Amirkabir University of Technology
(Tehran Polytechnic)

Electrical Engineering Faculty

Control Department

MSc Program

Assignment 03

through

System Identification

Course

by

Mohammad Azimi - 402123100

mohammadazimi2000@aut.ac.ir

Course Lecturer

Dr. Mehdi Karrari

Question 1

a

The system dynamics are extracted as follows:

$$u - k_2 y_2 + k_2 y_1 - b_2 \dot{y}_2 + b_2 \dot{y}_1 = m_2 \ddot{y}_2$$

$$k_2 y_2 - k_2 y_1 + b_2 \dot{y}_2 - b_2 \dot{y}_1 - k_1 y_1 - b_1 \dot{y}_1 = m_1 \ddot{y}_1$$

Taking Laplace transform from both equations gives us:

$$\begin{Bmatrix} -(k_2 + b_2 s)Y_1 + (k_2 + b_2 s + m_2 s^2)Y_2 & = U \\ -(k_2 + b_2 s + k_1 + b_1 s + m_1 s^2)Y_1 + (k_2 + b_2 s)Y_2 & = 0 \end{Bmatrix}$$

Hence, the transfer function of the system for Y_1 *and* U derives as:

$$\frac{Y_1}{U} = \frac{k_2 + b_2 s}{(m_1 m_2)s^4 + (b_1 m_2 + b_2 m_1 + b_2 m_2)s^3 + (k_1 m_2 + k_2 m_1 + k_2 m_2 + b_1 b_2)s^2 + (b_2 k_1 + b_1 k_2)s^1 + (k_1 k_2)}$$

Accordingly, the controllable canonical state space form of the system could be represented as:

$$\dot{X} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{(k_1 k_2)}{m_1 m_2} & -\frac{(b_2 k_1 + b_1 k_2)}{m_1 m_2} & -\frac{(k_1 m_2 + k_2 m_1 + k_2 m_2 + b_1 b_2)}{m_1 m_2} & -\frac{(b_1 m_2 + b_2 m_1 + b_2 m_2)}{m_1 m_2} \end{bmatrix} X + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} U$$

$$Output = [k_2 \quad b_2 \quad 0 \quad 0]X$$

```
clear; clc;

load t
load u
load y

syms m1 m2 b1 b2 k1 k2 s

% Check out the REPORT file for the dynamics derives of the system.

A = [0, 1, 0, 0;
     0, 0, 1, 0;
     0, 0, 0, 1;
     -(k1*k2)/(m1*m2), -(b2*k1 + b1*k2)/(m1*m2), -(m2*(k1 + k2) + m1*k2 + b1*b2)/(m1*m2),
-(m2*(b1 + b2) + m1*b2)/(m1*m2)];
B = [0; 0; 0; 1];
C = [k2/(m1*m2), b2/(m1*m2), 0, 0];

%%
% Forward Euler Discretization
```

```
T=0.1
```

```
T = 0.1000
```

```
Ad = T*A + eye(length(A))
```

Ad =

$$\begin{pmatrix} 1 & \frac{1}{10} & 0 & 0 \\ 0 & 1 & \frac{1}{10} & 0 \\ 0 & 0 & 1 & \frac{1}{10} \\ -\frac{k_1\,k_2}{10\,m_1\,m_2} & -\frac{b_1\,k_2+b_2\,k_1}{10\,m_1\,m_2} & -\frac{m_2\,(k_1+k_2)+b_1\,b_2+k_2\,m_1}{10\,m_1\,m_2} & 1-\frac{b_2\,m_1+m_2\,(b_1+b_2)}{10\,m_1\,m_2} \end{pmatrix}$$

```
Bd = T*B
```

```
Bd = 4×1

        0
        0
        0
   0.1000
```

```
Cd = C
```

