

## GE420 Laboratory Assignment 6

### DC Motor Discrete Transfer Function Identification

#### Goals for this Lab Assignment:

Use a least-squares solution to identify the discrete open-loop transfer function of the DC motor with added flywheel

#### DSP/BIOS Objects Used:

Clock

#### Daughter Card Library Functions Used:

readEnc1, setEPWM3A, setDAC1

#### Matlab Functions Used:

GE420\_serialread, GE420\_serialwrite

#### Prelab:

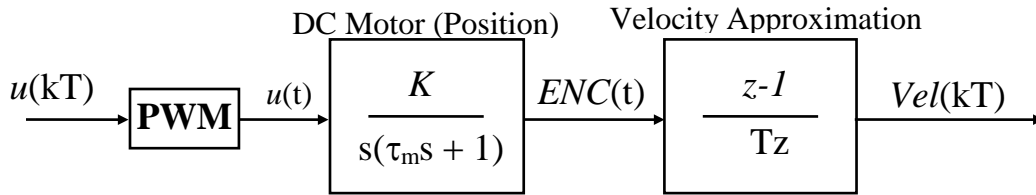
In MATLAB, solve the following over-determined set of linear equations for a,b,c,d using a least squares solution.

$$\begin{array}{rcccccccc} 12 & + & a*32 & = & b*45 & + & c*12 & + & d*2 \\ 44 & + & a*100 & = & b*13 & + & c*17 & + & d*21 \\ 2 & + & a*16 & = & b*19 & + & c*11 & + & d*43 \\ 32 & + & a*111 & = & b*112 & + & c*33 & + & d*23 \\ 112 & + & a*82 & = & b*54 & + & c*12 & + & d \\ 84 & + & a*29 & = & b*101 & + & c*88 & + & d*73 \\ 9 & + & a*24 & = & b*92 & + & c*72 & + & d*81 \\ 14 & + & a*234 & = & b*87 & + & c*37 & + & d*63 \\ 35 & + & a*11 & = & b*39 & + & c*19 & + & d*53 \end{array}$$

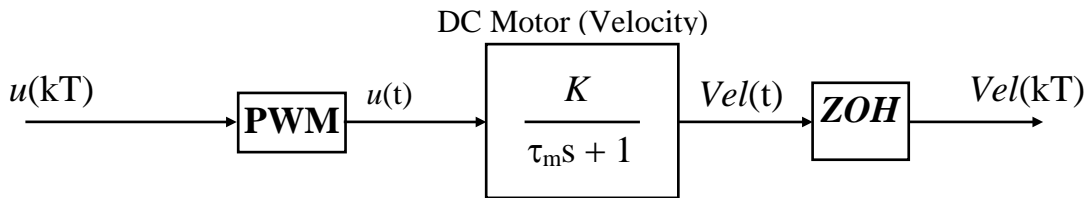
Type “help slash” to see how to solve an over-determined set of equations in MATLAB. Show that the answer is  $a = -0.1111$ ,  $b = 0.2603$ ,  $c = 0.8810$ , and  $d = -0.6174$  by printing the commands and their output in MATLAB. Note, these numbers and equations were just pulled out of the air as an exercise to familiarize you with a least-squares solution.

#### Laboratory Exercise

In this lab we are going to introduce you to the process of system identification, that is, to the task of identifying the *parameters* of an unknown plant of assumed dynamical structure. We will use the DC motor with attached flywheel as our plant to identify. Below is the open loop block diagram of the system. Figure 1 shows the full block diagram of the system. The DC motor only has angular feedback (optical encoder) so you will approximate its velocity with the backwards difference rule. Figure 2 shows a simplified block diagram of the system. Here we are assuming that our velocity approximation is exact and cancels the integrator in Figure 1’s DC motor position transfer function. This will help simplify the identification but still allow you to identify  $K$  and  $\tau_m$ . In general, the task of identification consists of applying carefully chosen inputs and measuring the plant output(s). In our case, the model is very simple and we will find that a step input works very well.



**Figure 1: Full Open-Loop Block Diagram**



**Figure 2: Simplified Open-Loop Block Diagram**

Using Figure 2 as a guide, the discrete transfer function can be derived by assuming a *zero-order hold* (ZOH) and taking the z-transform of the continuous system:

$$\frac{Vel(z)}{U(z)} = (1 - z^{-1}) \mathcal{Z} \left\{ \frac{1}{s} \frac{Vel(s)}{U(s)} \right\} \quad (6.1)$$

Then

$$\frac{Vel(z)}{U(z)} = \frac{K[1 - e^{-T/\tau_m}]}{z - e^{-T/\tau_m}}, \quad (6.2)$$

and corresponding difference equation is

$$Vel(kT) = c_1 \cdot Vel(kT - T) + c_2 \cdot u(kT - T) \quad (6.3)$$

where:

$$c_1 = e^{-T/\tau_m}, \quad c_2 = K[1 - e^{-T/\tau_m}]. \quad (6.4)$$

Our goal then is to identify the two model parameters,  $c_1$  and  $c_2$ . To do this, you will apply an open-loop step input to the DC motor and store the motor's velocity response to an array. In MATLAB, you will use this response data with equation 6.3 to form an over-determined set of equations that are a function of  $c_1$  and  $c_2$ . Using least-squares regression, you will then solve for  $c_1$  and  $c_2$ .

