Reconstructing continuous-time descriptions of a two-input/two-output linear dynamic system from impulse response data

Camila Correa-Jullian[1], Arturo Flores Alvarez[1]

[1] Department of Mechanical and Aerospace Engineering, University of California, Los Angeles (UCLA),

Abstract

This manuscript presents the reconstruction of a state-space model from the pulse response of the system, which consists of two-input/two-output channels with 1 bandpass filter, 2 resonators, and 1 unknown system. Moreover, relying on a thorough analysis of the frequency response and pole-zeros plots of the input/output channels, we identified the connection topology of the elements of the system. Finally, the bandpass filter's frequency response is studied using the Nyquist–Shannon sampling theorem.

1 Problem statement

1.1 Objectives & Approach

The general objectives of the project are summarized as follows:

- Learn about sample-and-hold models of linear systems.
- Extracting models of discrete-time state-space system from pulse response data.
- Application to a two-input/two-output physical system (various circuit "subsystems" connected in a topology) to determine continuous-time descriptions of the subsystems.

To achieve this goal, the following tasks have been set:

- Task 1: Model identification
- Task 2: Transmission zeros of the MIMO model and individual I/O channels
- Task 3: Block diagram derived from analysis of SISO channels
- Task 4: Continuous-time model of the bandpass subsystem

The remainder of the document consists of Section (2) where each task's results are described and discussed. Brief conclusions are presented in Section (3). Additionally, the referenced code is available in the Appendix (4). Here, specific functions are presented in (4.1), plotting functions in (4.2), and the main body of the code in (4.3).

2 Results and Analysis

In this section, we will present the plots obtained on each task and we will discuss their importance.

2.1 Task 1: Model Identification

The objective of this task is to perform a series of analyses to describe the system from the impulse response data using a Henkel matrix approach. For this purpose, the following steps are performed

1. Task 1.1: Analyze the resulting Hankel matrix for different state dimensions to perform a time-domain identification of the system. This analysis consists of the following steps:
    - Construct H_{100}. This is implemented through the function *HankelMatrix* described in Listing 1. To obtain the non-singular values, an SVD for H_{100} is performed. Figure (1) presents the first 40 non-singular values of this Hankel matrix. This first result is coherent with the results of Figure 5 in the project prompt and shows that there are seven dominant singular values. This information is later used to define

the optimal dimension of the realization for our reconstructed state-space model.

- Obtain the H_{100} for each $n_s = \{6, 7, 10, 20\}$ state dimension. This is implemented through the function *ModelReduction* in Listing 2. This function performs an SVD of the H_{100} matrix and then selects the blocks corresponding to each state dimension. It also returns the reconstructed Hankel matrix for each dimension.
- For each Hankel matrix, obtain the Controllability $\mathcal{C}$ and Observability O matrices for each state dimension n_s. This is performed through the *SquaredFactorization* function described in Listing 3. This function receives the SVD decomposition for each matrix and implements Equation (1) below.

$$O_n = U_1 \Sigma_{n_s}^{1/2} \quad \mathcal{C}_n = \Sigma_{n_s}^{1/2} V_1^T \tag{1}$$

- Obtain the matrices B, C for each state dimension n_s. These are obtained as the first n_s columns and rows, respectively, corresponding to each $\mathcal{C}$ and O matrix computed in the previous step. Refer to Listing 4 and the function *BC Extraction*.
- Compute the corresponding A matrix for each state dimension using equation 2. Here, $O^\dagger = (O^* O)^{-1} O^*$ is the pseudo left inverse of O, $\mathcal{C}^\dagger = \mathcal{C}^* (\mathcal{C}\mathcal{C}^*)^{-1}$ is the pseudo right inverse of $\mathcal{C}$, and $\tilde{H}$ is the shifted Hankel matrix.

$$A = O^\dagger \tilde{H} C^\dagger \tag{2}$$

- Verify that systems are asymptotically stable. Note that for discrete systems, this requires $|\lambda_k| < 1, \forall \lambda_k(A_s)$. Our result for the command *max(abs(eig(A)))* is 0.9137 then, we confirm that the system is asymptotically stable. This command is implemented in lines 148-153 of Listing 15.

2. Task 1.2: Compare measurement data to impulse response simulated from different state dimensions. In-built Matlab functions (*ss*, *impulse*) are used to simulate the impulse response (refer to lines 162-165 in Listing 15. Figures (2) and (3) provide plots for the impulse response of the two input channels u_1, u_2 simulated for each model of dimension n_s overlayed on the measured impulse response. From Figure (2), it can be observed that for $n_s = 6$, the simulated impulse response does not replicate the measured data provided. However, for $n_s \geq 7$, the simulated impulse response is an adequate representation of the measured data. From Figure (3), we see that the impulse for $n_s = 10, 20$ is pretty similar to the response of $n_s = 7$, then the model $n_s = 7$ can be regarded as the most optimal among all of the models in representing the system's dynamics.

3. Task 1.3-1.4: Compare the magnitude and phase of the empirical frequency response for each state dimension. The model frequency response is obtained through Equation (3), where $t_s = 1/40$ is the sample period in seconds and $\omega \varepsilon [0, \omega_{nyq}]$ is a frequency bounded by the Nyquist frequency $\omega_{nyq} = 2\pi 20 \quad rad/s$.

$$H(j\omega) = C(e^{j\omega t_s} I - A)^{-1} B + D \tag{3}$$

Figures (4) and (5) provide plots for the magnitude and phase of the empirical impulse response for each reduced state dimension model. These are overlayed over the estimated empirical frequency response calculated from the measured data. These figures provide additional arguments to define that a state-space representation of $n_s = 7$ is sufficient to replicate the measured data.

Refer to Listing 15 for details about the implementation of this task.

2.2 Task 2: Transmission zeros of the MIMO model and individual I/O channels

The objective of this task is to analyze the poles and transmission zeros of the two-input/two-output model and individual I/O channels. For this task, only the model corresponding to the state dimension $n_s = 7$ is considered. The following steps are implemented:

1. Task 2.1-2.2 Analyze the model's transmission zeros and visualize the relation with the system's eigenvalues. A system's transmission zeros can

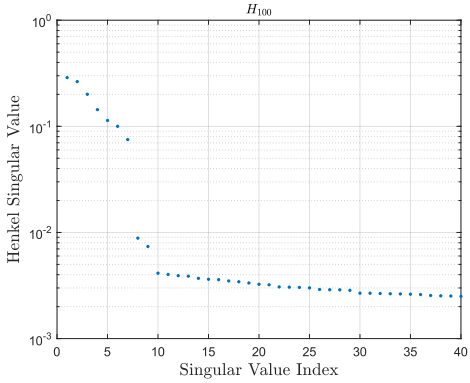
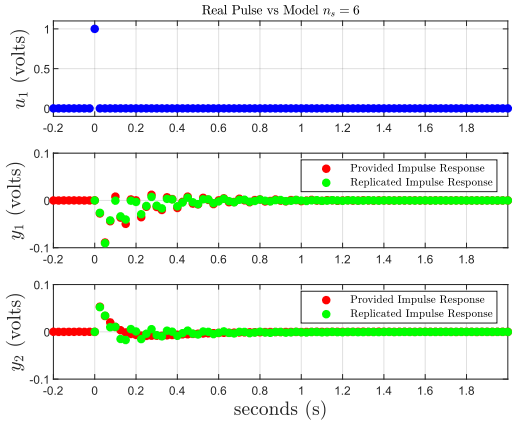
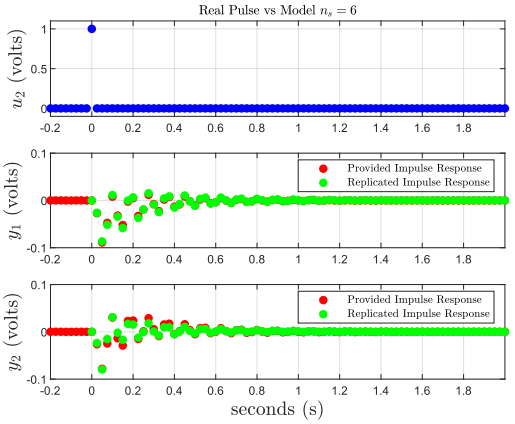


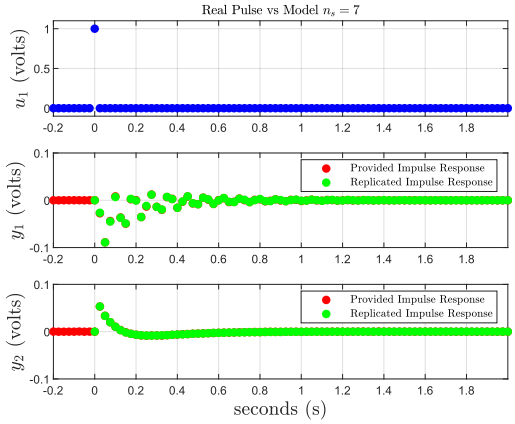
Figure 1: *Task 1.1: Hankel Singular Values.*



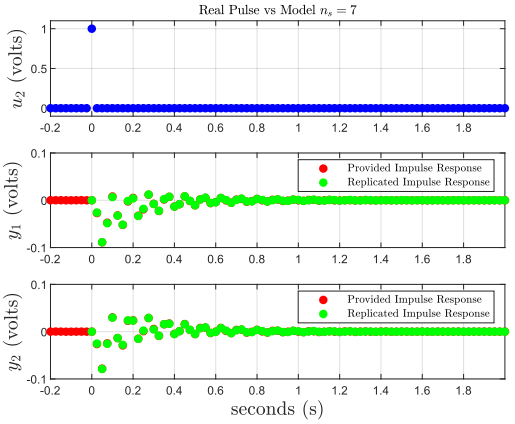
(a) *Input Channel 1, Model dimension $n_s = 6$.*



(b) *Input Channel 2, Model dimension $n_s = 6$.*



(c) *Input Channel 1, Model dimension $n_s = 7$.*

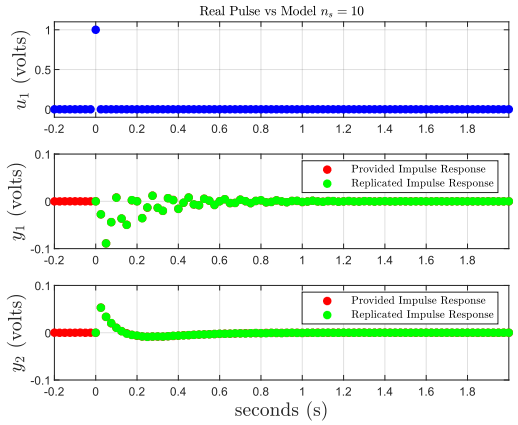


(d) *Input Channel 2, Model dimension $n_s = 7$.*

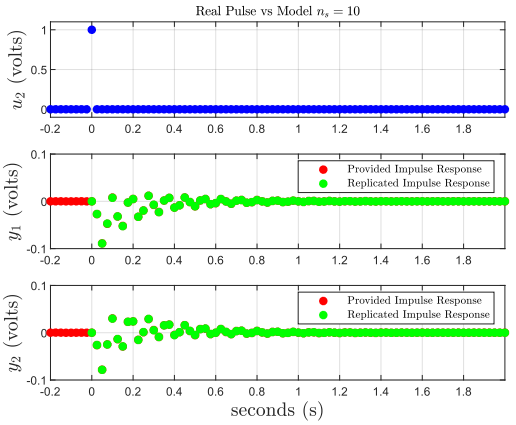
Figure 2: *Comparison of measured data and simulated impulse response for different state dimensions (to be continued).*

be computed from the Generalized Eigenvalue Problem, as stated in Equation (4). The values of $\{z, \mathbf{w}, \mathbf{c}\}$ are derived from equation (5).