Cd =

$$\begin{pmatrix} \frac{k_2}{m_1\,m_2} & \frac{b_2}{m_1\,m_2} & 0 & 0 \end{pmatrix}$$

```
syms z;

% forming the transfer function
G = Cd*inv(z*eye(length(Ad)) - Ad)*Bd;

fprintf("\n\n G = \n")
```

```
 G =
```

```
pretty(G)
```

```
k2   10 b2 (z - 1)

-- + -------------

#1         #1



where



   #1 == 100 b1 b2 - 10 b1 k2 - 10 b2 k1 - 1000 b1 m2 - 1000 b2 m1 - 1000 b2 m2 + k1 k2 + 100 k1 m2 +
100 k2 m1 + 100 k2 m2 + 10000 m1 m2
```

```
                          2               2               3               3               2               3
2              2

         - 3000 b1 m2 z  - 3000 b2 m1 z  + 1000 b1 m2 z  + 1000 b2 m1 z  - 3000 b2 m2 z  + 1000 b2 m2 z  +
100 k1 m2 z  + 100 k2 m1 z
```

```
                         2                 2                 3                 4

         + 100 k2 m2 z  + 60000 m1 m2 z  - 40000 m1 m2 z  + 10000 m1 m2 z  - 200 b1 b2 z + 10 b1 k2 z + 10
b2 k1 z + 3000 b1 m2 z + 3000 b2 m1 z
```

```
                                                                                               2

         + 3000 b2 m2 z - 200 k1 m2 z - 200 k2 m1 z - 200 k2 m2 z - 40000 m1 m2 z + 100 b1 b2 z
```

```
%% System Estimation

N = length(y);
n = 4;
m = 2;
U = arx_U_builder(n,m,u,y);

fprintf("\n\n\n")
fprintf("\n\n U = \n")
```

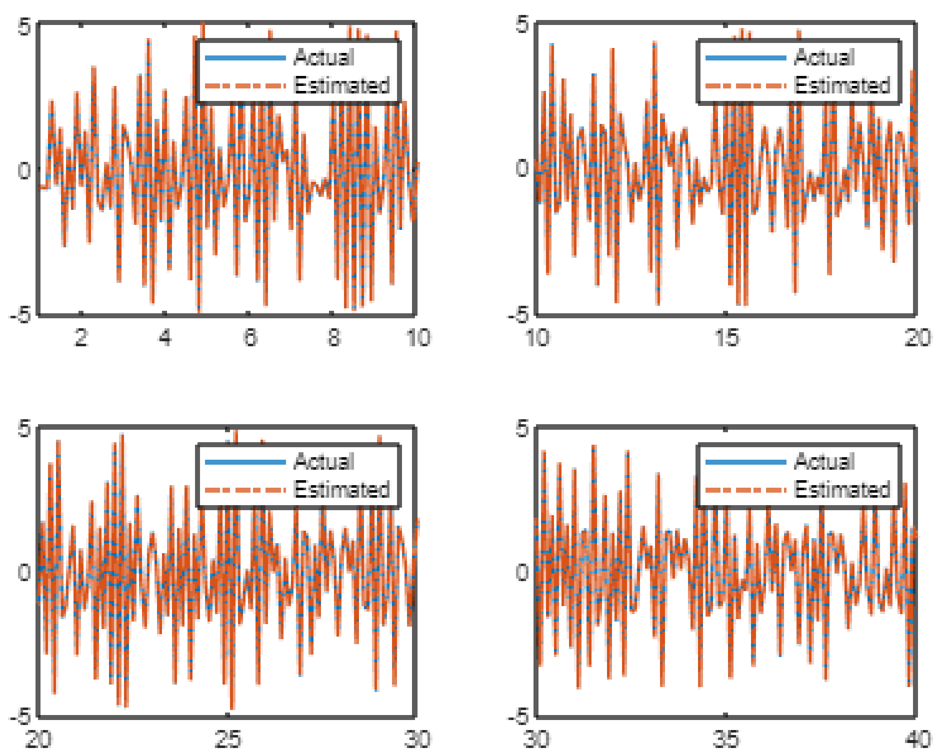
```
 U =
```

```
disp(U(1:10,:))
```

```
         0         0         0         0         0         0
   -0.0037         0         0         0         0         0
   -0.0123   -0.0037         0         0         0         0
    0.0083   -0.0123   -0.0037         0   -1.5000         0
    1.5010    0.0083   -0.0123   -0.0037   -1.5000   -1.5000
    0.0650    1.5010    0.0083   -0.0123   -1.5000   -1.5000
    1.0211    0.0650    1.5010    0.0083   -1.5000   -1.5000
    0.4967    1.0211    0.0650    1.5010   -1.5000   -1.5000
    0.7378    0.4967    1.0211    0.0650   -1.5000   -1.5000
    0.6598    0.7378    0.4967    1.0211   -1.5000   -1.5000
```

```
% Solving for Linear Least Squares
theta_hat = inv(U'*U)*U'*y;
y_hat = U*theta_hat;
```

```
figure()

subplot(2,2,1)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([1,10])

subplot(2,2,2)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([10,20])

subplot(2,2,3)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([20,30])

subplot(2,2,4)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([30,40])
```




```
%% Evaluation Metrics
disp("----------------Model Evaaluation Report-------------------")
```

```
----------------Model Evaaluation Report-------------------
```

```
SST = sum((y-mean(y)).^2);
SSE = sum((y-y_hat).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 0.1486299
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.0001453
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 0.9999739
```

```
disp("============================================================")
```

```
============================================================
```

D:

Yes. Since equations are at least equalt to the number of unkonwn parameters, it is possible to find a unique solution.

