Exp 6: Motor Spin Down Test

In this experiment, you will be exposed to a practical example of a first order dynamic system – the decay response of a DC motor provided with an initial angular velocity. The goal of the experiment is to find out which friction model best describes the friction observed in the given DC motor.

A schematic of the DC motor is shown below (Dorf and Bishop, Modern Control Systems)

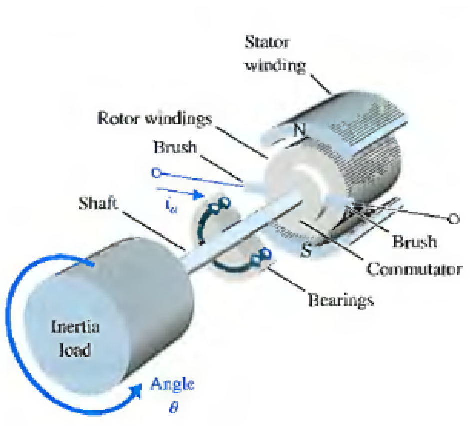

Figure 1: Schematic of a DC Motor

The equation motion of the DC motor is given by

$$I\dot{\omega}(t) = T(t) - T_f(t) - T_L(t)$$

where I is the total moment of inertia reflected to the motor axis, $\omega(t)$ angular speed, $T(t) = K_T i_a(t)$ is the motor torque, $T_f(t)$ is the torque due to friction and $T_L(t)$ is the load torque. In our case $T_L(t) = 0$. Once you disconnect the power supply, $T(t) = 0$ and we get

$$I\dot{\omega}(t) = -T_f(t) \tag{1}$$

The friction torque can be modeled using different models:

1. Viscous damping. Here we assume

$$T_f(t) = c_1\omega(t)$$

2. Coulomb damping. Here we assume

$$T_f(t) = c_2$$

3. Combined damping. Here we assume

$$T_f(t) = c_3\omega(t) + c_4$$

Here c_1, c_2, c_3 and c_4 are positive constants. As the goal of the experiment is to find out which friction model best describes the friction in the real system, i.e. we need to estimate the constants $c_i, i = 1, \dots, 4$ depending on the model we choose. To do this we need a way to estimate the angular speed of the motor once you disconnect the power supply. We estimate the angular speed by measuring the back emf generated in the motor, $e_b(t)$. The back emf is directly proportional to the angular speed of the motor, $\omega(t)$ and is can be written as

$$e_b(t) = K_b\omega(t)$$

where K_b is the constant of proportionality. Hence by measuring $e_b(t)$ one can estimate the angular speed of the motor. To estimate the constants in the different friction models, you will first solve Eq. (1) assuming different friction models. You will then estimate the constants for a particular friction model by using the analytical solution, your measured data and curve fitting.

During the lab: You will be given a DC motor (Maxon A-max 110947) mounted on a stand, an additional mass to be affixed to the motor shaft, and the tools required for the mechanical assembly. You will also be provided with a power supply and a USB-based Arduino setup for data acquisition. You will need to have MATLAB installed and running on your workstation. Attach the cylindrical mass to the motor, and affix the motor mount on the table. Connect one of the motor terminals to the analog input terminal of the Arduino, and the other one to ground (GND). Start the program Arduino IDE, and set it to record and display the input voltage. Power the motor by connecting the 5V supply from the Arduino to the motor terminal. Make sure that this is the same terminal that is connected to A0 (Fig. 2).

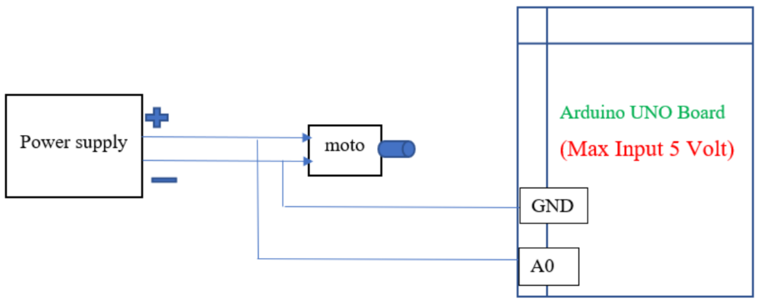

Figure 2: Circuit Diagram

You should be able to observe the voltage applied to the motor on the computer monitor. Disconnect the power to the motor. The Arduino IDE will now display the back EMF of the motor. The speed of the motor is proportional to the back EMF of the motor – this speed constant, or voltage constant, can be found in the motor datasheet. To record data for the experiment, connect the power supply to the motor, and allow the motor to reach steady-state. In Arduino IDE, start streaming and click on write to file to record data. Disconnect the power to the motor by removing one of the terminals of the wire, and record the data till the motor gradually spins down.

