MAE270A Project

Prof. M'Closkey

Due date: 5pm, Dec. 13, 2024

Guidelines

Submit a PDF report organized according to the "tasks" outlined in this project description. The main results are shown in figures, however, write a short narrative for each task and answer any questions posed in the project description. You can choose up to one partner for this project and only a single report needs to be submitted. Clearly note yourself and partner on the report cover sheet. The Matlab code that you write to complete each task must be included in the report appendix. The code must be clearly commented.

Project objectives

- Learn about sample-and-hold models of linear systems.
- Identify empirical frequency responses from input-output data sequences.
- Estimating discrete-time state-space models from the pulse response estimate.
- Application to a three-input/two-output physical system created by connecting "subsystems" in specific a topology.

1 Discrete-time systems

1.1 Pulse response of discrete-time system

Consider a multi-input/multi-output, discrete-time linear system with m outputs and q inputs,

$$\begin{aligned} \mathbf{x}[k+1] &= A\mathbf{x}[k] + B\mathbf{u}[k] \\ \mathbf{y}[k] &= C\mathbf{x}[k] \end{aligned} \tag{1}$$

where the state dimension is n_s so $A \in \mathbf{R}^{n_s \times n_s}$, $B \in \mathbf{R}^{n_s \times q}$ and $C \in \mathbf{R}^{m \times n_s}$. It is assumed the "feedthrough" matrix $D \in \mathbf{R}^{m \times q}$ is zero. This system may have been derived from the testing of a continuous-time system as noted in Sec. 1.2, or, it may be a digital filter that is implemented within signal processing equipment.

We will assume the state-space matrices have real elements. It is useful to refer to the individual "channels" of the input $\mathbf{u}[k]$ as follows,

$$\mathbf{u}[k] = \begin{bmatrix} u_1[k] \\ u_2[k] \\ \vdots \\ u_q[k] \end{bmatrix}.$$

Define the discrete-time *unit pulse* as the scalar-valued signal

$$\delta[k] = \begin{cases} 1 & k = 0 \\ 0 & k > 0 \end{cases}$$

Using this notation, the rth column of the *pulse response* sequence $\{h[k]\}$, where $h[k] \in \mathbf{R}^{m\times q}$, is obtained when $\mathbf{x}[0] = 0$ and the input is given by

$$\begin{aligned} u_p &= 0,\ p \neq r \\ u_r &= \delta, \end{aligned}$$

The pulse response is the discrete-time analog of the impulse response of a continuous-time system. It is straightforward to show

$$\begin{aligned} &h[0] = 0 \in \mathbf{R}^{m\times q}, \\ &h[k] = CA^{k-1}B \in \mathbf{R}^{m\times q},\ k = 1, 2, 3, \ldots \end{aligned} \tag{2}$$

1.2 Sampling a continuous-time state-space system

This section shows how "sampling" a continuous-time system produces a discrete-time system. Consider testing a continuous-time system, denoted "System", in Fig. 1. The discrete-time input sequence is denoted $\mathbf{u}[k]$, $k \in \mathbb{Z}$. In practice, $\mathbf{u}[k]$ is generated by signal processing hardware for some finite range of k and is generally under control of the engineer[1]. In modern test equipment the discrete-time signal $\mathbf{u}[k]$ is converted into a continuous-time signal $\mathbf{u}(t)$ by a digital-to-analog convertor (DAC). The DAC implements a *zero-order hold* which extends the value of the $\mathbf{u}[k]$ over the time interval $[kt_s, (k+1)t_s)$. In other words, $\mathbf{u}(t) = \mathbf{u}[k]$, $t \in [kt_s, (k+1)t_s)$. In the pulse response testing discussed in Sec. 1.1, $k = 0$ refers to the sample at which the input pulse occurs. The sample period is denoted t_s. Let $p(t)$ represent a continuous-time rectangular pulse of height "1" and width t_s,

$$p(t) = \begin{cases} 1 & t \in [0, t_s] \\ 0 & t \notin [0, t_s] \end{cases}. \tag{3}$$

Then, the continuous-time input to the system can be expressed

$$\mathbf{u}(t) = \sum_{k=-\infty}^{\infty} \mathbf{u}[k]p(t - kt_s). \tag{4}$$

Thus, the continuous-time input $\mathbf{u}(t)$ is exactly known for all t and can be determined from $\mathbf{u}[k]$.

The continuous-time signal $\mathbf{y}(t)$ is the output of the system. This signal is converted to a discrete-time signal through the process of *sampling*. In practice, sampling is implemented with a integrated circuit called an analog-to-digital convertor (ADC). A trigger signal controls the ADC and tells it when to capture a sample of the continuous time signal. Thus, if the ADC is triggered on the time grid $\{0, t_s, 2t_s, 3t_s, \ldots\}$, then a discrete-time output sequence is defined as $\mathbf{y}[k] = \mathbf{y}(kt_s)$, $k = 0, 1, 2, 3, \ldots$.

[1] The square brackets in $\mathbf{u}[k]$ denote it is a discrete-time signal and the sample index is determined by an integer k. This notation may seem more cumbersome, however, we will need to refer to individual elements of $\mathbf{u}[k]$ and using subscripts for this purpose makes more sense

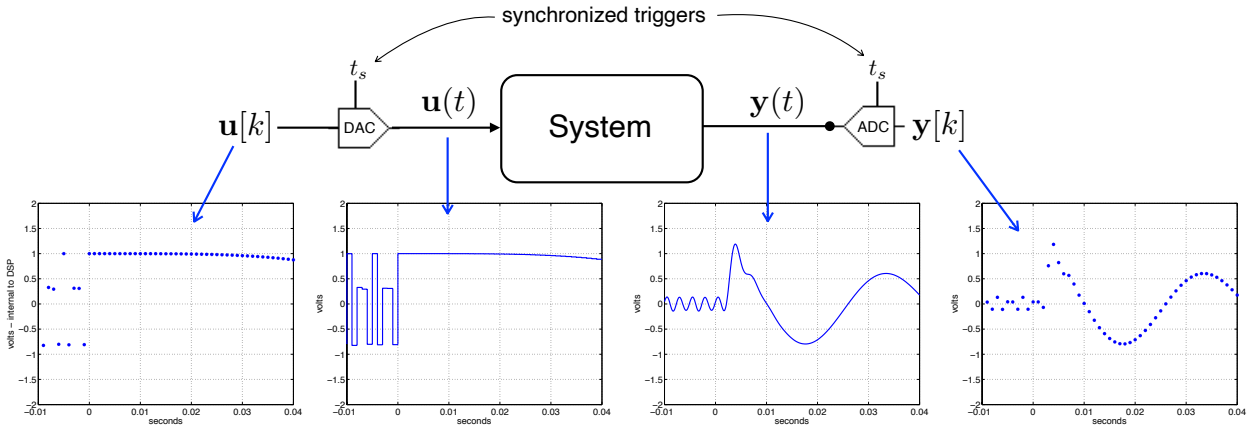

Figure 1: Testing a system with modern signal processing equipment.