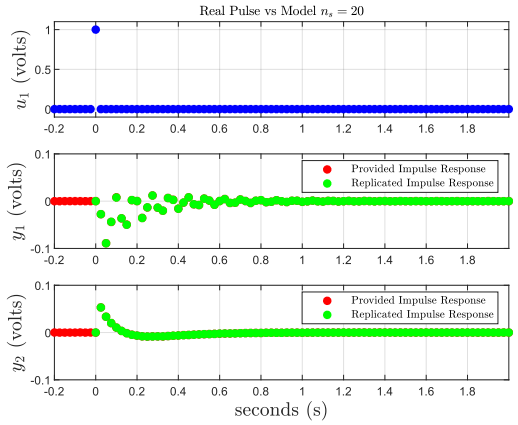
$$\begin{bmatrix} A & B \\ -C & -D \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{w} \end{bmatrix} = z \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{w} \end{bmatrix} \tag{4}$$



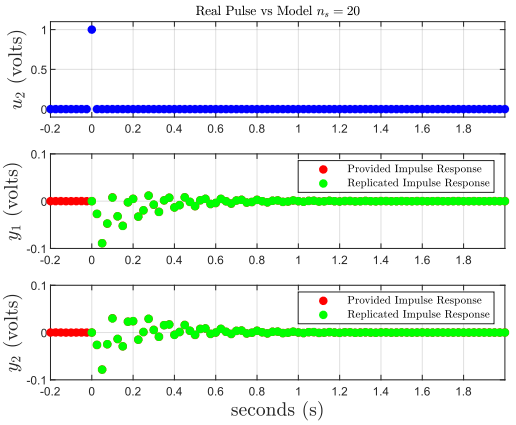
(a) *Input Channel 1, Model dimension $n_s = 10$.*



(b) *Input Channel 2, Model dimension $n_s = 10$.*



(c) *Input Channel 1, Model dimension $n_s = 20$.*



(d) *Input Channel 2, Model dimension $n_s = 20$.*

Figure 3: *Comparison of measured data and simulated impulse response for different state dimensions (continued).*

$$S(z) = \begin{bmatrix} \mathbf{c} \\ \mathbf{w} \end{bmatrix} = \begin{bmatrix} zI - A & -B \\ C & D \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{w} \end{bmatrix} = 0 \quad (5)$$

Table 1 shows the obtained transmission zeros z for the $n_s = 7$ model. Note that $|z_1| > 1$ is the unstable zero, while all the others are asymptotically stable zeros. Figure (6) plots the location of each transmission zero and pole $\lambda(A)$ (eigenvalue) relative to a unit circle. Note that the model is asymptotically stable, as all $\lambda(A)$ lie within the unit circle (black dashed line).

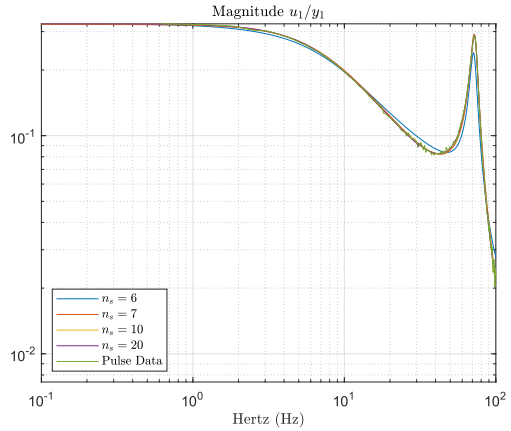
2. Task 2.3 Simulate the two-input/two-output model using $x_0 = \mathbf{c}$ as the initial condition and $\mathbf{u}[k] = \mathbf{w}z^k$ as input. The simulated output is expected to be zero for all samples as described by Equation (6).

Table 1: *System poles and transmission zeros*

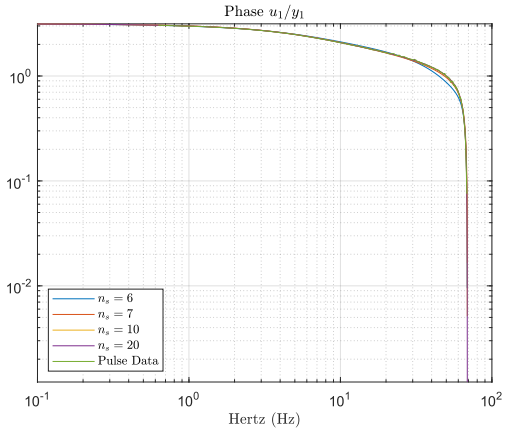
Poles λ		Transmission zeros z	
λ_1	-0.1881 + 0.8924i	z_5	-0.3241 + 0.0000i
λ_2	-0.1881 - 0.8924i	z_4	0.6078 - 0.6740i
λ_3	-0.2138 + 0.8890i	z_3	-0.6078 + 0.6740i
λ_4	-0.2138 - 0.8890i	z_2	0.9988 + 0.0000i
λ_5	0.7699 + 0.0000i	z_1	-2.6310 + 0.0000i
λ_6	0.8258 + 0.0000i	-	
λ_7	0.8686 + 0.0000i	-	

$$\mathbf{y}_k = C\mathbf{c}z^k + D\mathbf{w}z^k = \begin{bmatrix} C & D \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{w} \end{bmatrix} z^k = 0 \quad (6)$$

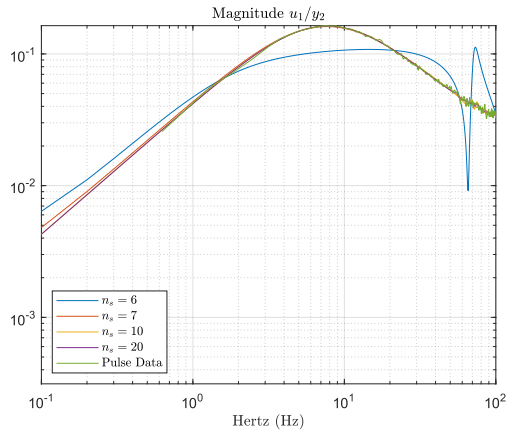
Figure (7) provides the elements of the simu-



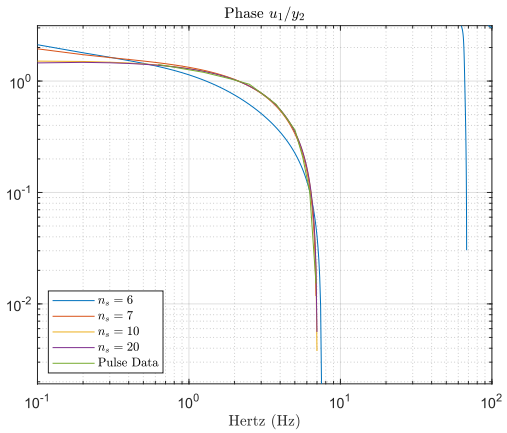
(a) *Empirical magnitude, Model dimension* $n_s = 6$.



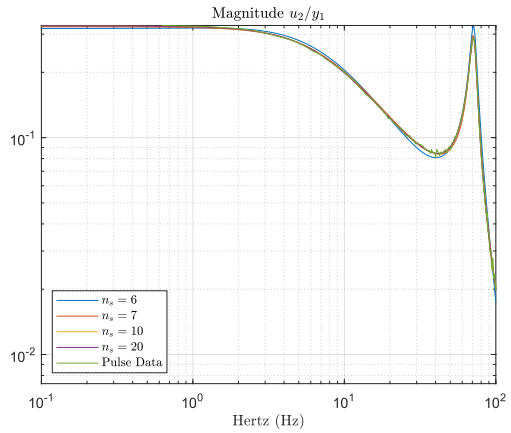
(b) *Empirical phase, Model dimension* $n_s = 6$.



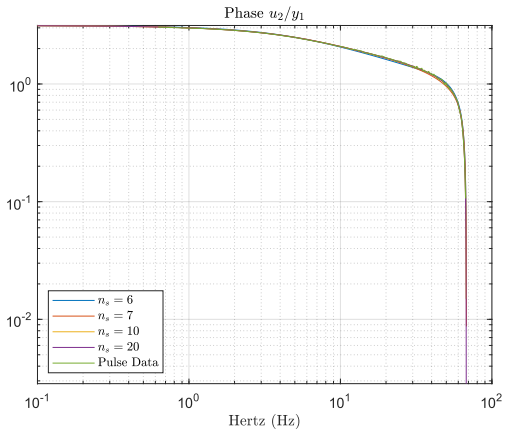
(c) *Empirical magnitude, Model dimension* $n_s = 7$.



(d) *Empirical phase, Model dimension* $n_s = 7$.



(e) *Empirical magnitude, Model dimension* $n_s = 10$.

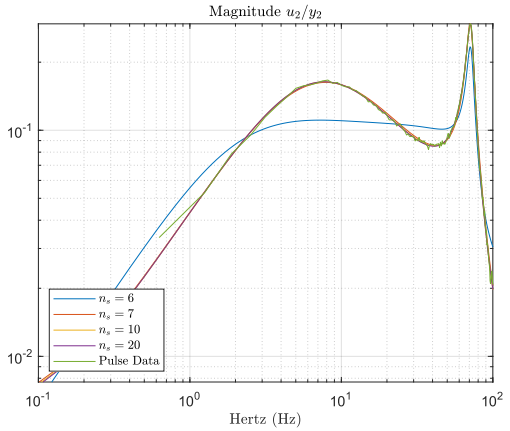


(f) *Empirical phase, Model dimension* $n_s = 10$.

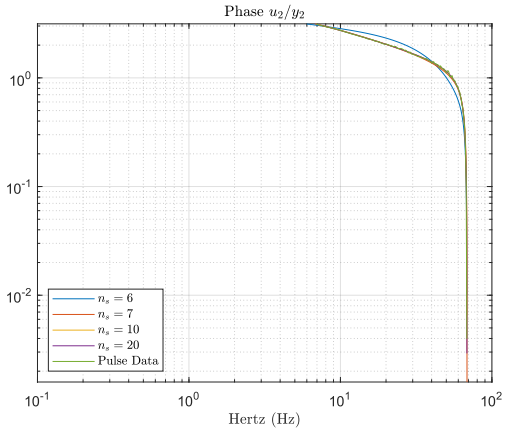
Figure 4: *Empirical frequency response for each reduced state dimension model (to be continued).*

lated input sequence and corresponding output sequence. Note that 7b shows that the output sequence is of the order of 10^{-16} and validates the fact that for a specific initial condition and an input, we can create a zero output response.

