Notebook 1: System Identification

Written by:
Riley Kenyon & Gabe Rodriguez
ECEN 5038: Control Systems Lab

Feb. 4, 2020

Contents

1 Problem Statement

The rotary servo base unit offered by Quanser is a fundamental component to subsequent rotary systems used in this lab. The motor is controlled by a DAC and power amplifier to provide the voltage range (-10V, 10V). There are two sensors on the motor module: a tachometer measures angular velocity and an encoder measures absolute angular position. The resolution of the encoder is 4096 pulses per revolution which provides a discernible difference in position of 0.088°. The tachometer is filtered through the power amplifier and is measured to a 1:1 ratio of voltage to angular velocity in radians per second. Both sensor signals are connected to an ADC and are imported into Matlab/Simulink using the Quarc software offered by Quanser. Figure 1 references the typical setup of the system.

In this notebook we experimentally identify the model of the system for both position and velocity. The methods used to identify parameters are time- and frequency-based system identification. The notebook ultimately concludes by comparing and validating the time constants and DC gain of the system.

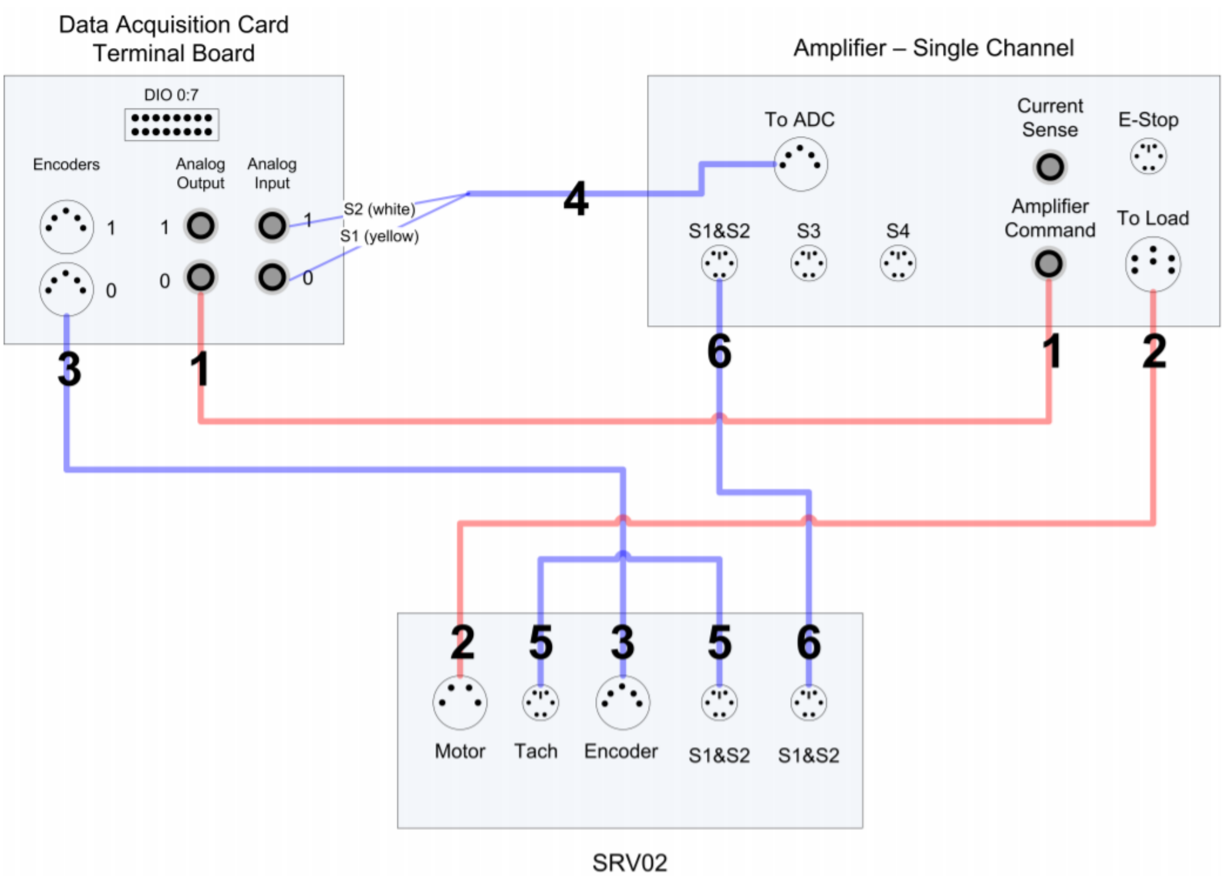

Figure 1: Connecting the SRV02 to a single channel amplifier and two channel DAQ (Image courtesy of the SRV02 User Manual). For the purposes of the lab, the tachometer output is connected directly to the S1&S2 port of the amplifier.