It is assumed that the DAC and ADC are synchronized so that $\mathbf{u}[k]$ and $\mathbf{y}[k]$ are associated with same point in time, i.e. kt_s.

Suppose the continuous-time system is described by linear, time-invariant state-space equations

$$\begin{aligned} \dot{\mathbf{x}} &= A_c\mathbf{x} + B_c\mathbf{u} \\ \mathbf{y} &= C_c\mathbf{x} + D_c\mathbf{u}. \end{aligned} \tag{5}$$

The subscript "c" just denotes these state-space data are associated with the *continuous-time* system. Given the block diagram in Fig. 1, it is possible to determine a discrete-time state-space system that exactly relates the input and output *samples* $\mathbf{u}[k]$ and $\mathbf{y}[k]$. To determine the difference equation it is sufficient to relate the values of $\mathbf{x}$ at $t = kt_s$ and $t = (k+1)t_s$. This is accomplished using the closed-form solution that was derived in the notes,

$$\mathbf{x}[k+1] = e^{A_c t_s}\mathbf{x}[k] + \int_{kt_s}^{(k+1)t_s} e^{A_c((k+1)t_s - \tau)} B_c \mathbf{u}(\tau) d\tau,$$

where $\mathbf{x}[k] = \mathbf{x}(kt_s)$. Changing variables in the integral yields,

$$\mathbf{x}[k+1] = e^{A_c t_s}\mathbf{x}[k] + \int_0^{t_s} e^{A_c(t_s - \tau)} B_c \mathbf{u}(\tau + kt_s) d\tau,$$

Note $\mathbf{u}(\tau + kt_s) = \mathbf{u}[k]$, $\tau \in [0, t_s]$, so

$$\begin{aligned} \mathbf{x}[k+1] &= e^{A_c t_s}\mathbf{x}[k] + \int_0^{t_s} e^{A_c(t_s - \tau)} B\mathbf{u}[k] d\tau \\ &= e^{A_c t_s}\mathbf{x}[k] + \int_0^{t_s} e^{A_c(t_s - \tau)} B_c d\tau\, \mathbf{u}[k] \\ &= e^{A_c t_s}\mathbf{x}[k] - A_c^{-1}\left(I - e^{A_c t_s}\right) B_c \mathbf{u}[k] \end{aligned}$$

It is assumed A_c is invertible (no zero eigenvalues). If this is not the case then the integral must be determined another way. Thus, the discrete-time equations are

$$\begin{aligned} \mathbf{x}[k+1] &= A\mathbf{x}[k] + B\mathbf{u}[k] \\ \mathbf{y}[k] &= C\mathbf{x}[k] + D\mathbf{u}[k] \end{aligned} \tag{6}$$

where

$$A = e^{A_c t_s}, \quad B = -A_c^{-1}\left(I - e^{A_c t_s}\right) B_c, \quad C = C_c, \quad D = D_c. \tag{7}$$

The frequency response of the discrete-time system is given by

$$H[\omega] = C(e^{j\omega t_s} I - A)^{-1} B + D$$

Although not derived, $H[\omega]$ is the discrete-time Fourier transform (assuming asymptotically stability) of the system's *pulse response* given by (2).

2 Empirical frequency response estimates

Empirical frequency response estimates (EFRE) are a non-parametric representation of the input-output behavior of a system. EFRE can be obtained in a variety of ways:

1. Sinewave testing builds up a frequency-by-frequency estimate of the system's frequency response (the test inputs are sinusoids).
2. Impulse testing is common in modal analysis: an impact hammer applies a pulse-like force to the structure under test; the structure response is measured (typically with accelerometers) and the Fourier transforms of the output measurements are normalized by the Fourier transform of the input pulse to yield multi-frequency estimates of the frequency response.
3. Testing with "persistent" inputs like random signals, chirps, binary sequences, etc. to which spectral estimation techniques are applied to eventually recover the frequency response.

There is no one "best" experimental technique to obtain an EFRE.

2.1 Spectral density estimation from measurement data

EFRE using "random" test inputs will be brief discussed. There are a many interesting details that are skipped. Suppose a linear system is subjected to a persistent input, u (the single-input/single-output continuous-time case is considered). The output is $y = h * u$, where h is the system's impulse response and "$*$" denotes convolution. If u is bounded, as it is in any practical scenario, then y is bounded when the system is asymptotically stable. Correlation functions are defined for these signals:

$$\begin{aligned}
\text{Auto-correlations:} \quad R_{uu}(t) &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} u(\tau) u^*(\tau - t) d\tau \\
R_{yy}(t) &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} y(\tau) y^*(\tau - t) d\tau, \\
\text{Cross-correlations:} \quad R_{yu}(t) &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} y(\tau) u^*(\tau - t) d\tau \\
R_{uy}(t) &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} u(\tau) y^*(\tau - t) d\tau.
\end{aligned} \tag{8}$$

(the superscript u^* denotes the complex-conjugate, not convolution). These definitions may be applied to a wide class of persistent signals. For non-periodic "random" signals, it is assumed that the signal

properties are *translation invariant*. In other words, the absolute time is of no consequence in the formula, only the relative lag between the signals in the integrand is important.

