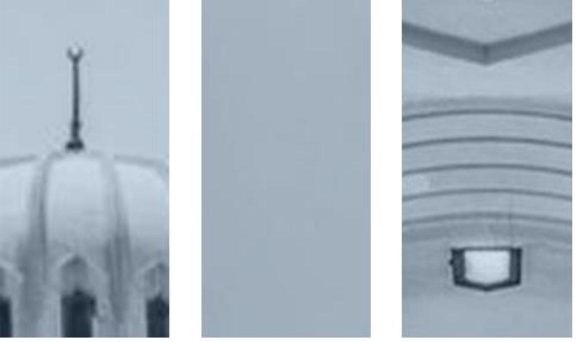


RESEARCH & PROJECT SUBMISSIONS

Program:

Course Code: CSE 271

Course Name: System Dynamics & Control Components

Examination Committee

Dr. Hossam Abdelmunim

Dr. Mohamed Sobh

Ain Shams University

Faculty of Engineering

Spring Semester – 2020

Student Personal Information for Group Work

Student Names:	Student Codes:
Loay Abdalla Youssef	1701043
Amr Ehab Abdelaziz	1700923
Amr Ahmed Mohamed Fathy	1700918
Loay Anwar Abdelrazek Hegazy	1701040
Loay Khaled Mohamed Abdo	1701042

Plagiarism Statement

I certify that this assignment / report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they are books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment / report has not been previously been submitted for assessment for another course. I certify that I have not copied in part or whole or otherwise plagiarized the work of other students and / or persons.

Signature/Student Name: **Loay Abdalla Youssef**
Amr Ehab Abdelaziz
Amr Ahmed Mohamed Fathy
Loay Anwar Abdelrazek Hegazy
Loay Khaled Mohamed Abdo

Date: 10/6/2020

Abstract

In Engineering problems an LTI system is studied and represented in a differential equation of inputs and outputs of the system. In this project we built a program to solve the differential equations numerically and represent the system states, plot them and plot the system input and output allowing users to control the system and predict its response to different inputs (unit step or unit impulse).

01 *System Simulation & Model*

To reach state space model from input output differential equation:

$$a_n\, y^{(n)} + a_{n-1}\, y^{(n-1)} + .. + a_1\, \dot{y} + a_0\, y = b_n\, u^{(n)} + b_{n-1}\, u^{(n-1)} + .. + b_1\, \dot{u} + b_0\, u$$

Assume $x_1 = y + \beta_0 u \quad \rightarrow \quad y = x_1 - \beta_0 u \quad \rightarrow \quad \dot{y} = \dot{x}_1 - \beta_0 \dot{u} \rightarrow \cdots$

$x_2 = \dot{x}_1 + \beta_1 u \rightarrow \dot{x}_1 = x_2 - \beta_1 u \quad \rightarrow \quad \ddot{x}_1 = \dot{x}_2 - \beta_1 \dot{u} \quad (First\ State\ Equation)$

$\vdots \qquad\qquad \vdots$

$x_n = \dot{x}_{n-1} + \beta_{n-1} u \quad \rightarrow \quad \dot{x}_{n-1} = x_n - \beta_{n-1} u \qquad ((n-1)th\ State\ Equation)$

To get the nth order: First substitute from the y equation in the differential equation to get equation a differential equation of x_1 and its derivatives.

Ex: for the nth order: $x_1 = y + \beta_0 u \quad \rightarrow \quad y = x_1 - \beta_0 u$

$\rightarrow \quad \dot{y} = \dot{x}_1 - \beta_0 \dot{u} \quad \rightarrow \quad \ldots \rightarrow \ y^{(n)} = x^{(n)} - \beta_0 u^{(n)}$

$$a_n\left(x_1^{(n)} - \beta_0 u^{(n)}\right) + \cdots + a_1(\dot{x}_1 - \beta_0 \dot{u}) + a_0(x_1 - \beta_0 u) = b_n\, u^{(n)} + \cdots + b_1 \dot{u} + b_0 u$$

Second: from the state equations substitute in the differential equation starting from $\dot{x}_1 = x_2 - \beta_1 u$ and so on till we get equation of $\dot{x}_n$ and the state variables and $\beta_0 \rightarrow \beta_{n-1}$

Third: Equate the coefficients input derivatives from the L.H.S and R.H.S to eliminate the input derivatives and get the values of betas.

$\beta_0 = -\frac{b_n}{a_n} \quad , \quad \beta_1 = \frac{1}{a_n}\left(-b_{n-1} + \frac{a_{n-1}}{a_n} * b_n\right)$

$\beta_2 = \frac{1}{a_n}(-b_{n-2} - \frac{a_{n-1}}{a_n}\left(-b_{n-1} + \frac{a_{n-1}}{a_n} * b_n\right) + a_{n-2} * b_n)\,,\ \ldots.$

Fourth: Get $\dot{x}_n$ as a function of state variables of $x_1\ to\ x_n$ and β.

$\dot{x}_1 = x_2 - \beta_1 u$

$$\dot{x}_2 = x_3 - \beta_2 u\ , \dot{x}_i = x_{i+1} - \beta_i u\ , \ldots, \quad \dot{x}_n = \frac{1}{a_n}(-\sum_{k=1}^{k=n} a_{k-1} x_k + u(b_0 + \sum_{k=0}^{k=n-1} a_k \beta_k))$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ -\frac{a_0}{a_n} & \dots & \frac{-a_{n-2}}{a_n} & \frac{-a_{n-1}}{a_n} \end{bmatrix}, B = \begin{bmatrix} -\beta_1 \\ \vdots \\ -\beta_{n-1} \\ \frac{b_0 + a_0\beta_0 + a_1\beta_1 + \dots + a_{n-1}\beta_{n-1}}{a_n} \end{bmatrix}, C = [1 \quad 0 \quad 0 \quad \dots \quad 0], D = [-\beta_0]$$

02 *Numerical Approximations*

To solve the system of first order differential equations, the method of backward difference is used to approximate the first derivative $\dot{x} = \frac{x(k)-x(k-1)}{T}$ where T is the size of the step used.

The system of linear equations is then solved using Gauss-Seidel method which is a numerical, iterative method of finding the solution.

First, from each equation, one variable is separated to the LHS.

Iterative substitution is then made to find the next value for each variable and use it in the next iteration. Initial conditions are guessed for each variable.

$$x_1(1) = 0 \, , x_2(1) = 0 \, \dots . \, x_n(1) = 0$$

Formulas for x_i

If i is less than n

$$x_i(k) = T(x_{i+1}(k-1) + B(i,1)u) + x_i(k-1)$$

If i is equal to n

$$x_n(k) = \frac{1}{1 - T * A(n,n)} \left[T * \left(B(n-1,1)u + \sum_{i=1}^{i=n-1} A(n,i)x_i(k) \right) + x_n(k-1) \right]$$

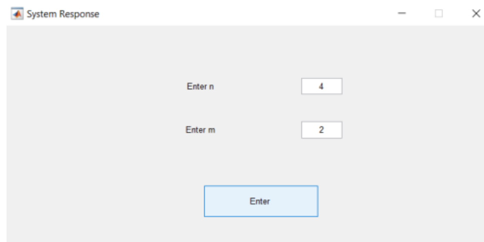
$$y(k) = x_1(k) + D(1,1)u$$

Impulse response is calculated as the derivative of the step response of the system.

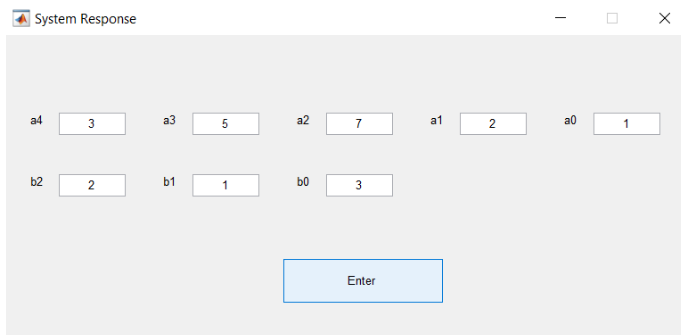
$$I(k) = \frac{y(k) - y(k-1)}{T}$$

03 *Algorithm & User Journey*

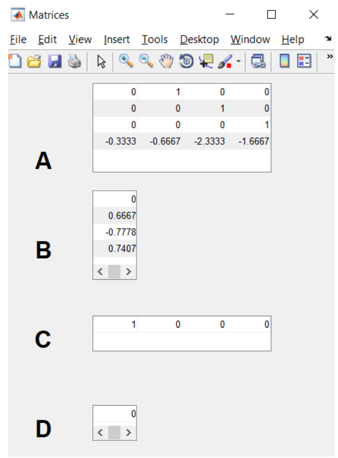
1) n, and m are taken from the user using GUI



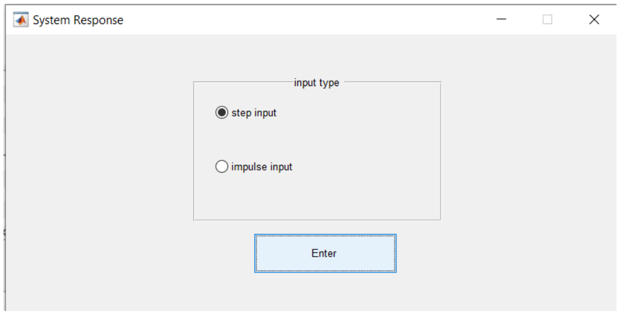
2) The GUI shows the user slots to enter a's and b's of the number they choose.



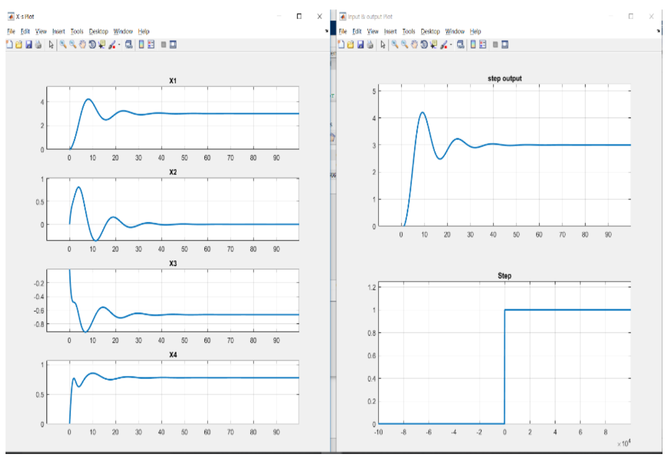
3) a's and b's are stored in two arrays.
4) The needed β's are calculated based on the user input (n).
5) The state space matrices are calculated from the previous formulas, and shown in the GUI.



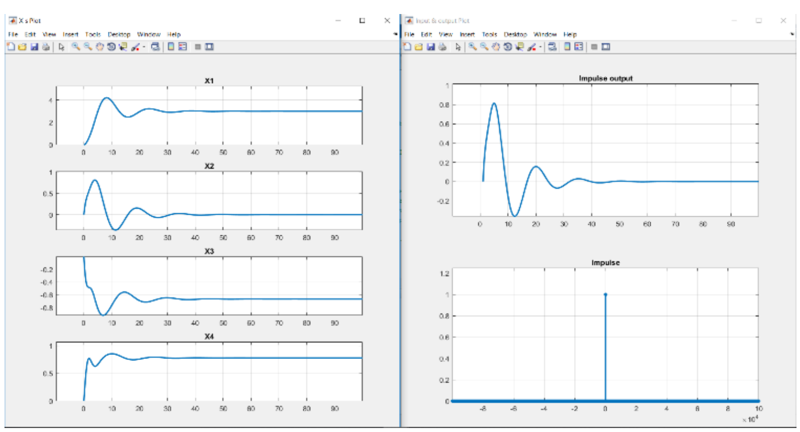
6) Initial values of x_1 to x_n and y are set to zero.
7) The update loop runs to find 'N' points on $x_1(t), x_2(t) \ldots x_n(t), y(t), I(t)$ with a step 'T' between these points and stores them in arrays.
8) The user chooses the input type in a GUI.



9) If they choose step response a plot showing, step input, step response of the system and system states appears in two figures.



10) If they choose impulse response a plot showing, impulse input, impulse response of the system and system states of the step appears in two figures.

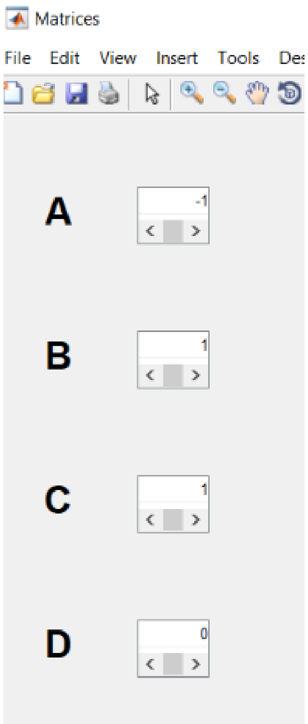


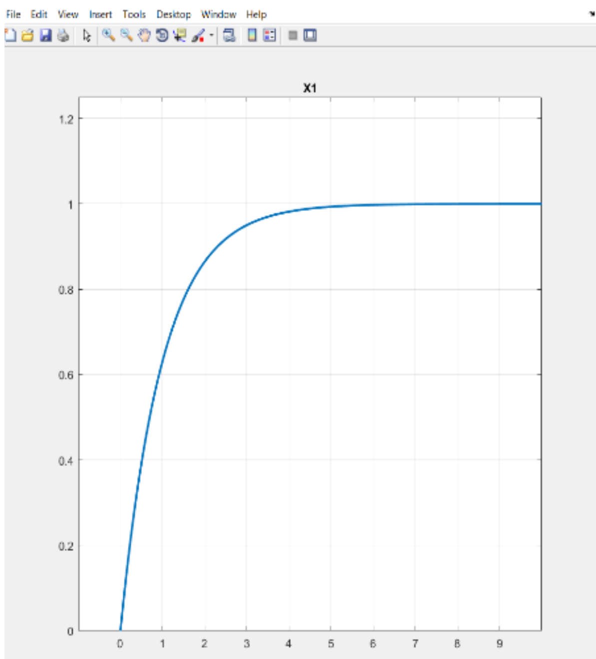
04

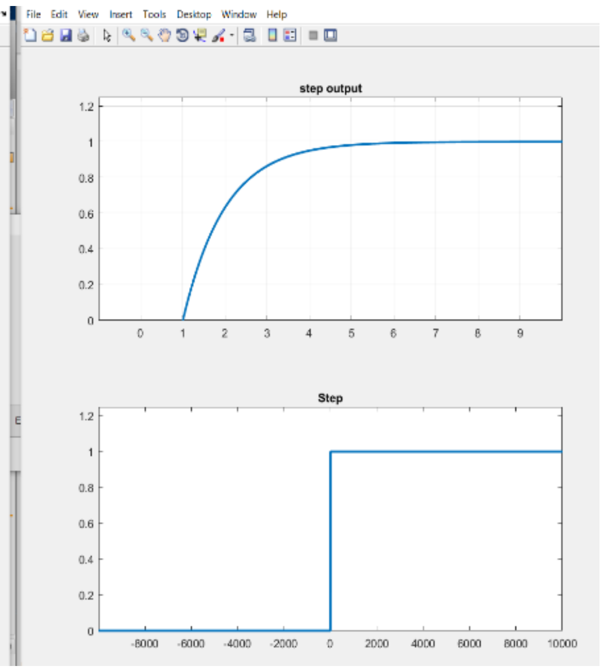
Experimental Results

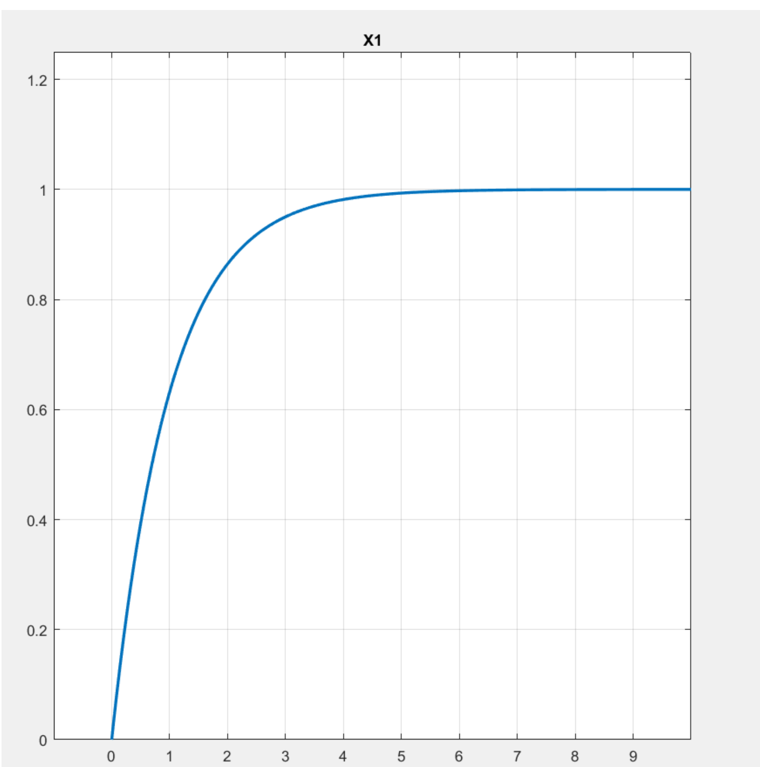
First Order Systems:

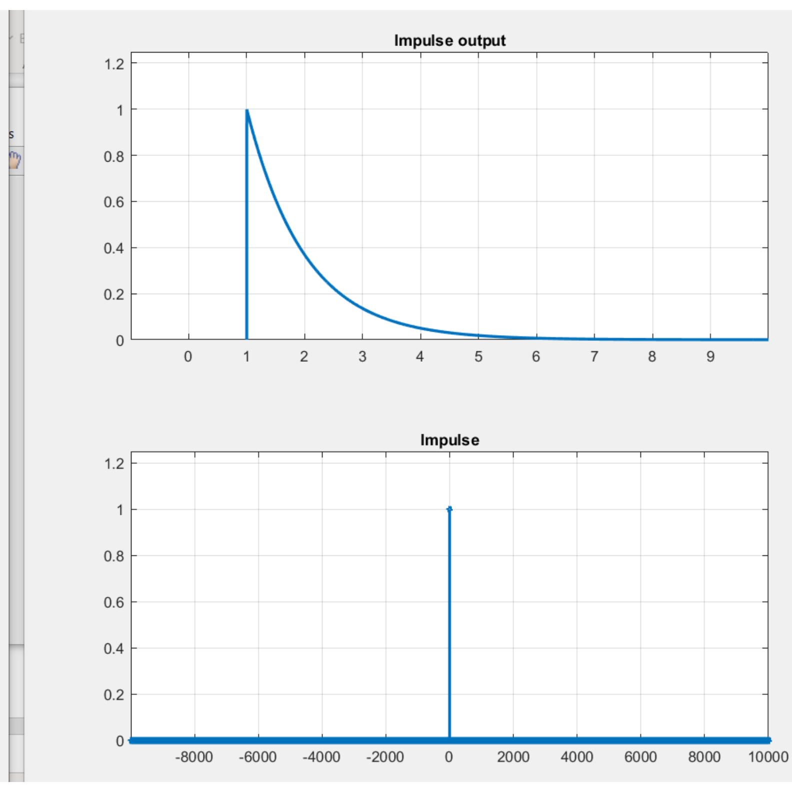
1) $\dot{y} + y = u$



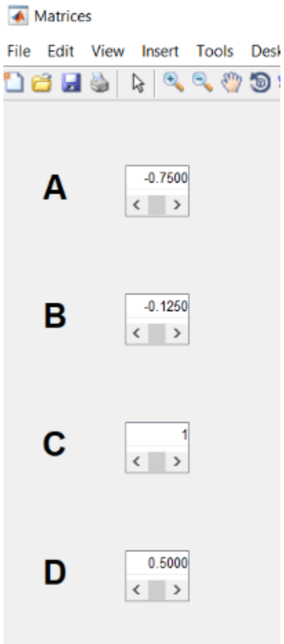


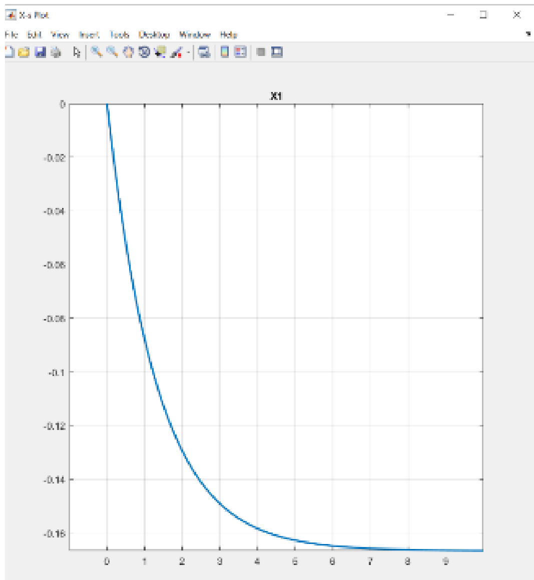


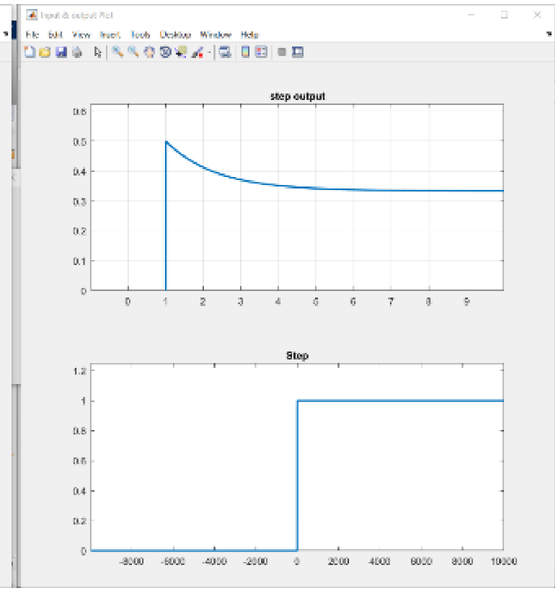


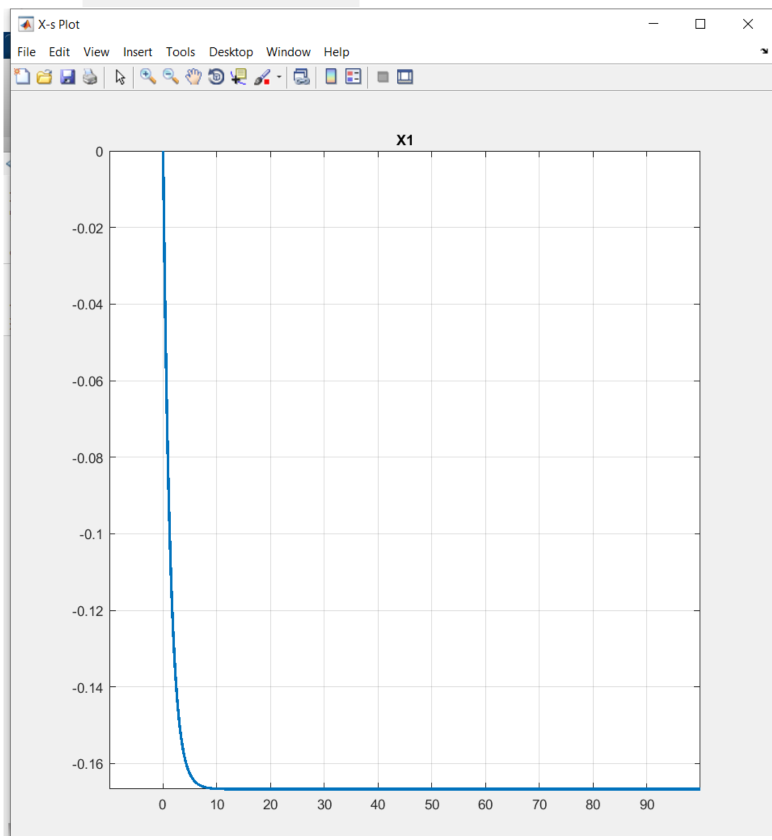


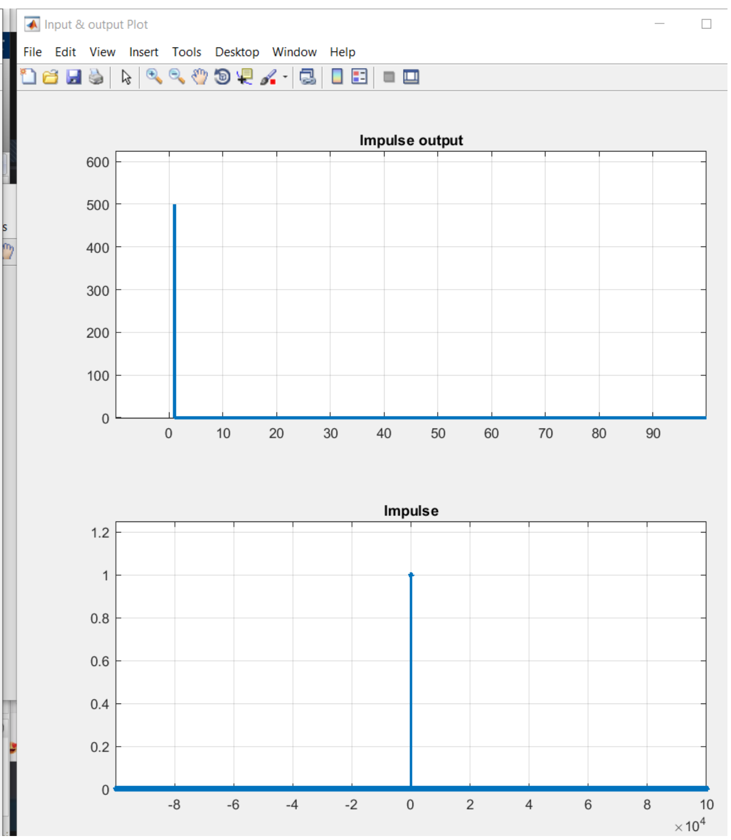
2) $4\dot{y} + 3y = 2\dot{u} + u$



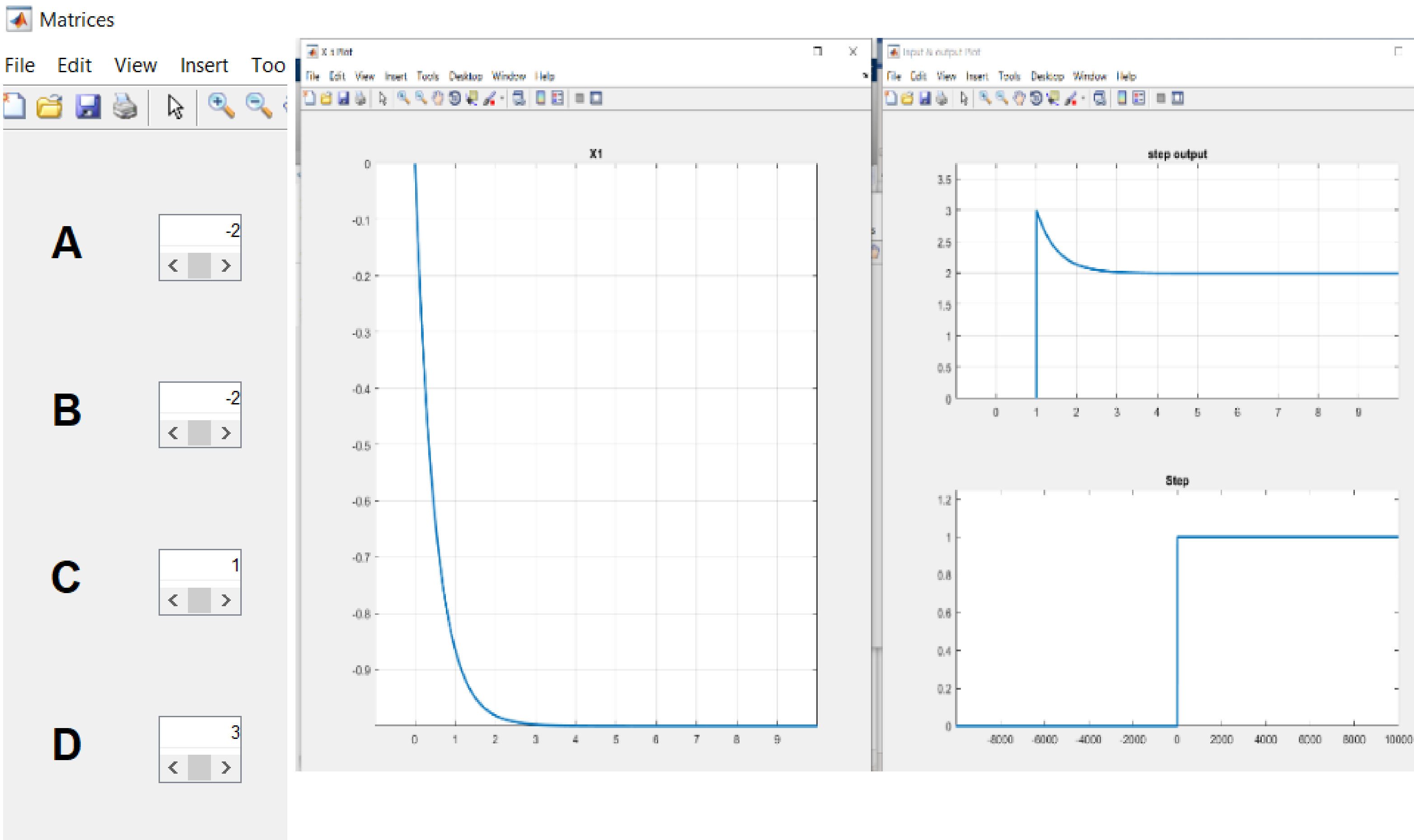


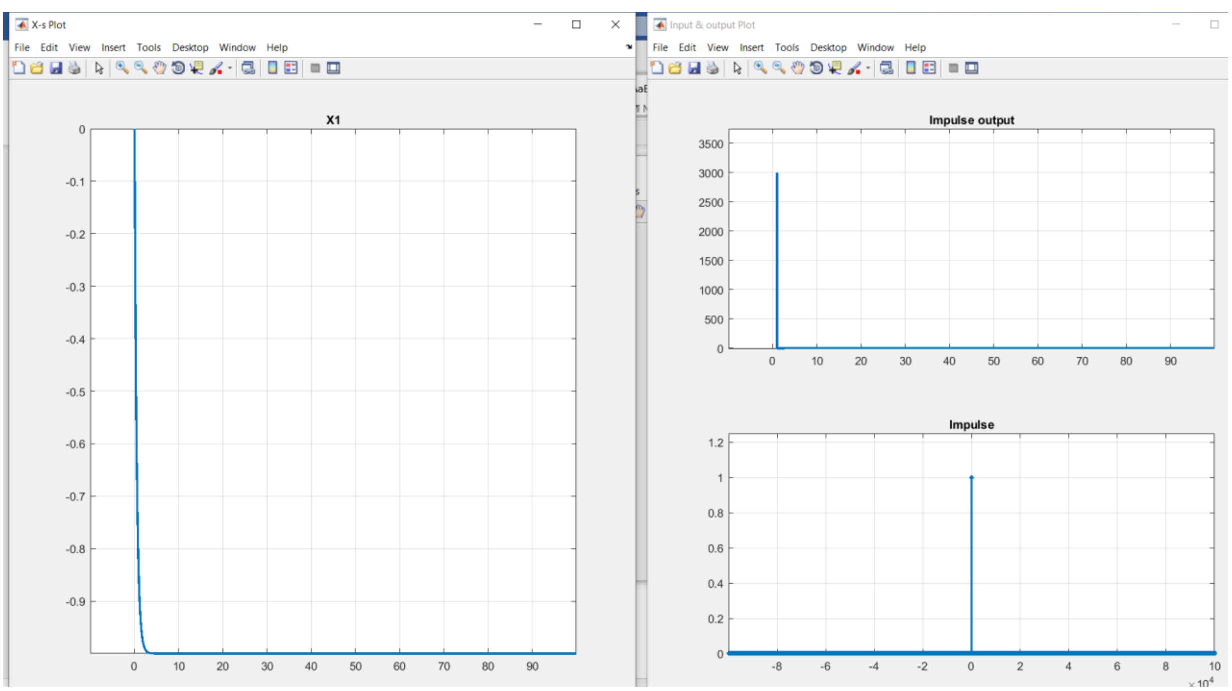




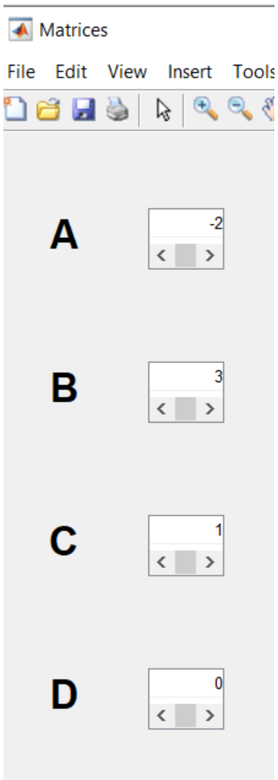


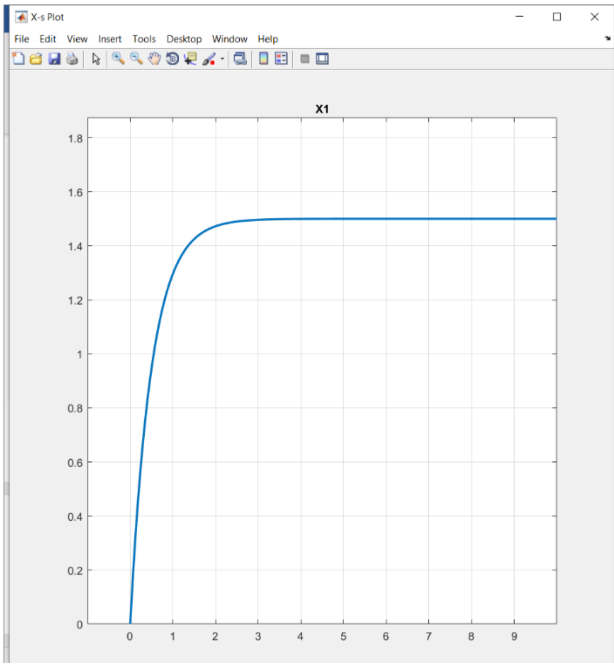
3) $\dot{y} + 2y = 3\dot{u} + 4U$

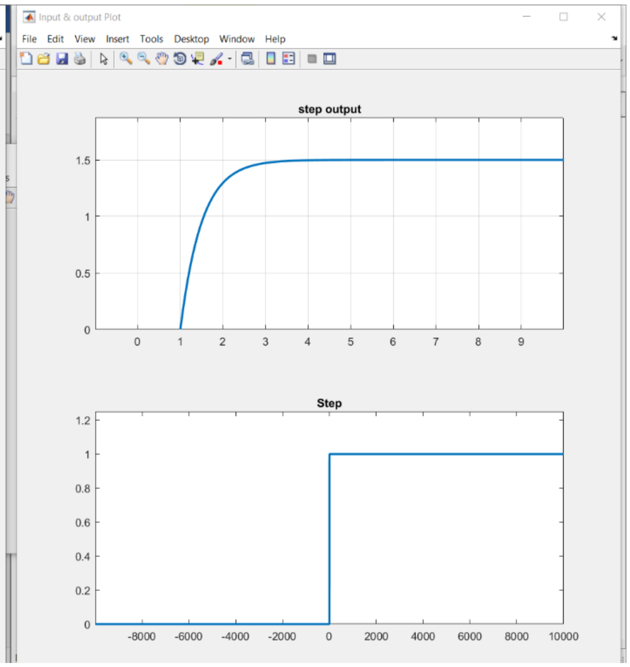


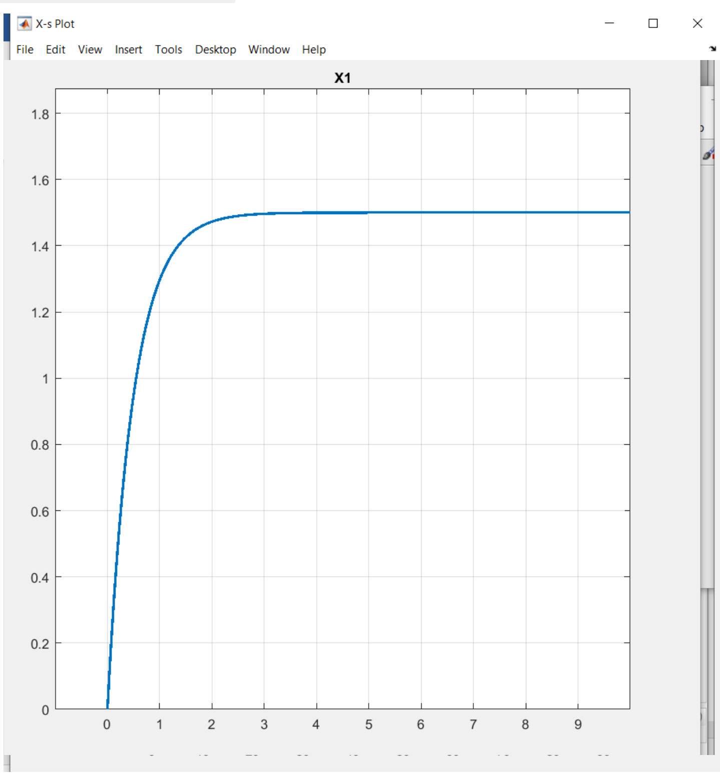


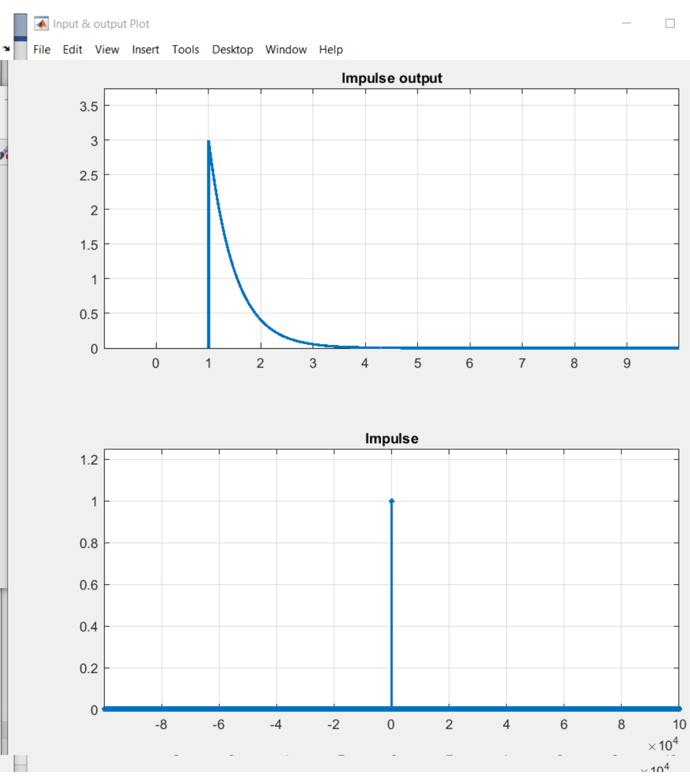
4) $\dot{y} + 2y = 3u$


Matrices
File Edit View Insert Tools
A
-2
B
3
C
1
D
0

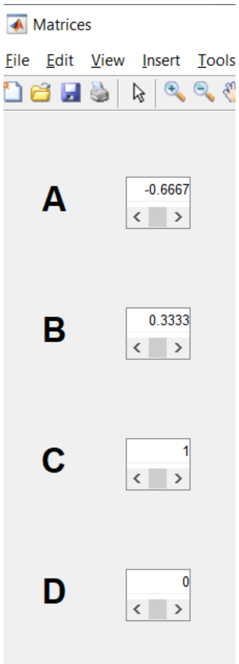