2 Theory

2.1 Theoretical Model

A motor can be modeled as a coiled wire that produces a back voltage from an inertia-resisting motion. In terms of electrical components, this includes a resistance and inductance, modeled by an RL circuit. The mechanical components include an inertia (J) and damping (μ). General relations are found in Eqs. (1) and (2). The coupling that occurs between the components is defined using coefficients k_m and k_τ. The back voltage is defined in Eq. (4) and the torque-current relation is defined by Eq. (3).

$$J\ddot{\theta} = \tau - \mu\dot{\theta} \tag{1}$$

$$V_{in} = Ri + L\frac{di}{dt} + V_m \tag{2}$$

$$\tau = k_\tau i \tag{3}$$

$$V_m = k_m\dot{\theta} \tag{4}$$

2.1.1 Transfer Function

Using the Laplace transform of the aforementioned equations, the voltage-position relationship of the system can be distilled to a first-order transfer function, shown in Eq. (7). As seen in the simplification, the inductance of the motor is assumed negligible. From Eq. (7), the damping coefficient (μ) is lumped into the rest of the time constant (τ) during system identification. In conclusion, the two transfer functions used to model the system are Eqs. (6) and (7) with τ representing the time constant and K being the DC gain.

$$\frac{\Omega}{V_{in}} = \frac{\frac{k_\tau}{RJ}}{s + \frac{k_\tau k_m + R\mu}{RJ}} \tag{5}$$

$$\frac{\Omega}{V_{in}} = K\frac{1}{\tau s + 1} \tag{6}$$

$$\frac{\Theta}{V_{in}} = K\frac{1}{s(\tau s + 1)} \tag{7}$$

2.1.2 State Space

Another method of system representation is through state space. The state variables x are defined as angular position, velocity, and current. The following expressions utilize the differential equations describing the motor for representing the position-voltage relation in state space.

$$x_1 = \theta \tag{8}$$

$$x_2 = \dot{\theta} \tag{9}$$

The resultant system of equations matches the representation by the transfer function for voltage to position of the system.

$$\dot{x_1} = x_2 \tag{10}$$

$$\dot{x_2} = -(\frac{k_\tau k_m + R\mu}{RJ})x_2 + (\frac{K_\tau}{RJ})V_{in} \tag{11}$$

2.2 System Identification Methods

2.2.1 Time Domain

A step response is a time domain method for system identification, specifically for low order systems. Two parameters can be obtained from the step response: the rise time (t_r) and the steady state value (y_{ss}). Rise time can be related to the time constant of the system, and the steady state value is equivalent to the DC gain for a unit step input. The relation between rise time and time constant is approximately $t_r = 2.2\tau$.

2.2.2 Frequency Domain

An alternative method better suited for higher order systems is using the frequency domain. Assuming the system is a linear time invariant (LTI) system, the frequency of an input signal is equivalent to the frequency of the output from the system. Additionally, by using the superposition principle associated with linear systems, these frequencies can be analyzed individually and summed together. In essence, this method can be utilized by creating a multi-sine input signal and determining the output frequencies and magnitudes.

Two parameters determine the resolution and maximum frequency that can be identified from an experiment: the sampling period (dt) and experiment time (T). Sampling period determines the Nyquist frequency, or the maximum frequency content that can be identified from a signal. This limitation is due to aliasing; the signal frequency can only be reconstructed if the rate of sampling is twice the maximum frequency of the signal. If one samples less than the Nyquist frequency, the signal appears to have lower frequency content. On the opposite end, the experiment time T determines the frequency resolution that can be resolved during reconstruction.

If the experiment is conducted correctly according to sampling and experiment time, the data collected can be converted to the frequency domain via a Fourier transform. The transfer function of the system can then be determined by taking the ratio of magnitudes for the input and output signals. The system must reach steady state for this method to work, otherwise the ratio will contain excessive noise due to the transient response of the system. This is accomplished by only recording data for the last half of an experiment. To keep the same T, the overall length of the multi-sine input must be doubled. After post-processing the ratio of magnitudes, the resultant plot is the Bode magnitude of the system.