Sample code for curve fitting: Here is a sample code to fit the experimental data to the viscous damping model using 'fminsearch' function in Matlab. You can also use the curve fitting function if you wish. The analytical solution to the differential equation

$$I\frac{d\omega}{dt} + c_1\omega = 0$$

is

$$\omega = \omega_o \exp\left(\frac{-tc_1}{I}\right)$$

Here ω_o is the angular speed when $t = 0$. We will be estimating the damping coefficient c_1 to find the best curve to the data.

```
%ME370 Lab
%Created by Salil S. Kulkarni and Darshan Shah— March — 2023
 clear all
 close all
 options = optimset('Display','iter');
%parameters for DC motor (part no.110947)
```

```
lfKbrpmpV = 1180; %rpm/V
lfKbradps = 2*pi/60*lfKbrpmpV; %in rad/s

Imotor = 10.9; %gcm^2
Imotor = 10.9/1000*(1e-2)^2;

%Moment of inertia of the Aluminum cylinder
%I have guessed the values of R and H - use the values that you have measured
R = 2/100;
H = 1.5/100;
rho = 2700; %density of Aluminum
mass = pi*R^2*H*rho;
IAl = mass*R^2/2;
I = Imotor+IAl;

%read the data that you have stored
rglfData = readmatrix('Data.xlsx');
%use the data from the point it starts to decay - need to remove the steady
%state readings

%size of the data - number of rows, columns
[iR,iC] = size(rglfData);

%convert millisec to seconds
rglfTime = rglfData(:,1)/1000; %time in sec

%shift the time axis so that time starts at 0.0;
rglfTime = rglfTime(:) - rglfTime(1,1)*ones(iR,1);

%convert the voltage to angular speed (rad/s)
rglfOmega = rglfData(:,2)*lfKbradps;

%This is the value of Omega at instant it starts decaying
Omega0 =rglfOmega(1);

%initial guess for the fitted parameter
x0 = 0.0;

%here we are finding the unknown constant using fminsearch - an unconstrained
%minimization function in matlab
%you can also use the curve fitting function
[bestx,fval,exitflag,output] = ...
fminsearch(@(x)ViscousDamping(x,rglfTime,rglfOmega,I,Omega0),x0,options);

%solution returned by the optimization routine
c = bestx;
disp(' The value of the damping coefficient is:');
disp(c)

%compare the fitted fucntion with the actual data
figure(1);
```

```
plot(rglfTime,rglfOmega,'*');
hold on
y = Omega0*exp(-rglfTime*c/I);
plot(rglfTime,y,'r','LineWidth',2)
grid on
xlabel('Time (s)')
ylabel('Angular speed (rad/s)')
legend('Data','Fitted Function')
title('Viscous Damping Model')
set(gca,'fontsize', 16)

%the function to be fitted - corresponds to viscous damping
function y = ViscousDamping(x,tdata,ydata,I,Omega0)
          c = x;
          y = sum( (ydata - Omega0*exp(-tdata*c/I)).^2);
end
```

Here is the fitted model

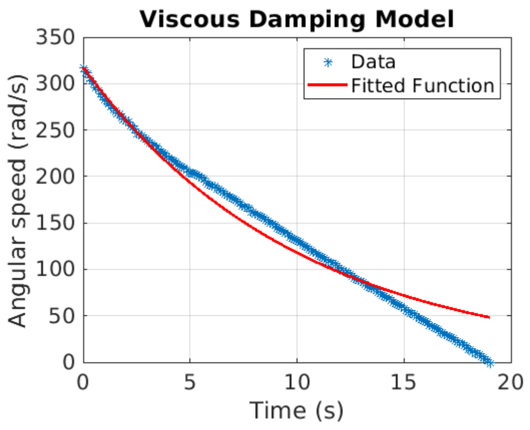

Figure 3: Viscous Damping Model