. Procedure

To begin we first build a DC motor step response model into Simulink. The input is a step and our output is the position and speed read in by the encoder. The model is seen below:

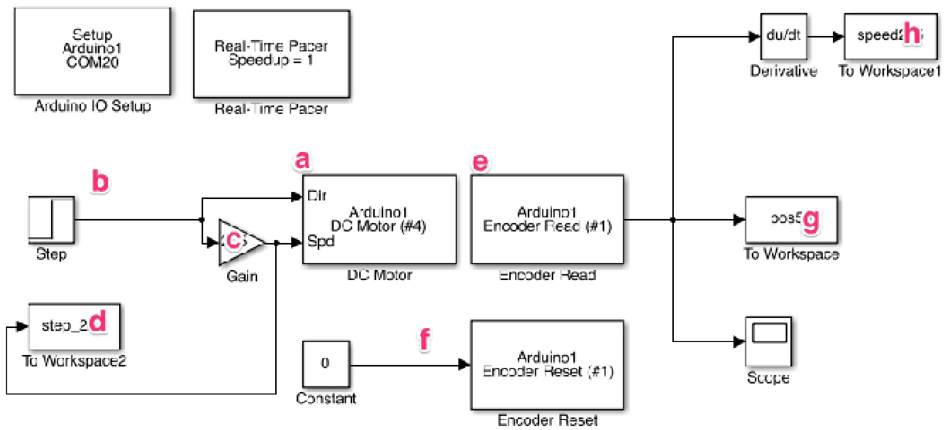

Figure 1: Simulink model for control and reading in data from the sensor

After we pass our step input into the model we can start reading in data from the sensor. Since the DC motor takes in to inputs we have a gain block that gives us a predetermined speed. For an analysis we export our position data to our workspace. Outputting to the workspace will allow us to plot our position with respect to time. We also take the derivative of our position data so that we have the speed of the motor inside our workspace. The setup of the hardware is the same as previous experiment.

For the experiment we range our step response from 50, 150 to 255. To analyze the data we plot our speed and position with respect to each one of our inputs. From these plots we can find a first order transfer function. The transfer function can be found by hand using an estimate of the DC gain (or steady state value $H(0)$), and the time constant (τ). From these two values we can find the first order transfer function:

$$H(s) = \frac{k}{s+a} \rightarrow \frac{k}{s+\frac{1}{\tau}}$$

$$H(0) = \frac{k}{a}$$

Now we introduce a new toolbox in MatLab the system identification toolbox. The toolbox makes MatLab find the transfer function automatically from a series of data. Using our speed data we can compare our hand calculated transfer function to a transfer function produced by MatLab. Finding the second order transfer function by hand would be more involved but the system identification toolbox allows us to construct the transfer function at the push of the button. Therefore we compare our first order transfer function to our first order.

III. Results

Below is the step response for the inputs of 50, 150 and 255. It is important to note these are the speeds that the motor will rotate.