3. Task 2.4 Visualize the relation between the simulated system's eigenvalues and transmission zeros



(a) *Empirical magnitude, Model dimension $n_s = 20$.*



(b) *Empirical phase, Model dimension $n_s = 20$.*

Figure 5: *Empirical frequency response for each reduced state dimension model (continued).*

for each channel. Figure (8) provides the individual frequency responses and pole-zero plots for each input/output channel combination. Likewise, we will provide the Henkel matrix for each channel and analyze their results. For the u_1/y_1 we have the Figure (8c) where there are 3 poles, and this Henkel matrix

$$H_{11} = \begin{bmatrix} 0.1741 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1167 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0910 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 \end{bmatrix} \tag{7}$$

where we can see that there are 3 dominant singular values. For the u_1/y_2 we have the Figure (8g) where there are 2 poles, and this Henkel matrix

$$H_{21} = \begin{bmatrix} 0.0830 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0352 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0008 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0005 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0001 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0001 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 \end{bmatrix} \tag{8}$$

where we can see that there are 2 dominant singular values. For the u_2/y_1 we have the Figure (8d) where there are 3 poles, and this Henkel matrix

$$H_{12} = \begin{bmatrix} 0.1764 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1161 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0919 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 \end{bmatrix} \tag{9}$$

where we can see that there are 3 dominant singular values. For the u_2/y_2 we have the Figure (9d) where there are 4 poles, and this Henkel matrix

$$H_{22} = \begin{bmatrix} 0.1325 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1180 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0527 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0316 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 \end{bmatrix} \tag{10}$$

where we can see that there are 4 dominant singular values. Notice that there are very small singular values in each matrix, however, these are neglectable because the data we are using is from a real system, which is affected by noise. Therefore, the pole-zero cancellation is not perfectly done.

4. Task 2.5 Compare the relationship between the system's eigenvalues and transmission zeros for $n_s = 8$. Figure (9) provides the individual frequency responses and pole-zero plots for each input/output channel combination for this system state-space representation. As it can be observed in Figures (9a) and (9b), the magnitude of the frequency response of the $n_s = 8$ model is similar to that of the model $n_s = 7$. Indeed, minor discrepancies may be observed at the lower and higher end of the frequencies. Additionally, Figures (9c) and (9d) show that by increasing the system's dimension, the added poles are in close proximity to a zero. Hence, it can be determined that a higher state-space representation $n_s > 7$ may not greatly influence the I/O properties of the system when compared to the $n_s = 7$ model

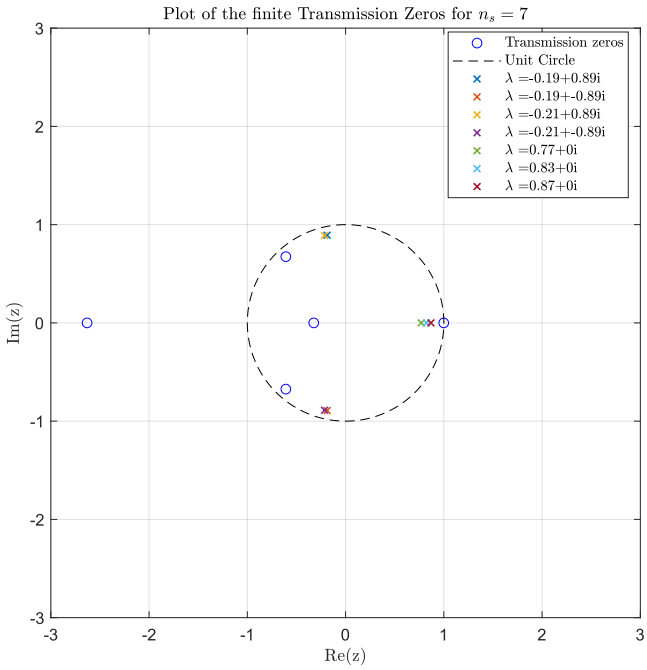
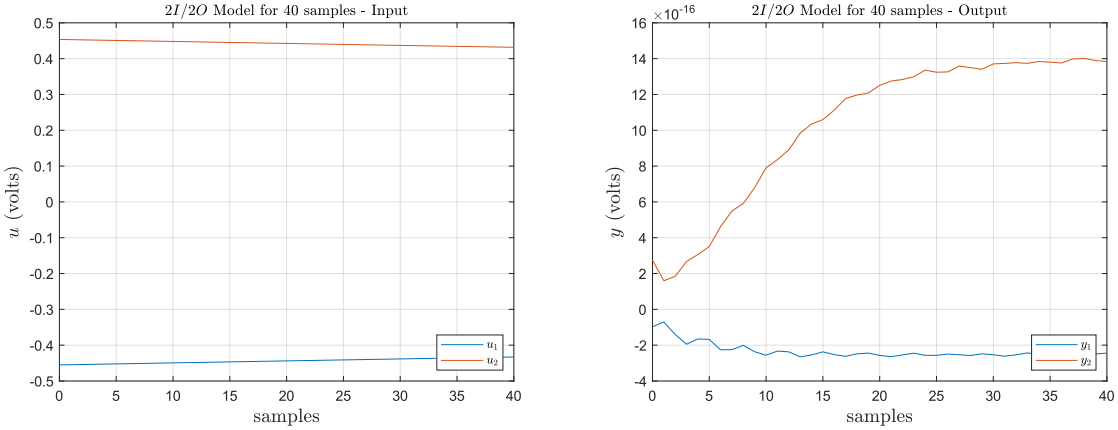


Figure 6: *Pole-Zero visualization for $n_s = 7$.*



(a) *Simulated input sequence.* ***(b)*** *Simulated output sequence.*

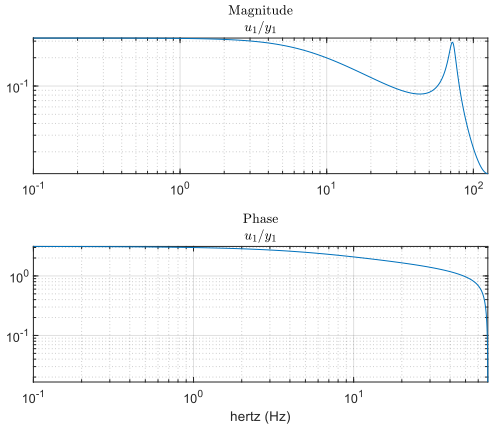
Figure 7: *Simulated input/output sequence at transmission zero $z \sim 1$.*

Refer to Listing 16 for details about the implementation of this task.

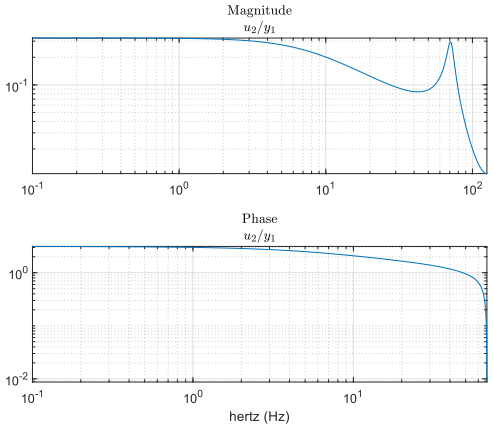
2.3 Task 3: Block diagram derived from analysis of SISO channels

Based on the model for $n_s = 7$, the objective of this task is to determine the physical structure that produced the measured data. For this purpose, these steps are followed

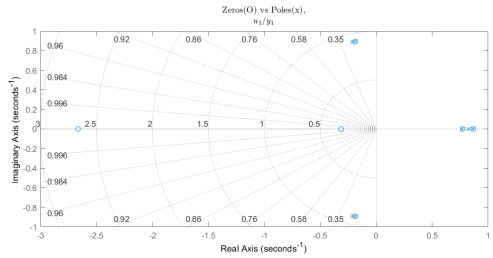
1. Compute continuous-time (CT) eigenvalues for the system $n_s = 7$. The CT eigenvalues are computed as described in Equation (11), where λ_d, λ_c represent the discrete-time (DT) and CT eigenvalues, respectively (refer to function *d2c pole* in Listing 7. Note that the representation of λ_c is not unique, so the smallest value for $\mathbf{Im}(\lambda_c)$ is



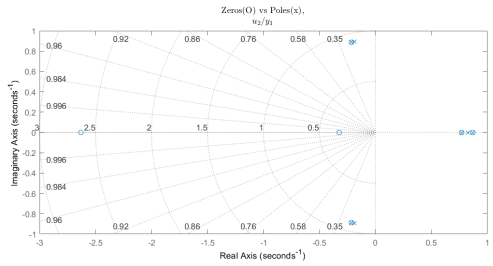
(a) *Frequency response for: Input 1, Output 1.*



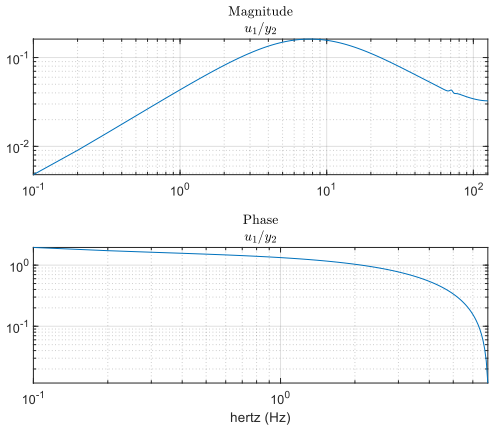
(b) *Frequency response for: Input 2, Output 1.*



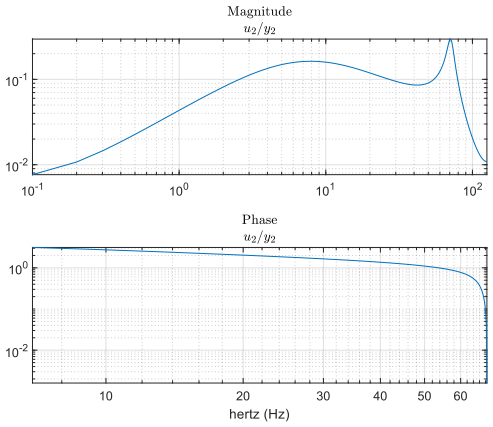
(c) *Pole-Zero plot for: Input 1, Output 1.*



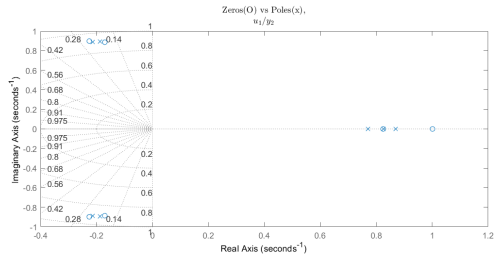
(d) *Pole-Zero plot for: Input 2, Output 1.*



