Midterm Project

The long-term objective of this project is to design an output feedback control system that can regulate the responses of a multi-input multi-output dynamic system. Recall from your previous state-space applied controls course that before you design an output feedback controller, you first need a controllable and observable state-space model of the open-loop plant.

For this project, we assume that nothing specific is known about the system dynamics; however, we do know that it has **two outputs** and **two inputs**, and we know that the system is open-loop stable. The Midterm Project will focus on developing and validating an empirical discrete-time state-space model of the dynamic system using system identification methods from the course notes. The Final Project will focus on developing and validating an output feedback controller.

For this project, you must consider the open-loop plant as a "black box" dynamic system. This dynamic system already has a sample rate of 20 Hz and includes appropriate anti-aliasing filters. You are also given the following:

Control Inputs: u_1 and u_2

- Control input signals will saturate or clip at 10 volts, i.e. $|u_m| \leq 10V$
- Control signals must not saturate the DAC's for more than 0.1% of the samples
- You can ignore DAC amplitude quantization for this assignment

Sensor Responses: y_1 and y_2

- Sensor signals will saturate or clip at 10 volts, i.e $|y_i| \leq 10V$
- Sensor signals must not saturate the ADC's for more than 0.1% of the samples
- You can ignore ADC amplitude quantization for this assignment
- Each sensor signal is corrupted with stationary random noise n_1 and n_2

The dynamic system for this project is embedded in the Matlab p-file: **`s19_plant.p`**, which is a compiled binary function. This p-file can be called like any ordinary Matlab function. To obtain a response matrix **`y`** caused by an excitation vector **`u`**, the call structure in Matlab is given by:

```
y = s19_plant(u);
```

where N is the number of samples, **`y`** is a $2\times N$ matrix of sensor responses, **`u`** is a $2\times N$ matrix of the excitation input. This particular plant will only allow you to collect limited number of samples in one batch. You may not use more than one batch of data!

In order to estimate the open-loop dynamics of this MIMO plant (i.e. System ID), you will need to probe the dynamic system with an appropriate excitation. There are four input-output paths in the open-loop dynamic system that must be identified. From the class notes, you know that you will need to make a number of decisions, and your first attempt may not be what is needed. You should use a trial-and-error approach. It is very common in this project to re-run your entire program many times until all of your settings produce acceptable results. It is also very common in this project to obtain slightly different results every time you run the program, even if you don't make any changes to your settings. This is exactly what happens in a real world test, and is very different from a conventional academic problem.

In many of the tasks below, you must compute and plot frequency domain information. All frequency domain plots must at least range from 0.01 Hz to the Nyquist frequency; however, you will have to determine an appropriate frequency bin resolution. The frequency limits should be identical and use a log scale.

This project must be completed in Matlab, and your project submission must use the Live Script feature in Matlab, which will be used to generate a PDF document that includes your Matlab code as well as any formatted plots and numerical outputs. Each of the numbered sections below must appear in your Matlab script in the order listed.

Section 1: Construct an excitation for system identification. Compute and plot the properly scaled power spectrum for the excitation. Demonstrate that your excitation signal does not saturate the DAC by plotting your entire excitation in the time domain.

(1 Figure only)

Section 2: Apply the excitation to the dynamic system to obtain the noisy response signals. Estimate and document the SNR (dB) for each of the two paths. HINT: Your System ID results will be much better if you attempt to maximize the SNR; however, you are not allowed to saturate (i.e. clip) either the excitations or the responses!

(Display properly labeled SNR results to the command window)

Section 3: Compute and plot the properly scaled power spectrum for the responses. On these same plots, compute and plot the properly scaled power spectrum for the noise signals. Demonstrate that the measured sensor responses do not saturate the ADC's by plotting the responses in the time domain. The spectrum of y_1 and n_1 must be in one axes, and the spectrum of y_2 and n_2 must be in a second axes. The time series plot of y_1 and n_1 must be in one axes, and the time series plot of y_2 and n_2 must be in a second axes. You must have four separate axes in a 2x2 grid on a single figure.

(1-2 Figure(s) only)

Section 4: Apply an appropriate frequency domain estimation technique from the course to estimate frequency response functions for each open-loop path. Estimate the coherence associated with each path. This data will be plotted in a later section.