An interesting and useful property of linear systems is the following:

$$R_{yu} = h * R_{uu}. \tag{9}$$

In other words, if the signal R_{uu} is applied as an input, then the output is R_{yu}. This relation is the basis for determining the system's impulse response from test data generated with random inputs. For example, if R_{uu} is impulse-like, then R_{yu} is an estimate of the system's impulse response. Computing the Fourier transform of (9) yields,

$$S_{yu} = \hat{h} S_{uu}$$

where S_{uu} is the Fourier transform of R_{uu}, S_{yu} is the Fourier transform of R_{yu}, and $\hat{h}$ is the Fourier transform of the system's impulse response, i.e. the frequency response. These transforms are the mean-square auto-spectral density of u and cross-spectral density of y and u, respectively. These are also called power spectral densities. Note that the frequency response is equal to,

$$\hat{h} = S_{yu} S_{uu}^{-1} \tag{10}$$

The ratio is formed only when S_{uu} has sufficient power at a given frequency. This is the basis for estimating a system's frequency response from test data:

1. apply a random input to the system
2. gather enough data to reliably estimate R_{yu} and R_{uu}
3. compute the Fourier transforms of R_{yu} and R_{uu} to get the power spectral densities S_{yu} and S_{uu}
4. estimate $\hat{h}$ from (10)

It is fair to ask why not simply use an impulsive input so that $y \approx h$. The answer is that the measurement y typically contains "noise" and effects of other disturbances acting on the system. Thus, a more realistic model for the signal y is

$$y = h * u + n$$

where n captures the presence of noise and disturbances. In this case,

$$R_{yu} = h * R_{uu} + R_{nu}$$

In most cases, we can arrange $R_{nu} = 0$, that is, the noise and input are *uncorrelated* so the relation (9) still holds even in the presence of disturbances and noise. This is not the case if we selected u to be impulsive because there is no averaging to remove the effects of disturbances. Furthermore, there are limitations on the amplitude of input signals that can be physically applied to a system so in practice an impulsive input will be limited in energy and consequently may produce a "weak" response. This limitation is removed when a persistent test input is selected.

There is one more innovation that occurred in the 1960's with the advent of inexpensive signal processing equipment[2]: S_{uu} and S_{yu} are *directly estimated*, therefore bypassing the correlation estimates. In other words, the frequency response is estimated using the following modified procedure:

[2] A revolution in signal processing occurred in the 1960's when the *fast Fourier transform* (FFT) algorithm was discovered/rediscovered and digital computers based on integrated circuits where becoming more widely distributed. This allowed the "real-time" estimation of spectral densities with special-purpose test equipment.

1. apply a random input to the system
2. gather enough data to reliably estimate S_{yu} and S_{uu}
3. estimate $\hat{h}$ from (10)

2.2 Pulse response estimates

In sampled-data systems, the test data are records of discrete-time input-output data. Thus, the power spectral densities of these discrete-time signals are computed. Matlab has functions which estimate power spectral densities of discrete-time signals. The function used in this project is `cpsd`. Suppose scalar-valued time series are given by the Matlab variables `u` and `y`, which are the input and output date vectors, respectively. Let `fs`, `win` and `nfft` represent the sample frequency associated with the data, the data-tapering window and length (number of points) of the data sub-record upon which the spectral estimates are based. Then, the following Matlab code estimates S_{uu} and S_{yu} on a specific grid of frequencies:

```
ts = 1/50; % sample period in seconds
fs = 1/ts; % sample rate in hertz
nfft = 250; % sub-record length nfft/fs = 5 seconds
win = hamming(nfft); % use Hamming data tapering window

[Suu,f] = cpsd(u,u,win,[],nfft,fs,'twosided'); % auto-spectral density of u
[Syu,f] = cpsd(y,u,win,[],nfft,fs,'twosided'); % cross-spectral density of y and u
```

The system's frequency response is then estimated on the frequency grid associated with the spectral densities (the `f` variable returned from `cpsd`) by simply forming the ratio `H = Syu./Suu`. The `twosided` option is convenient because `H` is determined on a frequency grid that is compatible with the *inverse fast Fourier transform* implemented in Matlab. Thus, an estimate of system's pulse response is simply `h = ifft(H)`. To conclude, a simple way to recover an estimate of the discrete-time system's pulse response is to follow these steps:

1. Apply a random input to the system and gather sufficient data for low-variance estimates of the spectral densities.
2. Suppose the data vectors are denoted `u` and `y` in Matlab (assumed to be a single-input/single-output system for now). Estimate the power spectra according to

   ```
   [Suu,f] = cpsd(u,u,win,[],nfft,fs,'twosided');
   [Syu,f] = cpsd(y,u,win,[],nfft,fs,'twosided');
   ```

3. Estimate the frequency response `H = Syu./Suu`.
4. Estimate the pulse response `h = ifft(H)` with associated time vector `t = [0:length(h)-1]/fs;`

The next section shows how to estimate a state-space model from the pulse response.

3 Time-domain model identification from pulse response estimates

The discrete-time system (1) has associated observability and controllability matrices,

$$\mathcal{O}_n = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n_s-1} \end{bmatrix} \in \mathbf{R}^{mn_s \times n_s}, \quad \mathcal{C}_n = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n_s-1}B \end{bmatrix} \in \mathbf{R}^{n_s \times n_s q},$$

which are assumed to be full rank, i.e. the model is controllable and observable. The system has m outputs and q inputs.

A finite number of terms from the pulse response sequence can be organized into an $mn \times nq$ *Hankel matrix*, denoted M_n,

$$M_n = \begin{bmatrix} h[1] & h[2] & h[3] & \cdots & h[n] \\ h[2] & h[3] & h[4] & \cdots & h[n+1] \\ h[3] & h[4] & h[5] & \cdots & h[n+2] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h[n] & h[n+1] & h[n+2] & \cdots & h[2n-1] \end{bmatrix} \in \mathbf{R}^{mn \times nq}. \tag{11}$$

This matrix uses $2n-1$ samples of the pulse response data. A distinguishing feature of the Hankel matrix is that the elements of each anti-diagonal block are the same. By virtue of it's definition, the Hankel matrix can be expressed as the following product,

$$M_n = LR,$$

where

$$L = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n_s-1} \\ CA^{n_s} \\ \vdots \\ CA^{n-1} \end{bmatrix} \in \mathbf{R}^{mn \times n_s},$$

$$R = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n_s-1}B & A^{n_s}B & \cdots A^{n-1}B \end{bmatrix} \in \mathbf{R}^{n_s \times nq}.$$

The observability and controllability matrices are embedded in L and R, respectively, so long as $n \geq n_s$. Thus, we need to use at least $2n_s$ data points to form M_n and $\tilde{M}_n$ (defined below). We will assume we have enough data to meet this constraint. Because rank $\mathcal{O}_n = n_s$ and rank $\mathcal{C}_n = n_s$ then rank $L = n_s$ and rank $R = n_s$, thus,

$$\text{rank } M_n = n_s,$$

In other words, the rank of the Hankel matrix constructed from the pulse response is equal to the state dimension. This assumption on the rank not only requires that enough points be available so that the

column and row dimensions of H_n are no less than n_s, but it also requires that certain "structural" properties of the system be satisfied –these details are discussed in the notes under the subjects of observability and controllability. If the state-space system undergoes a change of coordinates using $T \in \mathbf{R}^{n_s \times n_s}$, $\det T \neq 0$,

$$\begin{aligned} \mathbf{z}[k+1] &= T^{-1}AT\mathbf{z}[k] + T^{-1}B\mathbf{u}[k] \\ \mathbf{y}[k] &= CT\mathbf{z}[k] \end{aligned}$$

Note that L and R transform to

$$L \mapsto \tilde{L} = LT, \quad R \mapsto \tilde{R} = T^{-1}R.$$

Of course, the impulse response remains the same and so does the Hankel matrix $M_n = LR = \tilde{L}\tilde{R}$.