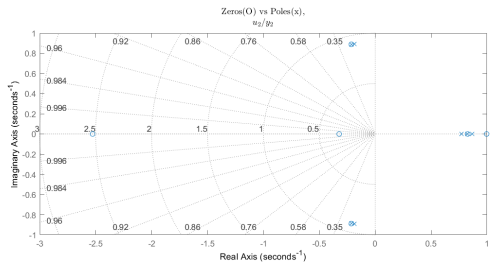
(e) *Frequency response for: Input 1, Output 2.*



(f) *Frequency response for: Input 2, Output 2.*

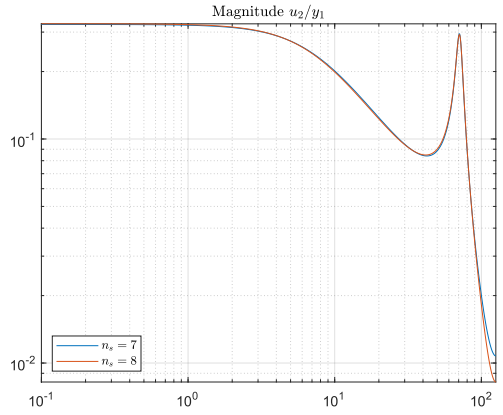


(g) *Pole-Zero plot for: Input 1, Output 2.*

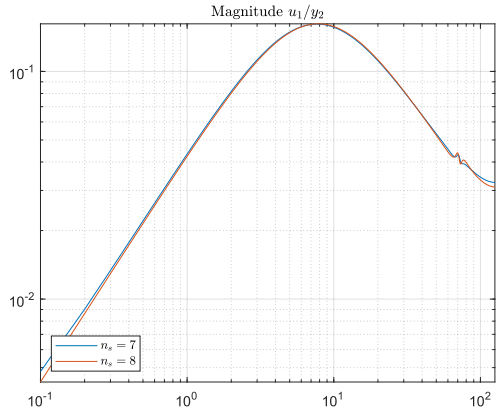


(h) *Pole-Zero plot for: Input 2, Output 2.*

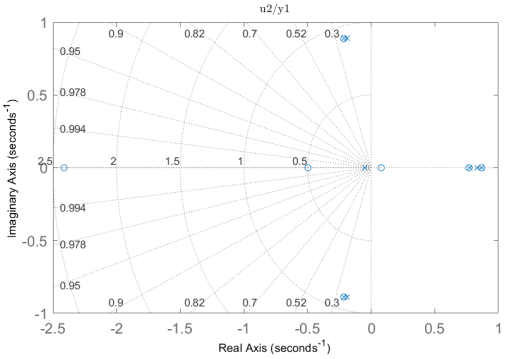
Figure 8: *Frequency response and pole-zero plots for each input/output channel.*



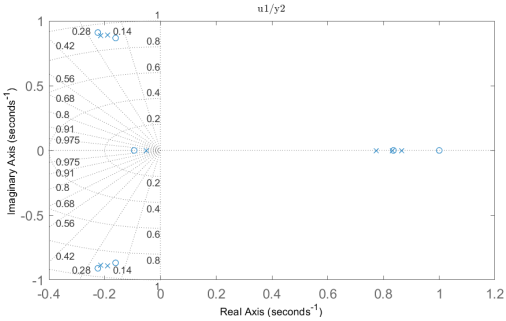
(a) *Frequency response for: Input 2, Output 1.*



(b) *Frequency response for: Input 1, Output 2.*



(c) *Pole-Zero plot for: Input 2, Output 1.*



(d) *Pole-Zero plot for: Input 1, Output 2.*

Figure 9: *Frequency response and pole-zero plots for each input/output channel for $n_s = 8$.*

Table 2: *Discrete-time and continuous-time eigenvalues*

Discrete-Time λ		Continuous-Time λ	
λ_{d1}	-0.1881 + 0.8924i	λ_{c1}	-3.6842 +71.1394i
λ_{d2}	-0.1881 - 0.8924i	λ_{c2}	-3.6842 -71.1394i
λ_{d3}	-0.2138 + 0.8890i	λ_{c3}	-3.5800 +72.2737i
λ_{d4}	-0.2138 - 0.8890i	λ_{c4}	-3.5800 -72.2737i
λ_{d5}	0.7699 + 0.0000i	λ_{c5}	10.4604 + 0.0000i
λ_{d6}	0.8258 + 0.0000i	λ_{c6}	-7.6568 + 0.0000i
λ_{d7}	0.8686 + 0.0000i	λ_{c7}	-5.6364 + 0.0000i

retained. Table 2 presents the DT and corresponding CT eigenvalues.

$$\lambda_d = e^{\lambda_c t_s} \tag{11}$$

2. Identify eigenvalues related to two damped resonators. Note that from Table 2 it can be seen that these correspond to $\lambda_{c1,2}$ and $\lambda_{c3,4}$ because they are a complex conjugate pair of poles. The RES1 corresponds to the pair $\lambda_{c3,4}$ with a corresponding natural frequency of $\omega_{RES1} = 11.5168\ rad/s$, while RES2 is described by $\lambda_{c1,2}$ and its natural frequency $\omega_{RES1} = 11.3374\ rad/s$. We selected this order for the Resonators because the prompt states that the resonator with a higher frequency is RES1.

3. Relate the three remaining eigenvalues $\lambda_{c5}, \lambda_{c6}, \lambda_{c7}$ with additional physical blocks of the system: a band-pass filter (BP) and unknown filter (UN). To determine which poles are assigned to each subsystem we need to analyze in which channel the zero in 1 appears. Since this is the case for channel u_1/y_2, the poles in this channel $\lambda_{c5}, \lambda_{c7}$ are from the bandpass filter. This

result leads us to conclude that the remaining pole λ_{c6} is from the UN system.

4. Determine the physical structure of the system. Based on the previous analysis of the plots, the required connections are represented in Figure (11) as an initial approach to solving this task. Since we are limited to using one block of each subsystem in each channel, we will resort to summing conjunctions and multiple connections to discover the topology of the system. The system blocks have been arranged as shown in Figure (12) to represent the physical system topology.

From the previous analysis, we have clearly shown that channels u_1/y_1 and u_2/y_1have the UN subsystem, therefore we will use these channels to obtain the frequency response of the UN system and prove that is a low-pass filter. First, we assume that the input u_1 is zero, then the transfer function is presented below

$$\begin{aligned} U_2(RES1)BP &= Y_{22} \\ U_2(RES2)UN &= Y_{12} \end{aligned} \tag{12}$$

then we can divide these two equations

$$\frac{BP}{UN} = \frac{Y_{22}}{Y_{12}} \tag{13}$$

but the frequency response of BP is Y_{21}, Then the final expression for the frequency response for the UN system is the following

$$UN = Y_{21}\frac{Y_{12}}{Y_{22}} \tag{14}$$

which is implemented in Listing 17, lines 64-65. These results are plotted and validated in Figure (10) which provides the comparison between the proposed and real frequency response for the unknown system. Since we can see that there is a considerable peak in low frequency and a plateau for frequencies higher than 100 Hz, this is the frequency response that corresponds to a low-pass filter.

Refer to Listing 17 for details about the implementation of this task.

2.4 Task 4: Continuous-time model of the bandpass subsystem

The objective of this task is to determine a continuous-time (CT) model for the bandpass subsystem BP. Based on the previous analysis and the system's physical architecture determined in Figure (12), the following steps are implemented:

1. Obtain a ZOH/sampled model for the bandpass filter. For this purpose, the correct column of B (first column) and the correct row of C (second row) are extracted to perform a *balanced realization*.

2. Provide graphs with the DT/CT model's frequency response magnitude and phases. Figure (13) presents the comparison of continuous-time and discrete-time frequency response of the bandpass subsystem.

3. Use the sampling theorem to transform the CT frequency response to DT and eliminate the aliasing effect around 20 Hz.

   For this task, we followed 2 approaches to reduce the realization dimension. This first one corresponds to a Henkel matrix reduction model considering just the first two dominant singular values. This approach corresponds to CT1 and ZOH1 in the legend of each plot, and you can see this implementation in Listing (18), lines 37-55. On the other hand, we computed the balanced realization of the system and just keep the first 2 dimensions using the model reduction theorem. It is worth mentioning that to compute the Gramians for this discrete system we used the discrete version of the Lyapunov equation. This approach corresponds to CT2 and ZOH2 in the legend, and you can see this implementation in Listing (18), lines 56-81. From the zoom-in Figure (13) we can see that the balanced realization approach is a bit closer to the expected result for the DT frequency response. We were able to compare both approaches for this project.

Refer to Listing 18 for details about the implementation of this task

3 Conclusions

We have successfully implemented all the tasks of this project. Here are the main takeaways of this project