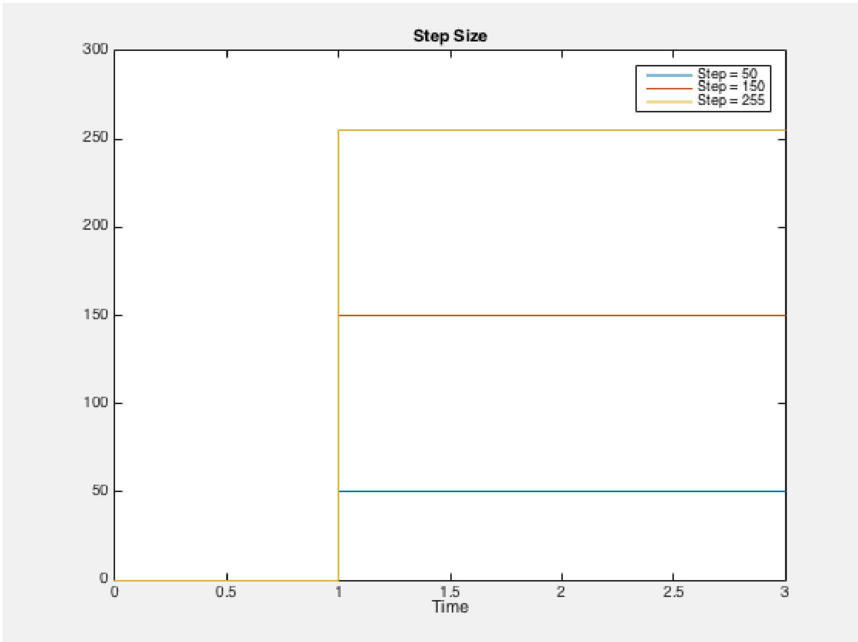

Figure 2: Plot of the Different Inputs

As we can see, the inputs are exactly what we are after so we can be sure that our results will be accurate.

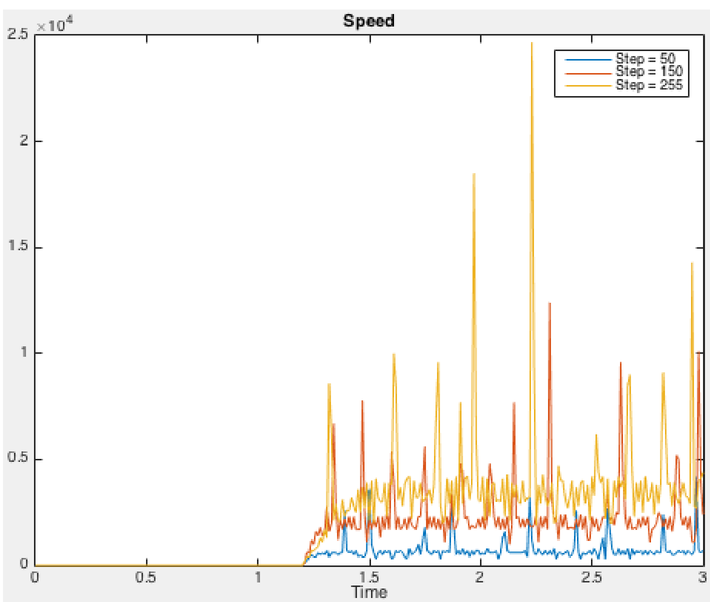

Figure 3: Speed of the Motor with Respect the Input Size

From this plot we can see that with smaller input sizes to the system, the motor moves more slowly. We also notice a large amount of noise in these responses.

We can also see that there is a transient response and a steady state response. The larger the input, the longer it takes for the motor to reach full speed. In the steady state region, we notice that a larger input corresponds to a much noisier signal.

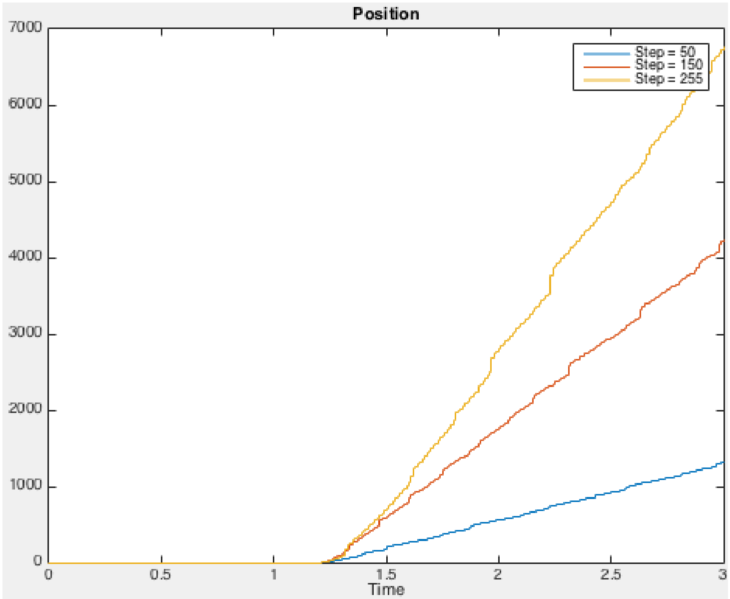

Figure 4: Position Plot

The plot above shows the position of the motor with respect to the size of the input. As stated previously a larger input means more velocity. As each input was applied to the motor for three seconds, it makes sense that the motor rotated the furthest for the largest input.