Now consider the situation in which the pulse response samples are provided, for example, by the procedure outlined in Sec. 2.2. In this case, M_n can be formed from $\{h[k]\}$, $k = 1, 2, \ldots$, according to (11). Compute an SVD of M_n,

$$M_n = U\Sigma V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_{n_s} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U_1 \Sigma_{n_s} V_1^T,$$

where $U_1 \in \mathbf{R}^{mn \times n_s}$, $\Sigma_{n_s} \in \mathbf{R}^{n_s \times n_s}$, $V_1 \in \mathbf{R}^{nq \times n_s}$. L and R can be determined from the SVD, however, the factorization is not unique, for example,

$$\begin{aligned} &L = U_1 \text{ and } R = \Sigma_{n_s} V_1^T, \\ \text{or } &L = U_1 \Sigma_{n_s} \text{ and } R = V_1^T, \\ \text{or } &L = U_1 \Sigma_{n_s}^{\frac{1}{2}} \text{ and } R = \Sigma_{n_s}^{\frac{1}{2}} V_1^T, \\ \text{or } &L = U_1 W \text{ and } R = W^{-1} \Sigma_{n_s} V_1^T, \text{ for any invertible } W. \end{aligned}$$

The choice of factorization just corresponds to a *specific choice of coordinates* in which the state-space matrices are to be expressed. Whatever factorization is used, C is selected to be the first m rows of L and B is selected to be first q columns of R. To recover A, a new Hankel matrix is computed,

$$\tilde{M}_n = \begin{bmatrix} h[2] & h[3] & h[4] & \cdots & h[n+1] \\ h[3] & h[4] & h[5] & \cdots & h[n+2] \\ h[4] & h[5] & h[6] & \cdots & h[n+3] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h[n+1] & h[n+2] & h[n+3] & \cdots & h[2n] \end{bmatrix} = LAR.$$

Once L and R have been determined from M_n using the desired factorization, a left inverse of L, denoted $L^\dagger$, and a right inverse of R, denoted $R^\dagger$, are used to compute A in the same coordinates as C and B,

$$A = L^\dagger \tilde{M}_n R^\dagger.$$

The left and right inverses from the pseudo-inverse expressions can be used. For example, the SVD of M_n conveniently allows specification of left and right inverses using the matrices from the decomposition: if $L = U_1$ and $R = \Sigma_{n_s} V_1^T$, then $L^\dagger = U_1^T$ and $R^\dagger = V_1 \Sigma_{n_s}^{-1}$.

In practice, the Hankel matrix is always full rank when using real data so the one way to estimate a model is to analyze the singular values of M_n and make a judgement on a suitable model order. For

example, if there is a notable "jump" in the singular values (like in Fig. 6), then a sensible approach is to retain the largest singular values and truncate those after the "jump". Thus, although M_n is full rank, it is approximated by a rank n_s matrix, where n_s is the *selected* model order. In other words, n_s is not *a priori* specified –it is determined from analysis of the data, and once determined, M_n is replaced by the "optimal" rank n_s approximation

$$M_n \approx U_1 \Sigma_{n_s} V_1^*, \text{ where } \Sigma_{n_s} = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & & \\ \vdots & & \ddots & \\ 0 & & & \sigma_{n_s} \end{bmatrix}$$

where U_1 and V_1 represent the first n_s columns of U and V, respectively (where U and V are the unitary matrices from an SVD of M_n).

3.1 A note on the Gramian relations

Recall the observability and controllability gramians using the first n samples of the impulse response are defined as

$$G_o[n] = \sum_{k=1}^{n} (A^T)^{k-1} C^T C A^{k-1} = L^T L$$
$$G_c[n] = \sum_{k=1}^{n} (A)^{k-1} B B^T (A^T)^{k-1} = R R^T$$

If we make the following choices for L and R,

$$L = U_1 \Sigma_{n_s}^{\frac{1}{2}}, \quad R = \Sigma_{n_s}^{\frac{1}{2}} V_1^T$$

then the gramians are *balanced*, in other words, $G_o[n] = G_c[n] = \Sigma_{n_s}$, because

$$G_o[n] = L^T L = \left(U \Sigma_{n_s}^{\frac{1}{2}} \right)^T U \Sigma_{n_s}^{\frac{1}{2}} = \Sigma_{n_s}$$
$$G_c[n] = R R^T = \left(\Sigma_{n_s}^{\frac{1}{2}} V^T \right) \left(\Sigma_{n_s}^{\frac{1}{2}} V^T \right)^T = \Sigma_{n_s}$$

Now if the system is asymptotically stable we can take the limit $p \to \infty$, then

$$G_o[\infty] = \sum_{k=1}^{\infty} (A^T)^{k-1} C^T C A^{k-1} \tag{12}$$
$$G_c[\infty] = \sum_{k=1}^{\infty} (A)^{k-1} B B^T (A^T)^{k-1} \tag{13}$$

and these gramians satisfy the corresponding discrete-time Lyapunov equations,

$$A^T G_o[\infty] A - G_o[\infty] = -C^T C$$
$$A G_c[\infty] A^T - G_c[\infty] = -B B^T.$$

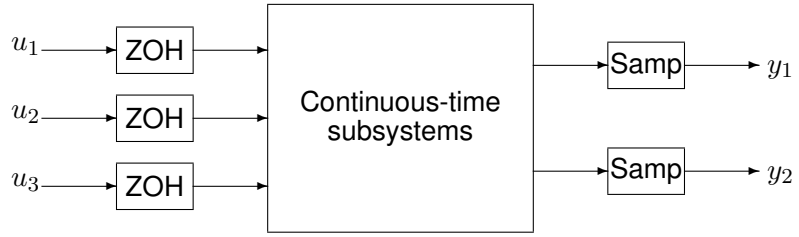

Figure 2: Block diagram associated with testing a connection of a continuous-time subsystems with three inputs and two outputs.

It's not possible to let $p \to \infty$ since only a finite amount is data is collected, however, if the Hankel matrix is constructed from essentially the entire transient pulse response data record, then for all practical purposes

$$G_o[n] \approx G_o[\infty] \text{ and } G_c[n] \approx G_c[\infty].$$

Thus, these coordinates represent the traditional "balanced realization," however, it is not necessary to decompose the Hankel matrices so that balanced coordinates are used.