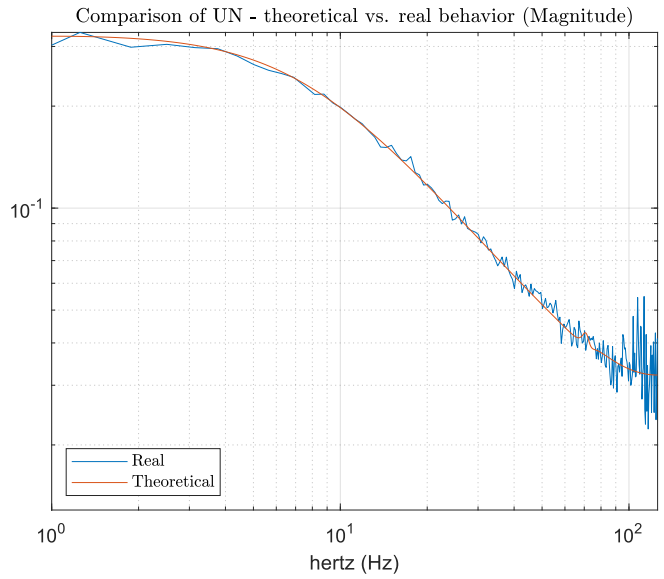


Figure 10: *Magnitude of the frequency response of the unknown system.*

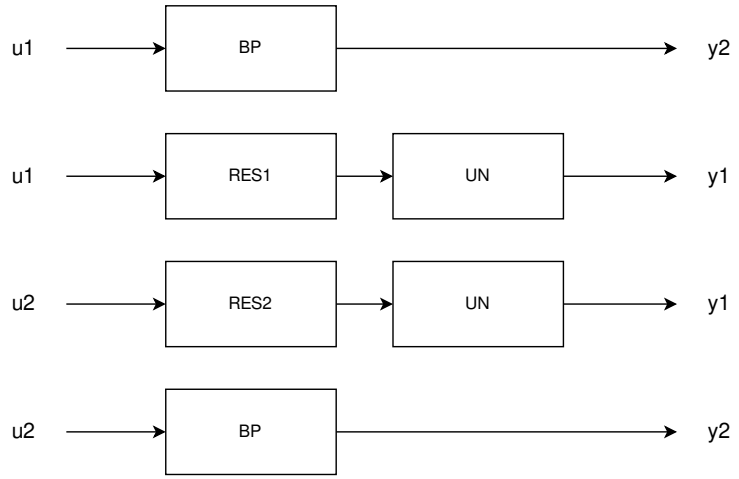


Figure 11: *Diagram of required system connections.*

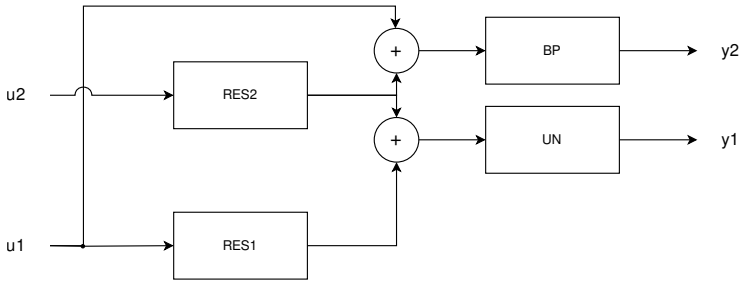
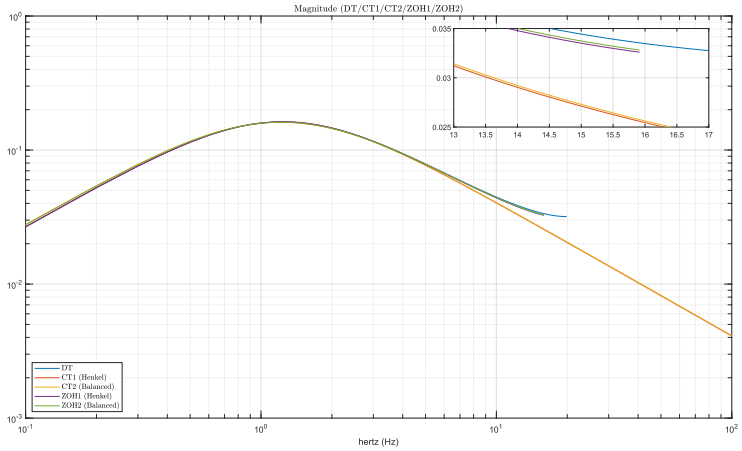
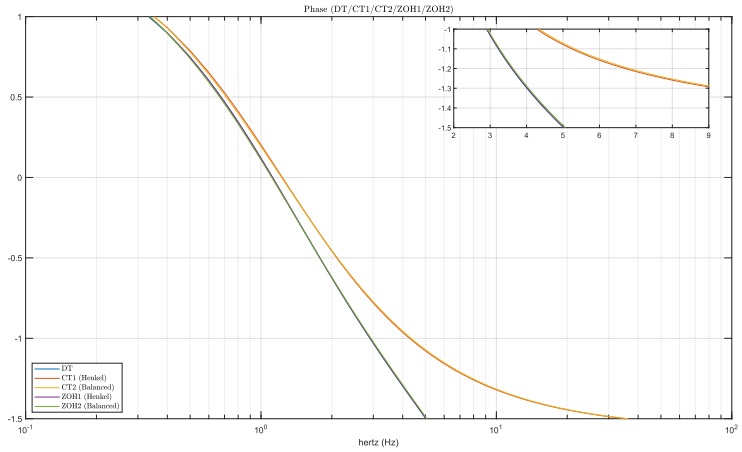


Figure 12: *Proposed system architecture.*



(a) *Frequency response magnitude.*



(b) *Frequency response phase.*

Figure 13: *Comparison of continuous-time and discrete-time frequency response of the bandpass subsystem.*

- The dominant Henkel singular values provide insights about the dimension of a most efficient realization. Even though we can consider the complete dimension realization, we can just reduce the dimension to the number of dominant singular values in a system. This will make a more efficient representation and reduce the computation cost.
- The most efficient dimension for the state space realization according to the pulse data provided is $n_s = 7$
- The $n_s = 7$ system has eigenvalues inside the unit circle, which proves stability for a discrete system.
- The UN subsystem is a low-pass filter.
- Since we are using real data taken from a system, the zero pole cancellation occurs when a zero and a pole are in close proximity.
- Pole-zero plots and frequency response are powerful tools for systems identification.
- In order to do a successful implementation of the balance realization, we used the discrete version of the Lyapunov equation.
- The balance realization approach is more effective in terms of reducing the dimension of a system (compared to a Henkel matrix approach) and reproducing the the system dynamics.

4 Appendix

4.1 Functions

Listing 1: *Hankel Matrix Function*

```
function HM=Hankel_Matrix(y11,y12,y21,y22,n,st)
%u1(y11,y21)
%u2(y12,y22)
for i=0:n-1
    for j=(st)+i:(n+st)+i-1
    HM_element=[y11(j) y12(j);
                y21(j) y22(j)];
    HM_row{1,j-i}=HM_element;
    end
    HM_cell{i+1,1}=cell2mat(HM_row);
end
HM=cell2mat(HM_cell);
end
```

Listing 2: *Model Reduction Function*

```
function [U_HM_Reduced,S_HM_Reduced,V_HM_Reduced,HM_Reduced]=Model_Reduction(HM,n)
    [U_HM,S_HM,V_HM]=svd(HM);
    U_HM_Reduced=U_HM(1:end,1:n);
    S_HM_Reduced=S_HM(1:n,1:n);
    V_HM_Reduced=V_HM(1:end,1:n);
    HM_Reduced=U_HM_Reduced*S_HM_Reduced*V_HM_Reduced';
end
```

Listing 3: *Squared Factorization Function*

```
function [On,Cn]=Squared_Factorization(U_HM_Reduced,S_HM_Reduced,V_HM_Reduced)
    On=U_HM_Reduced*S_HM_Reduced^(0.5);
    Cn=S_HM_Reduced^(0.5)*V_HM_Reduced';
end
```

Listing 4: *Extraction for (B,C) Matrices Function*

```
function [B,C]=B_C_Extraction(On,Cn,ns,n_inputs)
    C=On(1:n_inputs,1:ns);
    B=Cn(1:ns,1:n_inputs);
end
```

Listing 5: *Frequency Response Function*

```
function [y11_freq7,y21_freq7,y12_freq7,y22_freq7]=
                    comparison_frequency_response(A_Total,B_Total,C_Total,ts,models)
    for f=1:length(models)
        if f == 5
           break
```

```
        end
        L=10;
        d_om=1/L;
        om = 0:d_om:(1/ts)*pi-d_om;

        I_dimA=length(A_Total{1,f});
        y11_freq=zeros(1,length(om));
        y21_freq=zeros(1,length(om));
        y12_freq=zeros(1,length(om));
        y22_freq=zeros(1,length(om));
        for i=1:length(om)
            R_matrix= C_Total{1,f}*((exp(1i*om(i)*ts)*eye(I_dimA) - ...
                A_Total{1,f})^(-1))*B_Total{1,f};
            y11_freq(i)=R_matrix(1,1);
            y21_freq(i)=R_matrix(2,1);
            y12_freq(i)=R_matrix(1,2);
            y22_freq(i)=R_matrix(2,2);
        end
        plot_Frequency_Responses(y11_freq,y21_freq,y12_freq,y22_freq,om);
        if f==2 %Keep values for n_s = 7.
            y11_freq7 = y11_freq;
            y22_freq7 = y22_freq;
            y21_freq7 = y21_freq;
            y12_freq7 = y12_freq;
        end
    end
end
```

Listing 6: *Empirical Hankel Matrix Function*

```
function [U,S,V,H]=hankel_matrix_gen(A,B,C)
  n=length(A);
  for i=1:n
    CA_matrix{i,1}= C*A^(i-1);
    AB_matrix{1,i}= A^(i-1)*B;
  end
CA=cell2mat(CA_matrix);
AB=cell2mat(AB_matrix);
H=CA*AB;
[U,S,V]=svd(H);
end
```

Listing 7: *Discrete-to-Continuous Time Function*

```
function c_pole=d2c_pole(d_pole,ts)
    pole=log(d_pole);
    c_pole_real=real(pole)/ts;
    c_pole_imag=imag(pole)/ts;
    c_pole=c_pole_real+1i*c_pole_imag;
end
```

Listing 8: *Hankel Matrix Function, Task 4*

```
function HM=Hankel_Matrix_T4(y,n,st)
%u1(y11,y21)
%u2(y12,y22)
for i=0:n-1
    for j=(st)+i:(n+st)+i-1
```

```
    HM_element=[y(j)];
    HM_row{1,j-i}=HM_element;
    end
end
HM=cell2mat(HM_cell);
end
```

4.2 Plotting Functions

Listing 9: *Singular Values Plotting Function*

```
function ht=plot_sinval(S_HM,m)
    Y=S_HM';
    X=[1:1:m];
    figure(3)
    ht = plot(X,Y(1:m),'.', 'markersize', 8);
    set(gca, 'YScale', 'log')
    title('$H_{100}$','Interpreter','Latex');
    xlabel('Singular Value Index','FontSize',14,'Interpreter','Latex');
    ylabel('Hankel Singular Value','FontSize',14,'Interpreter','Latex');
    ylim([10^(-3) 1])
    xlim([0 m])
    grid on
end
```

Listing 10: *Frequency Response Plotting Function*

```
function plot_Frequency_Responses(y11_freq,y21_freq,y12_freq,y22_freq,om)
figure(1);
mag=abs(y11_freq);
loglog(om,mag);
title('Magnitude $u_1/y_1$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
grid on
axis tight
hold on

figure(2);
phase=angle(y11_freq);
loglog(om,phase);
title('Phase $u_1/y_1$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(3);
mag=abs(y21_freq);
loglog(om,mag);
title('Magnitude $u_1/y_2$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(4);
```

```
phase=angle(y21_freq);
loglog(om,phase);
title('Phase $u_1/y_2$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(5);
mag=abs(y12_freq);
loglog(om,mag);
title('Magnitude $u_2/y_1$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(6);
phase=angle(y12_freq);
loglog(om,phase);
title('Phase $u_2/y_1$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(7);
mag=abs(y22_freq);
loglog(om,mag);
title('Magnitude $u_2/y_2$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on

figure(8);
phase=angle(y22_freq);
loglog(om,phase);
title('Phase $u_2/y_2$','Interpreter','Latex')
legend({'$n_s=7$','$n_s=8$'},'Interpreter','Latex','Location','southwest')
axis tight
grid on
hold on
end
```

