MAE270A Linear Dynamic System
Project
Prof. M'Closkey

Zhijie Wang

2017-12-08

Table of Contents

MAE270A Project

Zhijie Wang

Introduction

A multi input/ multi-output, discrete-time linear system with 2 outputs and 2 inputs is defined as:

$$x_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k$$

where the state dimension is n_S so $A \in \mathbf{R}^{n_s \times n_s}, B \in \mathbf{R}^{n_s \times 2}$ and $C \in \mathbf{R}^{2 \times n_s}$. It is assumed the feedthrough matrix $D \in \mathbf{R}^{2\times 2}$ is zero. We assume the state space matrixes have real elements. The rth column of the pulse response sequence $\{h_k\}$, where $h_k \in \mathbf{R}^{2\times 2}$, is obtained when $x_0 = 0$ and the input is given by the sequence which the rth element of u_0 is 1 and all other elements are zero, and then the subsequent input samples are all zero. The pulse response is the discrete-time analog of the impulse response of a continuous-time system. It is straight forward to show:

$$h_0 = 0$$
$$h_k = CA^{k-1}B, k = 1,2,3 \dots$$

Task 1:

Problem 1-1:

The pulse response sequence can be organized into a Hankel matrix by using the input terms. First construct the Hankel Matrix as:

$$H = \begin{bmatrix} h_1 & h_2 & h_3 & \dots & h_n \\ h_2 & h_3 & h_4 & \dots & h_{n+1} \\ h_3 & h_4 & h_5 & \dots & h_{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_n & h_{n+1} & h_{n+2} & \dots & h_{2n-1} \end{bmatrix}$$

The Hankel Singular Value for H_{100} is plotted in the following figure:

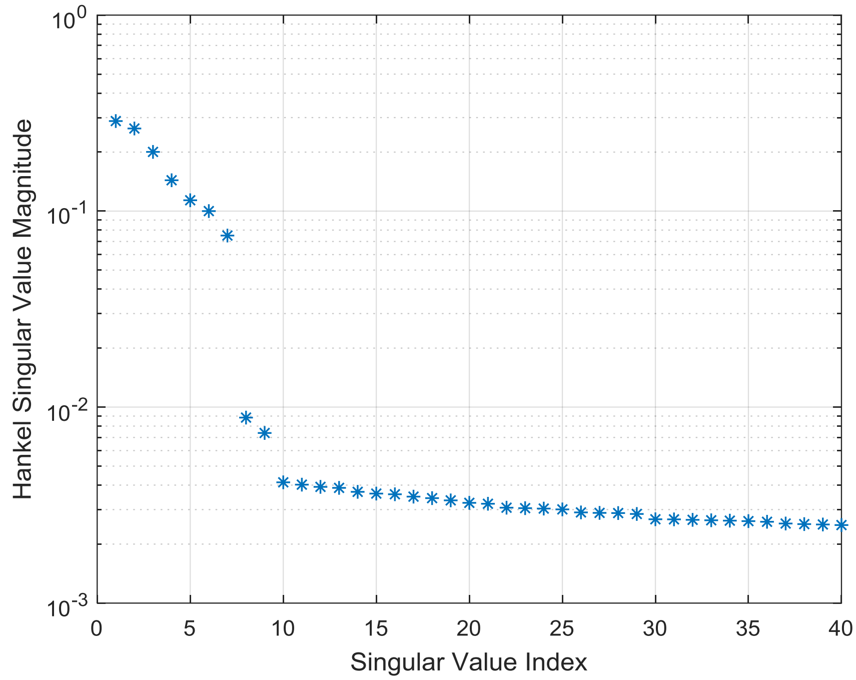

Figure 1. Hankel Singular Value for H_100

Then determine O and C from the Singular Value Decomposition of Hankel Matrix:

We choose $O_n = U_1$ and $C_n = \Sigma_{n_s} V_1^T$

And we can have the pseudoinverse as $O_n^\dagger = U_1^T$ and $C_n^\dagger = V_1 \Sigma_{n_s}^{-1}$

State matrix C is chosen from the first $[2 \times n_s]$ element of O_n and state matrix B is chosen from the first $[n_s \times 2]$ element of C_n.

A new Hankel Matrix is computed by starting from h_2, and A is determined using:

$$A = O_n^\dagger \tilde{H}_n C_n^\dagger$$

With pseudoinverse calculated above.

Then we test for the stabilty of the systme:

$$\max(|eig(A_6)|) = 0.9526$$

$$\max(|eig(A_7)|) = 0.9144$$

$$\max(|eig(A_{10})|) = 0.9141$$

$$\max(|eig(A_{20})|) = 0.9975$$

Because the max of absolute is within $[-1\ 1]$ bound. The systems are all asymptotically stable.

Problem 1-2:

Each model was simulated and compared to the measurement data. The measurement data is plotted in red below. Simulated data is plotted in black dot. Except ns = 6, all model has an inferior reproduction of the pulse response data, which are indistinguishable. Therefore, ns = 7 can be used as valid model for the system.

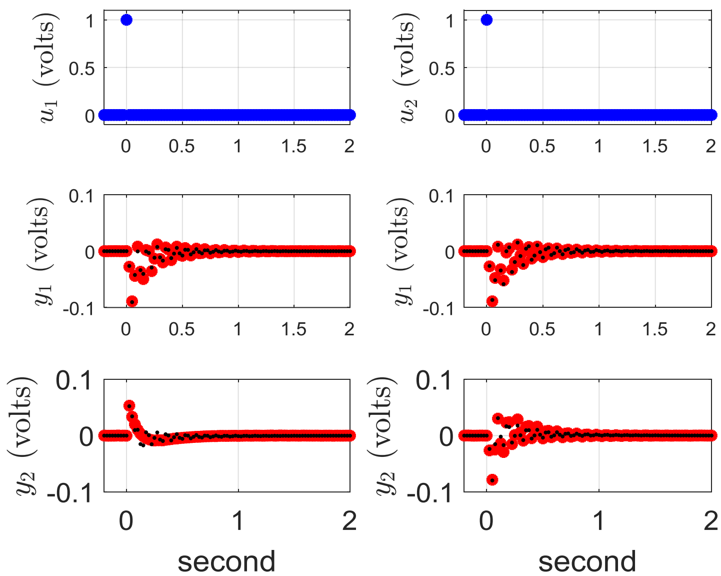

Figure 2. Comparison of 6-state model.