Now we will show the results of the estimated first order transfer function from the system identification toolbox.

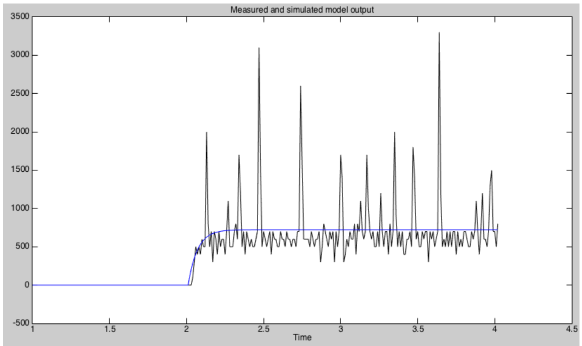

Figure 5: Step Size 50 First Order

```
tf1 =

  From input "u1" to output "y1":
    241.3
  ---------
  s + 16.68
```

Figure 6: Transfer Function for Step Size of 50

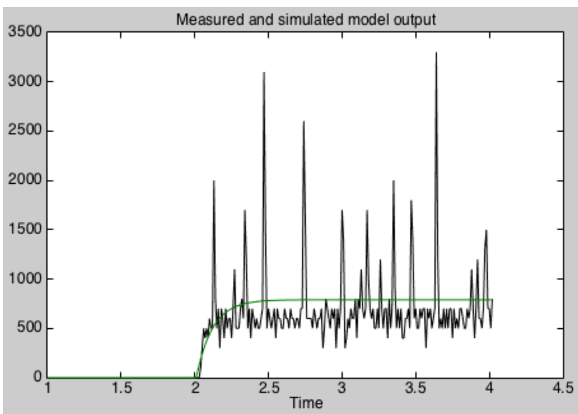

Figure 7: Step Size 150 First Order

```
tf2 =

  From input "u1" to output "y1":
   143.3
  --------
  s + 9.07
```

Figure 8: Transfer Function from MatLab Step Size 150

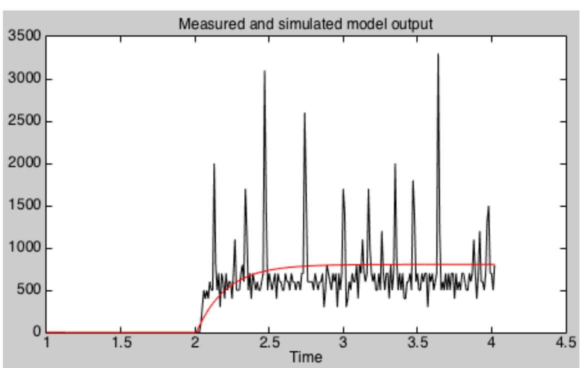

Figure 9: Model Output Step Size 255

```
tf3 =

  From input "u1" to output "y1":
    80.69
  ---------
  s + 5.012
```

Figure 10: Transfer Function Step Size 255

In the photos above it is important to note the first order characteristic of the model output. The transfer function agrees with the model output in the sense that it only has one pole. We know this by the fact that the denominator is first order.

Below are the plots that resulted in our estimation of tau and $H(0)$.