4 Project Tasks

4.1 Background: Three-input/two-output system composed of subsystems

Input-output data obtained by testing a three-input/two-output system with broadband random test signals can be downloaded from the course website. The system is constructed from four single-input/single-output "subsystems": two resonators and two low-pass filters. The subsystems are connected in a certain topology. You will determine a state-space representation of a discrete-time model that very closely reproduces the pulse response measurements. Then, analysis of the pole-zero structure of the model transfer functions will allow you to determine the connection topology. Other interesting properties of the system will also be analyzed. The block diagram is shown in Fig. 2.

4.1.1 Response to random inputs

Data sets can be downloaded from the course website. The data sequences are associated with a sample period of $t_s = 1/50$ second. The data sets are named

```
random_u1.mat
random_u2.mat
random_u3.mat
```

It is necessary to discuss individual channels of input and output so the following notation is adopted. The kth input sample of $\mathbf{u}$ is denoted $\mathbf{u}[k]$ but since it is necessary to refer to the first, second or third channel of the input, the elements of $\mathbf{u}[k]$ are labeled as follows,

$$\mathbf{u}[k] = \begin{bmatrix} u_1[k] \\ u_2[k] \\ u_3[k] \end{bmatrix}$$

and similarly for the output,

$$\mathbf{y}[k] = \begin{bmatrix} y_1[k] \\ y_2[k] \end{bmatrix}.$$

The system is tested input-by-input so that is why three data sets are given. In the Matlab code shown below the vector `y11` represents a set of output samples $\{y_1[k]\}$, when $\{u_1[k]\}$ is white noise and $u_2 = u_3 = 0$. The input variable in Matlab is `u11`. Similarly, `y21` represents a set of output samples $\{y_2[k]\}$ for the same input. These data are stored in `random_u1.mat`. When the second input channel is used to test the system, in other words, $\{u_2[k]\}$ is white noise and $u_1 = u_3 = 0$, then the corresponding outputs $\{y_1[k]\}$ and $\{y_2[k]\}$ are represented by the Matlab variables `y12` and `y22`, respectively and the input by variable `u22`. These data are stored in `random_u2.mat`. Finally, when the third input channel is used to test the system, in other words, $\{u_3[k]\}$ is white noise and $u_1 = u_2 = 0$, then the corresponding outputs $\{y_1[k]\}$ and $\{y_2[k]\}$ are represented by the Matlab variables `y13` and `y23`, respectively, and the input by the variable by `u33`. These data are stored in `random_u3.mat`. It is possible to test the system in which all input channels are "on", however, the non-parametric spectral estimation techniques described in Sec. 2.2 require more test data to produce low-variance estimates because for a given input, the other two inputs are "disturbances" whose effects are only removed by longer averaging. The following Matlab code loads the input-output data and graphs it. Note that only one input channel is "on" in each data set. In this problem, $m = 3$ (two outputs) and $q = 2$ (two inputs). This Matlab code produces Figs. 3, 4 and 5.

```
% load response data
load random_u1.mat
y11 = y1;
y21 = y2;
u11 = u1;

load random_u2.mat
y12 = y1;
y22 = y2;
u22 = u2;

load random_u3.mat
y13 = y1;
y23 = y2;
u33 = u3;

Ndat = length(u11);
t = [0:Ndat-1]*ts;

ax = [0 10 -6 6];

figure(1)
subplot(311)
plot(t,u11)
title('u_1 input')
grid on; axis(ax);  legend('u_1')
subplot(312)
plot(t,y11)
grid on; axis(ax); legend('y_1')
subplot(313)
plot(t,y21)
```

```
grid on; axis(ax);  legend('y_2')
xlabel('Time (s)')


figure(2)
subplot(311)
plot(t,u22)
title('u_2 input')
grid on; axis(ax);  legend('u_2')
subplot(312)
plot(t,y12)
grid on; axis(ax); legend('y_1')
subplot(313)
plot(t,y22)
grid on; axis(ax);  legend('y_2')
xlabel('Time (s)')


figure(3)
subplot(311)
plot(t,u33)
title('u_3 input')
grid on; axis(ax);  legend('u_3')
subplot(312)
plot(t,y13)
grid on; axis(ax); legend('y_1')
subplot(313)
plot(t,y23)
grid on; axis(ax);  legend('y_2')
xlabel('Time (s)')
```

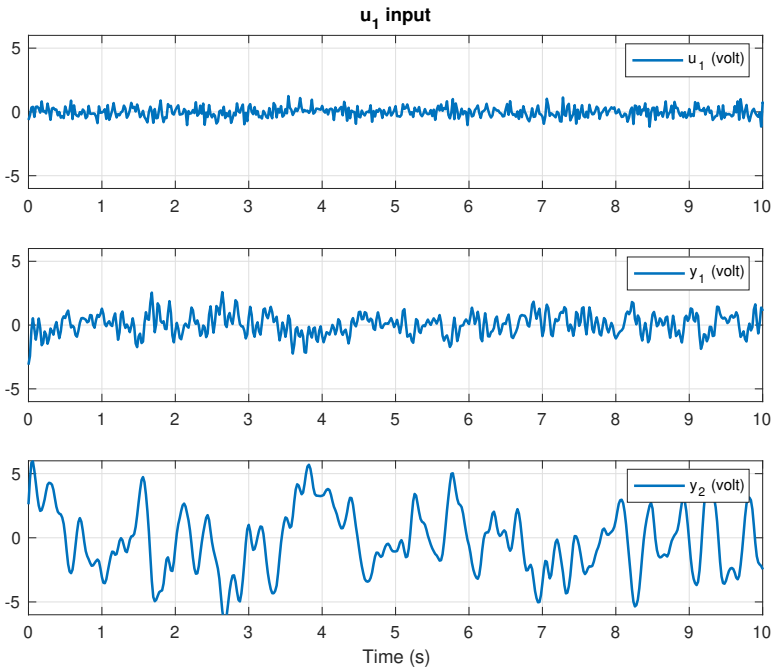

Figure 3: System response when u_1 is "white noise" and $u_2 = u3 = 0$. This is only a segment of the data -there is 250 seconds of data in each set. Also note that the data are given as individual points although Matlab is connecting points with lines.