X-s Plot
File Edit View Insert Tools Desktop Window Help
X1

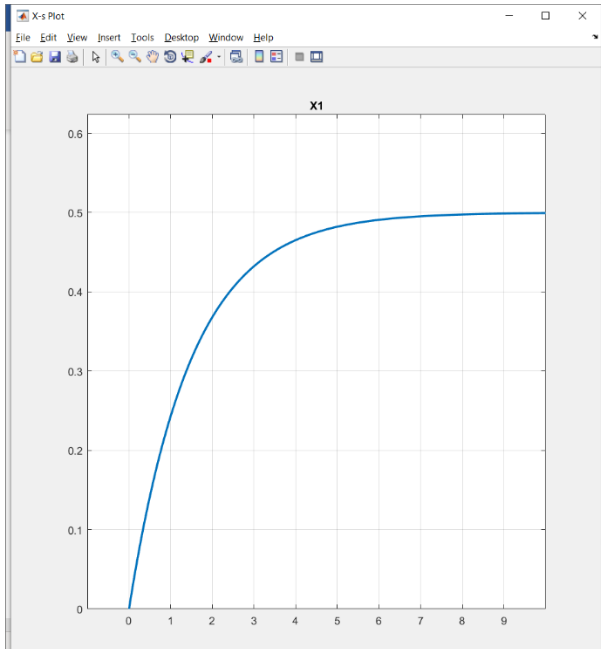

Input & output Plot
File Edit View Insert Tools Desktop Window Help
step output
Step

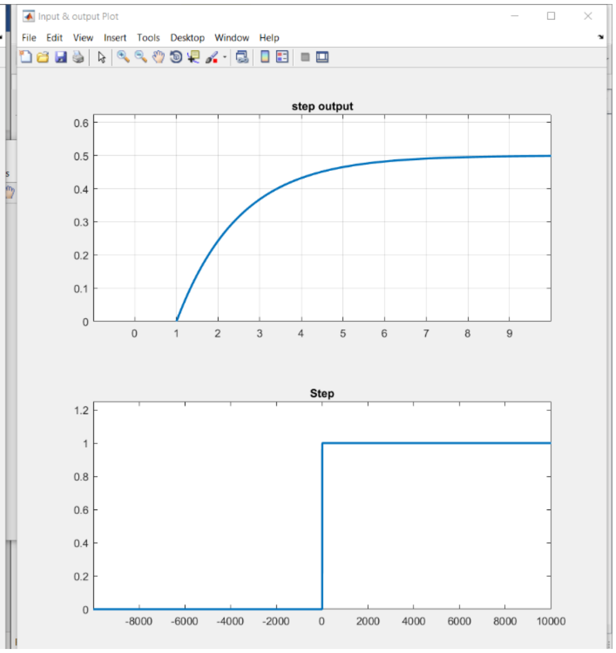

X-s Plot
File Edit View Insert Tools Desktop Window Help
X1

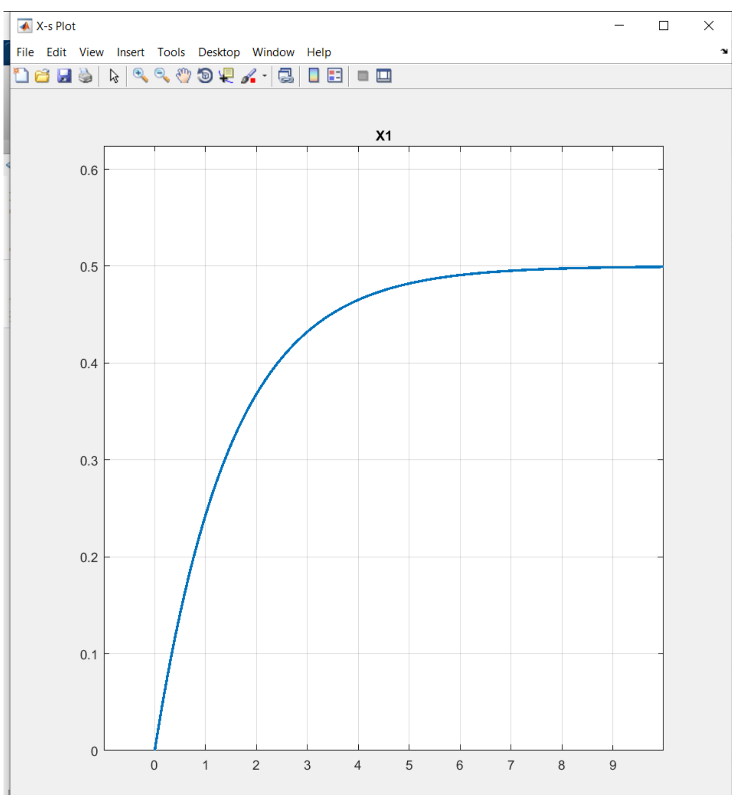

Input & output Plot
File Edit View Insert Tools Desktop Window Help
Impulse output
Impulse

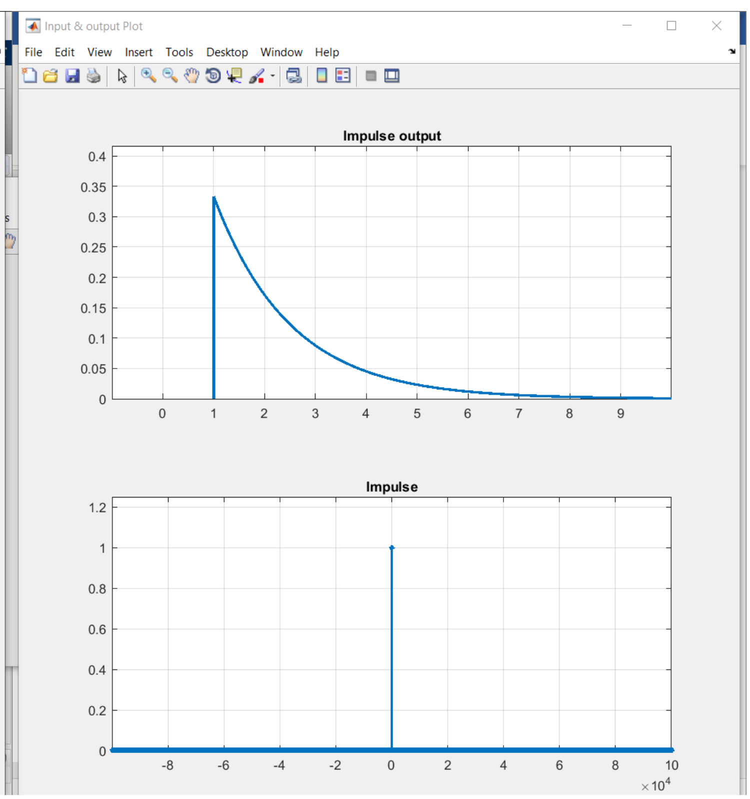
5) $3\dot{y} + 2y = u$


Matrices
File Edit View Insert Tools
A
-0.6667
B
0.3333
C
1
D
0


X-s Plot
File Edit View Insert Tools Desktop Window Help
X1

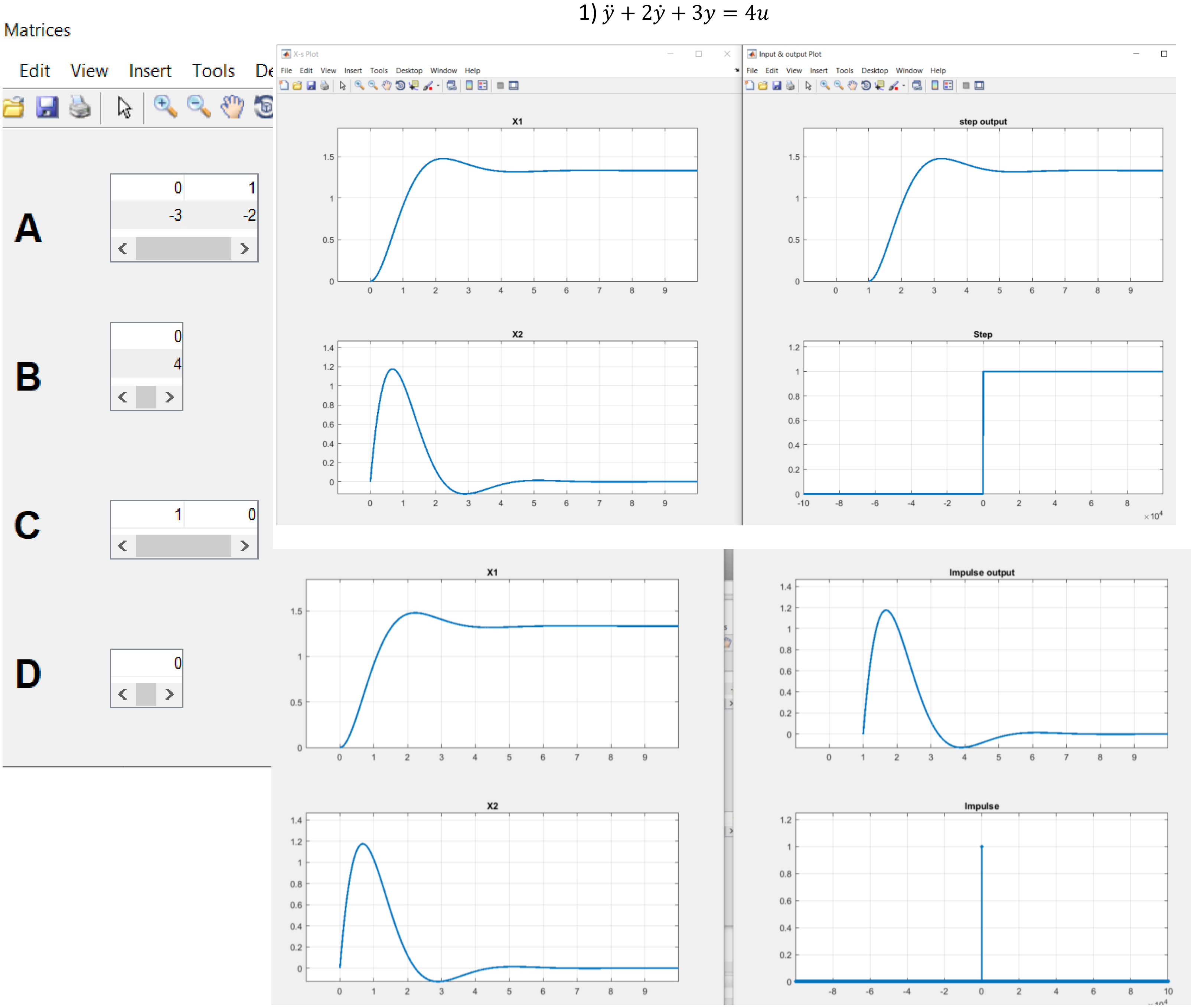

Input & output Plot
File Edit View Insert Tools Desktop Window Help
step output
Step