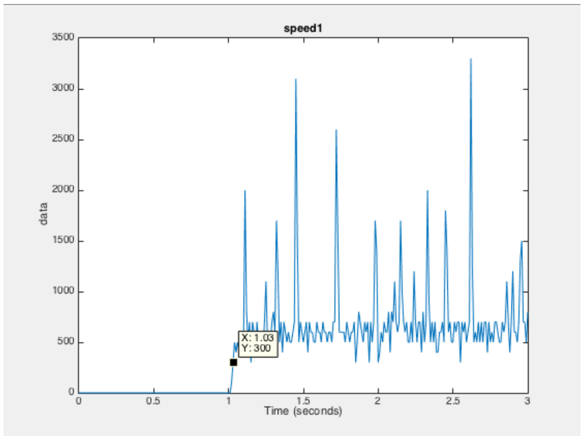

Figure 11: Step Size 50

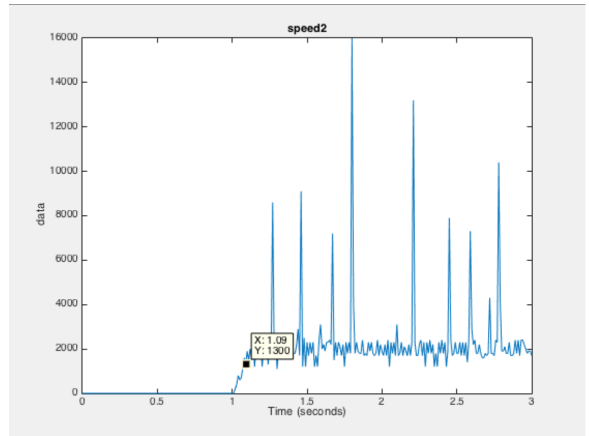

Figure 12: Step Size 150

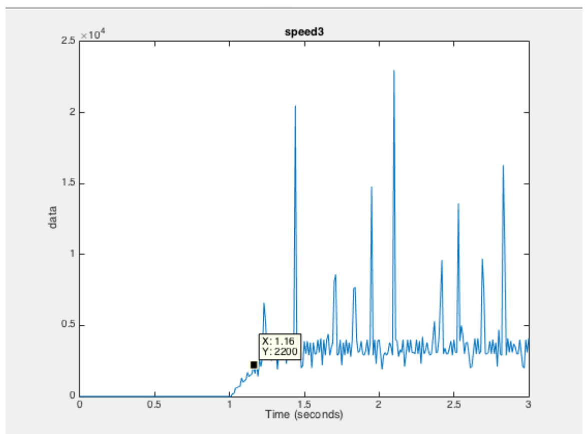

Figure 13: Step Size 255

In the plots above, we estimate the steady state value of the response. We take 63% of that value and locate the time, x, that corresponds with 63% of the steady state approximation. A table that compares the MatLab transfer function and the estimated transfer function are shown below:

$$1st\ Order\ Form: H(s) = \frac{k}{s+a} \rightarrow \frac{k}{s+\frac{1}{\tau}}$$

Table 1: Comparison of MatLab Results and Analytical Estimations (First Order)

Step Size	K (MatLab)	A (MatLab)	Tau (MatLab)	Output	63% Output	K (Hand)	A (Hand)	Tau (Hand)	K % Difference	Tau % Difference
5.00E+01	*2.41E+02*	*1.67E+01*	*6.00E-02*	*6.00E+02*	*3.78E+02*	*1.20E+01*	*2.00E+01*	*5.00E-02*	*95.03%*	*16.60%*
1.50E+02	*1.43E+02*	*9.07E+00*	*1.10E-01*	*2.00E+03*	*1.26E+03*	*1.33E+01*	*1.11E+01*	*9.00E-02*	*90.70%*	*18.37%*
2.55E+02	*8.07E+01*	*5.01E+00*	*2.00E-01*	*3.50E+03*	*2.21E+03*	*1.37E+01*	*6.25E+00*	*1.60E-01*	*82.99%*	*19.81%*

$$2nd\ Order\ Form: H(s) = \frac{k}{s^2+as+b}$$

Table 2: Comparison of MatLab Results and Analytical Estimations (Second Order)

Step Size	K (MatLab)	B (MatLab)	A (MatLab)
5.00E+01	*1.24E+04*	*4.34E+01*	*9.47E+02*
1.50E+02	*6.57E+03*	*4.43E+01*	*4.17E+02*
2.55E+02	*1.54E+03*	*1.76E+01*	*9.71E+01*

Now we will use the system identification toolbox to estimate the second order transfer function for each input. The model outputs, and the transfer functions are shown below.