2.2.3 Validation

To validate both proposed methods of system identification, we use a sample first-order system described by Eq. 12.

$$G(s) = \frac{1}{0.15s + 1} \tag{12}$$

The transfer function has a theoretical rise time of 2.2τ, or 0.33 s. One way of predicting rise time is to extrapolate the initial slope of the response until it crosses the steady state

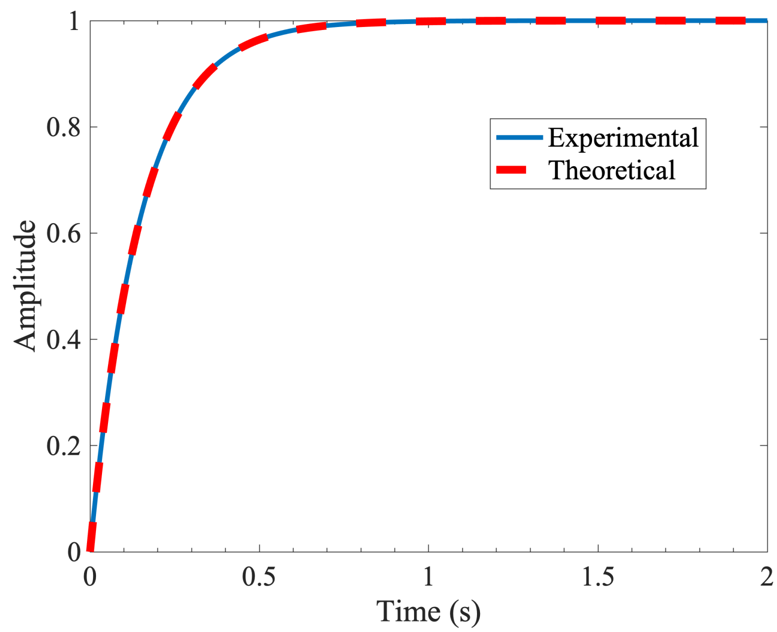

Figure 2: The example first order system with fitted parameters $K = 1$, $\tau = 0.15$ that match the known model parameters.

value. In the case of this system, the DC gain is one with a sampling period (dt) of 0.002s. A more accurate method to obtain the same information about the system is to fit the experimental response to the time based response shown in Eq. (13). The equation is obtained by taking the inverse Laplace transform of the transfer function in Eq. (6). Using this as a model, the experimental data can be fit to the function by minimizing the error of the non-linear regression. The fitting coefficients of the response are obtained using Matlab's built-in $nlinfit$ function; the coefficients represent the values for DC gain and the time constant. Using the sample system from Eq. (12), the time-based system identification approach is shown in Figure 2.

$$g(t) = 1 - e^{-t/\tau} \tag{13}$$

To validate the frequency-based method of identification, the same model from Eq. (12) is used. The input multi-sine is a summation of 150 sine waves, where the frequency of each wave is a multiple of the resolution determined by the experiment time (defined to be five seconds). Superimposing the harmonic frequencies of the sine waves can result in in the voltage saturation and the inability to identify unique sine curves. This offset is accounted for by introducing a unique phase angle for each wave. The sum of waves can be mathematically defined by

$$u = \sum_{i=0}^{150} \sin(\frac{2\pi i}{T} t + \phi_i), \tag{14}$$

where ϕ_i is either numerically random in the range of 2π or a chirp. A chirp is defined as

$$\phi_i = \frac{2\pi i}{N}, \tag{15}$$

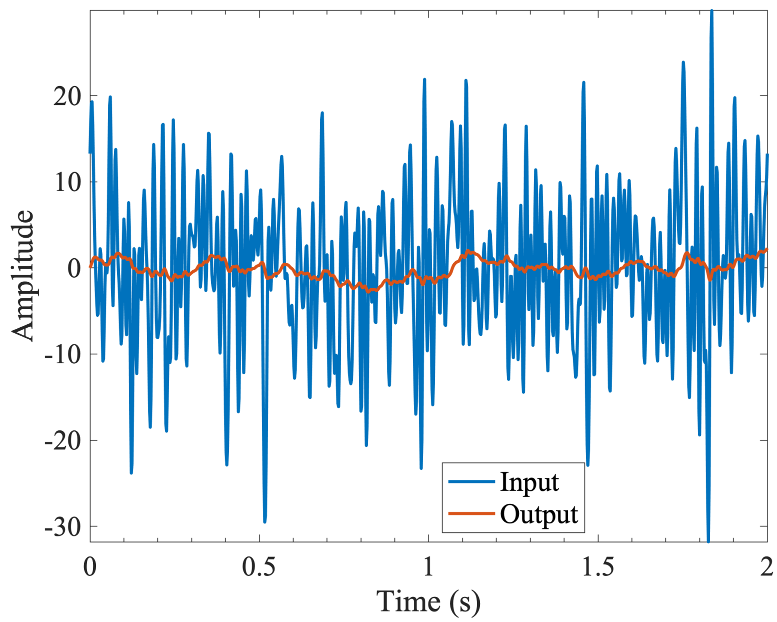

Figure 3: The input multi-sine and the corresponding response of the system are visualized here in the time domain. The input signal avoids saturation and the output is bounded.

where N equals the number of superimposed waves. The time representation of the input and output signals from the theoretical system are shown in Figure 3.