Listing 11: *Magnitude/Frequency Plotting Function*

```
function plot_frequencies_fft(y11f,y21f,y12f,y22f,om)
figure(1)
mag=abs(y11f);
loglog(om,mag);
title('Magnitude $u_1/y_1$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(mag) max(mag)])
grid on
hold on

figure(2)
```

```
phase=angle(y11f);
loglog(om,phase);
title('Phase $u_1/y_1$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(phase) max(phase)])
grid on
hold on

figure(3)
mag=abs(y21f);
loglog(om,mag);
title('Magnitude $u_1/y_2$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(mag) max(mag)])
grid on
hold on

figure(4)
phase=angle(y21f);
loglog(om,phase);
title('Phase $u_1/y_2$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(phase) max(phase)])
grid on
hold on

figure(5)
mag=abs(y12f);
loglog(om,mag);
title('Magnitude $u_2/y_1$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(mag) max(mag)])
hold on

figure(6)
phase=angle(y12f);
loglog(om,phase);
title('Phase $u_2/y_1$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(phase) max(phase)])
hold on

figure(7)
mag=abs(y22f);
loglog(om,mag);
title('Magnitude $u_2/y_2$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(mag) max(mag)])
hold on

figure(8)
phase=angle(y22f);
loglog(om,phase);
```

```
title('Phase $u_2/y_2$','Interpreter','Latex')
legend({'$n_s=6$','$n_s=7$','$n_s=10$','$n_s=20$','Pulse ...
    Data'},'Interpreter','Latex','Location','southwest')
axis([0.1 100 min(phase) max(phase)])
hold on
end
```

Listing 12: *Frequency Response Plotting Function, Task 2.5*

```
function comparison_frequency_response_25(A_Total,B_Total,C_Total,ts,models)
    for f=1:length(models)
        if f == 5
           break
        end
        L=10;
        dw=1/L;
        w = 0:dw:(1/ts)*pi-dw;
        om=w;
        I_dimA=length(A_Total{1,f});
        y11_freq=zeros(1,length(om));
        y21_freq=zeros(1,length(om));
        y12_freq=zeros(1,length(om));
        y22_freq=zeros(1,length(om));
        for i=1:length(om)
            R_matrix=C_Total{1,f}*((exp(1i*om(i)*ts)*eye(I_dimA)-
            A_Total{1,f})^(-1))*B_Total{1,f};

            y11_freq(i)=R_matrix(1,1);
            y21_freq(i)=R_matrix(2,1);
            y12_freq(i)=R_matrix(1,2);
            y22_freq(i)=R_matrix(2,2);
        end
        plot_Frequency_Responses(y11_freq,y21_freq,y12_freq,y22_freq,om);
    end
end
```

Listing 13: *Individual Channel Frequency Response Plotting Function*

```
function  S=individual_channels_plot(A_Task2,B_Task2,C_Task2,flag)
n1=randi([5 10]);
C_ch1=C_Task2(1,:);
C_ch2=C_Task2(2,:);
B_ch1=B_Task2(:,1);
B_ch2=B_Task2(:,2);
if flag=='11'
    [n_zeros,d_poles]= ss2tf(A_Task2,B_ch1,C_ch1,0);
     title_plot = strcat('$u_1 / y_1$');
     B = B_ch1;
     C = C_ch1;
elseif flag=='21'
    [n_zeros,d_poles]= ss2tf(A_Task2,B_ch1,C_ch2,0);
    title_plot = strcat('$u_1 / y_2$');
     B = B_ch1;
     C = C_ch2;
elseif flag=='12'
    [n_zeros,d_poles]= ss2tf(A_Task2,B_ch2,C_ch1,0);
```

```
    title_plot = strcat('$u_2 / y_1$');
    B = B_ch2;
    C = C_ch1;
elseif flag=='22'
    [n_zeros,d_poles]= ss2tf(A_Task2,B_ch2,C_ch2,0);
    title_plot = strcat('$u_2 / y_2$');
    B = B_ch2;
    C = C_ch2;
end

ts = 1/40;
dw = 0.001;
dom_t2 = (0:dw:(1/ts)/2-dw)/(2*pi);

sys = tf(n_zeros,d_poles);

figure(n1)
h1 = pzplot(sys);
title({'Zeros(O) vs Poles(x),',title_plot},'Interpreter','Latex')
grid on

[U,S,V,H]=henkel_matrix_gen(A_Task2,B,C);

end
```

Listing 14: *Bode Plot Plotting Function*

```
function bode_plot_mag_phase(y_freq7,flag)
n=randi([1 5]);
if flag=='11'
     title_plot = strcat('$u_1 / y_1$');
elseif flag=='21'
    title_plot = strcat('$u_1 / y_2$');
elseif flag=='12'
    title_plot = strcat('$u_2 / y_1$');
elseif flag=='22'
    title_plot = strcat('$u_2 / y_2$');
end

ts = 1/40;
L=10;
dw = 1/L;
dom_t2 = (0:dw:(1/ts)*pi-dw);

figure(n);
a1 = subplot(211);
mag=abs(y_freq7);
loglog(dom_t2 ,mag);
title({'Magnitude',title_plot},'Interpreter','Latex')
grid on
axis tight

figure(n);
a2 = subplot(212);
phase=angle(y_freq7);
loglog(dom_t2 ,phase);
xlabel('hertz (Hz)')
title({'Phase',title_plot},'Interpreter','Latex')
grid on
```

```
axis tight
% linkaxes([a1 a2],'x')
end
```

4.3 Code Implemented per Task

4.3.1 Task 1

Listing 15: *Code implemented for Task 1*

```
%% M270A - Fall 2022 Project - Task 1
clc
clear all
close all

%% Project Preliminaries
%Load impulses u1, u2 from provided .mat files and code
load u1_impulse.mat
y11 = u1_impulse.Y(3).Data;
y21 = u1_impulse.Y(4).Data;
u1 = u1_impulse.Y(1).Data; %%% note that the pulse magnitude is 5

u1_max=max(u1); %store max value of impulse u1
[¬,mi] = max(u1>0); %%% find index where pulse occurs

load u2_impulse.mat
y12 = u2_impulse.Y(3).Data;
y22 = u2_impulse.Y(4).Data;
u2 = u2_impulse.Y(2).Data;
u2_max=max(u2); %store max value of impulse u2

%%% remove any offsets in output data using data prior to pulse application
y11 = y11 - mean(y11([1:mi-1]));
y12 = y12 - mean(y12([1:mi-1]));
y21 = y21 - mean(y21([1:mi-1]));
y22 = y22 - mean(y22([1:mi-1]));

%%% rescale IO data so that impulse input has magnitude 1
y11 = y11/u1_max;
y12 = y12/u2_max;
y21 = y21/u1_max;
y22 = y22/u2_max;
u1 = u1/u1_max;
u2 = u2/u2_max;

ts = 1/40; %%%% sample period
N = length(y11); %%%% length of data sets
t = [0:N-1]*ts - 1;

% Store values for part 1.4
y11_org = y11;
y12_org = y12;
y21_org = y21;
y22_org = y22;
u1_org = u1;
u2_org = u2;

```

```
%System response for u2=0 ------------------------------------------------
figure(1);
subplot(311)
plot(t,u1,'b*','LineWidth',2)
title('Real Pulse vs Model $n_s = 20$','Interpreter','Latex')
ylabel('$u_1$ (volts)','FontSize',14,'Interpreter','Latex');
grid on
axis([-0.2 2 -0.1 1.1])

subplot(312)
plot(t,y11,'r*','LineWidth',2)
ylabel('$y_1$ (volts)','FontSize',14,'Interpreter','Latex');
grid on
axis([-0.2 2 -0.1 0.1])
hold on

subplot(313)
plot(t,y21,'r*','LineWidth',2)
ylabel('$y_2$ (volts)','FontSize',14,'Interpreter','Latex');
grid on
xlabel('seconds (s)','FontSize',14,'FontSize',14,'Interpreter','Latex')
%set(gca,'FontSize',14)
axis([-0.2 2 -0.1 0.1])
hold on

%System response for u1=0 ------------------------------------------------
figure(2);
subplot(311)
plot(t,u2,'b*','LineWidth',2)
title('Real Pulse vs Model $n_s = 20$','Interpreter','Latex')
ylabel('$u_2$ (volts)','FontSize',14,'Interpreter','Latex');
grid on
axis([-0.2 2 -0.1 1.1])
hold on

subplot(312)
plot(t,y12,'r*','LineWidth',2)
ylabel('$y_1$ (volts)','FontSize',14,'Interpreter','Latex');
axis([-0.2 2 -0.1 0.1])
grid on
hold on

subplot(313)
plot(t,y22,'r*','LineWidth',2)
ylabel('$y_2$ (volts)','FontSize',14,'Interpreter','Latex');
xlabel('seconds (s)','FontSize',14,'Interpreter','Latex')
axis([-0.2 2 -0.1 0.1])
grid on
hold on

%% 1.0 Extraction of relevant data
%Index [mi+1:end]
y11 = y11(mi+1:end);
y12 = y12(mi+1:end);
y21 = y21(mi+1:end);
y22 = y22(mi+1:end);
u1 = u1(mi+1:end);
u2 = u2(mi+1:end);

```

```
%% 1.1 System Identification + Henkel Matrices
clc
n_inputs=2; %MIMO inputs

% Variables
n=100; %for Hankel Matrix n=100
m=40; %Plot first 40 non-singular values

%Change indexes to shift Hankel Matrix
st_1=1; st_2=2;

%Optional, provide plots Hankel Matrix non-singular values
plot_flag_T1=1;

models=[6,7,10,20,8]; %n_s state dimensions
for k=1:length(models)
    % 1. Construct H100 and graph singular values.
    HM=Henkel_Matrix(y11,y12,y21,y22,n,st_1); %See function Henkel_Matrix
    %Optional, provide plots Hankel Matrix non-singular values
    if plot_flag_T1==1
        S_HM=svd(HM);
        ht=plot_sinval(S_HM,m); %Plotting function
    end
    %Compute the reduced state dimensions for n_s models. See function Model_Reduction
    [U_HM_Reduced,S_HM_Reduced,V_HM_Reduced,HM_Reduced]=Model_Reduction(HM,models(k));

    %Compute O&Cfor the reduced state dimensions. See function Squared_Factorization
    [On,Cn]=Squared_Factorization(U_HM_Reduced,S_HM_Reduced,V_HM_Reduced);

    %2. Obtain matrices A, B, C for each state dimension
    [B,C]=B_C_Extraction(On,Cn,models(k),n_inputs);
    HM_tilde=Henkel_Matrix(y11,y12,y21,y22,n,st_2);
    On_pi=pinv(On); %left inverse
    Cn_pi=pinv(Cn); %rigth inverse
    A=On_pi*HM_tilde*Cn_pi;

    %Store all computed A, B, C values in cells
    A_Total{1,k}=A;
    B_Total{1,k}=B;
    C_Total{1,k}=C;

    %Check if mod(eigs) <1 for asymp. stability
    max_eig = max(abs(eig(A)));
    if max_eig ≥ 1
        disp('The system is not asymptotically stable')
    end
end

%% 1.2 Impulse Responses
system=4; %Choose n_s = 7

%Construct D = 0 matrix
[r,c]=size(C_Total{1,system});
D = zeros(r,n_inputs);

%State-space model function for ts sample period
sys = ss(A_Total{1,system},B_Total{1,system},C_Total{1,system},D,ts);
[y_impulse,t_impulse]=impulse(sys,2); %Obtain impulse response
y_impulse=y_impulse*ts;
```

```
166
167 %System response for u2=0 ------------------------------------------------
168 figure(1);
169 subplot(312)
170 plot(t_impulse,y_impulse(:,1,1),'g*','LineWidth',2)
171 legend({'Provided Impulse Response','Replicated Impulse ...
        Response'},'Interpreter','Latex','Location','northeast')
172 grid on
173 hold on
174
175 subplot(313)
176 plot(t_impulse,y_impulse(:,2,1),'g*','LineWidth',2)
177 legend({'Provided Impulse Response','Replicated Impulse ...
        Response'},'Interpreter','Latex','Location','northeast')
178 grid on
179 hold on
180
181 %System response for u1=0 ------------------------------------------------
182 figure(2);
183 subplot(312)
184 plot(t_impulse,y_impulse(:,1,2),'g*','LineWidth',2)
185 legend({'Provided Impulse Response','Replicated Impulse ...
        Response'},'Interpreter','Latex','Location','northeast')
186 grid on
187 hold on
188
189 subplot(313)
190 plot(t_impulse,y_impulse(:,2,2),'g*','LineWidth',2)
191 legend({'Provided Impulse Response','Replicated Impulse ...
        Response'},'Interpreter','Latex','Location','northeast')
192 grid on
193 hold on
194
195 input('You can save the plot at this moment, press any key to continue!')
196
197 %% 1.3 Model Frequency Response
198 input('You can save the plot at this moment, press any key to continue!')
199 clc
200 close all
201
202 %Provide magnitude/phase frequency response plots; keep values for n_s = 7.
203 [y11_freq7,y21_freq7,y12_freq7,y22_freq7]=
204 comparison_frequency_response(A_Total,B_Total,C_Total,ts,models);
205
206 %% 1.4 Pulse Response Data
207 %Estimate empirical frequency response
208 y11f= fft(y11_org)./fft(u1_org);
209 y21f= fft(y21_org)./fft(u1_org);
210 y12f= fft(y12_org)./fft(u2_org);
211 y22f= fft(y22_org)./fft(u2_org);
212 N = length(y11f);
213 om=2*pi*[0:N-1]/(ts*N);
214 %Provide frequency response estimations overlayed on previous plots.
215 plot_frequencies_fft(y11f,y21f,y12f,y22f,om)
216
217 %% 1.x Saving data for the next Task
218 y_freq_ns7=[y11_freq7;y21_freq7;y12_freq7;y22_freq7];
219 save('y_freq_ns7.mat','y_freq_ns7')
220 State_Space_Realizations = {A_Total;B_Total;C_Total;D};
```

```
save('State_Space_Realizations.mat','State_Space_Realizations')
y_freq_data=[y11f;y21f;y12f;y22f];
save('y_freq_data.mat','y_freq_data')
save('y21.mat','y21')
```