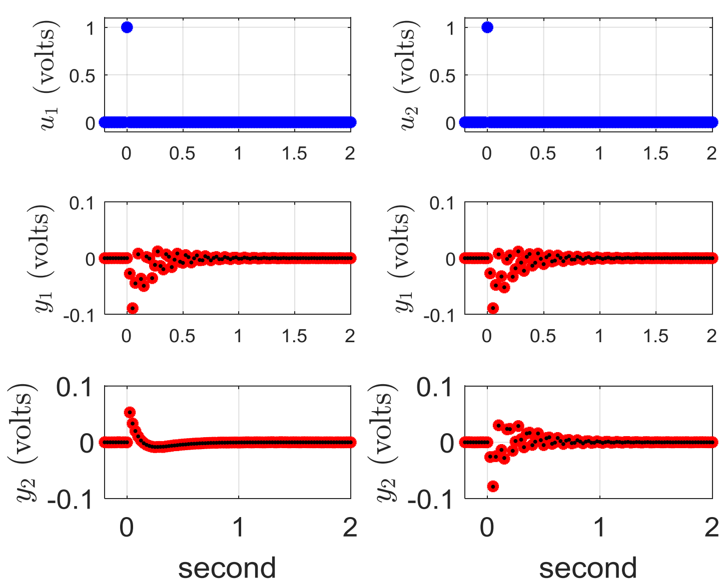

Figure 3. Comparison of 7-state model

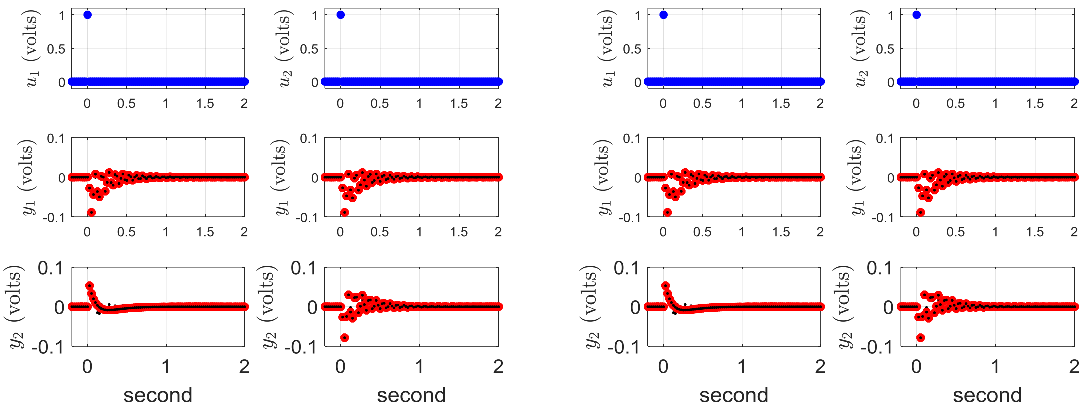

Figure 4. Comparison of 10-state model (on the left), and 20-state model (on the right)

Problem 1-3 and 1-4:

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

$$C\left(e^{jwt_s}I - A\right)^{-1}B + D$$

where $t_s = 1/40$ is the sample period in seconds. The pulse response data is used to directly estimate the frequency response. The estimated empirical frequency response magnitude and phase are overlaid on previous results. The phase and magnitude are shown in Figure 6:

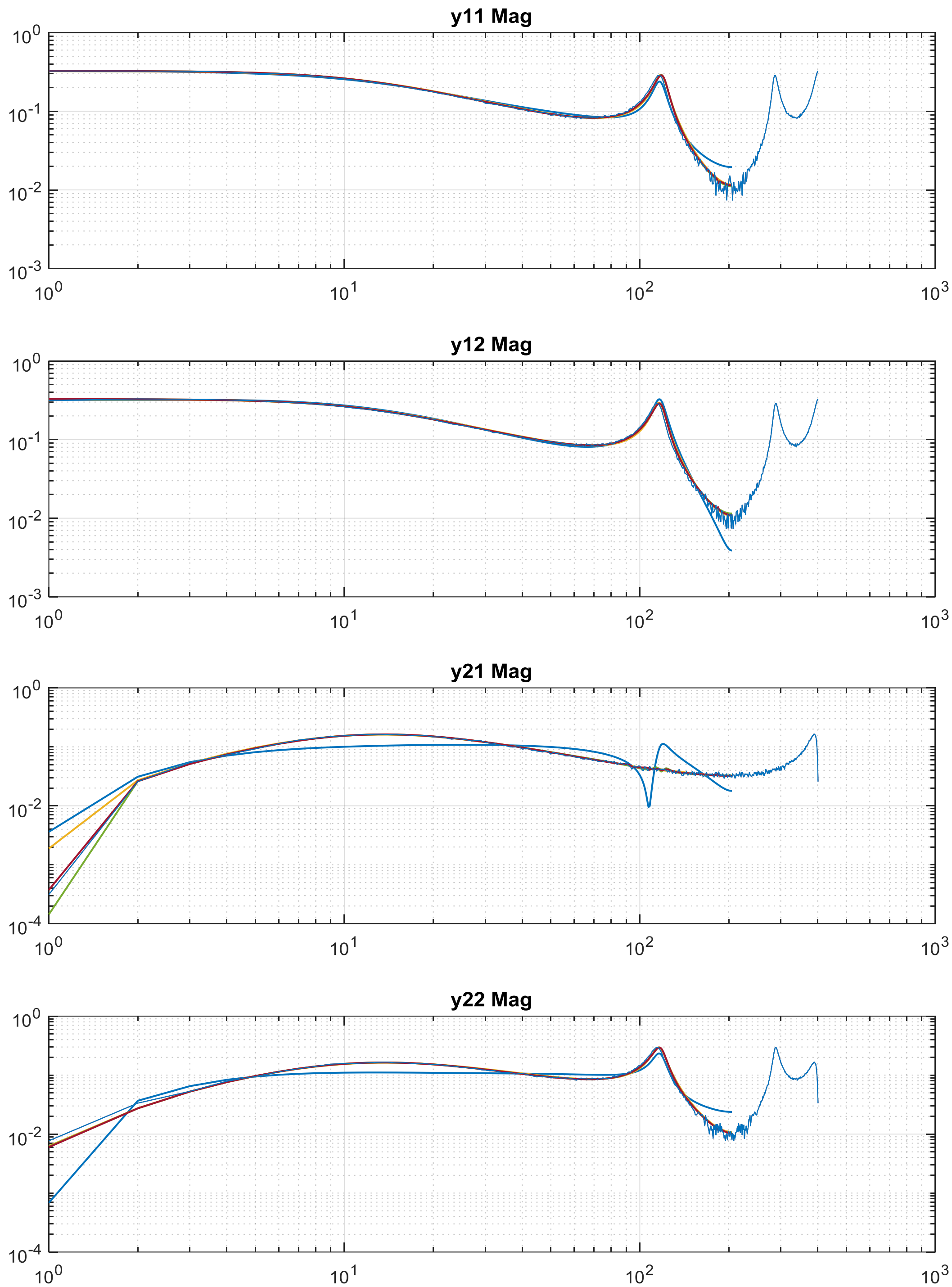

Figure 5. Magnitude Plot

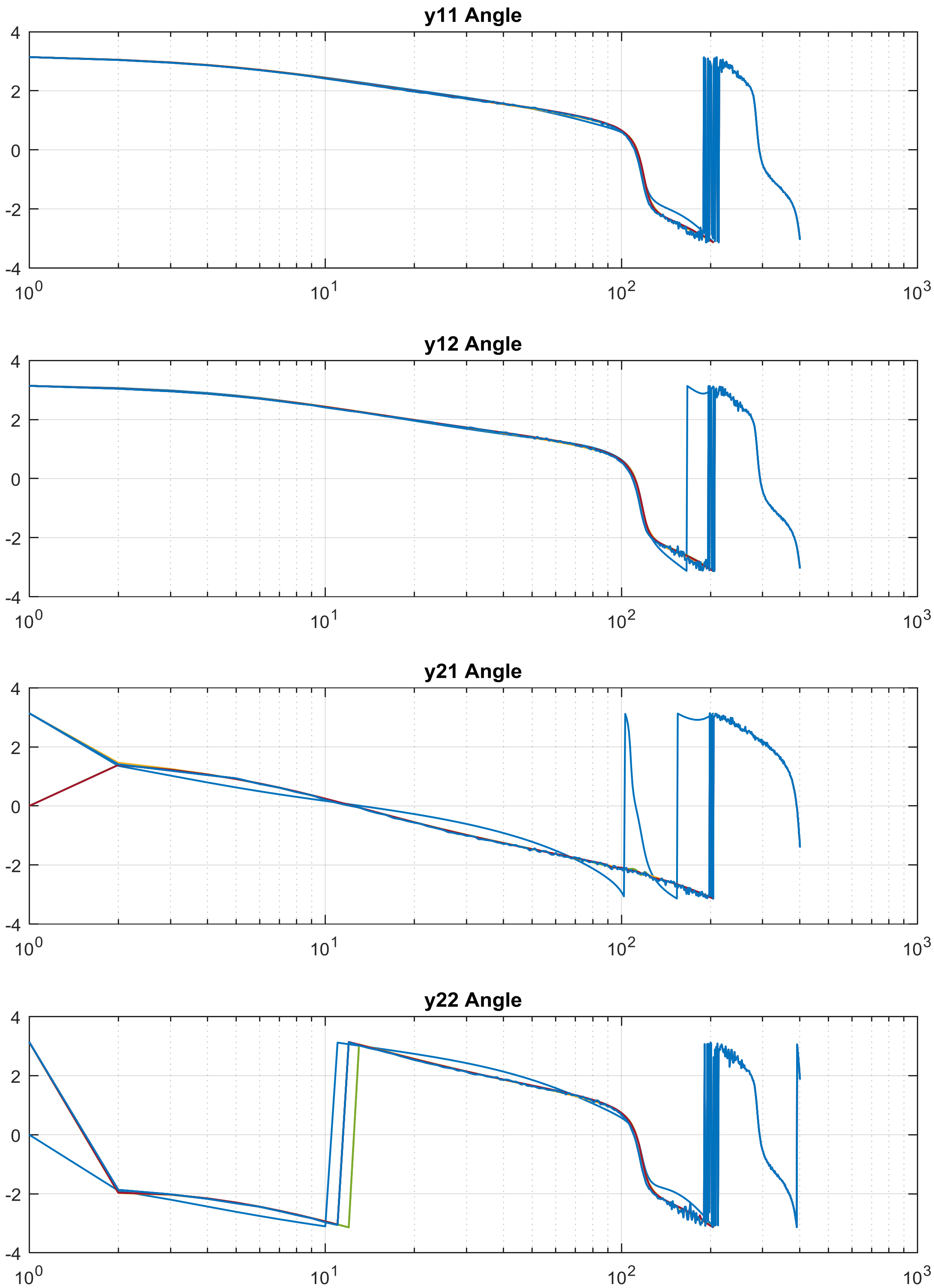

Figure 6. Phase Plot

We can see the three of the models (ns = {7, 10, 20}) match up very well with the empirical frequency response result, only ns = 6 doesn't.

<u>Task #2:</u>

Consider the 7-state model. We first confirm the normal rank of

$$\mathrm{rank}[S(\alpha)] = \mathrm{rank}\begin{bmatrix} \alpha I - A & -B \\ C & D \end{bmatrix} = 9$$

that is $\mathrm{rank}[S(\alpha)] = 9$ for almost all $\alpha \in \mathbf{C}$. A transmission zero of the system occurs at $z \in \mathbf{C}$ when $\mathrm{rank}[S(\alpha)] < 9$. In this case there exist non-zero vectors $c \in \mathbf{C^2}$ and $w \in \mathbf{C^2}$ such that

$$[S(\alpha)]\begin{bmatrix} c \\ w \end{bmatrix} = \begin{bmatrix} \alpha I - A & -B \\ C & D \end{bmatrix}\begin{bmatrix} c \\ w \end{bmatrix} = 0$$

The zeros can be computed from a generalized eigenvalue problem,

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

Problem 2-1:

There are only five finite transmission zeros of the 7-state model, with the remaining four zeros given as infinity:

$$z_p = [-2.5267 \quad 0.9964 \quad 0.8273 \quad -0.3224 \quad -0.2137]$$

$$p = 1,2,3,4,5$$

$|z_p| > 1$ is called an unstable zero because the input it generates is unbounded.

Problem 2-2:

The eigenvalues and transmission zeros of the 7-state model are graphed in the complex plane in the following plot.

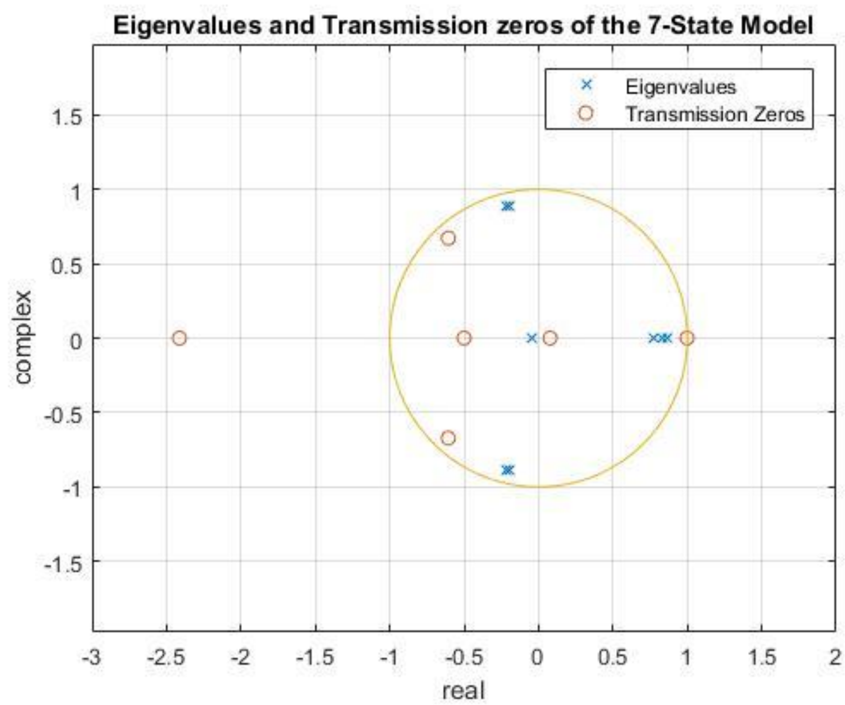

Figure 7. Eigenvalues and transmission zeros of the 7-state model

Problem 2-3:

Because there are two eigenvalues with imaginary part, we convert them from discrete time model to continuous time model

$$\lambda_c = \frac{\log(\lambda_d)}{ts}$$

and we get:

$$\lambda_1 = -3.6842 \pm 71.1394i$$

$$\lambda_2 = -3.58 \pm 72.2737i$$

There are **2** damped oscillators, their natural frequencies are:

$$\omega_1 = \sqrt{(-3.6842)^2 + (71.14i)^2} = 71.24 \text{ rad/s}$$

$$\omega_2 = \sqrt{(-3.58)^2 + (72.27i)^2} = 72.36 \text{ rad/s}$$

The calculated frequency is same as the natural frequency in the frequency response.

Problem 2-4:

The eigenvalues and transmission zeros are graphed in each individual channel below:

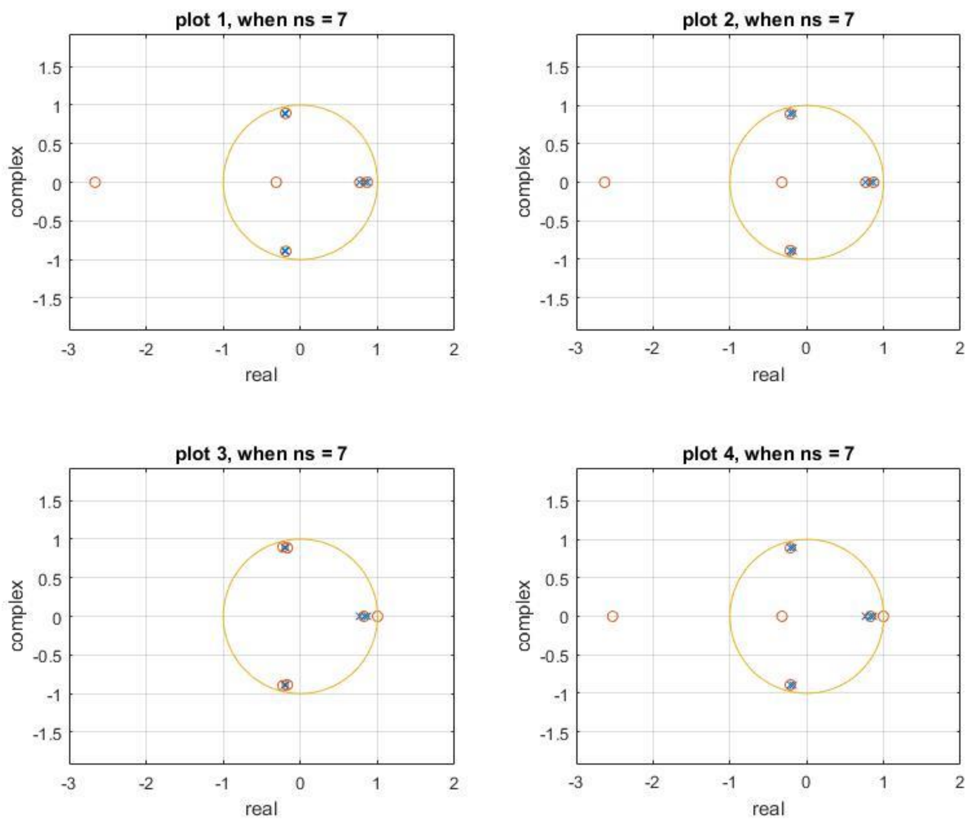

Figure 8. Eigenvalues and transmission zeros for each channel of the 7-state model

Their Hankel singular values are plotted in the following figures:

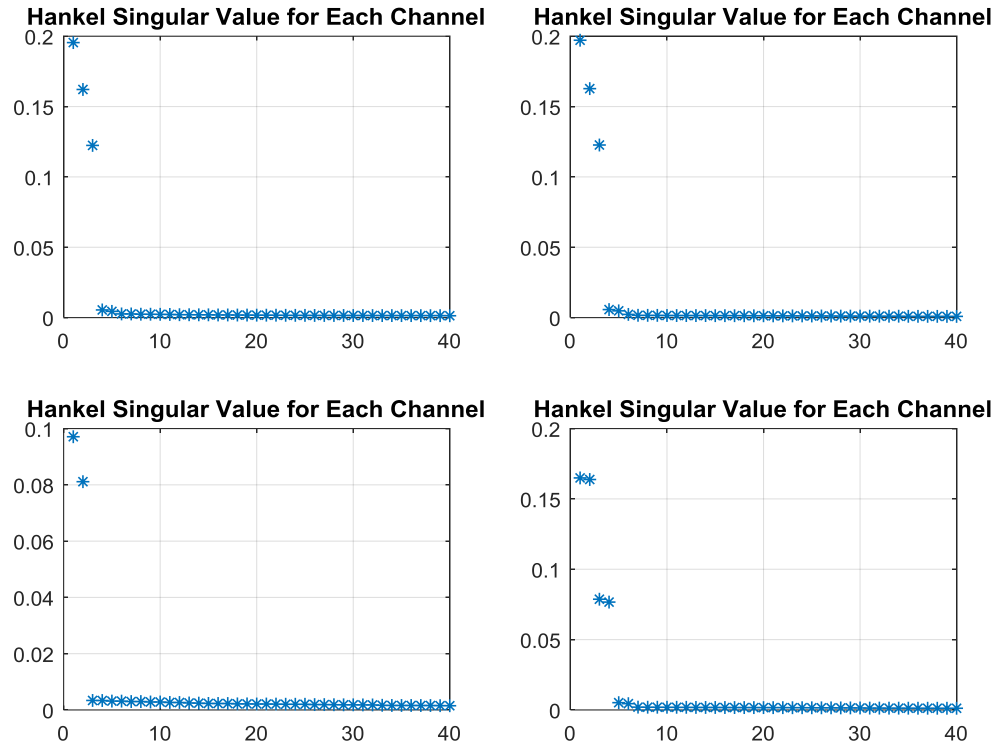

Figure 9. Hankel Singular Value for Each Channel

Problem 2-5:

Another pole-zero plot is created for 8-state model:

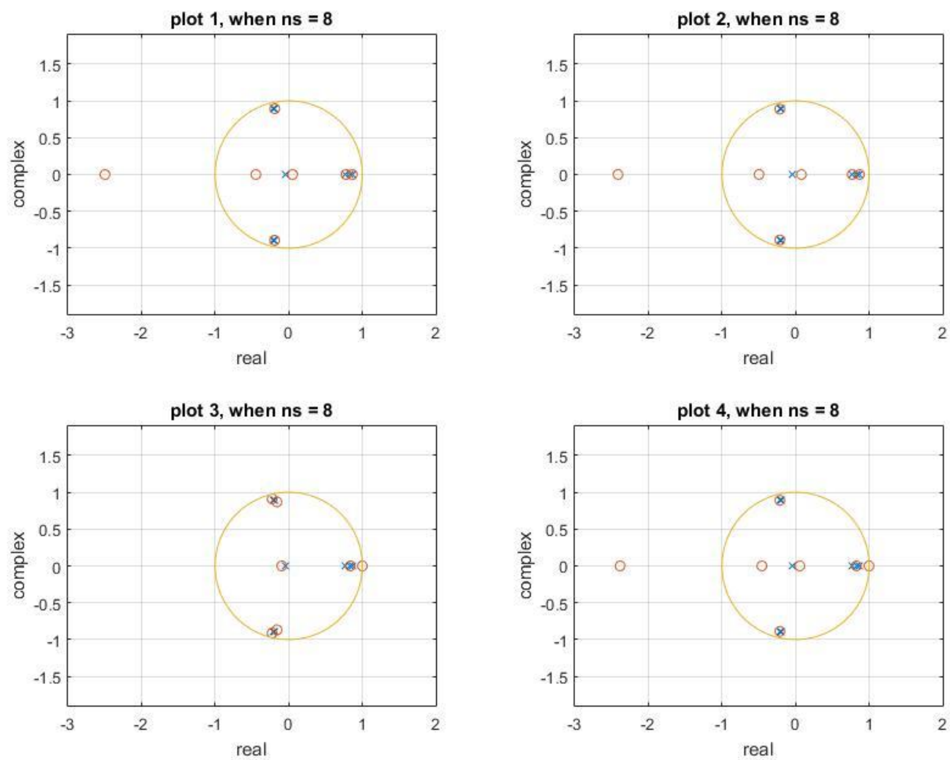

Figure 10. Eigenvalues and transmission zeros of the 8-state model

From the plot we can see that there are added pole-zero pair exists at the origin. Furthermore, added pole is accompanied by zero in close proximity and that this occurs for all channels.