The Fourier transform of the input and output signals are visualized in Figure 4 where the stems are located at multiples of $2\pi/T$ with a Nyquist frequency of π/dt. The Bode plot of the system is then found by taking the ratio of output to input magnitudes and plotting as a function of frequency. The resultant magnitude Bode plot is shown in Figure 5. From the Bode plot, the parameters of τ and K can be identified. The DC gain is taken from the Bode plot by the magnitude at zero frequency. In implementation, this is done by averaging the flat section shown before the cutoff frequency in the Bode plot. The cutoff frequency is defined as half power or $-3dB$ from the DC gain. In this example, the cutoff is found by determining the intersection of the plot with $-3dB$ due to the DC gain being one or $0dB$. The fit is shown in Figure 5 and matches the parameters of the known model.

3 Implementation

The time and frequency method validated in the previous section is used on the physical system to identify a first-order model for the voltage-velocity relationship and a second-order model for voltage-position relationship. Time-based identification is used to obtain a preliminary fit and act as a comparison for the frequency based method. The setup will be in open loop for the determining the voltage-velocity relationship.

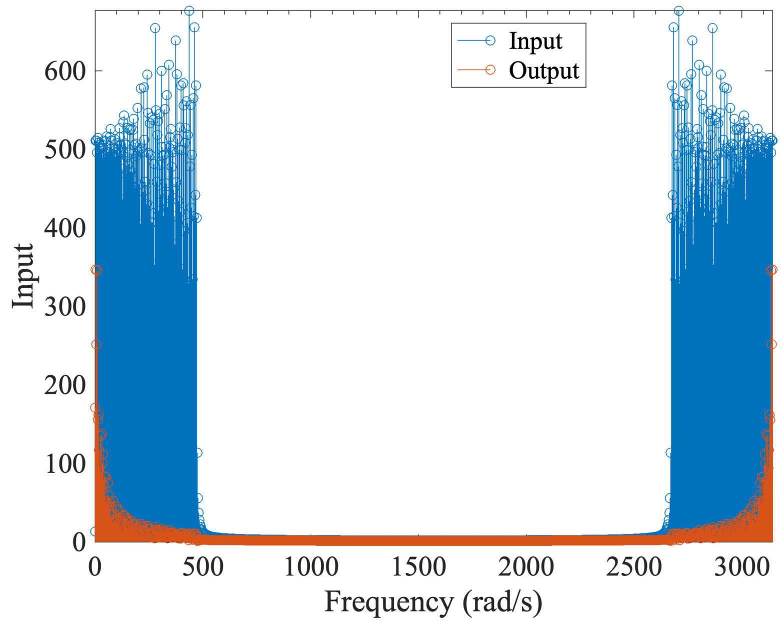

Figure 4: The Fourier transform of the input and output signals collocated at multiples of the frequency resolution $2\pi/T$ for the example system.

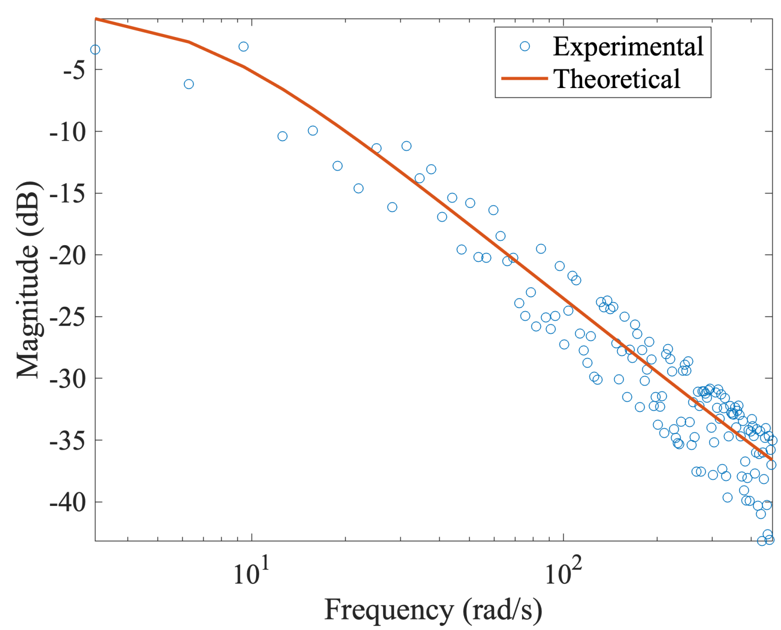

Figure 5: The Bode plot follows the discrete representation of the known model. From the figure, an estimate is made to identify the DC gain of the system by looking at the magnitude at $w = 0$. Note: The cutoff frequency of this example is $6.67\frac{rad}{s}$.