Question 2

```
clc; clear;

load t
load u
load y

N = length(y);
% trying orders of 1 to 6
for n=1:6
    m = n;

    U = arx_U_builder(n,m,u,y);

    fprintf("\n\n\n")
    fprintf("\n\n U = \n")

    disp(U(1:10,:))

    rnk = rank(U'*U);
    reference = min(size(U));
    if (rnk == reference)
        temp = n;
        fprintf(">>> Full Rank U for order of %d \n", n)
        disp("------------------------------------------")
    else
        fprintf(">>> Rank Deficient of U for order of %d \n", n)
        disp("------------------------------------------")
    end
end
```

```
 U = 

         0   -1.5000         0

         0   -1.5000   -1.5000

         0   -1.5000   -1.5000

         0   -1.5000   -1.5000

    1.5000   -1.5000   -1.5000

    0.0698   -1.5000   -1.5000

    1.0171   -1.5000   -1.5000

    0.4931   -1.5000   -1.5000

    0.7363   -1.5000   -1.5000

    0.6579   -1.5000   -1.5000

>>> Full Rank U for order of 1

------------------------------------------

 U = 
```

```
         0         0   -1.5000         0         0

         0         0   -1.5000   -1.5000         0

         0         0   -1.5000   -1.5000   -1.5000

         0         0   -1.5000   -1.5000   -1.5000

    1.5000         0   -1.5000   -1.5000   -1.5000

    0.0698    1.5000   -1.5000   -1.5000   -1.5000

    1.0171    0.0698   -1.5000   -1.5000   -1.5000

    0.4931    1.0171   -1.5000   -1.5000   -1.5000

    0.7363    0.4931   -1.5000   -1.5000   -1.5000

    0.6579    0.7363   -1.5000   -1.5000   -1.5000

>>> Full Rank U for order of 2

------------------------------------------

 U =

         0         0         0   -1.5000         0         0         0

         0         0         0   -1.5000   -1.5000         0         0

         0         0         0   -1.5000   -1.5000   -1.5000         0

         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000

    1.5000         0         0   -1.5000   -1.5000   -1.5000   -1.5000

    0.0698    1.5000         0   -1.5000   -1.5000   -1.5000   -1.5000

    1.0171    0.0698    1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.4931    1.0171    0.0698   -1.5000   -1.5000   -1.5000   -1.5000

    0.7363    0.4931    1.0171   -1.5000   -1.5000   -1.5000   -1.5000

    0.6579    0.7363    0.4931   -1.5000   -1.5000   -1.5000   -1.5000

>>> Full Rank U for order of 3

------------------------------------------

 U =

         0         0         0         0   -1.5000         0         0         0         0

         0         0         0         0   -1.5000   -1.5000         0         0         0

         0         0         0         0   -1.5000   -1.5000   -1.5000         0         0

         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000         0

    1.5000         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.0698    1.5000         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000
```

```
    1.0171    0.0698    1.5000         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.4931    1.0171    0.0698    1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.7363    0.4931    1.0171    0.0698   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.6579    0.7363    0.4931    1.0171   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000
```

```
>>> Full Rank U for order of 4
----------------------------------------
 U =

         0         0         0         0         0   -1.5000         0         0         0         0
0

         0         0         0         0         0   -1.5000   -1.5000         0         0         0
0

         0         0         0         0         0   -1.5000   -1.5000   -1.5000         0         0
0

         0         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000         0
0

    1.5000         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000
0

    0.0698    1.5000         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000

    1.0171    0.0698    1.5000         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000

    0.4931    1.0171    0.0698    1.5000         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000

    0.7363    0.4931    1.0171    0.0698    1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000

    0.6579    0.7363    0.4931    1.0171    0.0698   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000
```

```
>>> Rank Deficient of U for order of 5
----------------------------------------
 U =

         0         0         0         0         0         0   -1.5000         0         0         0
0         0         0

         0         0         0         0         0         0   -1.5000   -1.5000         0         0
0         0         0

         0         0         0         0         0         0   -1.5000   -1.5000   -1.5000         0
0         0         0
```

```
         0         0         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000
0         0         0

    1.5000         0         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000         0         0

    0.0698    1.5000         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000   -1.5000         0

    1.0171    0.0698    1.5000         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000   -1.5000   -1.5000

    0.4931    1.0171    0.0698    1.5000         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000   -1.5000   -1.5000

    0.7363    0.4931    1.0171    0.0698    1.5000         0   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000   -1.5000   -1.5000

    0.6579    0.7363    0.4931    1.0171    0.0698    1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -
1.5000   -1.5000   -1.5000

 >>> Rank Deficient of U for order of 6

 ------------------------------------------
```

```
fprintf(">>> So Taking n=%d as the proper rank for matrix U\n", temp)
```

```
 >>> So Taking n=4 as the proper rank for matrix U
```