(No figures and no output to the command window)

Section 5: Use the **`invfreqz`** function to estimate discrete time transfer function models for each of the two paths in the open-loop plant. You must choose exactly the same number of poles for each path. Store your estimated numerator and denominator polynomials in a Matlab cell array with appropriate dimensions to match the dynamic system dimensions. Convert the numerator and denominator cell arrays into a discrete-time transfer function LTI object. Convert this LTI object into a minimum realization using the **`minreal`** function.

(No figures and no output to the command window)

Section 6: Convert the minimum realization transfer function LTI object from Section 5 into a discrete-time state-space LTI object. Input the discrete-time state-space model to the **`balreal`** function in Matlab to generate a balanced realization and extract the Hankel singular values for the estimated model. Plot the Hankel singular values in a bar chart to determine an appropriate cutoff threshold for pruning non-contributing modes from the resulting LTI state-space object. Use the **`modred`** function in Matlab to generate a reduced order LTI discrete time state-space model. Read the Matlab help for the **`minreal`**, **`balreal`**, and **`modred`** functions to learn more about how these functions work.

(1 Figure only)

Section 7: Compute the z-domain eigenvalues (poles) of the LTI object result from Section 6. Compute the natural frequencies with units of Hz, and the damping ratios associated with every eigenvalue.

(Display numerical values to the command window)

Section 8: Compute the eigenvalues (poles) for both of the discrete time open-loop transfer function models you estimated in Section 5. Use the **`zgrid`** function in Matlab to generate a z-domain grid, and then plot each set of poles on this plot using clearly labeled markers for each of the estimated path models. There should be three sets of poles on the plot: one for the y_1 path (Section 5), one for the y_2 path (Section 5), and one for the final discrete-time state-space model (Section 6). All markers should be the same for the poles associated with the same path. Make sure to use **`axis equal`** to scale the plot correctly.

If your models are accurate enough, you should see tightly concentrated “clusters” of poles from each of the paths in the z-plane. Clusters indicate pole locations that are common between paths. Technically both paths should have the exact same poles since they are all associated with the same dynamic system. This may not be the case for your solution because the estimation process is a numerical approximation for each individual path. Some poles will not belong to any cluster and not all paths will show up in every cluster.

(1 Figure only)

Section 9: Compute the frequency response of the final discrete-time state-space LTI object result from Section 5 using the **`freqresp`** function. Compute the frequency response of the final discrete-time state-space LTI object result from Section 6 using the **`freqresp`** function.

(No figures and no output to the command window)

Section 10: Compare the results of your estimation in the frequency domain by generating dB magnitude vs. log frequency and phase (degrees) vs. log frequency plots (a.k.a. Bode diagrams) for each path. You must directly plot the estimated frequency response from Section 4 on top of the frequency response from the transfer function results in Section 5 and the frequency response from the state-space results in Section 6. There will be two figures and on each figure there will be three axes displayed in a 3x1 grid: one for coherence, one for phase, and one for magnitude. All three frequency response curves must be properly labeled and displayed for visual inspection.

If you have accurately modeled each path, then the magnitude error should not be larger than 3dB at large magnitudes and the phase error should not be larger than 10 degrees at large magnitudes. Note, if the frequency responses of your final state-space model do not match the estimated frequency response with sufficient accuracy, you must go back and re-process the data. A portion of your grade will be based on how well your final discrete-time state-space model matches the estimated frequency response because inaccurate model estimation cannot be used for designing a real-time controller in the Final Project.

SUBMISSION REQUIREMENTS:

- You must submit one PDF document generated from your Matlab LiveScript that displays your results with clearly labeled sections corresponding to the numbered sections above
- You must also submit one Matlab LiveScript file (2 document submissions total)
- Your total submission must be less than 2 Mb
- You must not upload the s19_plant.p file that was provided to you
- A portion of your grade will be based on the formatting of all of your figures. All figures must be presented as if they were being submitted to a Journal, so they must be "publication quality"
- Figures are NOT allowed to have shaded backgrounds
- Figures can not be larger than one 8.5"x11" page
- Figures must not be rotated 90 degrees on a page
- All text on all figures must be clearly readable (11 point font minimum)
- All signals and axes within a figure must be properly annotated (legend, axis labels, title, colors, and markers where appropriate)
- All frequency domain plots should show data from the minimum frequency bin size to the Nyquist frequency, and have frequency units of Hz (do not use radians/sec)
- If a figure contains more than one time-domain or frequency-domain plot, each set of axes must have the same y-axis limits for direct comparison
- If a figure contains more than one data set, you must include a legend that clearly identifies each data set with an appropriate label