3.1 Velocity Step Response

For the open loop experiment to identify the transfer function parameters, the experiment time is two seconds because the system pole is fast and reaches a steady-state quickly. The input signal is a one volt step occurring at a time of one second. By delaying the step, a voltage offset observed during the first second can be subtracted from the measurements taken during the step response. The offset is likely due to sensor noise or a bias. After processing the signal, a response is observed to begin at zero volts. Using the model of a first-order response shown in Eq. (13), the experimental data produced fitting coefficients of $\tau = 0.025s$, and a DC gain of $K = 1.344$. The system response and the fitted model are shown in Figure 6.

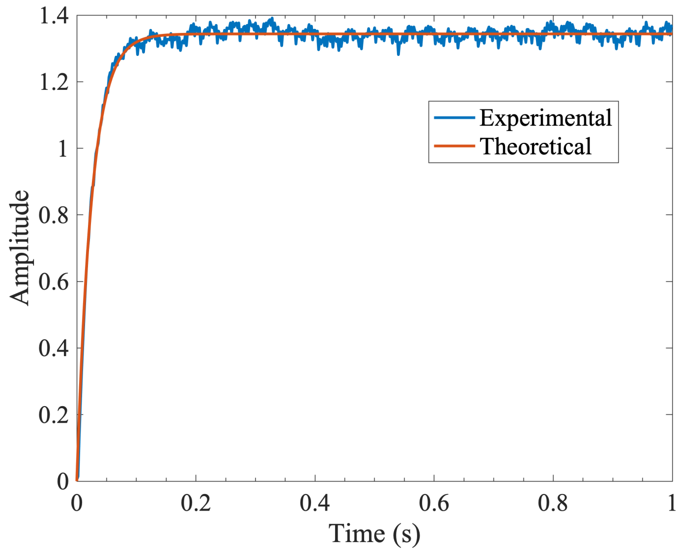

Figure 6: The step response of the motor's tachometer output using a one volt input step.

3.2 Velocity Bode Plot

The experiment length for the frequency-based identification is chosen to be five seconds, requiring an overall input of 10 seconds to avoid a transient response. The multi-sine is predetermined before passing the signal to the motor to avoid saturating the amplifier. The resultant signal spans the majority of the voltage range to obtain better resolution when converting the output signal from analog to digital. The experimental Bode plot of the system is overlaid with the Bode plot of the transfer function estimated by the frequency method and shown in Figure 7. The parameters obtained through the frequency-based identification in open loop for the voltage-velocity relationship are a time constant $\tau = 0.021s$ and DC gain of $K = 1.420$.

3.3 Angular Position Bode Plot

As shown in Eq. (7), there is an integrator for the theoretical voltage-position transfer function. To better obtain an identifiable bode plot for position, the system is put in closed

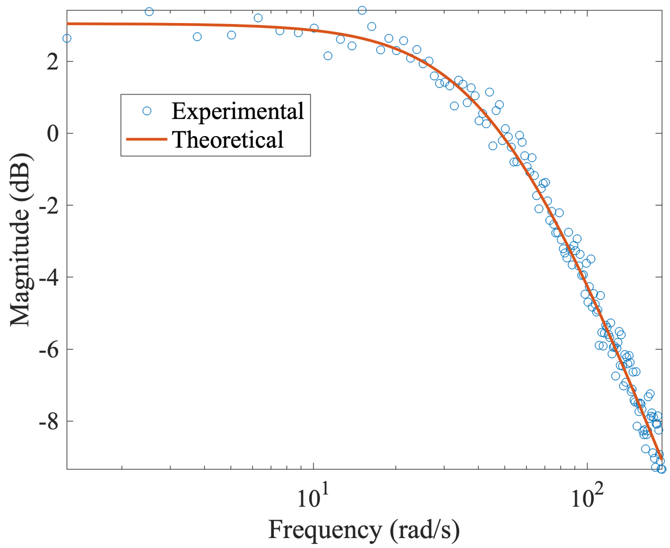

Figure 7: The experimental bode plot of the voltage-velocity relationship obtained from 150 superimposed sine waves.

loop and plotted against a theoretical model based on the parameters gained from the open loop configuration for velocity. If agreeable, the motor can be modelled using the time constant τ and DC gain K. The general form of the closed loop transfer function is shown in Figure (8).