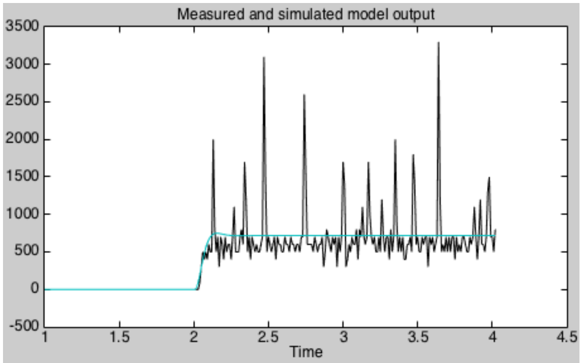

Figure 14: Second Order Transfer Function Step Size 50

```
tf4 =

  From input "u1" to output "y1":
         1.358e04
  ---------------------
  s^2 + 43.36 s + 946.5
```

Figure 15: Second Order Transfer Function Step 50

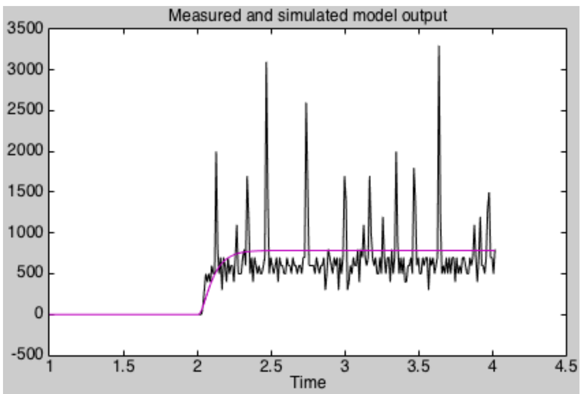

Figure 16: Second Order Transfer Function Step Size 150

```
tf5 =

  From input "u1" to output "y1":
           6568
  --------------------
  s^2 + 44.3 s + 417.4
```

Figure 17: Second Order Transfer Function Step 150

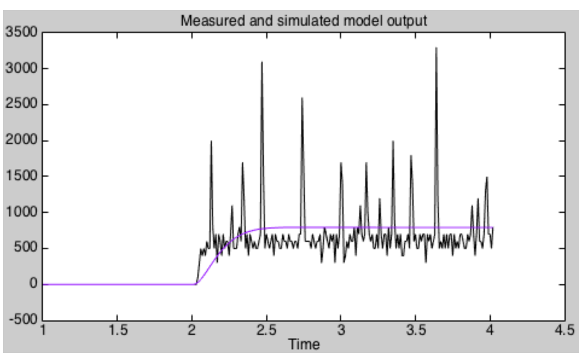

Figure 18: Second Order Transfer Function Step Size 255

```
tf6 =

  From input "u1" to output "y1":
           1540
  ---------------------
  s^2 + 17.61 s + 97.07
```

Figure 19: Second Order Transfer Function Step 255

It is important to note that the second order transfer function better capture the behavior of the transient response compared to the first order transfer function.

IV. Conclusion/Discussion

As we conclude the lab we found that we can find a transfer function for a DC motor. Referencing Table 1 we saw that as we increased our step size input our rise time decreased. The motor has faster accelerations as we increased our step size. We also saw that as the input increased our top speed also increased. When we compared an eyeball estimate to the MatLab results we found little changes in percent difference for the rise time. Our k values however varied drastically since we estimated the steady state output. Our output has lots of noise so it is not very clear on what the output truly is. The value MatLab interpreted to be steady state varied from what we took to be the steady state value. The noise occurs since we take the derivate of the position and when taking the derivate is amplifies the noise. Therefore, any jump in position correlates to large spikes in the speed.

By the end however, we found that MatLab makes good estimates on creating a representative transfer function for our DC motor. Using this Simulink model we program motor characteristics from a spec sheet into MatLab and produce a representative transfer function. Furthermore, we can use the transfer function is more complex systems that lets say represent the motor in a forklift. In the fork lift example, we can have the speed of the motor dependent on the weight of the load. Depending on the situation you may want drive your DC motor with a step size of 50 or 255.