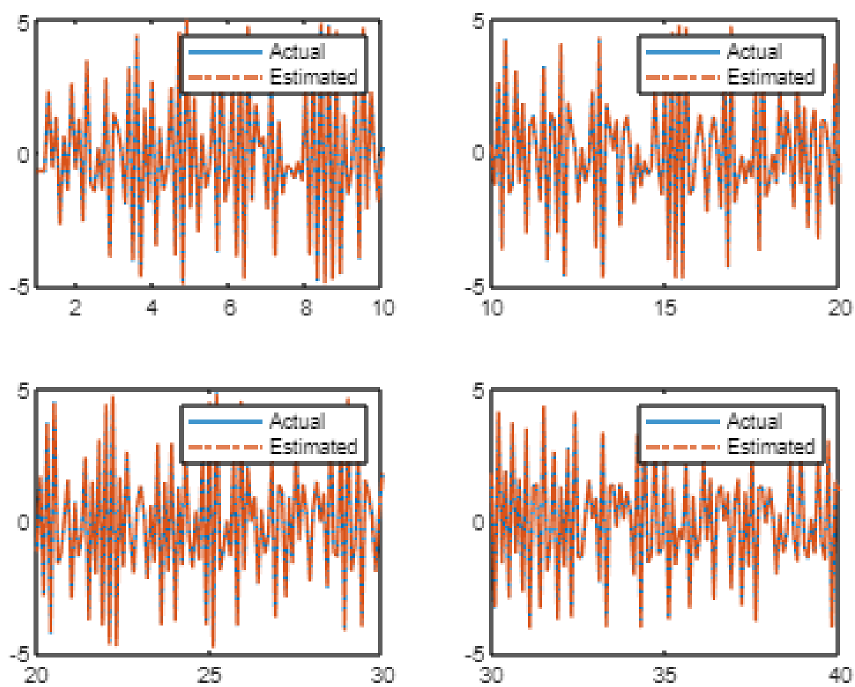
```
n = temp;

m = n;
U = arx_U_builder(n,m,u,y);
disp("U = ")
```

```
 U =
```

```
disp(U(1:10,:))
```

```
         0         0         0         0   -1.5000         0         0         0         0

         0         0         0         0   -1.5000   -1.5000         0         0         0

         0         0         0         0   -1.5000   -1.5000   -1.5000         0         0

         0         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000         0

    1.5000         0         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.0698    1.5000         0         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    1.0171    0.0698    1.5000         0   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.4931    1.0171    0.0698    1.5000   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.7363    0.4931    1.0171    0.0698   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000

    0.6579    0.7363    0.4931    1.0171   -1.5000   -1.5000   -1.5000   -1.5000   -1.5000
```

```
theta_hat = inv(U'*U)*U'*y;
y_hat = U*theta_hat;

figure()

subplot(2,2,1)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([1,10])

subplot(2,2,2)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([10,20])

subplot(2,2,3)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([20,30])

subplot(2,2,4)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([30,40])
```




```
%% Evaluation Metrics

disp("----------------Model Evaaluation Report-------------------")
```

```
----------------Model Evaaluation Report-------------------
```

```
SST = sum((y-mean(y)).^2);
SSE = sum((y-y_hat).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 0.0000000
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.0000000
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 1.0000000
```

```
disp("===========================================================")
```

```
===========================================================
```

The greater the R parameter, the better the model. Accordingly, this method seems to be a better option rather than the previous method.