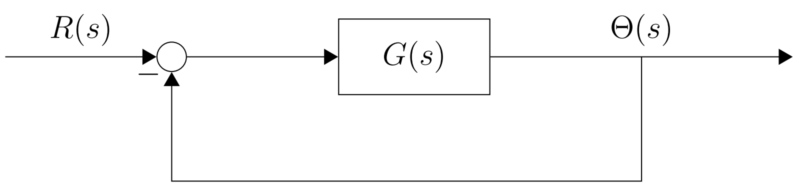

Figure 8: Unity Closed-Loop Feedback System. For analysis of reference and output, the controller ($C(s)$) both the disturbance (d) and noise (n) were neglected to obtain the transfer function. $R(s)$ represents the input voltage, $G(s)$ is the motor transfer function, and $\Theta(s)$ is the output angular position represented by voltage.

For a model of the voltage-position relationship, the closed loop transfer function can be represented by

$$T(s) = \frac{K}{s(\tau s + 1) + K} \tag{16}$$

where the parameters τ and K from the open loop configuration are also represented in closed loop. The bode plot of the closed loop system is shown in Figure 9 using the values of τ and K obtained from the system identification of the voltage-velocity relationship to plot the closed loop theoretical model.

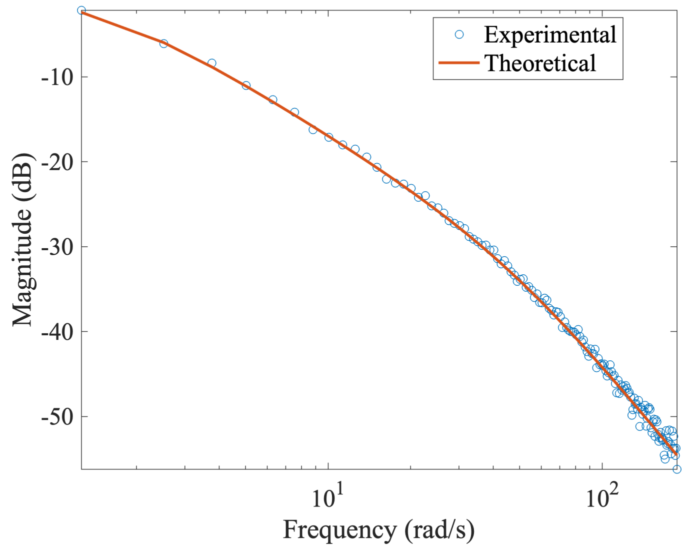

Figure 9: The closed loop bode plot describing the voltage-position relationship of the motor. The bode plot of the theoretical closed loop transfer function is provided based on the estimates of τ and K from the voltage-velocity model.

4 Conclusion

In this notebook, the theoretical model of the system was derived based on electro-mechanical equations of motion. It was deemed feasible to model the motor as a first-order system to relate input voltage to output velocity and lump parameters together to describe the time constant τ and DC gain K. The angular position relation of the motor with respect to input voltage is the integral of velocity, represented by a pure integrator in the transfer function. To guarantee stability and Bode plot readability, the position relationship was observed in closed loop. The preceding methods of identification produced an estimated time constant of $\tau = 0.021$ and DC term of $K = 1.420$. The variability between experiments deviate by 20.10 % for τ and 5.56 % for the K. The resultant transfer function we identified for the SRV02 motor from Quanser is described in Eq. (17).