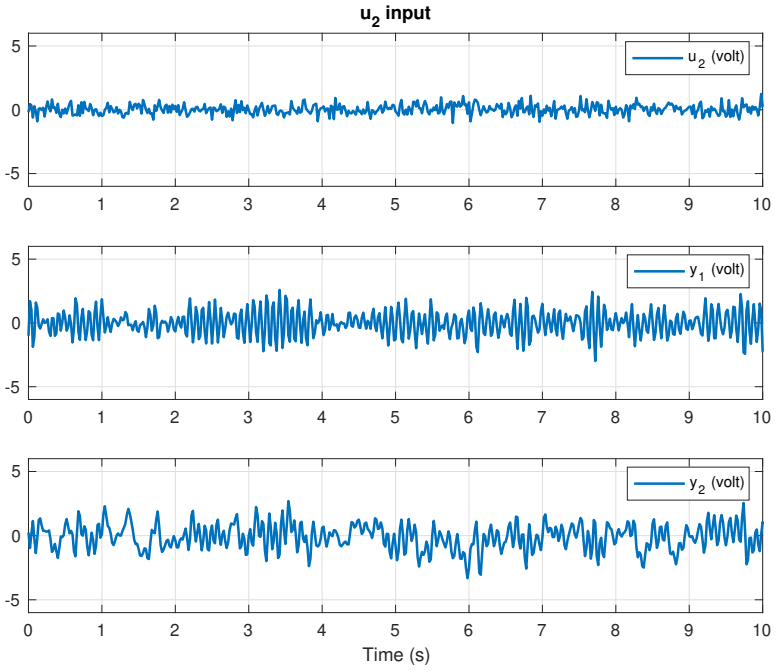

Figure 4: System response when u_2 is "white noise" and $u_1 = u_3 = 0$.

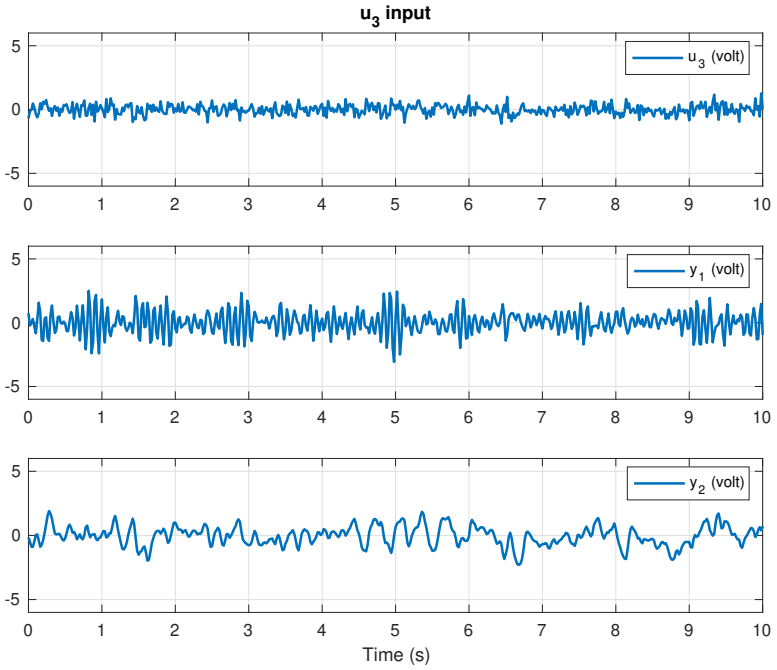

Figure 5: System response when u_3 is "white noise" and $u_1 = u_2 = 0$.

Task #1: Empirical frequency response estimates

Use the `cpsd` function in Matlab (see Sec. 2.2) to compute

- the auto-spectra of each input channel: $S_{u_1u_1}$, $S_{u_2u_2}$, $S_{u_3u_3}$
- the cross-spectra of each input channel: $S_{u_2u_1}$, $S_{u_3u_1}$, $S_{u_3u_2}$
- graph all six spectra in a log-log plot with `axis([0.1 fnyq 1e-5 1e-2])` and a frequency unit of hertz (the Nyquist frequency is denoted `fnyq` and the magnitudes of the cross-spectra are graphed since they are complex-valued)

Answer the following:

1. Show that the inputs are "white noise" because their auto-spectra are roughly constant over frequency.
2. Show that there is little correlation between the input signals because the magnitude of the cross-spectra are, on average, a magnitude smaller than the auto-spectra (more averaging, i.e. long data record, will further reduce the cross-spectra magnitudes)
3. Calculate the *mean square* value of each (time-domain) input sequence –these are the variances of the signals. Calculate the mean values of $S_{u_1u_1}$, $S_{u_2u_2}$, $S_{u_3u_3}$ and multiply by the signal sample rate of 50 Hz –these results should be very close to the variances obtained from the time-domain calculations.

4. Use the following notation for the requency response/transfer functions from individual input channels to individual output channels:

$$\begin{aligned} H_{11} &= \text{ output } y_1 \text{ due to input } u_1 \\ H_{21} &= \text{ output } y_2 \text{ due to input } u_1 \\ H_{12} &= \text{ output } y_1 \text{ due to input } u_2 \\ H_{22} &= \text{ output } y_2 \text{ due to input } u_2 \\ H_{13} &= \text{ output } y_1 \text{ due to input } u_3 \\ H_{23} &= \text{ output } y_2 \text{ due to input } u_3 \end{aligned}$$

   Estimate the frequency responses as follows:

$$\begin{aligned} H_{11} &= S_{y_1u_1}/S_{u_1u_1} \\ H_{21} &= S_{y_2u_1}/S_{u_1u_1} \\ H_{12} &= S_{y_1u_2}/S_{u_2u_2} \\ H_{22} &= S_{y_2u_2}/S_{u_2u_2} \\ H_{13} &= S_{y_1u_3}/S_{u_3u_3} \\ H_{23} &= S_{y_2u_3}/S_{u_3u_3} \end{aligned}$$

   Graph the magnitude of H_{11} and H_{21} in a single figure (all frequency response graphs should have magnitudes on log-log axes, frequency unit of hertz and `axis([0.1 fnyq 1e-3 1e2])`). Graph the phase of H_{11} and H_{21} in a single figure (all phase graphs should have linear-log axes, frequency unit of hertz and `axis([0.1 fnyq -200 200])`). These figures show how the system outputs respond to u_1.

   In another two figures, graph the magnitude and phase of H_{12} and H_{22}. These figures show how the system outputs respond to u_2. Finally, graph the magnitude and phase of H_{13} and H_{23} in a final set of figures. There should be a total of three figures showing the frequency response magnitudes and three figures showing the phases. These represent *empirical frequency response estimates* of the system.