Question 3

```
clc; clear;

% define the system
s = tf('s');
G = 5*(s - 1)*(s + 1)/(2*s^4 + 14*s^3 + 36*s^2 + 44*s + 24)
```

```
G =


               5 s^2 - 5

  ----------------------------------

  2 s^4 + 14 s^3 + 36 s^2 + 44 s + 24



Continuous-time transfer function.
```

```

% extract system poles
poles = pole(G);
```

The appropriate sampling time would be 10 times of the most dominant pole of the system. Note that the Ts = 1/(10*fs).

```
% find proper sampling time
Ts = 0.1/(abs(max(real(poles))))
```

```
Ts = 0.1000
```

White Noise Input

```

% define time space
t = 0:Ts:50;

% generate system input and output
N = length(t);
u = wgn(N,1,0);
y = lsim(G,u,t);

n=4;
m=3;
U = arx_U_builder(n,m,u,y);
disp("U = ")
```

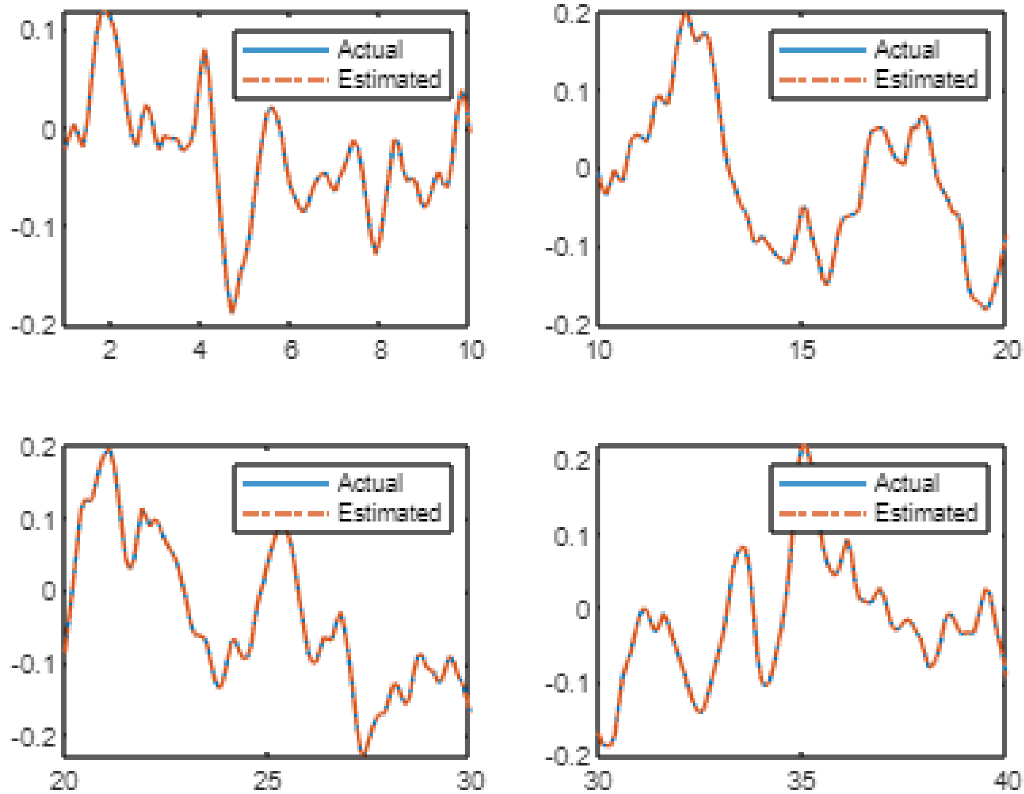
```
U =
```

```
disp(U(1:10,:))
```

```
         0         0         0         0    0.6737         0         0         0
```

```
         0         0         0         0   -0.6691    0.6737         0         0

   -0.0067         0         0         0   -0.4003   -0.6691    0.6737         0

   -0.0076   -0.0067         0         0   -0.6718   -0.4003   -0.6691    0.6737

    0.0023   -0.0076   -0.0067         0    0.5756   -0.6718   -0.4003   -0.6691

    0.0170    0.0023   -0.0076   -0.0067   -0.7781    0.5756   -0.6718   -0.4003

    0.0218    0.0170    0.0023   -0.0076   -1.0636   -0.7781    0.5756   -0.6718

    0.0241    0.0218    0.0170    0.0023    0.5530   -1.0636   -0.7781    0.5756

    0.0378    0.0241    0.0218    0.0170   -0.4234    0.5530   -1.0636   -0.7781

    0.0407    0.0378    0.0241    0.0218    0.3616   -0.4234    0.5530   -1.0636
```

```
theta_hat = inv(U'*U)*U'*y;
y_hat = U*theta_hat;

figure()
subplot(2,2,1)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([1,10])

subplot(2,2,2)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([10,20])

subplot(2,2,3)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([20,30])

subplot(2,2,4)
plot(t,y,t,y_hat,'-.')
legend('Actual','Estimated')
xlim([30,40])
```




```
%% Evaluation Metrics

disp("----------------Model Evaluation Report-------------------")
```

```
----------------Model Evaluation Report-------------------
```

```
SST = sum((y-mean(y)).^2);
SSE = sum((y-y_hat).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 0.0000011
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.0000000
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 0.9999998
```

```
disp("============================================================")
```

```
==========================================================
```