Task #3:

There appear two oscillators and tree poles in the system:

$$OSC1 = -0.1881 \pm 0.8924i$$

$$OSC2 = -0.2138 \pm 0.8890i$$

$$LP1 = 0.7699$$

$$LP2 = 0.8258$$

$$LP3 = 0.8686$$

The block diagram is graphed by analyzing the pole-zero cancellation, for example in u1y1 plot:

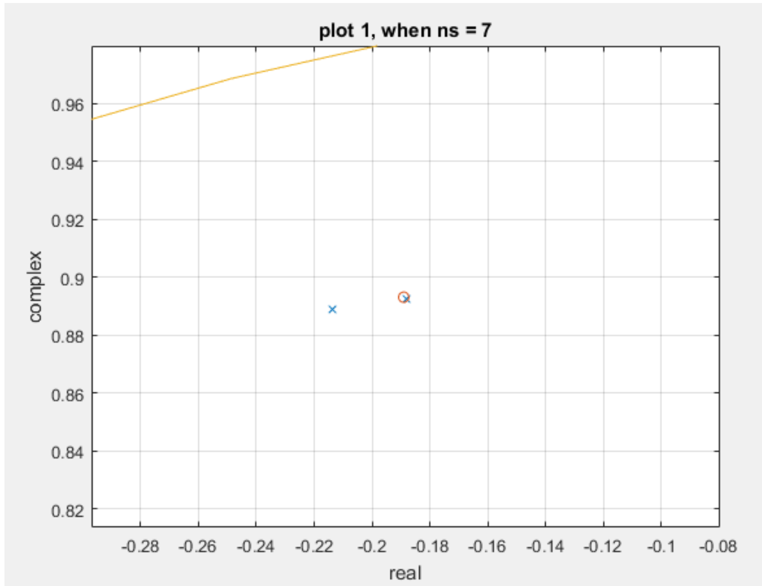

Figure 11. Pole-zero Cancellation for U1Y1

From the figure above, we can observe that the OSC1 is cancelled, and only OSC2 remains. The OSCs and LPs for each channel is generalized in the following table:

y1	y1	y2	y2
OSC2 = -0.2138+-0.8890i	OSC1 = -0.1881+-0.8924i	LP1 = 0.7699	OSC1 = -0.1881+-0.8924i
LP2 = 0.8258	LP2 = 0.8258	LP3 = 0.8686	LP1 = 0.7699
			LP3 = 0.8686
u1	u2	u1	u2

Figure 12. OSCs and LPs for each Channel

The block diagram is shown as following:

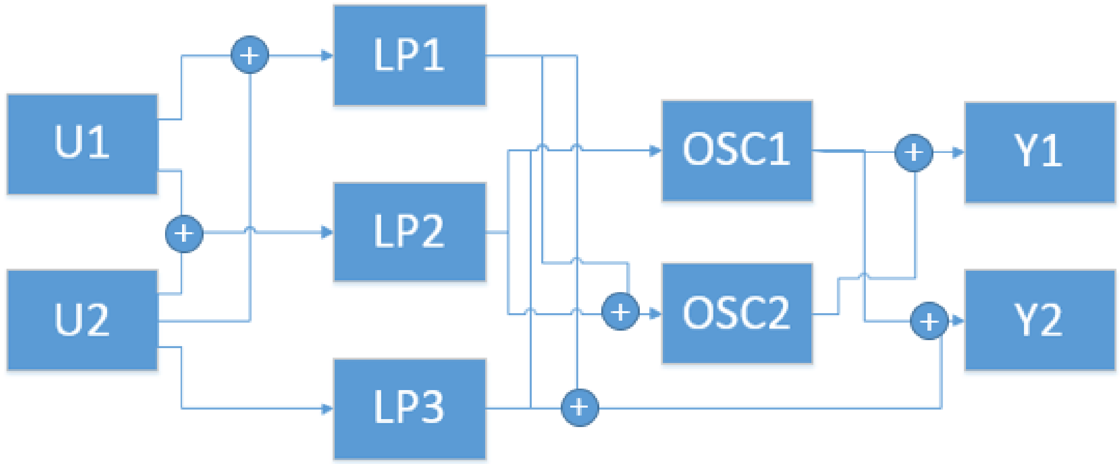

Figure 13. Block Diagram

The topology is not uniquely determined since the order of OSCs and LPs can't be decided.
Some channels have a zero at s = 1 this zero can be combined with one of the low-pass poles to create a high-pas filter. The low pass poles can be pair **LP1 or LP3** can be paired with this zero to make the high-pass filter.
The effects of the high-pass filter can be observed in the by the sharp rise in the u1y2 channel.

Task #4:

Problem 4-1:

We first verify that each input sequence is approximately zero mean. The average for $u1$ and $u1$ are:

$$E[u1] = -0.000966$$
$$E[u2] = 0.0013$$

Which are approximately zero.

Problem 4-2:

Estimates of the four entries of R_{uu} for indices $k \in [-200,\ 200]$ the entries versus lag factor $\tau \in [-5, 5]$ seconds are plotted below:

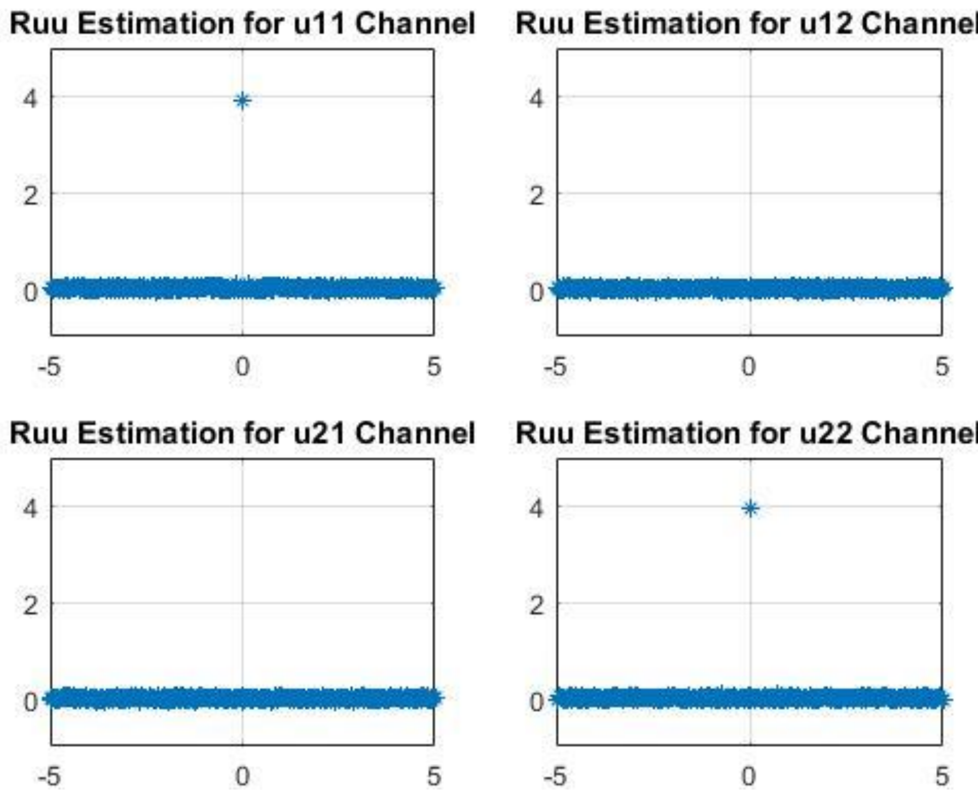

Figure 14. Approximation of four entries of R_uu

Problem 4-3:

Using the Auto-correlation formula of u:

$$R_{uu}[k] = \lim_{p\to\infty} \frac{1}{2p} \sum_{q=-p}^{p} u_{k+q} u_q^T \in \mathbf{R}^{n\times n}$$

here we set $p = 12{,}000$, which is almost half of the length of input.

We calculate the R_{uu} in Matlab, the result is:

$$R_{uu}[2] = \begin{bmatrix} 3.9856 & 0.0437 \\ 0.0437 & 4.0194 \end{bmatrix}$$

Which is approximately

$$\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$$

As desired.

Problem 4-4:

The variance of each channel of u is 4. We first estimate $R_{yu}[k]$ for $\tau \in [-0.2,2]$ seconds and then the first column of R_{yu} normalized by the variance of the first channel of u is graphed. We then graphed the second column of R_{yu} normalized by the variance of the second channel of u. Comparing results to the experimental impulse response obtained from the pulse response experiment, we can find they are basically the same.