X-s Plot
File Edit View Insert Tools Desktop Window Help
X1


Input & output Plot
File Edit View Insert Tools Desktop Window Help
Impulse output
Impulse
×10^4

Second Order Systems:

1) $\ddot{y} + 2\dot{y} + 3y = 4u$



2) $4\ddot{y} + 3\dot{y} + 2y = u$

A

0	1
-0.5000	-0.7500

B

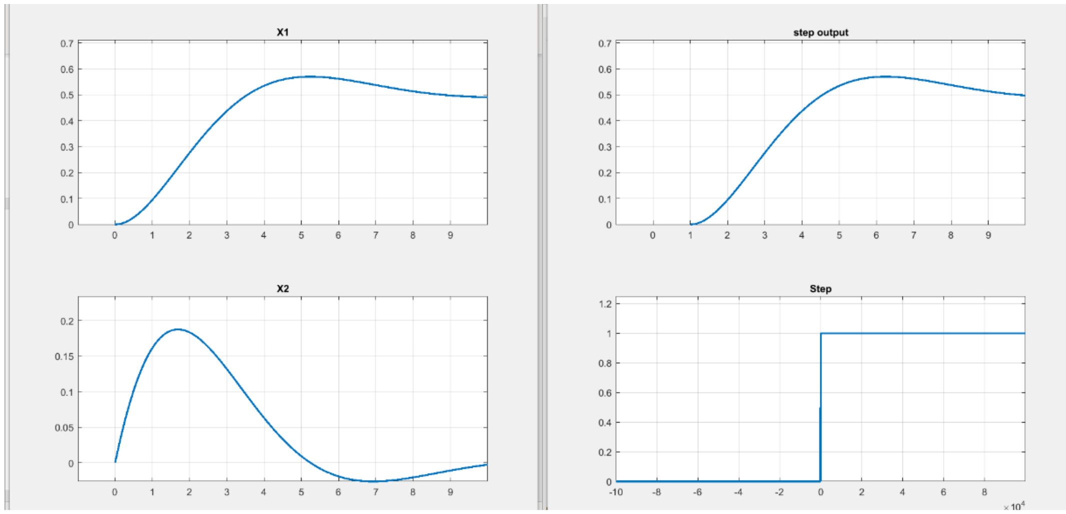
0
0.2500

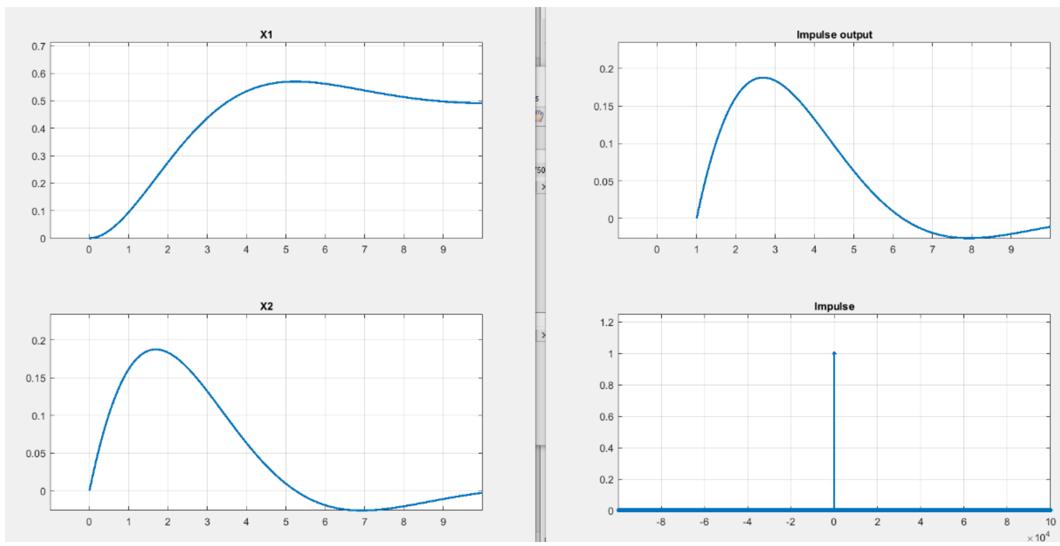
C

1	0

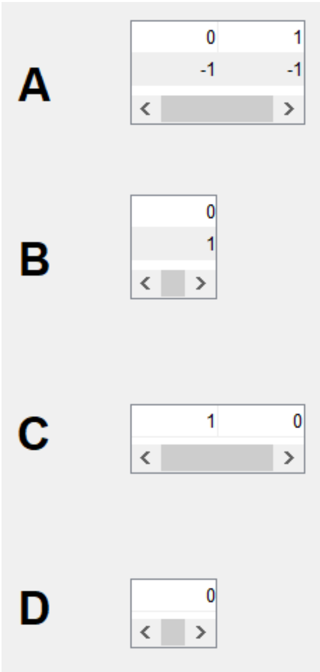
D

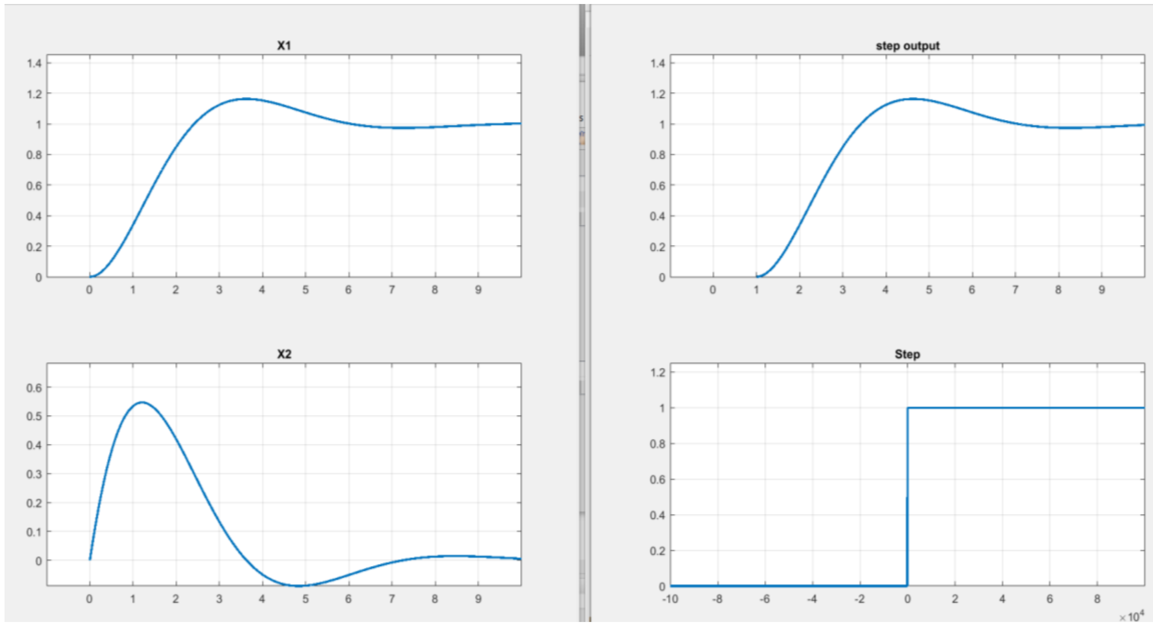
0

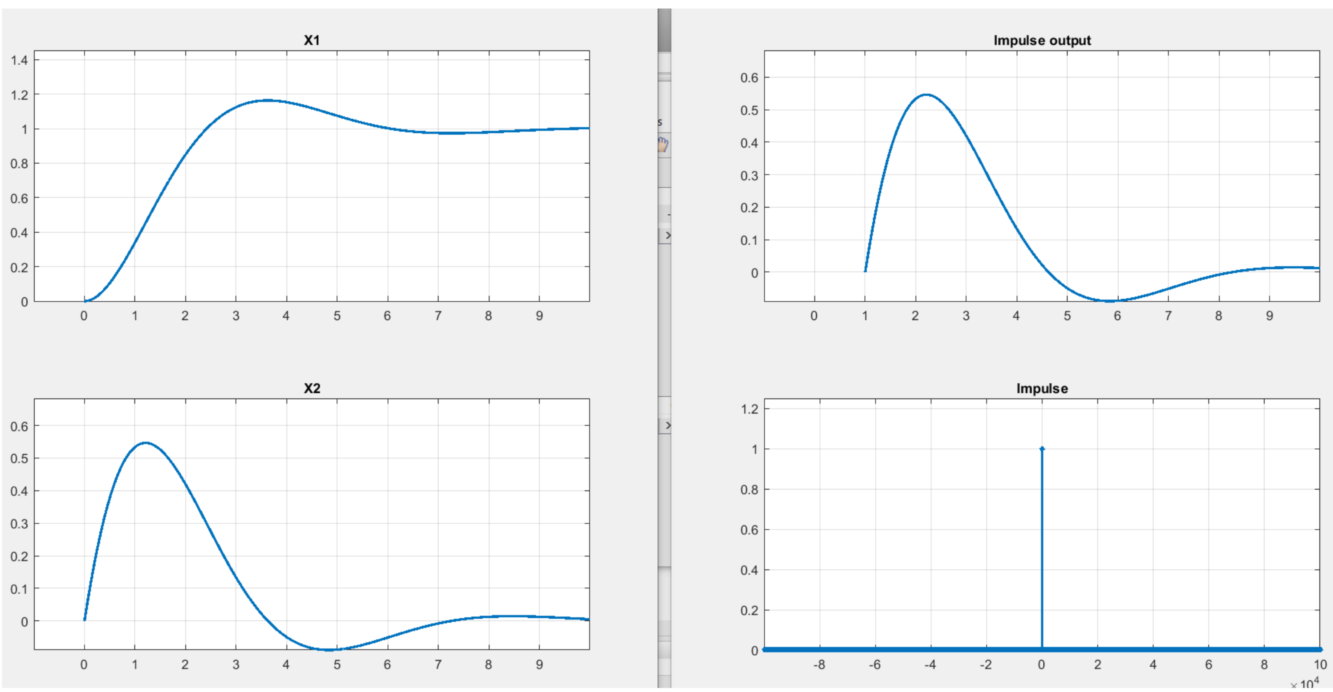