4.3.2 Task 2

Listing 16: *Code implemented for Task 2*

```
%% M270A - Fall 2022 Project - Task 2
clc
clear all
close all

%Load saved data from Task 1
load y_freq_ns7.mat
load State_Space_Realizations.mat
load y_freq_data.mat

%Select data corresponding to n_s = 7
y11_freq7=y_freq_ns7(1,:);
y21_freq7=y_freq_ns7(2,:);
y12_freq7=y_freq_ns7(3,:);
y22_freq7=y_freq_ns7(4,:);

%Recover A, B, C matrices
A_Total= State_Space_Realizations(1,1);
A_Total=A_Total{1,1};
B_Total= State_Space_Realizations(2,1);
B_Total=B_Total{1,1};
C_Total= State_Space_Realizations(3,1);
C_Total=C_Total{1,1};
D = State_Space_Realizations(4,1);
D = D{1,1};

%Recover empirical estimates of frequency response
y11f = y_freq_data(1,:);
y21f = y_freq_data(2,:);
y12f = y_freq_data(3,:);
y22f = y_freq_data(4,:);
ts = 1/40; %%%% sample period
%% 2.1 Transmission Zeros of the MIMO Model
selector=2; %System n_s=7
plot_flag_T2=1; %Optional, provide plots.

%Recover A, B, C matrices for n_s = 7
A_Task2=A_Total{1,selector};
B_Task2=B_Total{1,selector};
C_Task2=C_Total{1,selector};
D_Task2=D; % D = 0
I_dimA =eye(length(A_Task2)); %Identity matrix of appropriate dimension

%Generalized eigenvalue problem set-up-------------------------------------
LMat_eVal_Problem={A_Task2,B_Task2;
                   C_Task2,D_Task2};
RMat_eVal_Problem={I_dimA,zeros(size(B_Task2));
                    zeros(size(C_Task2)),zeros(size(D_Task2))};
```

```
LMat=cell2mat(LMat_eVal_Problem);
RMat=cell2mat(RMat_eVal_Problem);

%LMat*U_Task2-U_Task2*RMat*V_Task2
%U = eigenvectors; V = eigenvalues, z

%Obtain eigenvalues (V)
[U_Task2,V_Task2]=eig(LMat,RMat);
Transmission_zeros=diag(V_Task2);
for k=size(Transmission_zeros):-1:1
    if Transmission_zeros(k)==Inf || Transmission_zeros(k)== -1*Inf
       Transmission_zeros(k)=[];
    end
end

%Sort transmission zeros
TZ=sort(Transmission_zeros,'ComparisonMethod','abs');

%% 2.2 Eigenvalues & Transmission Zeros Plot
if plot_flag_T2==1
    real_eval=real(TZ);
    imag_eval=imag(TZ);
    circ = exp(1i*[0:360]*pi/180);
    plot(real_eval,imag_eval,'bo',real(circ),imag(circ),'k--');
    title('Plot of the finite Transmission Zeros for $n_s=7$','Interpreter','Latex')
    xlabel('Re(z)','Interpreter','Latex')
    ylabel('Im(z)','Interpreter','Latex')
    axis([-3 3 -3 3]);
    axis square
    grid on
    hold on
end

Eval_Task2 = eig(A_Task2);
for x=1:length(Eval_Task2)
    Sel_Eval=Eval_Task2(x);
    plot(real(Sel_Eval),imag(Sel_Eval),'x','Linewidth',1);
    eig_t2_el = strcat( '$\lambda$ =',num2str(round(real(Eval_Task2(x)),2)),
    '+',num2str(round(imag(Eval_Task2(x)),2)),'i');
    eig_t2{x+2} =  eig_t2_el;
    hold on
end
eig_t2{1}='Unit Circle';
eig_t2{2}='Transmission zeros';
legend(eig_t2,'Interpreter','latex')
input('You can save the plot at this moment, press any key to continue!');
%% 2.3 Transmission Zero Response of the MIMO Model
% Eigenvector close to 1 in position 3
close all
%Extract [c, w] from Generalized Eigenvalue Problem
c=U_Task2(1:7,3);
w=U_Task2(8:9,3);

x0_Task= c; %Given Initial Condition
%Initialization
samples_task2=100; %n = 40 samples required
u_in=zeros(2,1);
y_out=zeros(2,1);
%Simulation 2I2O Systems
```

```
for step=0:1:samples_task2
    u=w*TZ(4)^(step);
    x_n_1 = A_Task2*x0_Task+B_Task2*u;
    y=C_Task2*x0_Task;
    x0_Task=x_n_1;
    y_out=[y_out y];
    u_in=[u_in u];
end
step=0:1:samples_task2;

%Plot Output Sequence
figure(1)
plot(step,y_out(1,2:end),step,y_out(2,2:end));
title('$2I/2O$ Model for 40 samples - Output','Interpreter','Latex');
ylabel('$y$ (volts)','FontSize',14,'Interpreter','Latex');
xlabel('seconds (s)','FontSize',14,'Interpreter','Latex')
legend({'$y_1$','$y_2$'},'Interpreter','Latex','Location','southeast')
grid on

%Plot Input Sequence
figure(2)
plot(step,u_in(1,2:end),step,u_in(2,2:end))
title('$2I/2O$ Model for 40 samples - Input','Interpreter','Latex');
ylabel('$u$ (volts)','FontSize',14,'Interpreter','Latex');
xlabel('seconds (s)','FontSize',14,'Interpreter','Latex')
legend({'$u_1$','$u_2$'},'Interpreter','Latex','Location','southeast')
grid on

input('You can save the plot at this moment, press any key to continue!');

%% 2.4 Transmission Zeros of the MIMO Model
% '11' u1/y1 y11_freq7
% '21' u1/y2 y21_freq7
% '12' u2/y1 y12_freq7
% '22' u2/y2 y22_freq7

%Change the y_Task24 according to the desired analysis
y_Task24=y22_freq7; %Change the plotted input/output channel, e.g: u2/y2
flag_Task24='22';   %Change the plot identifier

S=individual_channels_plot(A_Task2,B_Task2,C_Task2,flag_Task24)
bode_plot_mag_phase(y_Task24,flag_Task24)
input('You can save the plot at this moment, press any key to continue!');

%% 2.5.A Comparison with Hs=8
clc
close all
%Recover A, B, C matrices for n_s = 8 (Version A)
selector_t25=5;
A_Task2_5=A_Total{1,selector_t25};
B_Task2_5=B_Total{1,selector_t25};
C_Task2_5=C_Total{1,selector_t25};
%Recover A, B, C matrices for n_s = 8 (Version B)
models_task_2_5=[7,8];
A_model_task_2_5={A_Total{1,2},A_Total{1,5}};
B_model_task_2_5={B_Total{1,2},B_Total{1,5}};
C_model_task_2_5={C_Total{1,2},C_Total{1,5}};

%Modified function to compare n_s = [7,8] only
```

```
comparison_frequency_response_25(A_model_task_2_5, B_model_task_2_5, ...
    C_model_task_2_5, ts,models_task_2_5);

input('You can save the plot at this moment, press any key to continue!');

%% 2.5.B  Comparison with Hs=8
close all
% CHANGE this D,1 D,2!
[y,u]= ss2tf(A_Task2_5,B_Task2_5,C_Task2_5,D,2);
% Do this for each channel, e.g, u1/y2, CHANGE this!
sys = tf(y(1,:),u(1,:));
figure(3)
h1_task2_5 = pzplot(sys);
grid on
% u1/y2, CHANGE this!
title('u2/y1','Interpreter','Latex');
set(gca,'FontSize',14)
```