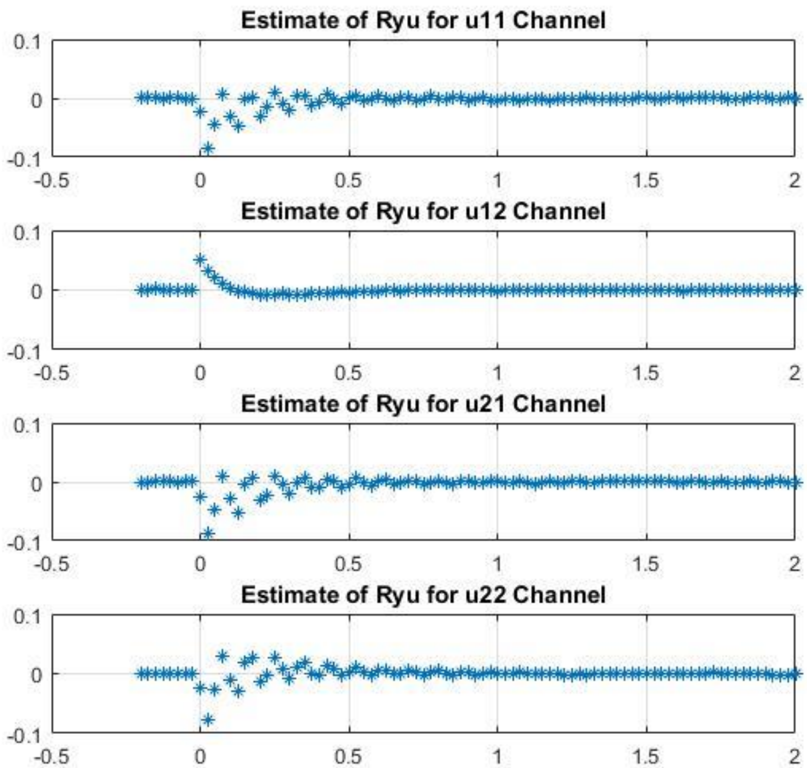

Figure 15. Estimation of Ryu

Task #5:

Problem 5-1:

The RMS value of the scaled output data y is calculated using:

$$\|y\|_{RMS}^{2} = \operatorname{tr}\left(\sum_{q=-\infty}^{\infty} h_q h_q^T\right)$$

The calculated RMS value is:

$$\|y\|_{RMS}^{1} = 0.2237$$

Problem 5-2:

$\|P\|_{H_2}$, where P is the 7-state model derived from the Hankel matrix analysis.

$$\|P\|_{H_2}^{2} = \operatorname{tr}(B^T G_O(\infty) B) = 0.2297$$

$$\|P\|_{H_2}^2 = \mathrm{tr}(CG_c(\infty)C^T) = 0.2297$$

The two values are the same.

Problem 5-3:

Approximate $\|P\|_{H_2}$ from equation below:

$$\|P\|_{H_2}^2 = \sum_{k=0}^{\infty} \|h_k\|_F^2 = 0.2298$$

The 4 calculated values from Task #5 are the same.

<u>Task #6:</u>

Problem 6-1:

Here we wright a Matlab function that accepts as inputs the state-space matrices of continuous-time system, upper and lower limits for the γ search, a tolerance, and the sample period, and then returns the H_∞ norm of the system computed to within the specified tolerance, and the approximate frequency at which the maximum gain is achieved.

$$D_\gamma = \gamma^2 I - D^* D$$

$$A_{clp} = \begin{bmatrix} A + BD_\gamma^{-1}D^*C & -BD_\gamma^{-1}B^* \\ C^*C + C^*DD_\gamma^{-1}D^*C & -A^* - C^*DD_\gamma^{-1}B^* \end{bmatrix}$$

Pseudocode:

- While count < Max Iterations
- Compute eigenvalue of Matrix
- Check for the existence of purely imaginary eigenvalue for A_{clp}
- If the purely imaginary eigenvalue exists, set <u>*lower bound*</u> = γ
- If not, set <u>*upper bound*</u> = γ
- Compute the frequency

The plot is shown in the following figure:

```
function [gam,fre] = hinfnormc(ll,ul,tolerance,A,B,C,D)
I        = eye(length(D));
ww       = 0; %number of lo with good
q        = 0;
N        = 1;
Nmax     = 1000;
zer      = 1e-08;
check    = 0;
while N < Nmax
    gam      = (ll+ul)/2;
    D1       = (gam)^2*I-D'*D;
    a        = A+B*inv(D1)*D'*C;
    b        = -B*inv(D1)*B';
    c        = C'*C+C'*D*inv(D1)*D'*C;
    d        = -A'-C'*D*inv(D1)*B';
    Aclp     = [a b; c d];
    reale    = real(eig(Aclp));
    image    = imag(eig(Aclp));
    eigd     = max(image);
    if(ul-ll)/2 < tolerance
        return
    end

    for ii = (1: length(real(eig(Aclp))))
        if (abs(reale(ii)) < zer && image(ii) > zer)
            check = 1;
        end
    end
    N   = N+1;

    if check == 1
    ll  = gam;
    else
        ul = gam;
    end
    check    = 0;
    term = (1+1j*eigd)/(1-1j*eigd);
    fre = log(term)/(ts*1j);
end
```

Figure 16. Function for Continuous-time System

Problem 6-2:

In order to compute the H_∞ for discrete time system. We first convert the state-space matrices into the continuous-time form and then apply the bisection search procedure over γ:

$$A_c = -(I + A)^{-1}$$

$$B_c = \sqrt{2}(I + A)^{-1}B$$

$$C_c = \sqrt{2}C(I + A)^{-1}$$

$$D_c = D - C(I + A)^{-1}B$$

```
function [gam,fre] = hinfnormd(ll,ul,tolerance,A,B,C,D,ts)
Ia  = eye(length(A));
Ac  = -(Ia-A)/(Ia+A);
Bc  = sqrt(2)*inv(Ia+A)*B;
Cc  = sqrt(2)*C*inv(Ia+A);
Dc  = D-C*inv(Ia+A)*B;

[gam,fre] = hinfnormc(ll,ul,tolerance,Ac,Bc,Cc,Dc);
end
```

Figure 17. Function for Discrete-time System

Problem 6-3:

Here we compute the H_∞ norm of the identified discrete-time model.

$$\gamma = 0.4693$$

$$\omega = 71.7152 \text{ rad/s}$$

Problem 6-4:

Here we compute the discrete-time frequency response of the model in the interval$[0, \omega_{nyq}]$, and then plot the singular values of the frequency response. The singular values I computed form the empirical frequency response data is overlaid.

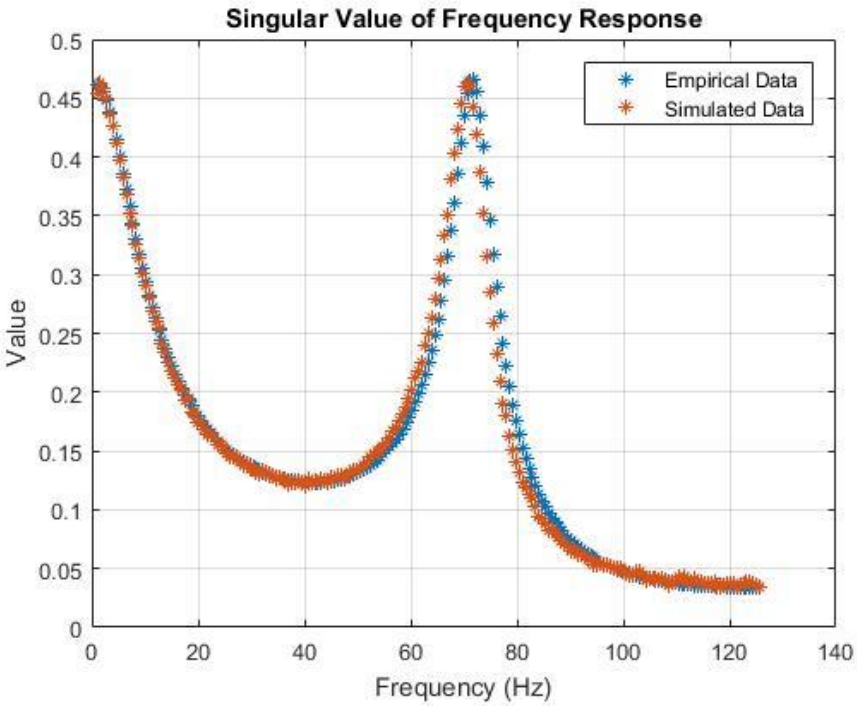

Figure 18. Singular Values of Frequency Responses

From the plots, we can observe that the model singular values are very close to the empirical singular values. The H_∞ norm computed from previous question locates the magnitude and corresponding frequency where the maximum singular value achieves its largest value.