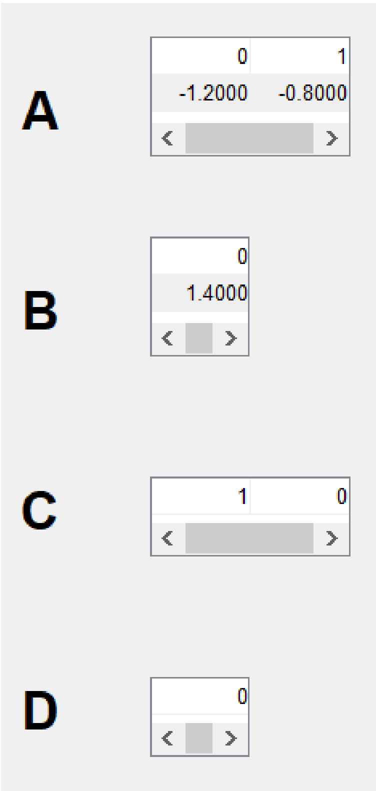
3) $\ddot{y} + \dot{y} + y = u$

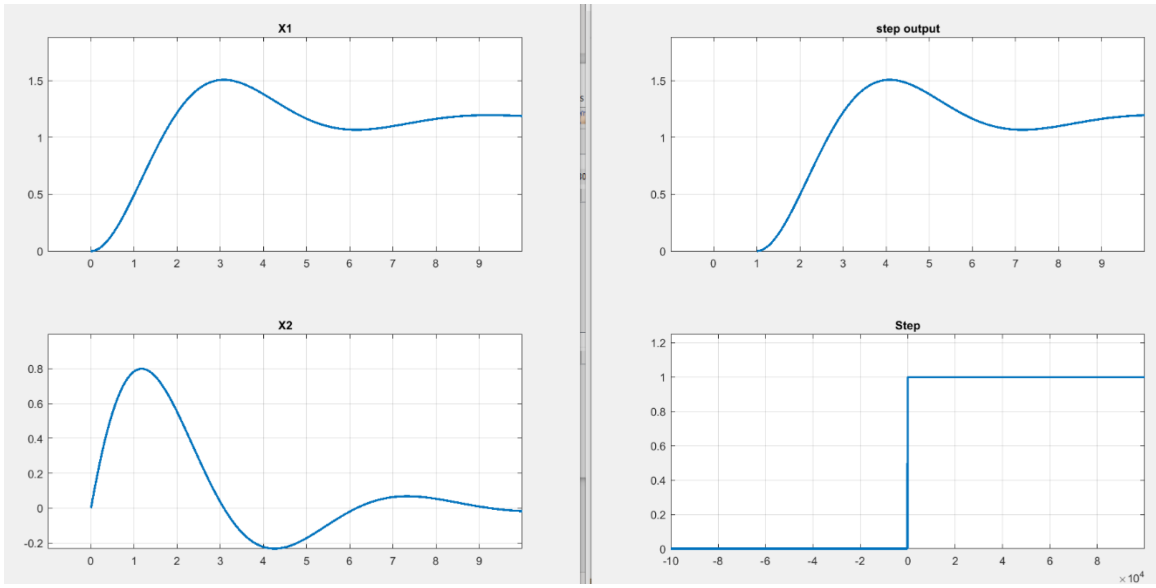


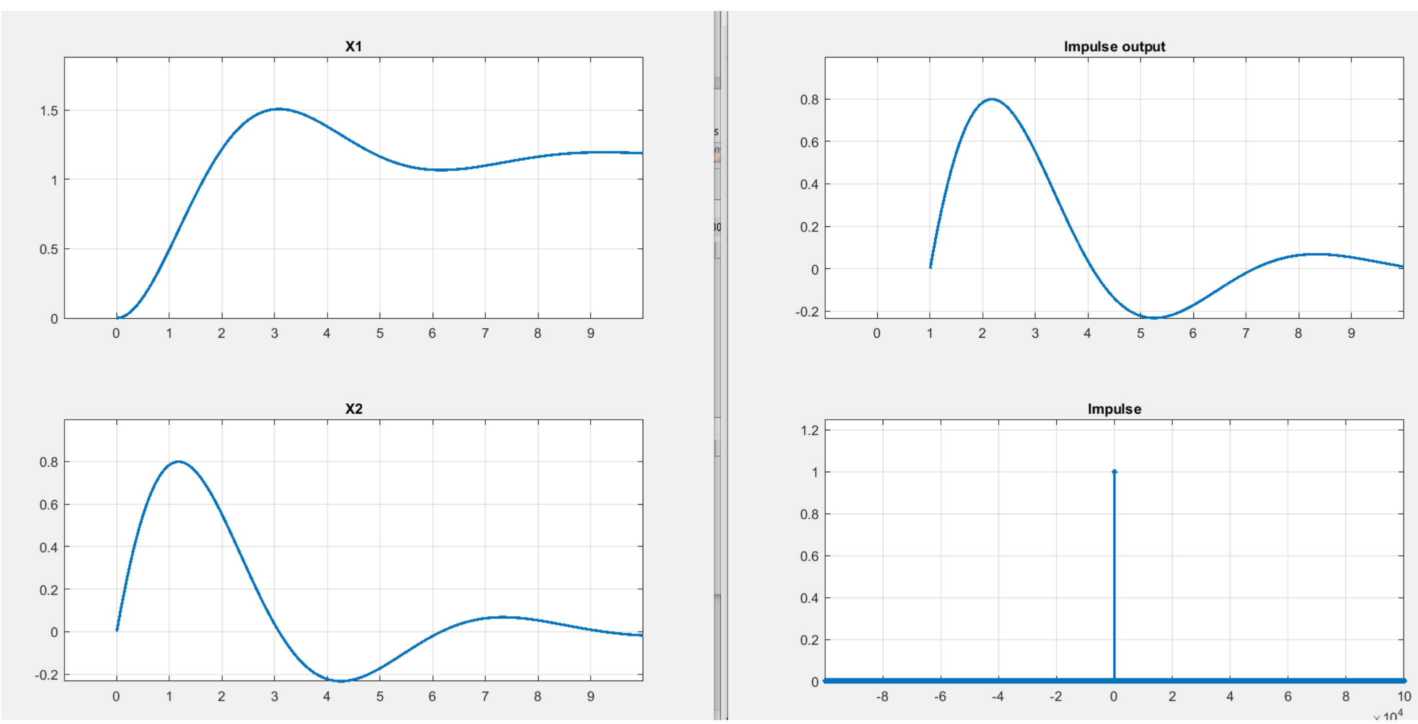




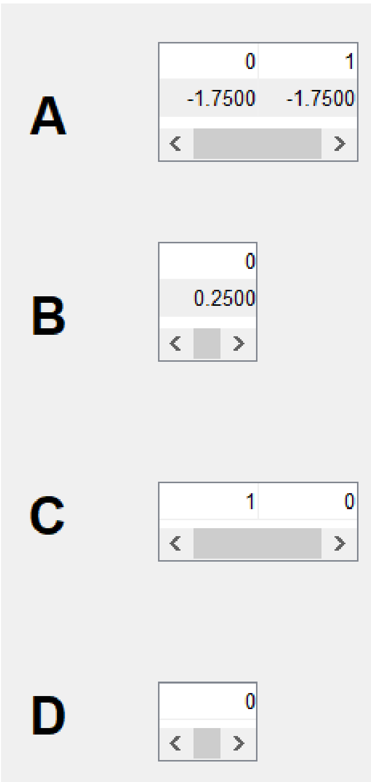
4) $5\ddot{y} + 4\dot{y} + 6y = 7u$

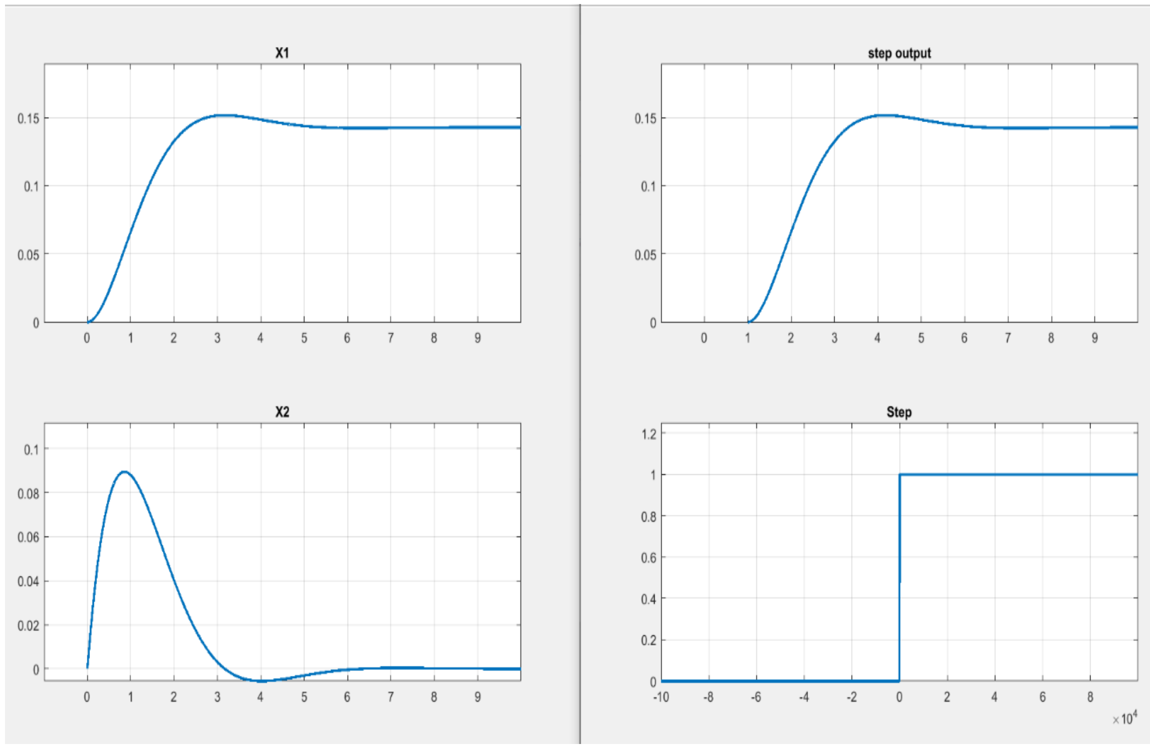


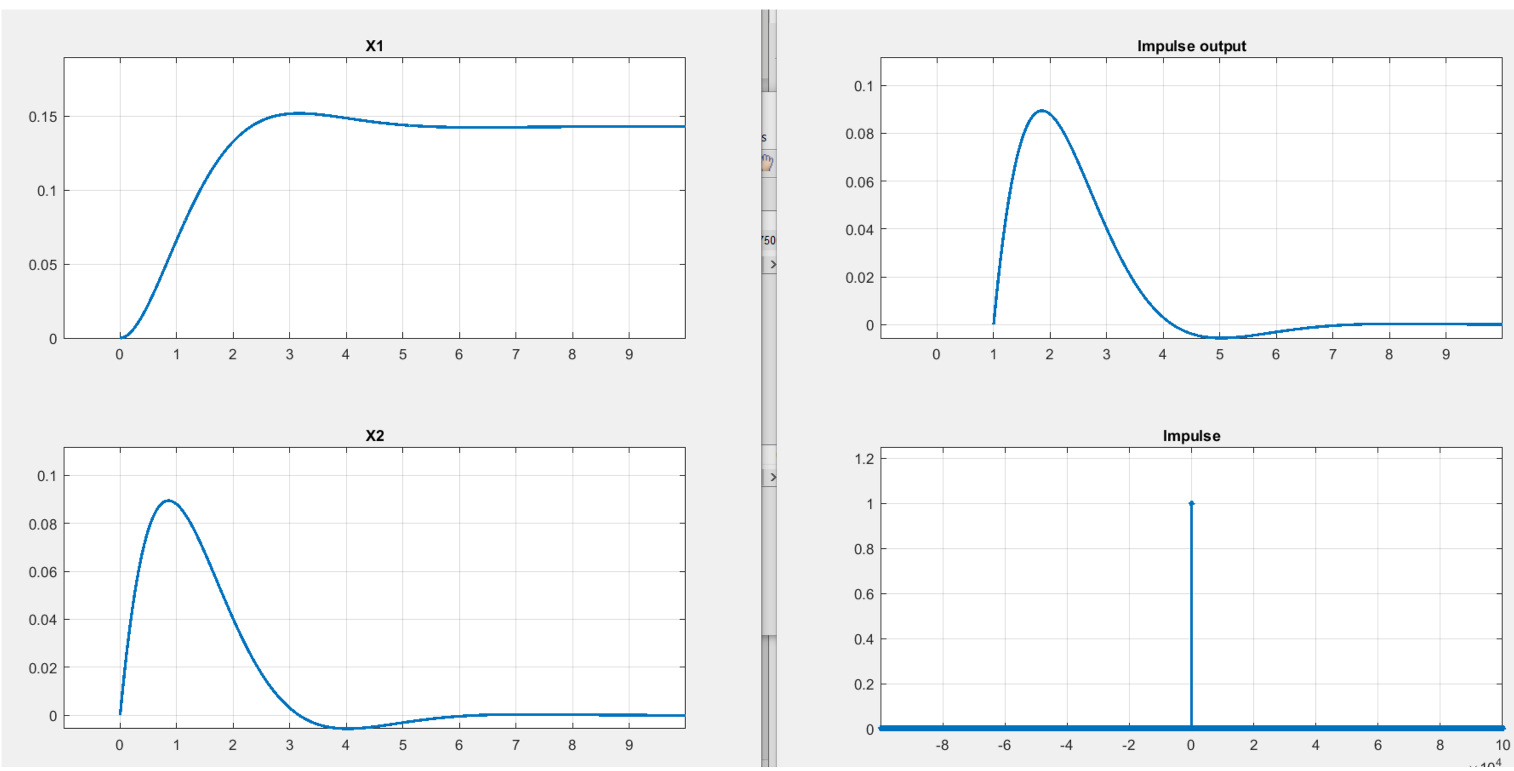




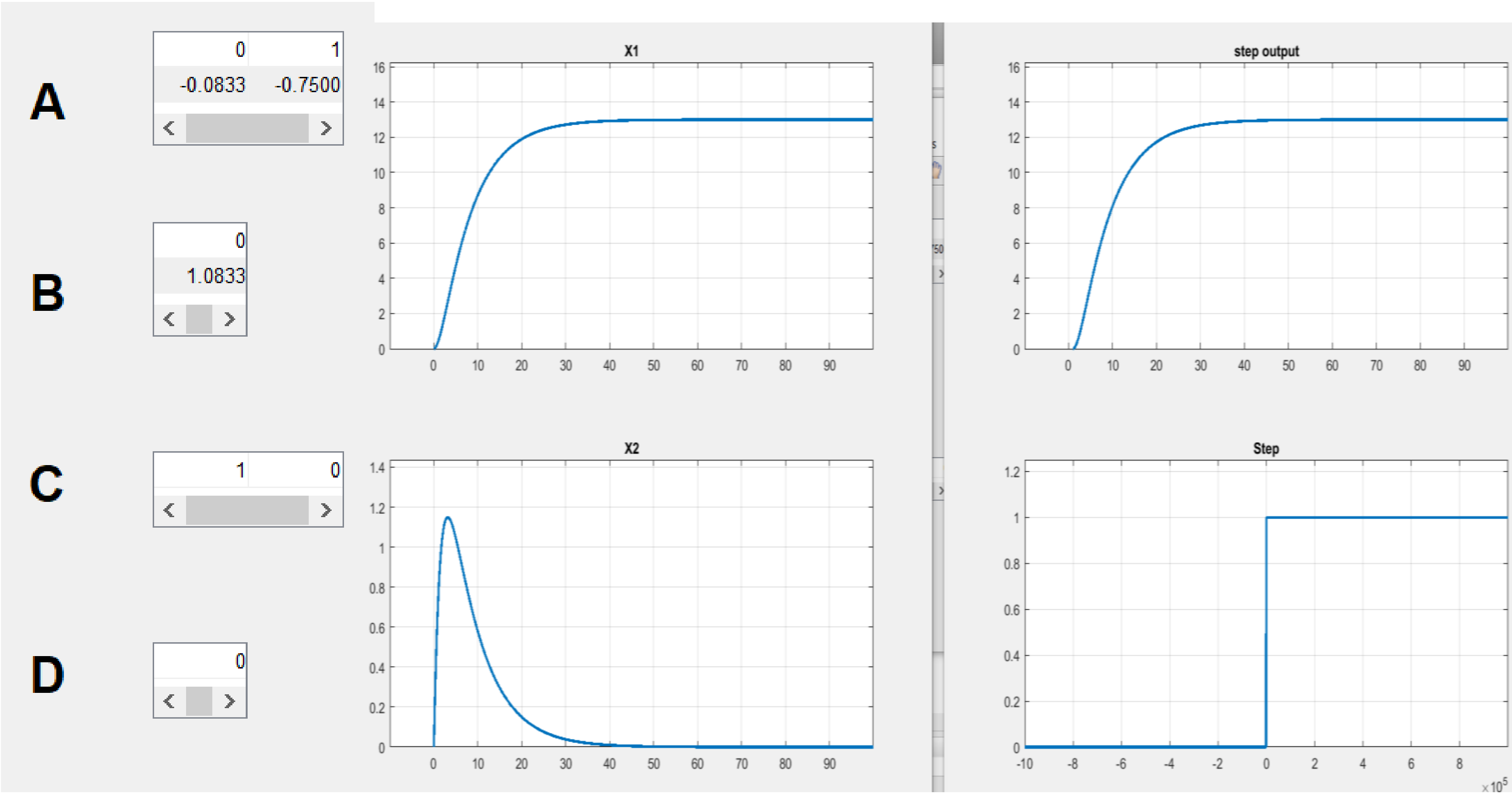
5) $4\ddot{y} + 7y + 7y = u$

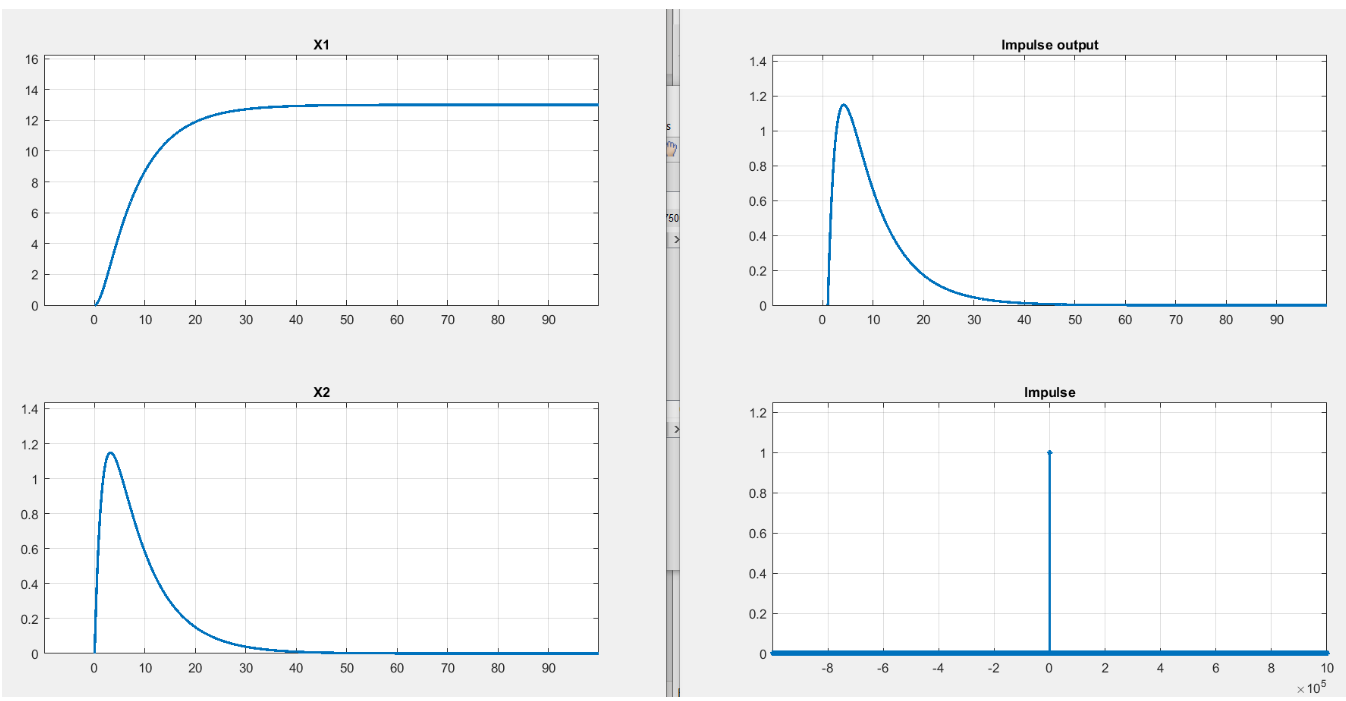




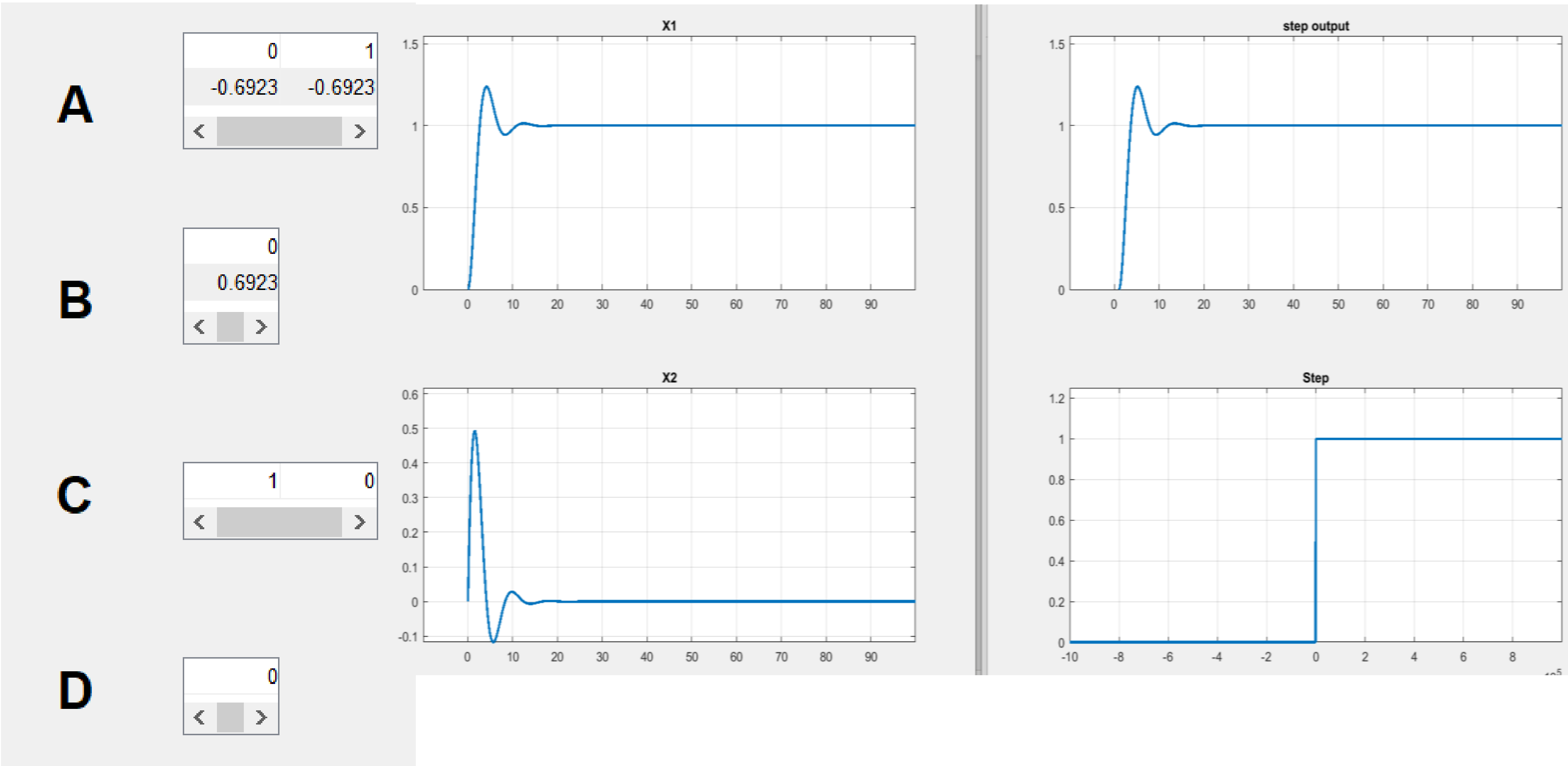