$$G(s) = \frac{1.420}{s(0.021s + 1)} \tag{17}$$

5 Appendix II: MATLAB Code

```
%% Initialize
close all;clear all; clc;

%% Load data

sim_idx   = 4;
sim_names = {'Code Validation','System Identification Omega',...
    'System Identification Theta Open Loop',...
    'System Identification Theta Closed Loop'};
filename = convertCharsToStrings(sim_names{sim_idx});

is_unity = true;
controller_mat = 'controller.mat';

% Define common parameters
dt = 0.002;             % Sampling time
N  = 150;               % Number of sine waves
font_size = 18;


% Define unique parameters
switch filename
    case 'Code Validation'
        K   = 1;                     % Fake step data
        tau = 0.15;
        T   = 2;
        A   = 1;
        is_closed = false;
        load fakeStepData.mat
    case 'System Identification Omega'
        K   = 1.4204;          % First round of ID
        tau = 0.02090368438074;
        T   = 5;
        A   = 30 / N;                     % Sine wave amplitude
        den = [tau 1];
        is_closed = false;
        load freqdata_T5_N150_A30.mat   % For sytemID_time (theta, v_in, velo
    case 'System Identification Theta Open Loop'
        K   = 1.4204;          % First round of ID
        tau = 0.02090368438074;
        T   = 5;
        A   = 30 / N;                % Sine wave amplitude
```

```
        den = [tau 1 0];
        load freqdata_T5_N150_A30.mat   % For postProcess.m
    case 'System Identification Theta Closed Loop'
        K   = 1.4204;        % First round of ID
        tau = 0.02090368438074;
        T   = 5;
        A   = 30 / N;                  % Sine wave amplitude
        den = [tau 1 0];
        if is_unity
            C = 1;
        else
            load(controller_mat);
        end
        load freqdata_closed_T5_N150_A30.mat   % For postProcess.m
end

% Parameter calulations
i = (1:N)';               % Sequence array
w = 2*pi/T*i;             % Frequency
% phi = 2*pi/N*i;         % Chirp
phi = 2*pi*rand(1,N)';    % Random Noise
num = K;

% Frequencies of fft
w_max  = max(w);
w_nyq  = pi / dt;
w_res  = 2 * pi / T;
w_plot = (0:w_res:2*w_nyq)';

% Check input
dt  = 0.002;
t   = (0:dt:T)';

% Create input multisine
u = A * sin(w * t' + phi);
u = sum(u)';

% Verify bounded between (-10,10) otherwise re-iterate
while any(u <= -10 & u >= 10)

    phi = 2 * pi * rand(1,N)';       % Random Noise
    u   = A * sin(w .* t' + phi);    % Generates all sine waves
    u   = sum(u)';                   % Sum of sine waves

end
```

```matlab
%% Initialization

close all

run initialize.m
t_save = t;
offset = 0.0220;        % Resting voltage

%% Fitting Omega

if strcmp(sim_names{sim_idx}, 'System Identification Omega')

        w_max2 = 20;

        idx = floor(length(velocity) / 2) + 1;
        y = velocity(idx:end) - offset;
        u = v_in(idx:end);
        t = t_save(1:length(y));
        w_plot = w_plot(1:length(y));
        plot_idxs = [4];

        Y = (fft(y)) / length(y) * 2; % Frequencies of output signal
        U = (fft(u)) / length(u) * 2;

        G_mag_exp = Y ./ U;
        G_mag_exp(1) = G_mag_exp(2);
        K2     = mean(abs(G_mag_exp(w_plot < w_max2)));


%         f = @(beta,t) beta(1) * (1 - exp(-t / beta(2)));
%         beta0 = [1;1];
%
%         beta = nlinfit(t,y,f,beta0);
%        K     = beta(1);
%         tau   = beta(2);

        fprintf('The fitted parameters are K = %.4f and tau = %.4f.\n', K, ta

        G_ol_th        = tf(K,[tau 1]);
        [G_ol_mag_th,~] = bode(G_ol_th, w_plot);
        G_ol_mag_th = squeeze(G_ol_mag_th);

%         figure3 = figure;
%         axes1    = axes('Parent',figure3);
%         hold(axes1,'on');
```

```
%           plot2 = plot(t,y);
%           plot1 = plot(t,f(beta,t));
%           set(plot1,'DisplayName','Theoretical', 'LineWidth',2);
%           set(plot2,'DisplayName','Experimental','LineWidth',2);
%           ylabel('Amplitude','HorizontalAlignment','center');
%           xlabel('Time (s)');
%           box(axes1,'on');
%           set(axes1,'FontName','Times New Roman','FontSize',font_size,...
%               'XMinorTick','on');
%           legend1 = legend(axes1,'show');
%           set(legend1,'FontSize',font_size,'Location','best');

        print(gcf,filename+'_theo_vs_exp.png','-dpng','-r300');

        figure;
        semilogx(w_plot,db(G_ol_mag_th))
        hold on;
        semilogx(w_plot,db(abs(G_mag_exp)))

        % sim_names{sim_idx} = 'System Identification Theta Open Loop';

end

%% Post Processing Script

switch sim_names{sim_idx}
    case 'Code Validation'
        y = lsim(tf(num,den), u, t);
        plot_idxs = [1, 2, 3];
        is_closed = false;
    case 'System Identification Theta Open Loop'
        t    = (0:dt:T-dt)';