PRBS Input

```
% input & output generation
u2 = prbs(5,N);
y2 = lsim(G,u2,t);

U_prbs = arx_U_builder(n,m,u2,y2);

disp("U = ")
```

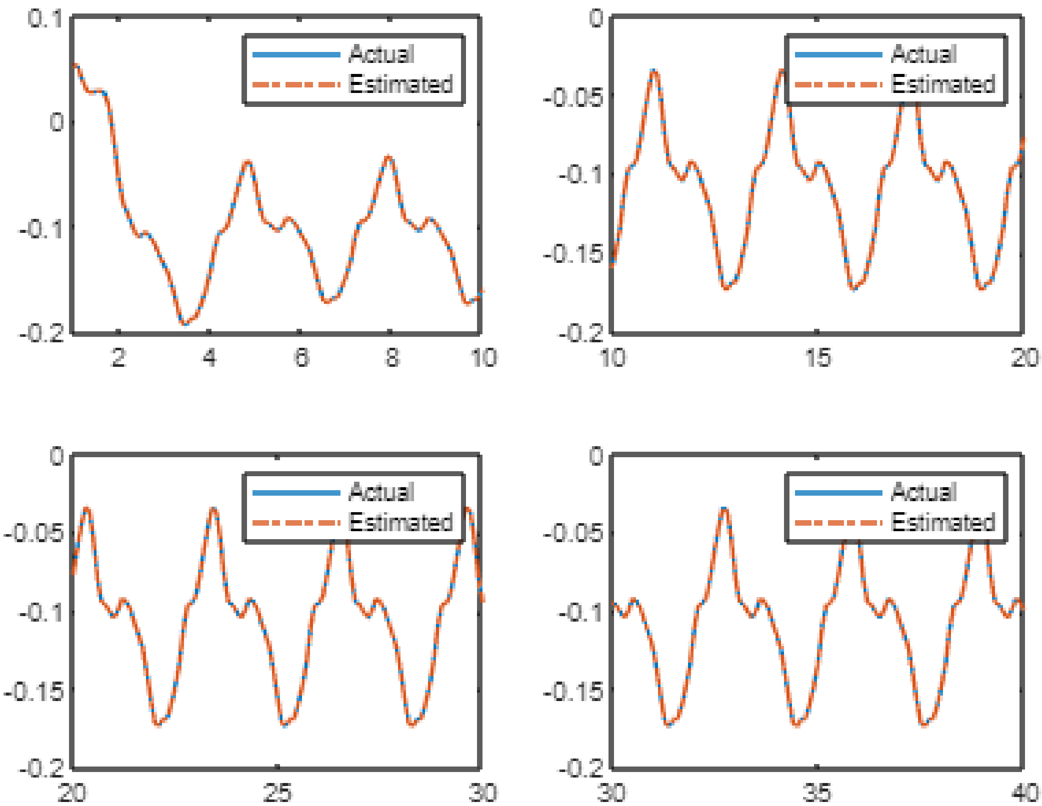
```
U =
```

```
disp(U_prbs(1:10,:))
```

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1.0000	0	0	0
0	0	0	0	0	1.0000	0	0
-0.0099	0	0	0	0	0	1.0000	0
-0.0212	-0.0099	0	0	1.0000	0	0	1.0000
-0.0234	-0.0212	-0.0099	0	0	1.0000	0	0
-0.0303	-0.0234	-0.0212	-0.0099	1.0000	0	1.0000	0
-0.0357	-0.0303	-0.0234	-0.0212	1.0000	1.0000	0	1.0000

```

theta_hat_prbs = inv(U_prbs'*U_prbs)*U_prbs'*y2;
y_hat_prbs = U_prbs*theta_hat_prbs;



figure()
subplot(2,2,1)
plot(t,y2,t,y_hat_prbs,'-.')
legend('Actual','Estimated')
xlim([1,10])

subplot(2,2,2)
plot(t,y2,t,y_hat_prbs,'-.')
legend('Actual','Estimated')
xlim([10,20])

subplot(2,2,3)
plot(t,y2,t,y_hat_prbs,'-.')
legend('Actual','Estimated')
```

```
xlim([20,30])

subplot(2,2,4)
plot(t,y2,t,y_hat_prbs,'-.')
legend('Actual','Estimated')
xlim([30,40])
```




```
%% Evaluation Metrics

disp("----------------Model Evaluation Report-------------------")
```

```
----------------Model Evaluation Report-------------------
```

```
SST = sum((y2-mean(y2)).^2);
SSE = sum((y2-y_hat_prbs).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('---(PRBS)---> SSE : %.7f \n', SSE);
```

```
---(PRBS)---> SSE : 0.0000003
```

```
fprintf('---(PRBS)---> MSE : %.7f \n', MSE);
```

```
---(PRBS)---> MSE : 0.0000000
```

```
fprintf('---(PRBS)---> R2  : %.7f \n', R2);
```

```
---(PRBS)---> R2  : 0.9999997
```

```

disp("===========================================================")
```

```
===========================================================
```