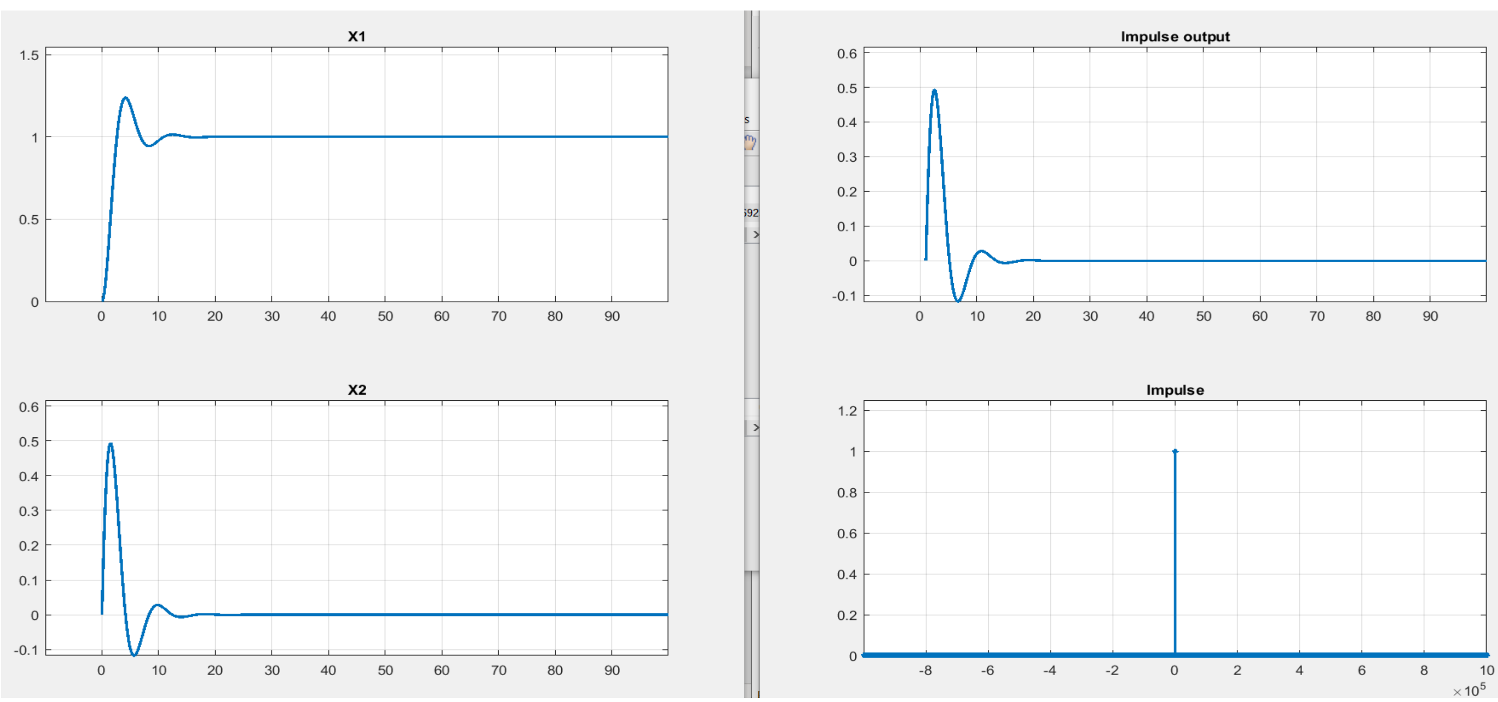
6) $12\ddot{y} + 9\dot{y} + y = 13u$



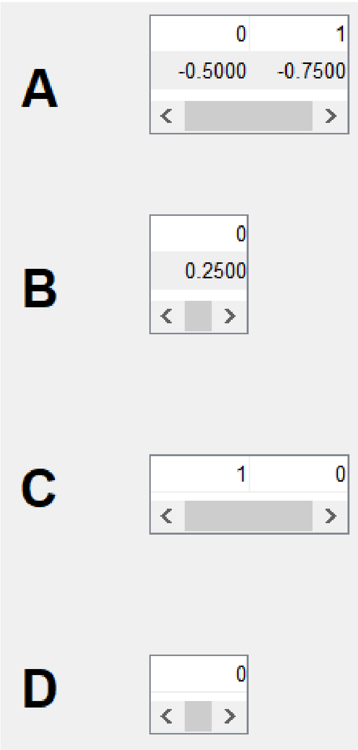


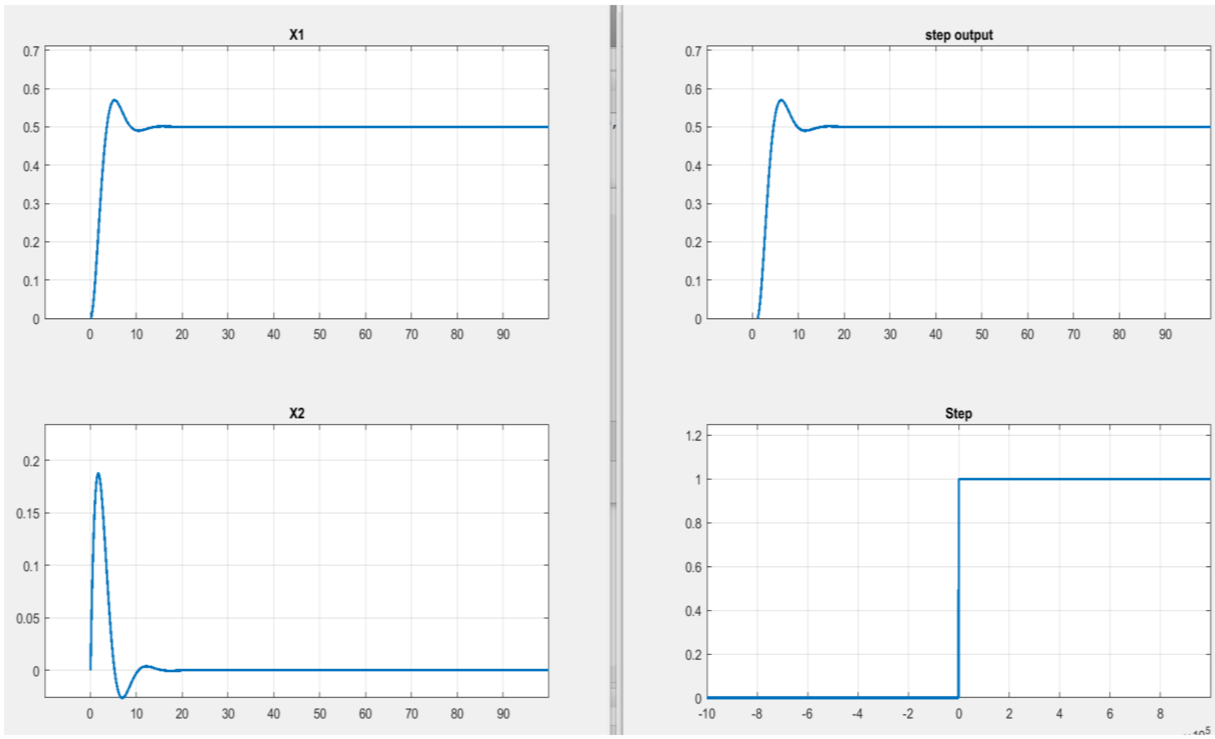
7) $13\ddot{y} + 9\dot{y} + 9y = 9u$

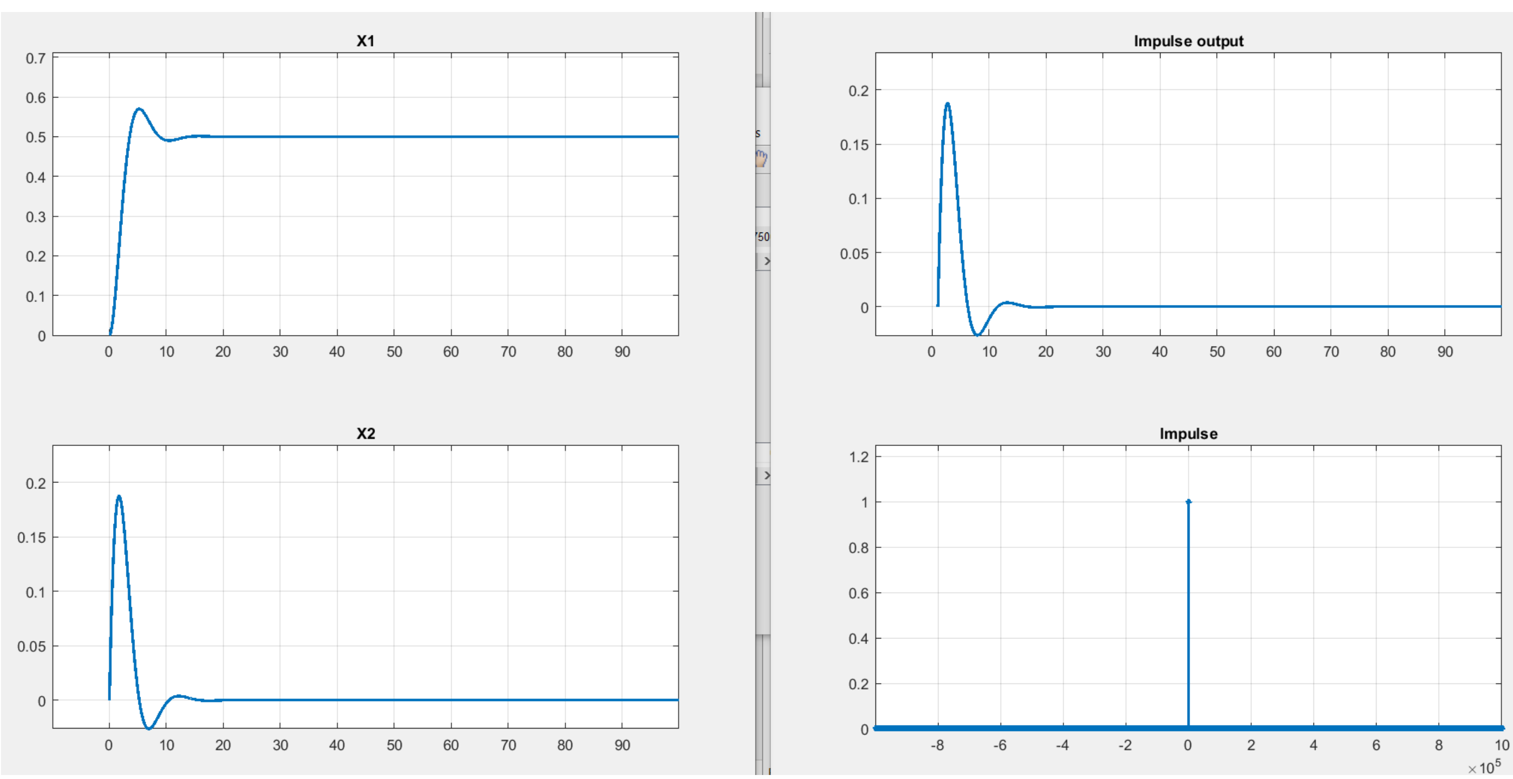




8) $8\ddot{y} + 6\dot{y} + 4y = 2u$







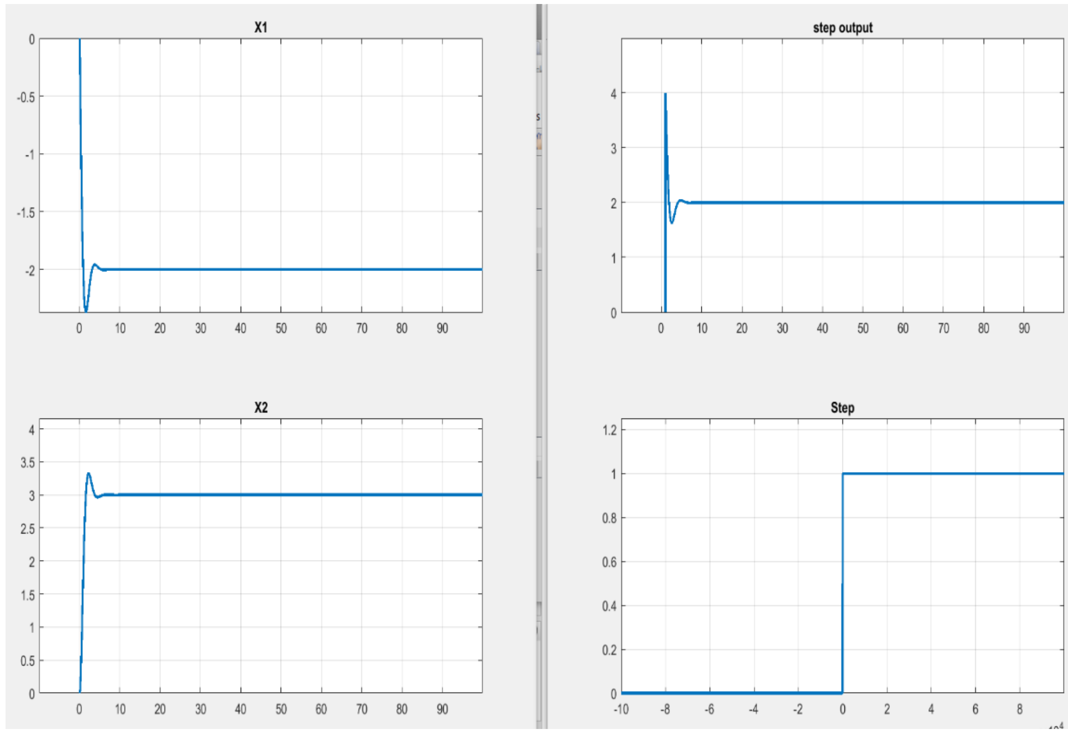
9) $\ddot{y} + 2\dot{y} + 3y = 4\ddot{u} + 5\dot{u} + 6u$

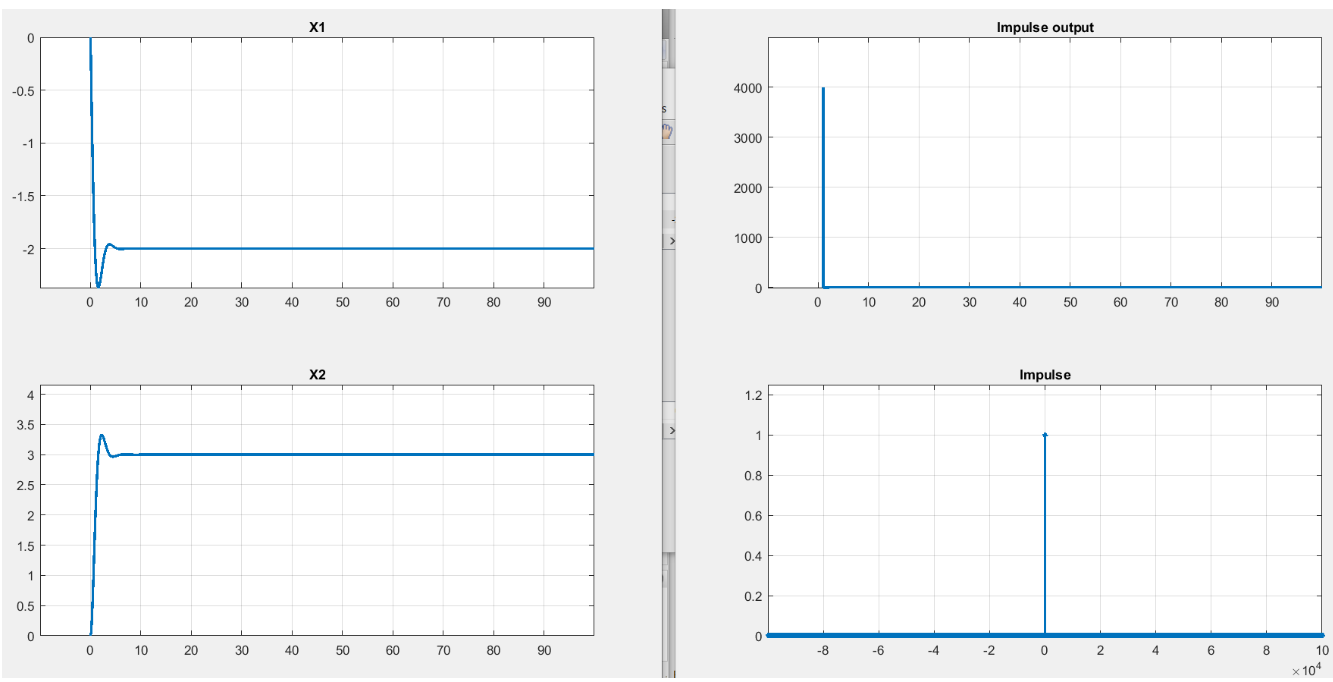
A $\begin{bmatrix} 0 & 1 \\ -3 & -2 \end{bmatrix}$