4.3.3 Task 3

Listing 17: *Code implemented for Task 3*

```
%% M270A - Fall 2022 Project - Task 3
clc
clear
close all

%Load saved data from Task 1
load y_freq_ns7.mat
load State_Space_Realizations.mat
load y_freq_data.mat

%Select data corresponding to n_s = 7
y11_freq7=y_freq_ns7(1,:);
y21_freq7=y_freq_ns7(2,:);
y12_freq7=y_freq_ns7(3,:);
y22_freq7=y_freq_ns7(4,:);

%Recover A, B, C matrices
A_Total= State_Space_Realizations(1,1);
A_Total=A_Total{1,1};
B_Total= State_Space_Realizations(2,1);
B_Total=B_Total{1,1};
C_Total= State_Space_Realizations(3,1);
C_Total=C_Total{1,1};
D = State_Space_Realizations(4,1);
D = D{1,1};

%Recover empirical estimates of frequency response
y11f = y_freq_data(1,:);
y21f = y_freq_data(2,:);
y12f = y_freq_data(3,:);
y22f = y_freq_data(4,:);
ts = 1/40; %%%% sample period
%% 3.1 Block diagram derived from analysis of SISO Channels
clc
syms ss
```

```
selector_t25=2; %System n_s=7
A_Task3=A_Total{1,selector_t25};
B_Task3=B_Total{1,selector_t25};
C_Task3=C_Total{1,selector_t25};

%Compute e-vec/e-vals for n_s = 7
[Vec_Task3,Val_Task3]=eig(A_Task3,'vector');
c_pole=zeros(size(Val_Task3));

for x=1:length(Val_Task3)
c_pole(x,1)=d2c_pole(Val_Task3(x),ts); %See function d2c_pole
end

%Discrete (Column 1) and Continuous-time Poles (Column 2:3)
poles=[Val_Task3 c_pole abs(c_pole)/(2*pi)]
%Underdamped
%RES1 = which has a slightly higher frequency 3 4
%RES2 = which has a slightly lower frequency  1 2
%% 3.3 Plot of UN
%Domain for the real behavior
N= length(y11f);
om_task3=2*pi*[0:N-1]/(ts*N);
%Domain for the theorical behavior
L=10;
dw=1/L;
w = 0:dw:(1/ts)*pi-dw;

%Calculation of UN
yun_real=(y21f).*(y12f./y22f);                  %real
yun_teo = (y21_freq7).*(y12_freq7./y22_freq7); %theorical
figure(1)
mag1=abs(yun_real);
mag2=abs(yun_teo);
loglog(om_task3,mag1,w,mag2);
title('Comparison of UN - theoretical vs. real behavior ...
    (Magnitude)','Interpreter','Latex')
legend({'Real','Theoretical'},'Interpreter','Latex','Location','southwest')
xlabel('hertz (Hz)')
axis([1 40*pi min(mag1) max(mag1)])
grid on
hold on
```

4.3.4 Task 4

Listing 18: *Code implemented for Task 4*

```
%% M270A - Fall 2022 Project - Task 4
clc
clear
close all

%Load saved data from Task 1
load y_freq_ns7.mat
load State_Space_Realizations.mat
load y_freq_data.mat
%Load relevant output channel
load y21.mat
```

```
12
13 %Select data corresponding to n_s = 7
14 y11_freq7=y_freq_ns7(1,:);
15 y21_freq7=y_freq_ns7(2,:);
16 y12_freq7=y_freq_ns7(3,:);
17 y22_freq7=y_freq_ns7(4,:);
18
19 %Recover A, B, C matrices
20 A_Total= State_Space_Realizations(1,1);
21 A_Total=A_Total{1,1};
22 B_Total= State_Space_Realizations(2,1);
23 B_Total=B_Total{1,1};
24 C_Total= State_Space_Realizations(3,1);
25 C_Total=C_Total{1,1};
26 D = State_Space_Realizations(4,1);
27 D = D{1,1};
28
29 %Recover empirical estimates of frequency response
30 y11f = y_freq_data(1,:);
31 y21f = y_freq_data(2,:);
32 y12f = y_freq_data(3,:);
33 y22f = y_freq_data(4,:);
34 ts = 1/40; %%%% sample period
35 %% 4 Band Pass Filter
36 %% 4.1 & 4.2 First Approach: Henkel matrices approach
37 % 1. Construct H100
38 HM_T4=Henkel_Matrix_T4(y21,100,1);
39 %Compute the reduced state dimensions for n=2 models. See function Model_Reduction
40 [U_HM_Reduced_T4,S_HM_Reduced_T4,V_HM_Reduced_T4,HM_Reduced_T4]=
41 Model_Reduction(HM_T4,2);
42 %Compute O&Cfor the reduced state dimension n=2. See function Squared_Factorization
43 [On_T4,Cn_T4]=Squared_Factorization(U_HM_Reduced_T4,S_HM_Reduced_T4,V_HM_Reduced_T4);
44
45 %2.Obtain matrices A, B, C for each state dimension
46 [B_T4,C_T4]=B_C_Extraction(On_T4,Cn_T4,2,1);
47 HM_tilde_T4=Henkel_Matrix_T4(y21,100,2);
48 On_pi_T4=pinv(On_T4); %left inverse
49 Cn_pi_T4=pinv(Cn_T4); %rigth inverse
50 A_T4=On_pi_T4*HM_tilde_T4*Cn_pi_T4;
51
52 %Discrete-2-Continuous
53 A_con_t4 = logm(A_T4)/ts;
54 B_con_t4 = -(A_con_t4^0-A_T4)^(-1)*A_con_t4*B_T4;
55 C_con_t4 = C_T4;
56 %% Second Approach : Balanced Realization
57 A_ns7=A_Total{1,2};
58 B_ns7=B_Total{1,2};
59 C_ns7= C_Total{1,2};
60 B_ns7=B_ns7(:,1);
61 C_ns7=C_ns7(2,:);
62 Gcd = dlyap(A_ns7,B_ns7*B_ns7'); %P
63 God = dlyap(A_ns7',C_ns7'*C_ns7); %Q
64 R = chol(Gcd);
65 Pass1= R*God*R';
66 [U_f,S_f,V_f]=svd(Pass1);
67 sig_check =sqrt(S_f);
68 Sig = sqrt(sig_check);
69 T= Sig*U_f'*(R^(-1))';
70
```

```
A_bar= T*A_ns7*T^(-1);
B_bar= T*B_ns7;
C_bar= C_ns7*T^(-1);

A_Bal= A_bar(1:2,1:2);
B_Bal= B_bar(1:2,1);
C_Bal= C_bar(1,1:2);

A_Balanced = logm(A_Bal)/ts;
B_Balanced = -(A_Balanced^0-A_Bal)^(-1)*A_Balanced*B_Bal;
C_Balanced = C_Bal;

%% DT-CT Comparison
% Discrete Domain
L=10;
dw=1/L;
w = 0:dw:(1/ts)/2-dw;
om_t3=(w);
BP_dis=zeros(1,length(om_t3));

% Continuous Domain
fq = [0.1:dw:100];
BP_con=zeros(1,length(fq));
BP_con2=zeros(1,length(fq));

%Frequency Response for both models
for i=1:length(om_t3)
            BP_dis(i)=C_T4*((exp(1i*om_t3(i)*2*pi*ts)*A_T4^(0)-A_T4)^(-1))*B_T4;
end
for i=1:length(fq)
    BP_con(i)=C_con_t4*((1i*fq(i)*2*pi*A_con_t4^(0)-A_con_t4)^(-1))*B_con_t4;
    BP_con2(i)=C_Balanced*((1i*fq(i)*2*pi*A_Balanced^(0)-A_Balanced)^(-1))*B_Balanced;
end

%Plot frequency response DT/CT
figure(1)
mag_t4=abs(BP_dis);
loglog(om_t3,mag_t4);
grid on
hold on

figure(2)
ang_t4=angle(BP_dis);
loglog(om_t3,ang_t4);
grid on
hold on

figure(1)
mag_t4_con=abs(BP_con);
loglog(fq,mag_t4_con);
grid on
 hold on

figure(2)
ang_t4_con=angle(BP_con);
loglog(fq,ang_t4_con);
grid on
hold on

```

```
figure(1)
mag_t4_con2=abs(BP_con2);
loglog(fq,mag_t4_con2);
grid on
 hold on

figure(2)
ang_t4_con2=angle(BP_con2);
loglog(fq,ang_t4_con2);
grid on
hold on

%% 4.3 Sampling Theorem
clc
dw=1/L;
w = [0.1:dw:100]; %Frequency grid
om_t43=(w);
BP_alias=zeros(1,length(om_t43));
BP_alias2=zeros(1,length(om_t43));
for i=1:length(om_t43)
    ft=0;
    ft2=0;
    for j=0:12
    ow_alias = om_t43(i)+(j-6)*(40*2*pi);

    term = (C_con_t4*((1i*(ow_alias)*A_con_t4^(0)-A_con_t4)^(-1))*B_con_t4)*
    (1/(1i*ow_alias))*(1-exp(-1i*ow_alias*ts));

    add =(1/ts)*term;
    ft = ft +add;

    term2 = (C_Balanced*((1i*(ow_alias)*A_Balanced^(0)-A_Balanced)^(-1))*B_Balanced)
    *(1/(1i*ow_alias))*(1-exp(-1i*ow_alias*ts));

    add2 =(1/ts)*term2;
    ft2 = ft2 +add2;
    end
    BP_alias(i)=ft;
    BP_alias2(i)=ft2;
end

figure(1)
mag_t43=abs(BP_alias);
loglog(om_t43/(2*pi),mag_t43);
grid on
hold on
%%%%

figure(2)
ang_t4_alias=angle(BP_alias);
loglog(om_t43/(2*pi),ang_t4_alias);
grid on
hold on

%% Second Approach: Balanced Realization

fig1 = figure(1)
mag_t432=abs(BP_alias2);
loglog(om_t43/(2*pi),mag_t432);
```

```
title('Magnitude (DT/CT1/CT2/ZOH1/ZOH2)','Interpreter','Latex')
xlabel('hertz (Hz)')
grid on
hold on
ylim([0.001 1])
xlim([0.1 100])
legend({'DT','CT1 (Henkel)','CT2 (Balanced)','ZOH1 (Henkel)','ZOH2 ...
    (Balanced)'},'Interpreter','Latex','Location','southwest')

ax_zoom2 = axes('Parent',fig1,'Position',[0.6 0.7 0.2800 0.2000]);
hold(ax_zoom2,'on'); box(ax_zoom2,'on')
loglog(om_t3,mag_t4,fq,mag_t4_con,fq,mag_t4_con2,om_t43/(2*pi),...
mag_t43,om_t43/(2*pi),mag_t432)
xlim(ax_zoom2,[13 17]);
ylim(ax_zoom2,[2.5*10^-2 3.5*10^-2]);
%set(gca,'FontSize',font_S)
grid on

fig2 = figure(2)
ang_t4_alias2=angle(BP_alias2);
loglog(om_t43/(2*pi),ang_t4_alias2);
title('Phase (DT/CT1/CT2/ZOH1/ZOH2)','Interpreter','Latex')
legend({'DT','CT1 (Henkel)','CT2 (Balanced)','ZOH1 (Henkel)','ZOH2 ...
    (Balanced)'},'Interpreter','Latex','Location','southwest')
xlabel('hertz (Hz)')
ylim([-1.5 1])
xlim([0.1 100])
grid on
hold on

ax_zoom2 = axes('Parent',fig2,'Position',[0.6 0.7 0.2800 0.2000]);
hold(ax_zoom2,'on'); box(ax_zoom2,'on')
loglog(om_t3,ang_t4,fq,ang_t4_con,fq,ang_t4_con2,om_t43/(2*pi),ang_t4_alias,om_t43/(2*pi),ang_t4_al
ylim(ax_zoom2,[-1.5 -1]);
xlim(ax_zoom2,[2 9]);
grid on



input('This is the last plot. You can save the plot at this moment, press any key to ...
    continue!');
```