Task #2: Pulse response estimates

The pulse responses of the system are simple to compute from the frequency response estimates. Pulse responses and frequency frequency responses are Fourier transform pairs, thus, the pulse response is estimated by computing the inverse Fourier transform of the corresponding frequency response. For discrete-time systems, the inverse *fast Fourier transform* algorithm can be applied to the frequency response estimates to generate pulse response estimates. The function to use in Matlab for this is `ifft`. It is necessary to create a time vector for the pulse response estimates. The pulse response estimate has exactly the same number of points as the frequency response from which it was derived -this is the `nfft` variable from Sec. 2.2. Since the first point in the pulse response estimate corresponds to $t = 0$ (when the pulse is applied), the time vector is computed as `t = [0:nfft-1]*ts;`.

Let the pulse response corresponding to H_{11} be denoted h_{11} and so forth. Create the following figures for the report:

- Graph h_{11} and h_{21} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph.
- Graph h_{12} and h_{22} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph.
- Graph h_{13} and h_{23} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph.

The pulse responses are used to estimate a parametric model in the next section. The pulse response of the MIMO system is denoted by h, in other words,

$$h = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \end{bmatrix}$$

Task #3: Hankel matrix analysis and parametric model

The state dimension is not known, however, one objective of the project is to the determine a state-space discrete-time system that can replicate the observed data. Only asymptotically stable models are considered since it clear from the test data that the system we are testing must be asymptotically stable. Also note that when using data sets from physical systems that M_n is invariably full rank because there is some noise in the measurement data and furthermore the system may be nonlinear even though it can be well-approximated with a linear model. Thus, although it appears that an n-dimensional model is required a lower rank approximation almost always produces a "better" model. For example, asymptotic stability of the identified model is not explicitly enforced and using high model dimensions can actually produce unstable models.

At the kth sample time, $\mathbf{y}[k]$ data in Fig. 3 form the first column of $h[k] \in \mathbf{R}^{2\times 3}$. Similarly, $\mathbf{y}[k]$ from Fig. 4 form the second column of $h[k]$ and $\mathbf{y}[k]$ from Fig. 5 form the third column of $h[k]$. The singular values of M_n, $n \in \{20, 40, 60, 80\}$, are shown in Fig. 6 where there appear to be 8 dominant singular values regardless of the dimension of M_n This suggests that a reasonable initial choice of model order is $n_s = 8$. You will select a handful of model orders and derive the corresponding state-space models from the Hankel matrix factorization and then compare your models in a few different ways.

1. Construct $M_{25} \in \mathbf{R}^{50\times 75}$ and graph its singular values (use the same axis type and range as Fig. 6). Compute models from M_{25} with the following state dimensions: $n_s \in \{7, 8, 10, 16\}$. There is no need to print out the state-space matrices since you will compute the model properties below. Calculate the `max(abs(eig(A)))` to confirm that all models of order $\{7, 8, 10\}$ are asymptotically stable but that order 16 is unstable (recall this is a discrete-time system so the eigenvalue condition is different than for a continuous-time system).

2. Simulate the impulse response of each model with order $\{7, 8, 10\}$ and compare it to the measurement data. For *each* of the model orders create the following figures

   - Graph h_{11} and h_{21} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph. Show the model result against the pulse response estimate.

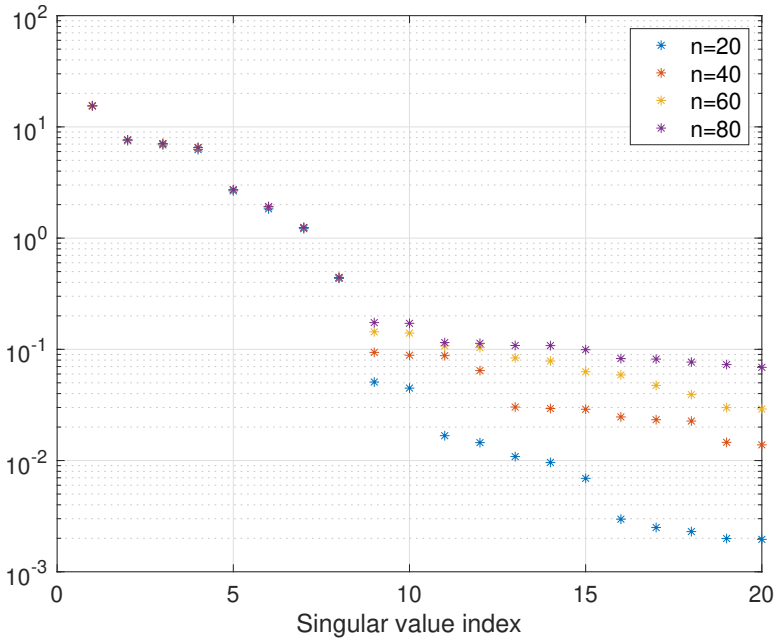

Figure 6: Singular values of M_{20} (blue), M_{40} (red), M_{60} (yellow) and M_{80} (purple). The first 20 singular values are shown for each case.

- Graph h_{12} and h_{22} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph. Show the model result against the pulse response estimate.
- Graph h_{13} and h_{23} in a single figure with `axis([0 3 -2 3])` and using the `subplot` feature so that each signal is shown in a separate part of the graph. Show the model result against the pulse response estimate.

Which model order has an inferior reproduction of the pulse response data? The remaining models are essentially indistinguishable, though, and could be used as a valid model for the system. Note that even if the physical system is asymptotically stable, there is no guarantee that the model produced from this process will be asymptotically stable (as shown by model order 16).

3. Another way to compare the model to the data is to compare the model frequency response to the empirical frequency response obtained from the pulse response data. The model frequency response is given by

$$C(e^{j\omega t_s}I - A)^{-1}B + D,$$

where $t_s = 1/50$ is the sample period in seconds. The frequency ω is in the interval $[0, \omega_{nyq}]$, where ω_{nyq} is the Nyquist frequency (equal to one half the sampling frequency, in other words, $\omega_{nyq} = 2\pi 25$ rad/s). Note that when evaluating the formula, ω should have units of radians-per-second, however, when graphing use the frequency unit of **hertz**. For *each* of the model orders create the following figures

- Graph the magnitude of H_{11} and H_{21} in a single figure with `axis([0.1 fnyq 1e-3 1e2])`. Compare the empirical frequency response to the model frequency response.