B $\begin{bmatrix} -3 \\ 0 \end{bmatrix}$

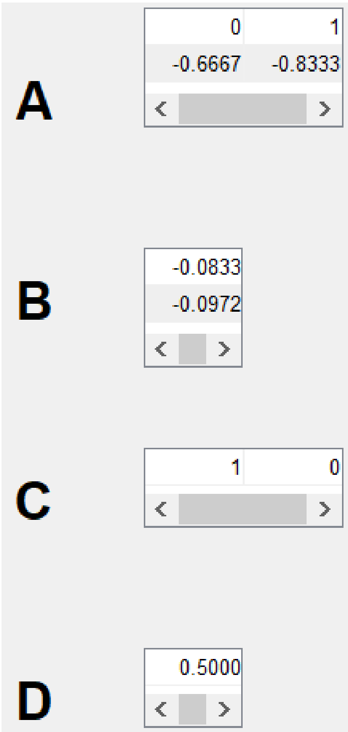
C $\begin{bmatrix} 1 & 0 \end{bmatrix}$

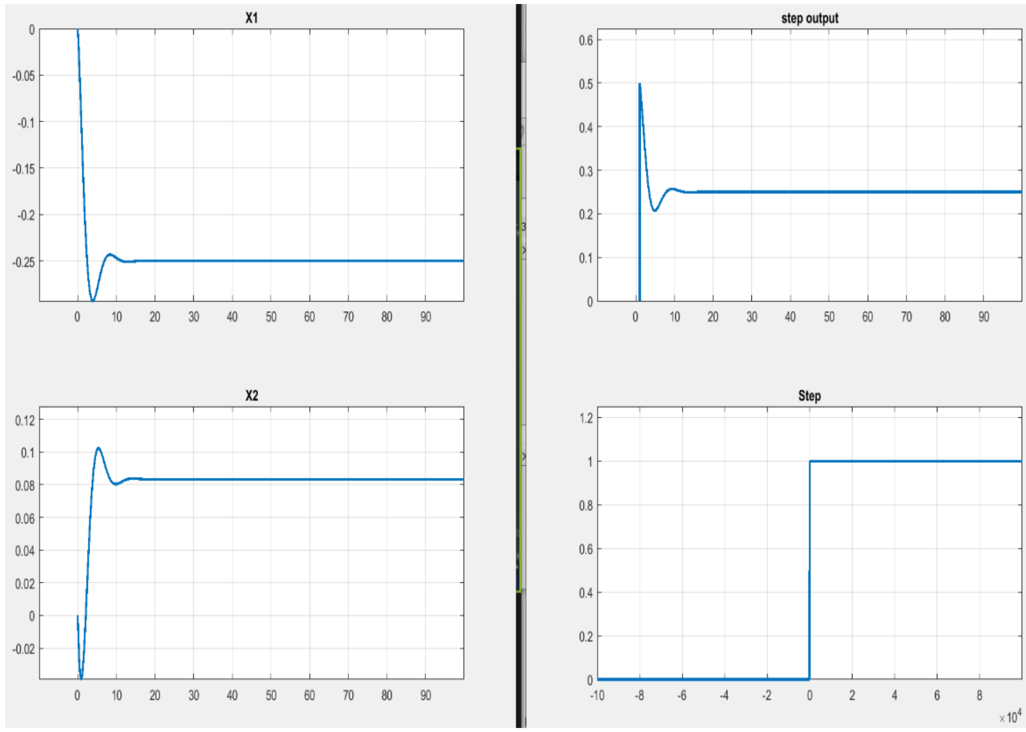
D $\begin{bmatrix} 4 \end{bmatrix}$

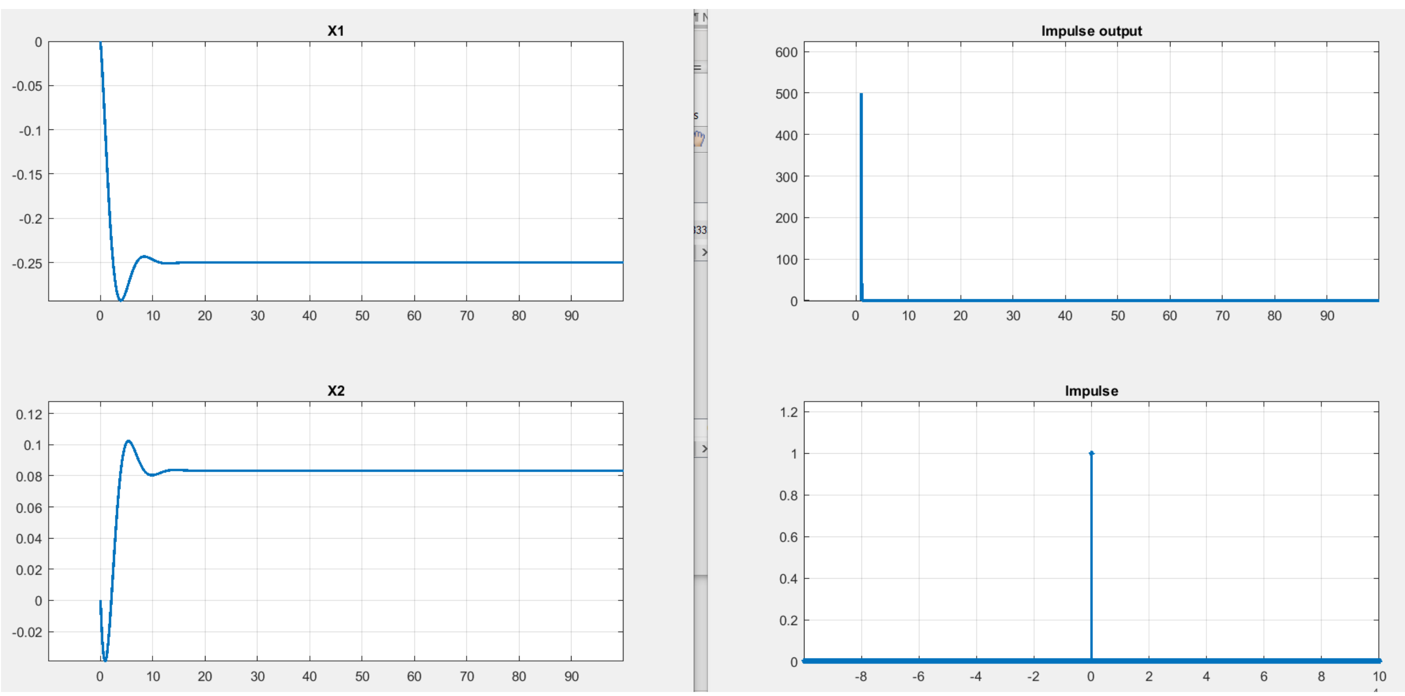




10) $6\ddot{y} + 5\dot{y} + 4y = 3\ddot{u} + 2\dot{u} + u$







Third Order Systems:

1) $\dddot{y} + 3y = 5\dot{u} + 6u$

A

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -3 & 0 & 0 \end{bmatrix}$$

B

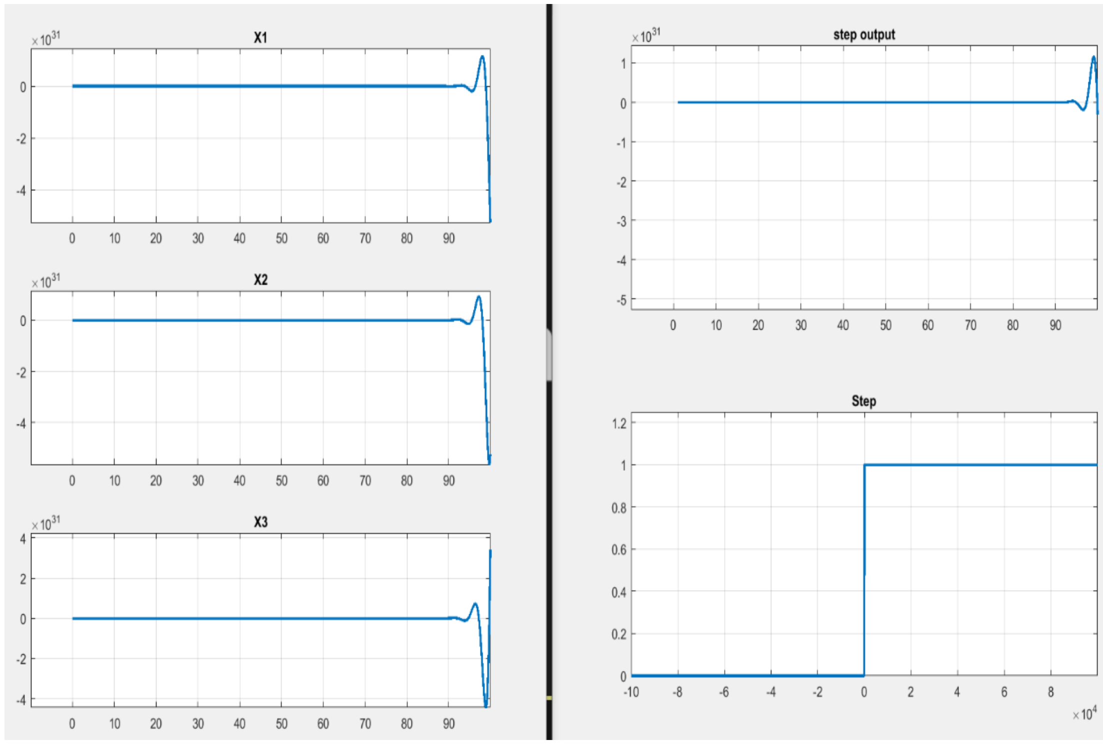
$$\begin{bmatrix} 0 \\ 5 \\ 6 \end{bmatrix}$$

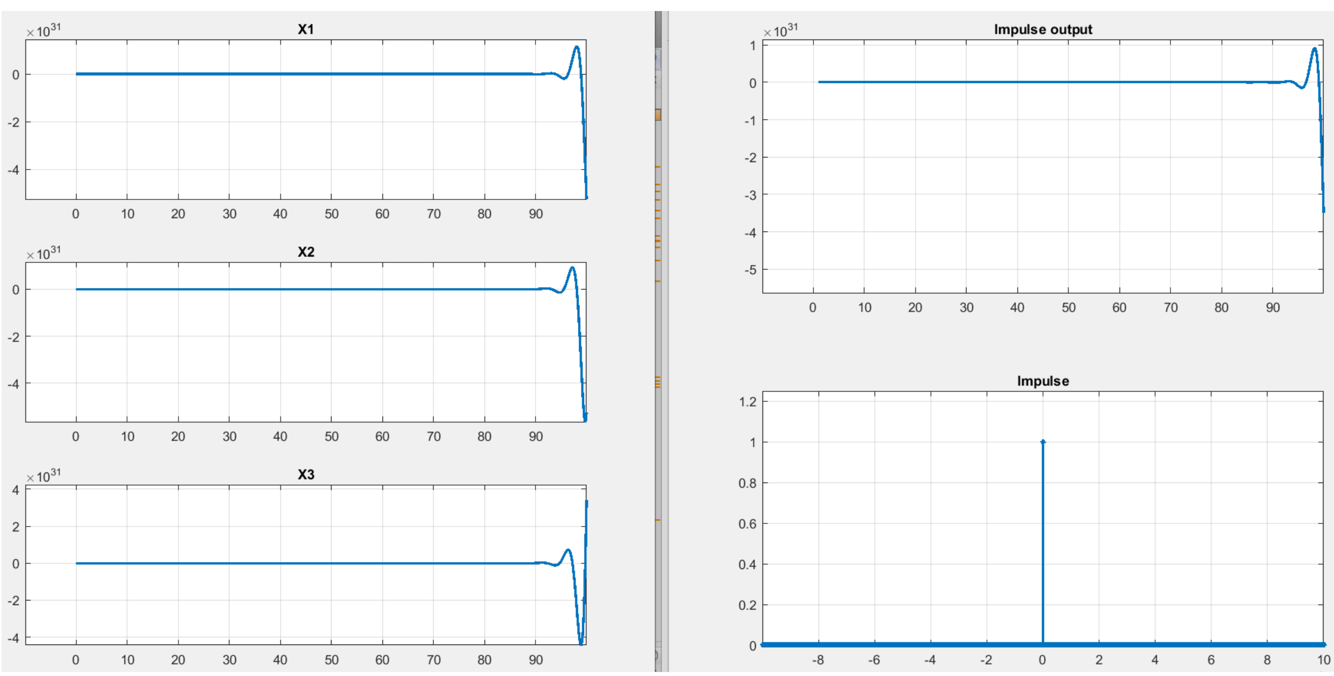
C

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

D

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





2) $\dddot{y} + 2\ddot{y} + 3\dot{y} + 4y = 5u$

A

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -4 & -3 & -2 \end{bmatrix}$$

B

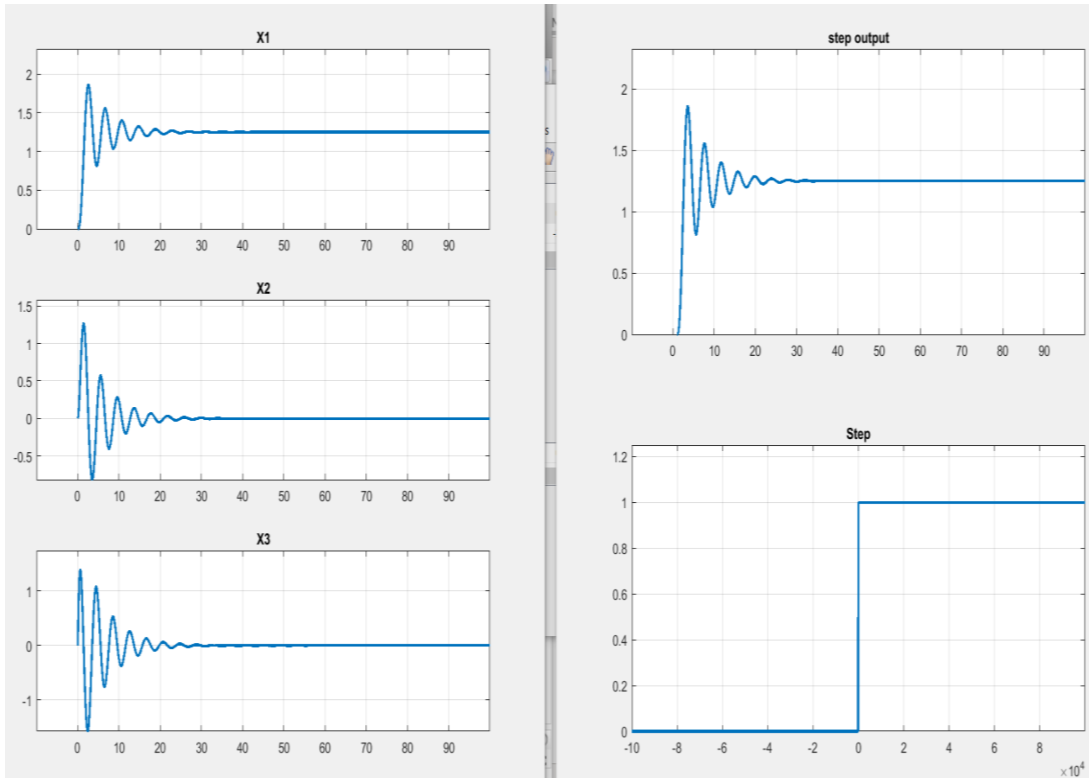
$$\begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix}$$