        idx  = length(v_in) - length(t) + 1;
        u    = v_in(idx:end);
        y    = theta(idx:end);
        G_ol = tf(num,den);
        plot_idxs = [2,4];
        is_closed = false;
    case 'System Identification Theta Closed Loop'
        t    = (0:dt:T-dt)';
        idx  = length(v_in) - length(t) + 1;
        u    = v_in(idx:end);
        y    = theta(idx:end);
        G_ol = tf(num,den);
        G_cl = C * G_ol / (1 + C * G_ol);
```

```
        plot_idxs = [2,4];
        is_closed = true;
end

% Fourier Transform
U = fft(u); % Frequencies of input signal
Y = fft(y); % Frequencies of output signal

% w_plot = w_plot(w_plot <= w_max);
% U      = U(1:length(w_plot));
% Y      = Y(1:length(w_plot));

% Obtain discrete bode plot of transfer function
G_ol = c2d(G_ol,dt,'zoh');
[G_ol_mag_th, ~] = bode(G_ol, w_plot);
G_ol_mag_th    = squeeze(G_ol_mag_th);

if is_closed
    G_cl = c2d(G_cl,dt,'zoh');
    [G_cl_mag_th, ~] = bode(G_cl, w_plot);
    G_cl_mag_th    = squeeze(G_cl_mag_th);
end

% Calculate experimental transfer function
G_mag_exp = Y./U;                 % Output/Input
G_mag_exp(abs(U) < 1e-3) = 0;     % Remove noise in input signal
G_mag_exp(1) = 0;                 % Remove non-existent DC signal

%% Plots

for plot_idx = plot_idxs
    switch plot_idx

        case 1 % Experimental Time Domain Plot (input vs. output)

            figure1 = figure;
            axes1   = axes('Parent',figure1);
            hold(axes1,'on');
            plot1 = plot(t,u);
            plot2 = plot(t,y);
            set(plot1,'DisplayName','Input', 'LineWidth',2);
            set(plot2,'DisplayName','Output','LineWidth',2);
            ylabel('Amplitude','HorizontalAlignment','center');
            xlabel('Time (s)');
            box(axes1,'on');
```

```
            set(axes1,'FontName','Times New Roman','FontSize',font_size,...
                'XMinorTick','on');
            legend1 = legend(axes1,'show');
            set(legend1,'FontSize',font_size,'Location','best');
            axis tight
            print(gcf,filename+'_time.png','-dpng','-r300');

        case 2 % Bode Plot Comparison

            idxs = w_plot < w_max;

            % Overlay theoretical and experimental
            figure2 = figure;
            axes1   = axes('Parent',figure2);
            hold(axes1,'on');
            if is_closed
                semilogx1 = semilogx(w_plot(idxs),[db(abs(G_cl_mag_th(idxs)))
                    db(abs(G_mag_exp(idxs)))]);
            else
                semilogx1 = semilogx(w_plot(idxs),[db(abs(G_ol_mag_th(idxs)))
                    db(abs(G_mag_exp(idxs)))]);
            end
            set(semilogx1(1),'DisplayName','Theoretical','LineWidth',2);
            set(semilogx1(2),'DisplayName','Experimental','Marker','o',...
                'LineStyle','none');
            ylabel('Magnitude (dB)','HorizontalAlignment','center');
            xlabel('Frequency (rad/s)');
            xlim(axes1,[3 500]);
            box(axes1,'on');
            axis tight
            set(axes1,'FontName','Times New Roman','FontSize',font_size,'XMi
                'XScale','log');
            legend2 = legend(axes1,'show');
            set(legend2,'FontSize',font_size,'Location','best');

            print(gcf,filename+'_freq.png','-dpng','-r300');

        case 3 % Fourier transform plot

            figure('Name','Fourier Transorms');
            subplot(2,1,1)
            stem(w_plot,abs(U));            % Create subplot of fft Y
            xlim([0 w_plot(end)]);
            ylabel('Input')
            subplot(2,1,2)
```

```
            stem(w_plot,abs(Y));            % Create subplot of fft U
            xlim([0 w_plot(end)]);
            xlabel('Frequency (rad/s)');
            ylabel('Input')

        case 4 % Theoretical vs. Experimental Time Domain Plot (step)

            if is_closed
                y_th = lsim(G_cl,u,t);
            else
                y_th = lsim(G_ol,u,t);
            end

            figure3 = figure;
            axes1   = axes('Parent',figure3);
            hold(axes1,'on');
            plot2 = plot(t,y);
            plot1 = plot(t,y_th);
            set(plot1,'DisplayName','Theoretical', 'LineWidth',2);
            set(plot2,'DisplayName','Experimental','LineWidth',2);
            ylabel('Amplitude','HorizontalAlignment','center');
            xlabel('Time (s)');
            box(axes1,'on');
            axis tight
            set(axes1,'FontName','Times New Roman','FontSize',font_size,...
                'XMinorTick','on');
            legend1 = legend(axes1,'show');
            set(legend1,'FontSize',font_size,'Location','best');

            print(gcf,filename+'_theo_vs_exp.png','-dpng','-r300');

    end
end
```