- Graph the phase of H_{11} and H_{21} in a single figure with `axis([0.1 fnyq -200 200])`. Compare the empirical frequency response to the model frequency response.
- Graph the magnitude of H_{12} and H_{22} in a single figure with `axis([0.1 fnyq 1e-3 1e2])`. Compare the empirical frequency response to the model frequency response.
- Graph the phase of H_{12} and H_{22} in a single figure with `axis([0.1 fnyq -200 200])`. Compare the empirical frequency response to the model frequency response.
- Graph the magnitude of H_{13} and H_{23} in a single figure with `axis([0.1 fnyq 1e-3 1e2])`. Compare the empirical frequency response to the model frequency response.
- Graph the phase of H_{13} and H_{23} in a single figure with `axis([0.1 fnyq -200 200])`. Compare the empirical frequency response to the model frequency response.

From the frequency response perspective, which model order has an inferior reproduction of the empirical frequency response? Which model orders yield almost indistinguishable frequency responses compared to the empirical data?

Task #4: Transmission zeros of individual I/O channels

The $n_s = \{8, 10\}$ models provide very accurate reproduction of the pulse response and frequency response compared to the data. The transmission zeros of each of the six scalar transfer functions are considered for these models. A transmission zero for a SISO system occurs at $z \in \mathbf{C}$ when there exist a vector $\mathbf{x}_0 \neq 0$ and a scalar $w \neq 0$ such that

$$\begin{bmatrix} zI - A & -B \\ C & D \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ w \end{bmatrix} = 0 \tag{14}$$

Let the input be $u[k] = wz^k$, $k = 0, 1, 2, \ldots$ (the input is scalar-valued because the SISO case is considered here). The state vector $\mathbf{x}_k = \mathbf{x}_0 z^k$ satisfies the difference equation with this input because

$$\mathbf{x}_0 z^{k+1} = A \underbrace{\mathbf{x}_0 z^k}_{\mathbf{x}[k]} + B \underbrace{x z^k}_{u[k]} \implies (zI - A)\mathbf{x}_0 - Bw = 0.$$

Furthermore, the output is zero for all samples because

$$\mathbf{y}[k] = C\mathbf{x}_0 z^k + Dwz^k = \begin{bmatrix} C & D \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ w \end{bmatrix} z^k = 0.$$

The zeros can be computed from a generalized eigenvalue problem,

$$\begin{bmatrix} A & B \\ -C & -D \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ w \end{bmatrix} = z \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ w \end{bmatrix}.$$

See the Matlab `eig` command for more information on solving this generalized eigenvalue problem. Answer the following:

1. In a figure graph the eigenvalues of the $n_s = 8$ model in the complex plane (refer to the `real` and `imag` command in Matlab). Denote the eigenvalues with an "`x`". Draw a circle of radius 1 (use the Matlab command `axis square` so the aspect ratio is one-to-one) and note that if your model is asymptotically stable (it should be) then the eigenvalues will lie within the unit circle.

2. Graph the eigenvalues of the $n_s = 10$ model in a new figure. Where are the extra eigenvalues located in the 10 state model?

3. For the $n_s = 8$ model, create six separate figures showing the eigenvalues (marked with "x" like in the previous figures) and the transmission zeros (marked with "o") calculated for *each individual channel*. The `inf` zeros can be ignored. Note that the eigenvalues will be the same in all cases, however, the transfer functions in each channel are different because the zeros are different. If a zero is in close proximity to an eigenvalue in a SISO transfer function then that eigenvalue is not a pole of the transfer function (the zero and eigenvalue effectively "cancel" each other). Carefully study each figure and note that although some zeros and poles cancel each other in certain channels, taken collectively, each eigenvalue will appear as a pole in at least one of the six SISO transfer functions.

4. Now consider the $n_s = 10$ model. This model is over-parameterized in some sense because its pulse response and frequency response are essentially identical to those associated with the $n_s = 8$ model yet it has two more states. Create another six figures of the eigenvalue and zero plots for each channel and show that the added eigenvalues are `always` accompanied by zeros in close proximity and that this occurs for all six channels. This creates an eigenvalue-zero cancellation in each channel so that the extra two eigenvalues in the $n_s = 10$ case do not have any impact on the input-output properties of the system compared to the $n_s = 8$ model. Thus, the $n_s = 8$ model can be considered a more efficient representation of the system given the available data.

Task #5: Block diagram derived from analysis of SISO channels

The test data were generated by connecting four subsystems in a certain topology. Each although each subsystem is a circuit with many components, its dominant behavior is governed by a two state model. This is why $n_s \geq 8$ is required for an accurate representation of the input-output properties of the 3-input/2-output system. The subsystems consist of two resonator circuits and two low-pass filter circuits.

The $n_s = 8$ model is used for the analysis in this task. A discrete-time eigenvalue λ_d can be converted into its "equivalent" continuous-time eigenvalues according to $\lambda_d = e^{\lambda_c t_s}$, where λ_c is the continuous-time eigenvalue. This is due to the fact that the "A" matrix for a sampled linear continuous-time system is given by $e^{A_c t_s}$ –see (7). Note that imaginary part of λ_c is not uniquely defined, so just choose the smallest value for the imaginary part of λ_c (this choice is justified as long as there is no aliasing of resonant frequencies during the data collection). **Make a table showing the eigenvalues of the discrete-time model and their corresponding continuous-time equivalents.** From the set of λ_c, show that there are two lightly damped resonators are in your model. What are their approximate natural frequencies in hertz? How do the frequencies compare with certain features in frequency response graphs?

The eigenvalues of each resonator are distinct (despite the fact that they are quite close) so give each complex conjugate pair of resonator eigenvalues a distinct label: "RES1" for the resonator with the slightly higher natural frequency and "RES2" for the other resonator. In addition to the two resonators, there are also two pairs of complex-conjugate eigenvalues. Each complex-conjugate pair is associated with a low-pass filter. Denote the more highly damped pair by "LP1" for "low-pass 1", and the other by "LP2".

You can use the following information to determine the connection topology of RES1, RES2, LP1 and LP2:

- Study which eigenvalues are not canceled by transmission zeros in the individual channels input-output channels (therefore, the subsystems with those eigenvalues as poles must be in the path from that given input to output).
- The low-frequency asymptotes of the frequency response magnitudes RES1 and RES2 are both "1". This information is required since two systems in a given input-output channel may be connected in series or in parallel (you need to justify which connection you make is the correct choice).
- Only three summing junctions are required to connect the subsystems.

Remark: The block diagram should consist of only the four blocks shown below and summing junctions. Each block is a single-input/single-output system so there should be only one arrow coming into, or leaving from, a block. In other words, arrange the blocks shown below with three summing junctions to "fill in" the "Continuous-time subsystems" block in Fig. 2 (the number of eigenvalues of each block are shown below for clarity but this can be omitted from your final block diagram):