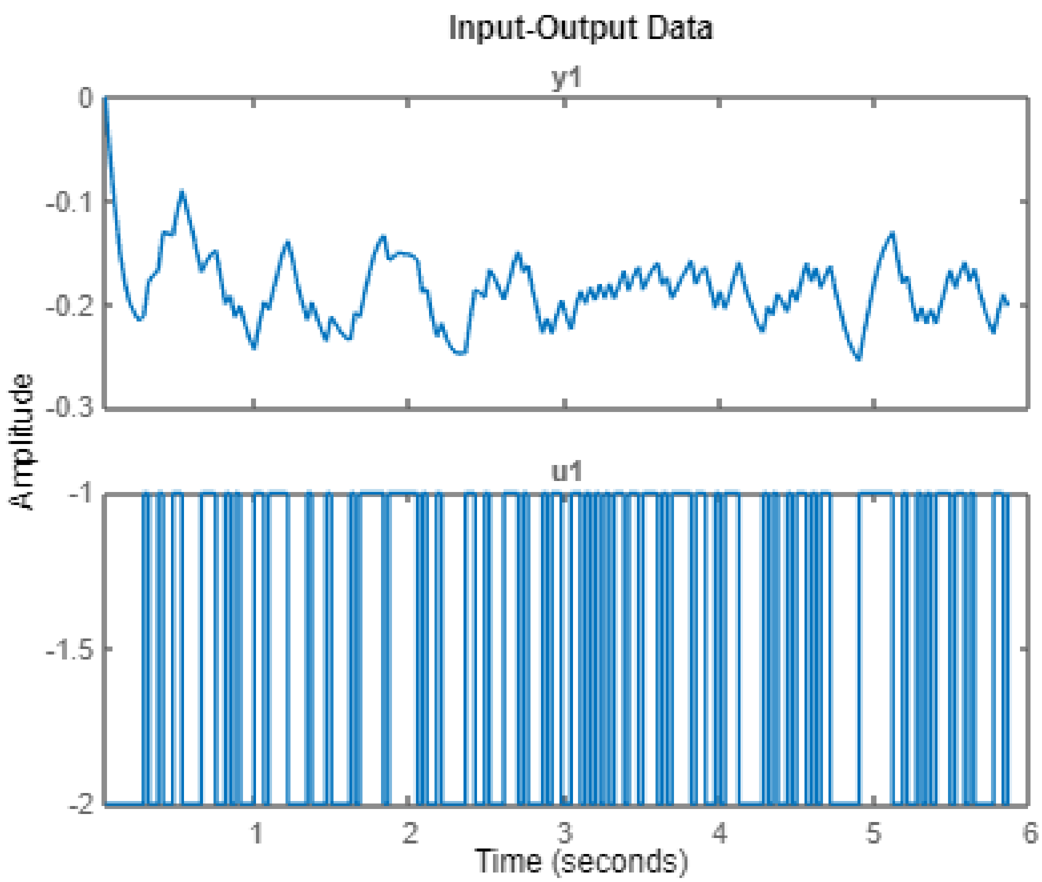
```

```

Question 4

```
clc; clear;

load 402123100
```

```
Warning: Updating objects saved with previous MATLAB version...
Resave your MAT files to improve loading speed.
```

```
y = id.y;
u = id.u;
N = length(y);

y_val = val.y;
u_val = val.u;

figure()
plot(id)
hold on
```