Appendix

```
clc; clear all; close all
tic
% ns = 7 for Task 2
for iii = 2:2
%% ns
ns      = [6,8,10,20];
ns      = ns(iii);

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
[m,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;

%%% 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/max(u1);
y12     = y12/max(u2);
y21     = y21/max(u1);
y22     = y22/max(u2);
u1      = u1/max(u1);
u2      = u2/max(u2);
ts      = 1/40; %%%% sample period
N       = length(y11); %%%% length of data sets
t       = [0:N-1]*ts - 1;
%% Task1_p1: Construct Hankel Matrix
cl      = 100;
bi      = 41;
H       = zeros(cl*2);
for i = 1:cl
   for j = 1:cl
      k   =i+j-1;
      H(2*i-1,j*2-1)=y11(k+bi); %hk(1,1)
      H(2*i-1,j*2)=y12(k+bi);   %hk(1,2)
      H(2*i,j*2-1)=y21(k+bi); %hk(2,1)
      H(2*i,j*2)=y22(k+bi); %hk(2,2)
   end
end
x       = 1:cl;
[U,S,V] =svd(H);
[T d]   = eig(H);

s       = diag(S);

[U,S,V] =svd(H);
Si      = zeros(cl*2);
for i = 1:ns
   Si(i,i) = S(i,i);
end
O       = U;
C       = Si*V';
Hh      = zeros(cl*2);

for i = 1:cl
   for j = 1:cl
      k   =i+j;
      Hh(2*i-1,j*2-1)=y11(k+bi); %hk(1,1)
      Hh(2*i-1,j*2)=y12(k+bi);   %hk(1,2)
      Hh(2*i,j*2-1)=y21(k+bi); %hk(2,1)
      Hh(2*i,j*2)=y22(k+bi); %hk(2,2)
   end
end
U       = U(:,1:ns);
V       = V(:,1:ns);
Si      = Si(1:ns,1:ns);

A       = U'*Hh*(V*inv(Si));
B       = C(1:ns,1:2);
C       = O(1:2,1:ns);
D       = zeros (2);


h = zeros(2,2*cl);
   for k = 1:cl
      a          = C*A^(k-1)*B;
      h(1,2*k-1)  = a(1,1);
      h(2,2*k-1)  = a(2,1);
      h(1,2*k)    = a(1,2);
      h(2,2*k)    = a(2,2);

      qa(k+bi)    = a(1,1);
      qb(k+bi)    = a(2,1);
      qc(k+bi)    = a(1,2);
      qd(k+bi)    = a(2,2);
   end

%Stability Check
scheck      = max(abs(eig(A)));
%% Task1_p2: Plot Response

%plot1(u1,y11,y21,y12,y22,u2,qa,qb,qc,qd,t)

%% Task1_p3:
I   = eye(ns);
%sample period
ts  = 1/40; %s
%Nyquist frequency
is  = 205;

w   = 20; %hz
w   = w*2*pi; %rad/s
w   = linspace(0,w,is);
% model frequency resposne
t   = 0;

for i = 1:is %size(100)
   fr_s    = (C*inv(exp(1j*w(i)*ts)*I-A)*B); %+D Transfer
Function
   a11(i)  = fr_s(1,1);
   a12(i)  = fr_s(1,2);
   a21(i)  = fr_s(2,1);
   a22(i)  = fr_s(2,2);
   frs_s{i}   = svd(fr_s);
end
      mag_a11 = abs(a11);mag_a12 = abs(a12);mag_a21 =
abs(a21);mag_a22 = abs(a22);
      ang_a11 = angle(a11);ang_a12 = angle(a12); ang_a21
= angle(a21); ang_a22 = angle(a22);
```

```
    y11f    = fft(y11)./fft(u1);
    N       = length(y11f);
    om1     = [0:N-1]/(ts*N);
    y21f    = fft(y21)./fft(u1);
    N       = length(y21f);
    om2     = [0:N-1]/(ts*N);
    y12f    = fft(y12)./fft(u2);
    N       = length(y12f);
    om3     = [0:N-1]/(ts*N);
    y22f    = fft(y22)./fft(u2);
    N       = length(y22f);
    om4     = [0:N-1]/(ts*N);

     %plot2(mag_a11,mag_a12,mag_a21,mag_a22,y11f,y1
2f,y21f,y22f,ang_a11,ang_a12,ang_a21,ang_a22)
end
for i = 1:is
  frs_e{i} = svd ([y11f(i)    y12f(i);
                  y21f(i)    y22f(i)]);
end
%% Task 2_p1:
%confirm rank(S) = 9
for al = 1:100
  S1 = [   al*eye(ns)-A       -B;
           C                D];
  rank(S1);
end

S2  =[  A    B;
    -C   -D];
I   = [eye(size(A))  zeros(ns,2);
     zeros(2,ns) zeros(2,2)];
[v2 d2] = eig(S2,I);
%% Task 2_p2:
figure
plot(eig(A) ,'x')
hold on
plot(diag(d2),'o')
xlim([-3 2])

xlabel('real')
ylabel('complex')
axis equal
grid on
circle(1)
title('Eigenvalues and Transmission zeros of the 7-State
Model')
%% Task 2_p3:

for jj = 1:4

B1  = B(:,1);
B2  = B(:,2);
C1  = C(1,:);
C2  = C(2,:);
D1  = [0];

%     cl  = 100;
%     H   = zeros(2*cl);
%     for i = 1:cl
%         for j   = 1:cl
%             k   = i+j-1;
%             H(2*i-1,2*j-1)  =y(k+bi);
%             H(2*i-1,2*j)    =y(k+bi);
%             H(2*i,2*j-1)    =y(k+bi);
%             H(2*i,2*j)      =y(k+bi);
%         end
%     end
%
%     H   = downsample(H,2);
%     H   = downsample(H',2);
%     H   = H';

%     x   = 1:cl;
%     [U,S,V]=svd(H);
%     [T d] = eig(H);
%     Si  =diag(S);
%     Si  = diag(Si(1:ns));
%     U   = U(:,1:ns);
%     V   = V(:,1:ns);
%     O   = U;
%     C   = Si*V';

% Hh = zeros(cl*2);
% for i = 1:cl
%     for j = 1:cl
%         k=i+j;
%         Hh(2*i-1,j*2-1)=y(k+bi); %hk(1,1)
%         Hh(2*i-1,j*2)=y(k+bi);   %hk(1,2)
%         Hh(2*i,j*2-1)=y(k+bi); %hk(2,1)
%         Hh(2*i,j*2)=y(k+bi); %hk(2,2)
%     end
% end
% Hh = downsample(Hh,2);
% Hh = downsample(Hh',2);
% Hh = Hh';
% Define New A,B,C, and D


switch jj
  case 1
    S2=[ A      B1;
        -C1    -D1];
  case 2
    S2=[ A      B2;
       -C1     -D1];
  case 3
    S2 =[A      B1;
        -C2    -D1];
  case 4