**Procedure:**

1. Create a new DSP application that outputs a constant step value to the DC motor. This application should also sample the optical encoder of the DC motor and use the radian values to estimate the speed of the DC motor in units of radians/second. For the first 2 seconds of your run, store both the output value and velocity value in separate arrays. The size of these arrays will need to be changed as you change sample rates for the different identification runs. Note: The Matlab function ‘GE420\_serialread’ and the DSP given support software, only allows 1000 floating -point values to be uploaded in a single call to GE420\_serialread. For that reason you will need two 1000 element arrays to store all the data for a 2 second run sampling at 1ms. Your TA will explain more about this issue.

Recall how to declare variables and arrays you want to have access to in Matlab, e.g.:

```
#pragma DATA_SECTION(amp, ".my_vars")
float amp = 1;

#pragma DATA_SECTION(testarray, ".my_arrs")
float testarray[100];
```

2. In MATLAB, use the function **GE420\_serialread** (remember ‘help GE420\_serialread’) to up load your data to MATLAB’s workspace.
3. Plot the motor’s response. From the plot find the value for K (remember you are applying a step of 5). Also using the 63% rule, find an approximate value for  $\tau_m$ . Record these values in the table below.
4. Using equation 6.3, starting with k = 2 and ending with k = 2000, compile the data into 1999 equations that are a function of  $c_1$  and  $c_2$ . i.e.:

$$\begin{aligned}
 y(2) &= c_1 \cdot y(1) + c_2 \cdot u(1) \\
 y(3) &= c_1 \cdot y(2) + c_2 \cdot u(2) \\
 &\vdots \\
 y(n) &= c_1 \cdot y(n-1) + c_2 \cdot u(n-1)
 \end{aligned}$$

Solve for  $c_1$  and  $c_2$  by least-squares regression. Solve for K and  $\tau_m$  using the equations for  $c_1$  and  $c_2$ . Record these values in the table below with 6 decimal places of precision for  $c_1$  and  $c_2$ .

5. Repeat these steps for each of the 3 settings given in the table, but modify the length of your dataset so that approximately 2 seconds are taken regardless of the sample period.

U	Ts	c1	c2	$\tau_m$ calculated from c1 and c2	K calculated from c1 and c2	$\tau_m$ From Plot	K From Plot
<b>5 Unit Step</b>	1 ms						
<b>5 Unit Step</b>	5 ms						
<b>5 Unit Step</b>	15 ms						

--	--	--	--	--	--	--	--

**Lab Check Off:**

1. Complete the table above. Do K and  $\tau_m$  agree with the different identification methods (and with your expectations)?
  
2. Now assume that the motor transfer function is third-order and has the form:

$$\frac{Y(z)}{U(z)} = \frac{a_1 \cdot z^2 + a_2 \cdot z + a_3}{z^3 + a_4 \cdot z^2 + a_5 \cdot z + a_6}, \quad (6.2)$$

Pick your data run that used a 5 ms sample period and use it to identify the parameters  $a_1 - a_6$ . The procedure will be very similar to your previous identification. You will just need to modify the A and B matrices for the least squares solution. How does this higher order model compare to the lower order model? Answer this question by producing step responses and bode plots of both your original low order identified transfer function and this new higher order transfer function. Also compare the poles of each transfer function. Can you explain the faster set of poles found in the higher order model? Hint: Look at the plot of the velocity data uploaded for the 5ms run. Turn in these plots and the M-file used to run this identification.

3. For this question I would like you to investigate the sensitivity in the numerics of the plant model in the z domain as sample rate is increased. To do this use a hypothetical motor with  $K = 60$  and  $\tau_m = .3$ . Use the steps below to help you in this task.
  - a. Use the MATLAB script below to help you.
  - b. Form the continuous transfer function in Matlab. Note where the continuous pole is located and compare it to the poles found after a numerical error has been introduced to the discrete plant model.
  - c. Find the discrete transfer function using 0.01 seconds as the sample period.
  - d. Add a numerical error of 0.009 to the discrete pole. Convert the transfer function back to a continuous transfer function and compare to the initial transfer function in step b. (Use the following Matlab code:)

```

%EXAMPLE m-FILE
%Set up transfer function
num=[60];
den=[.3 1];
sys=tf(num,den)
%Now use c2d to map your transfer function to the z-domain
sysz1=c2d(sys,.01,'zoh')
%Now add a plant numerical error by adding .009 to your discrete model.
sysz1.den{1} = sysz1.den{1} + [0 0.009]
%Now use d2c to map your transfer function back to the s-domain

```

```
sysc1=d2c(sysz1)
```

- e. Repeat the same process for a sample period of 0.001 seconds. Repeat again for two additional sample frequencies to help you visualize the changes that occur as sample rate varies.
- f. What can you say about the sensitivity of transfer function coefficients at different sample rates? Make sure to keep this in mind when performing your simulations in the upcoming labs.