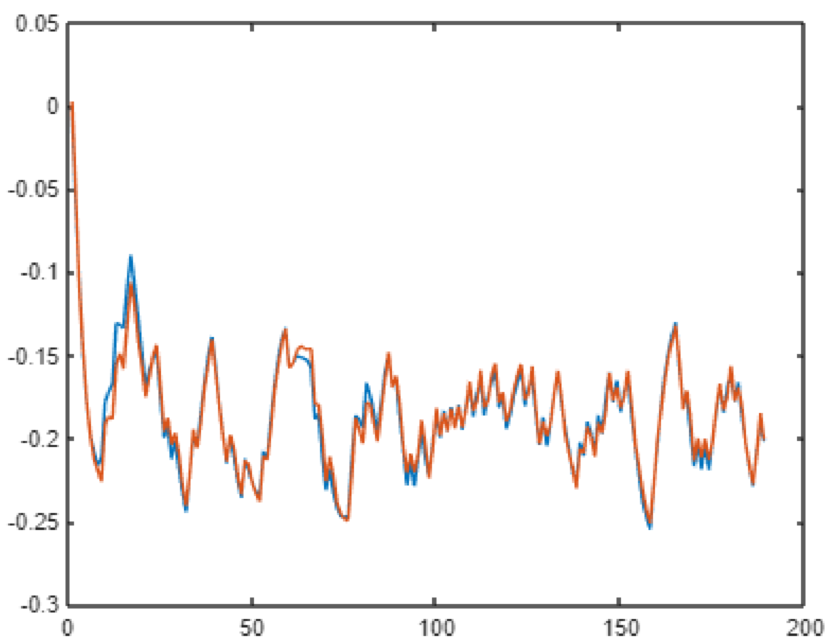

```
na = 2;
nb = 1;

U = arx_U_builder(na,nb,u,y);
disp("U = ")
```

```
U =
```

```
disp(U(1:10,:))
```

```
         0         0   -2.0000         0

   -0.0006         0   -2.0000   -2.0000

    0.0582   -0.0006   -2.0000   -2.0000

    0.1070    0.0582   -2.0000   -2.0000

    0.1489    0.1070   -2.0000   -2.0000

    0.1783    0.1489   -2.0000   -2.0000

    0.1983    0.1783   -2.0000   -2.0000

    0.2084    0.1983   -2.0000   -2.0000

    0.2162    0.2084   -1.0000   -2.0000

    0.2113    0.2162   -2.0000   -1.0000
```

```
theta_hat = inv(U'*U)*U'*y;
y_hat = U*theta_hat;
figure()
plot(y,'-')
hold on
plot(y_hat, '-')
```




```
disp("----------------Model Evaluation Report-------------------")
```

```
 ----------------Model Evaluation Report-------------------
```

```
SST = sum((y-mean(y)).^2);
SSE = sum((y-y_hat).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 0.0071355
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.0000378
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 0.9678453
```

```

disp("===========================================================")
```

```
===========================================================
```

Bonus Question

```
model = arx([y_val,u_val], [na nb 1]);

% compare([y_val, u_val], model)

y_pred = lsim(model, u_val);

disp("----------------Model Evaluation Report-------------------")
```

```
----------------Model Evaluation Report-------------------
```

```

SST = sum((y_val-mean(y_val)).^2);
SSE = sum((y_val-y_pred).^2);

R2 = 1 - (SSE/SST);

MSE = SSE/N;

fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 0.0640527
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.0003389
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 0.9107713
```

```
disp("===========================================================")
```

```
===========================================================
```

`arx` command provides a model estimation with 0.91. However, R2 value is higher in the hand-written algorithm of ours. Note that both of the are gretaer than 0.9 and they are proper.

Question 5

```
clear; clc

load t
load u
load y

N = length(y);
n = 0;
R2 = 0;

minimum_acceptable_R2 = 0.9025
```

```
minimum_acceptable_R2 = 0.9025
```

```
while (R2<minimum_acceptable_R2)
    % increase the order
    n = n+1;
    m = n-1;

    % build up U matrix
    U = arx_U_builder(n,m,u,y);

    % solve for he linear least squares
    theta_hat = inv(U'*U)*U'*y;
    y_hat = U*theta_hat;

    % evaluate the model
    SST = sum((y-mean(y)).^2);
    SSE = sum((y-y_hat).^2);

    R2 = 1 - (SSE/SST);

    MSE = SSE/N;
end
disp("----------------Model Evaaluation Report-------------------")
```

```
----------------Model Evaaluation Report-------------------
```

```
fprintf('The proper order of estimation n = %d \n\n', n);
```

```
The proper order of estimation n = 2
```

```
fprintf('------> SSE : %.7f \n', SSE);
```

```
------> SSE : 18.5076393
```

```
fprintf('------> MSE : %.7f \n', MSE);
```

```
------> MSE : 0.7711516
```

```
fprintf('------> R2  : %.7f \n', R2);
```

```
------> R2  : 0.9977278
```

```


disp("===========================================================")
```

```
===========================================================
```