    S2 = [A     B2;
        -C2    -D1];
end

I = [eye(size(A))  zeros(ns,1);
    zeros(1,ns)    zeros(1,1)];
[v2 d2] = eig(S2,I);

%% Task2_p4:
subplot(2,2,jj)
plot(eig(A),'x','LineWidth',1);
hold on
plot(diag(d2),'o','LineWidth',1);

xlim([-3 2])
ylim([-1.5 1.5])
xlabel('real')
ylabel('complex')
title(['plot ' num2str(jj) ', when ns = ' num2str(ns)])
axis equal
grid on
circle(1);
hold on
end
%% Task4_p1
load u_rand.mat
```

```
y1 = u_rand.Y(3).Data;
y2 = u_rand.Y(4).Data;
u1 = u_rand.Y(1).Data;
u2 = u_rand.Y(2).Data;

N = length(y1);
t = [0:N-1]*ts - 1;

p = 12000;
u = [u1; u2];
y = [y1; y2];

mean(u1);
mean(u2);

%% Task4_p2:
% a = zeros(1,401);b = zeros(1,401);c = zeros(1,401);d =
zeros(1,401);
%
% for k = 1:401
%     sm = zeros(2);
%     for q = 202:2*p-202
%         sm = sm + u(:,q+k-201)*(u(:,q))' ;
%     end
%     matr = 1/(2*p)*sm;
%     a(k) = matr(1,1);b(k) = matr(1,2);c(k) = matr(2,1);d(k) =
matr(2,2);
% end
%
% figure
% x = linspace(-5,5,401);
% plot (x,a,'*','LineWidth',1)
% hold on
% plot (x,b,'*','LineWidth',1)
% hold on
% plot (x,c,'*','LineWidth',1)
% hold on
% plot (x,d,'*','LineWidth',1)
% hold on

%% Task4_p3
% Ruu = [0 0 ; 0 0];
% for q = 1:24001
%     Ruu = (Ruu + u(:,q)*(u(:,q))');
% end
% (1/(2*p))*Ruu

%% Task4_p4
% a = zeros(1,89);b = zeros(1,89);c = zeros(1,89);d =
zeros(1,89);
%
% for k = 1:89
%     sm = zeros(2);
%     for q = 89:2*p-89
%         sm = sm + y(:,q+k-8)*(u(:,q))' ;
%     end
%     matr = 1/(2*p)*sm;
%
%     a(k) = matr(1,1)/var(u1);b(k) = matr(1,2)/var(u2);c(k) =
matr(2,1)/var(u1);d(k) = matr(2,2)/var(u2);
% end
%
% x = linspace(-0.2,2,89);
% figure
% subplot(4,1,1)
% plot (x,a,'*')
% hold on
% grid on
%
% subplot(4,1,2)
% plot (x,c,'*')
% hold on
% grid on
%
% subplot(4,1,3)
% plot (x,b,'*')
% hold on
% grid on
%
% subplot(4,1,4)
% plot (x,d,'*')
% hold on
% grid on

%% Task5_p1

ns  = 7;
cl  = 100;
bi  = 41;
H   = zeros(cl*2);
for i = 1:cl
    for j = 1:cl
            k=i+j-1;
            H(2*i-1,j*2-1)=y11(k+bi); %hk(1,1)
            H(2*i-1,j*2)=y12(k+bi);   %hk(1,2)
            H(2*i,j*2-1)=y21(k+bi); %hk(2,1)
            H(2*i,j*2)=y22(k+bi);
    end
end

x       = 1:cl;
[U,S,V] =svd(H);
[T d]   = eig(H);
Si      = zeros(cl*2);
for i = 1:ns
    Si(i,i) = S(i,i);
end
O = U;
C = Si*V';

Hh = zeros(cl*2);
for i = 1:cl
    for j = 1:cl
            k=i+j;
            Hh(2*i-1,j*2-1)=y11(k+bi); %hk(1,1)
            Hh(2*i-1,j*2)=y12(k+bi);   %hk(1,2)
            Hh(2*i,j*2-1)=y21(k+bi); %hk(2,1)
            Hh(2*i,j*2)=y22(k+bi); %hk(2,2)
    end
end
U       = U(:,1:ns);
V       = V(:,1:ns);
Si      = Si(1:ns,1:ns);

A       = U'*Hh*(V*inv(Si));
B       = C(1:ns,1:2);
C       = O(1:2,1:ns);
D       = zeros(2);

load u_rand.mat
y1 = u_rand.Y(3).Data/2;
y2 = u_rand.Y(4).Data/2;
u1 = u_rand.Y(1).Data/2;
u2 = u_rand.Y(2).Data/2;

ts = 1/40;
```

```
N = length(y1);
t = [0:N-1]*ts - 1;

p = 12000;
u = [u1; u2];
y = [y1; y2];

%%Problem 1
Ruu     = [0 0 ; 0 0];
for q = 1:24001
    Ruu = Ruu + y(:,q)*(y(:,q))';
end
y_rms   = sqrt(diag((1/(2*p))*Ruu));
RMS     = norm(y_rms);

%% Task5_p2
sys     = ss(A,B,C,D,ts);
Gc      = gram(sys,'c');
Go      = gram(sys,'o');
PH2_1   = norm(sqrt(diag(B'*Go*B)));
PH2_2   = norm(sqrt(diag(C*Gc*C')));

%% Task5_p3
%experimental norm
 PH2_3  = 0;
for i = 1:401
    n = norm([y11(i) y12(i);y21(i) y22(i)],'fro');
    PH2_3 = PH2_3+n^2;
end
PH2_3   = sqrt(PH2_3);
PH2_3   = norm(PH2_3);

%% Task6_1
ll  = 0.0000; %lower limit
ul  = 5; %upper limit
tolerance = 1e-5;

[gam_c,eigd]= hinfnormc(ll,ul,tolerance,A,B,C,D);
[gam_d,fre] = hinfnormd(ll,ul,tolerance,A,B,C,D,ts);

%Hankel singular value
max(eig(Gc*Go));

%% Task6_4
% Singular value of the frequency response.
    frs_s   = cell2mat(frs_s);
    frs_e   = cell2mat(frs_e);
    figure
    plot(frs_s(1,:),'*')
    hold on
    plot(frs_e(1,:),'*')
toc
```

```
function [gam,eigd] =
hinfnormc(ll,ul,tolerance,A,B,C,D)
I       = eye(length(D));
ww      = 0; %number of lo with good
q       = 0;
N       = 1;
Nmax    = 1000;
zer     = 1e-08;
check   = 0;
while N < Nmax
    gam     = (ll+ul)/2;
    Dl      = (gam)^2*I-D'*D;
    a       = A+B*inv(Dl)*D'*C;
    b       = -B*inv(Dl)*B';
    c       = C'*C+C'*D*inv(Dl)*D'*C;
    d       = -A'-C'*D*inv(Dl)*B';
    Aclp    = [a b; c d];
    reale   = real(eig(Aclp));
    image   = imag(eig(Aclp));
    eigd    = max(image);
    if(ul-ll)/2 < tolerance
        return
    end

    for ii = (1: length(real(eig(Aclp))))
        if (abs(reale(ii)) < zer && image(ii) > zer)
            check = 1;
        end
    end
    N   = N+1;

    if check == 1
    ll  = gam;
    else
        ul = gam;
    end
    check   = 0;
end
```