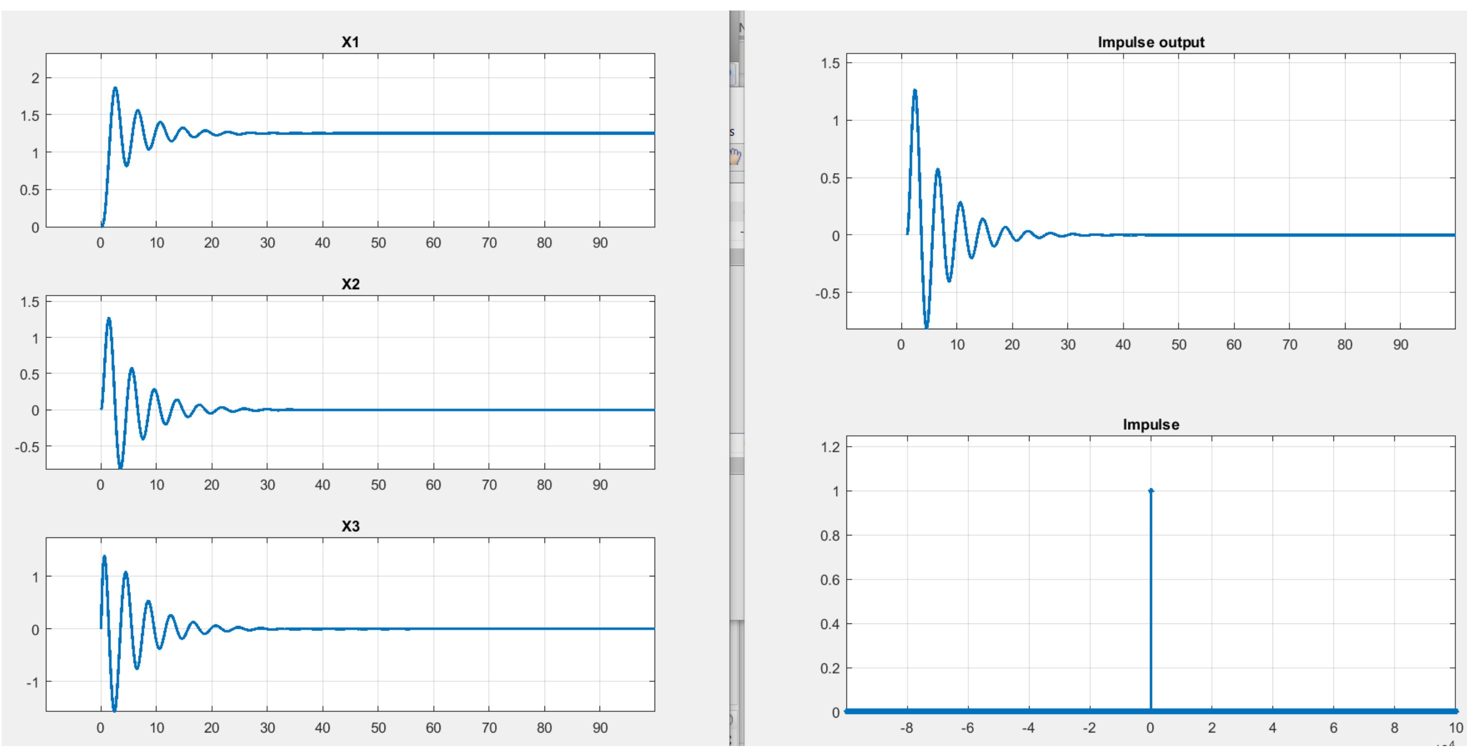
C

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

D

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





3) $5\dddot{y} + 4\ddot{y} + 3\dot{y} + 2y = u$

A

0	1	0
0	0	1
-0.4000	-0.6000	-0.8000

B

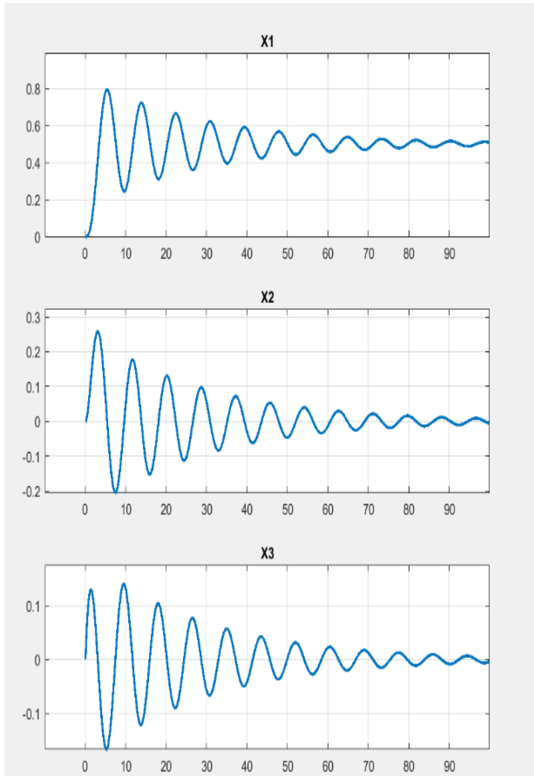
0
0
0.2000

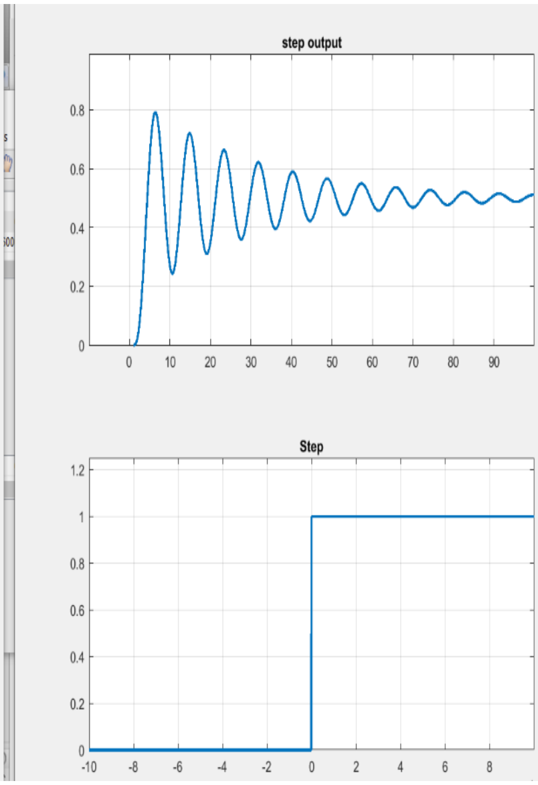
C

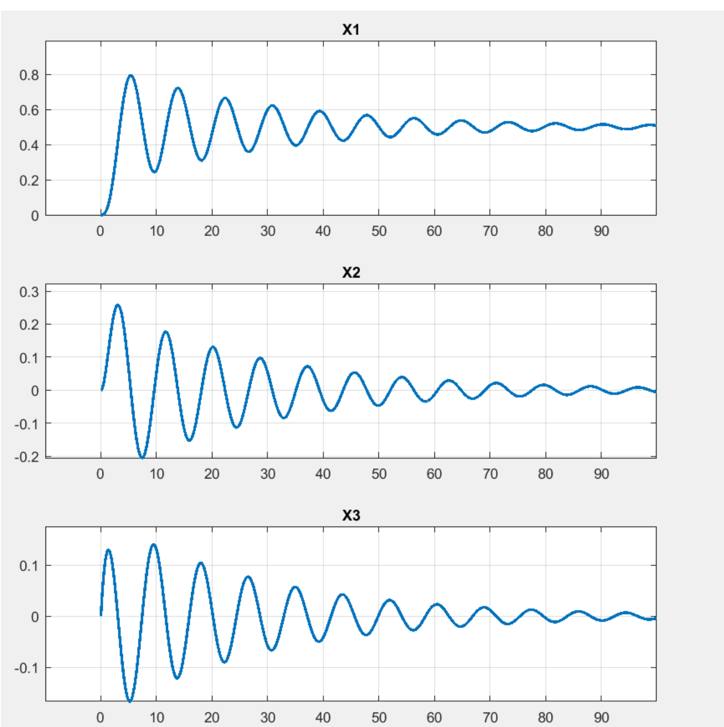
1	0	0

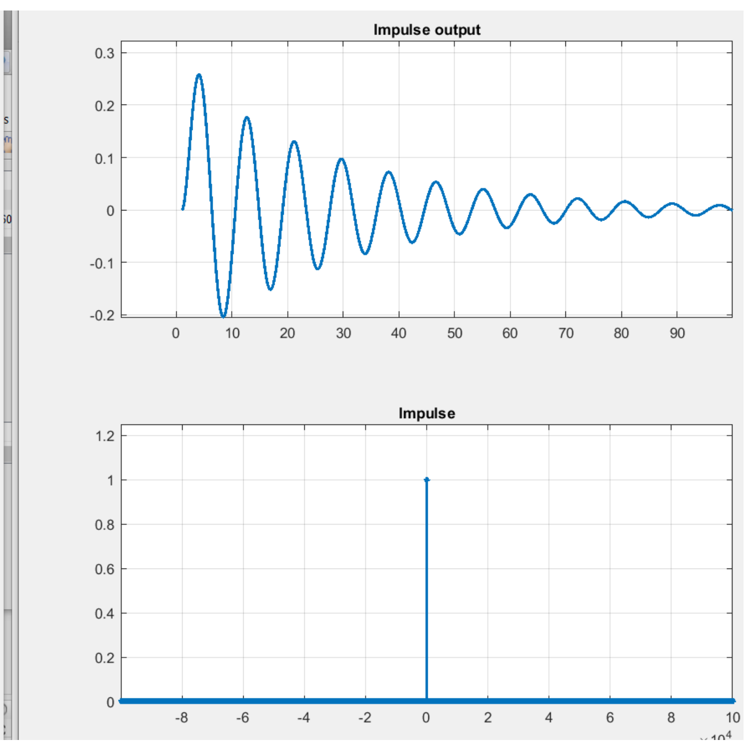
D

0









4) $23\dddot{y} + 5\ddot{y} - 2\dot{y} - 86y = -14u$

A

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 3.7391 & 0.0870 & -0.2174 \end{bmatrix}$$

B

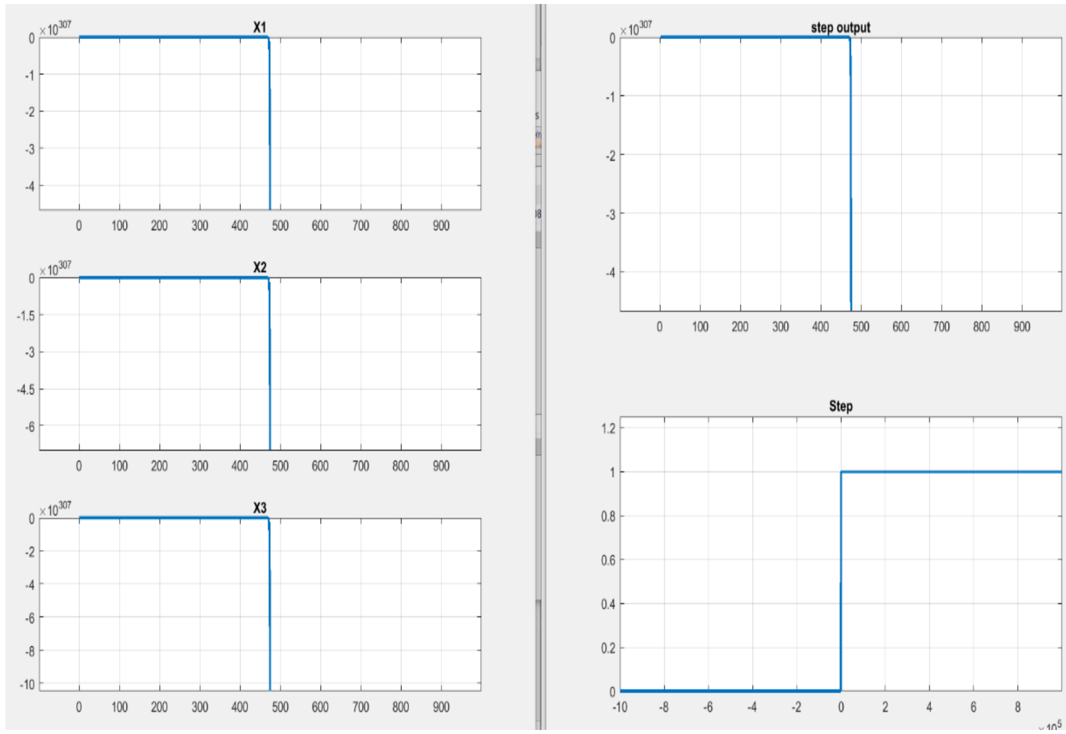
$$\begin{bmatrix} 0 \\ 0 \\ -0.6087 \end{bmatrix}$$

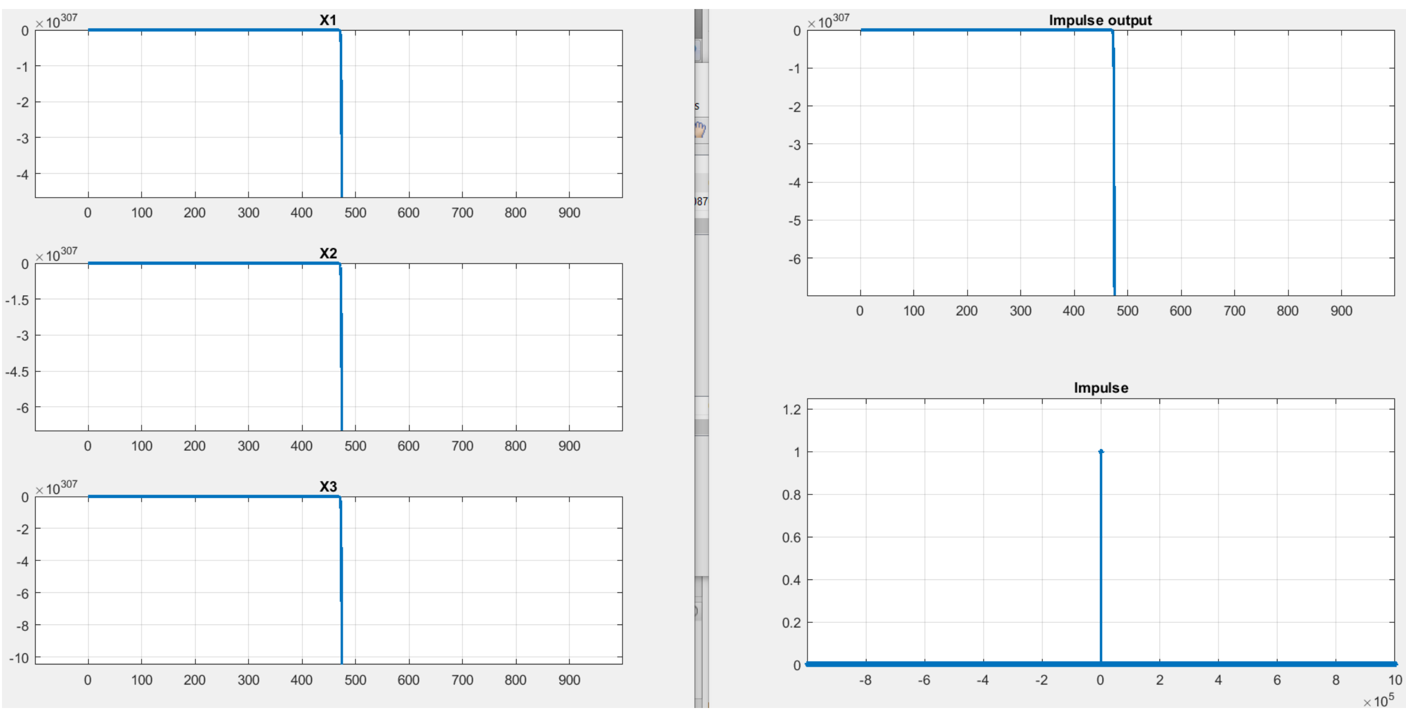
C

$$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

D

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





5) $-68\dddot{y}+42\ddot{y}-96\dot{y}-26y=-84u$

A

0	1	0
0	0	1
-0.3824	-1.4118	0.6176

B

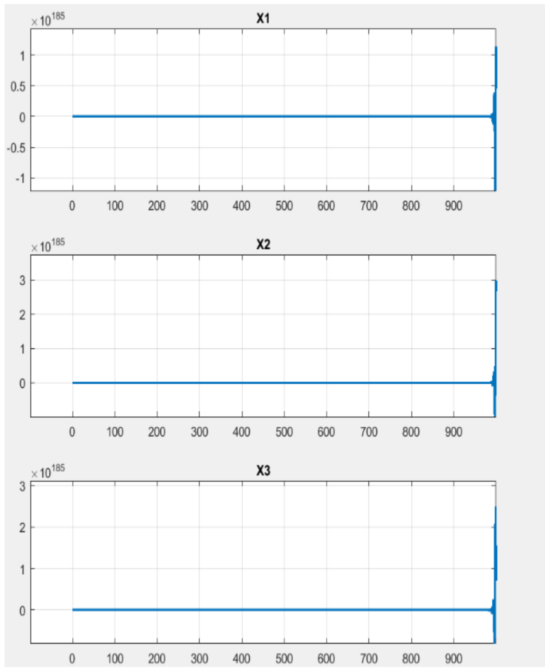
0
0
1.2353

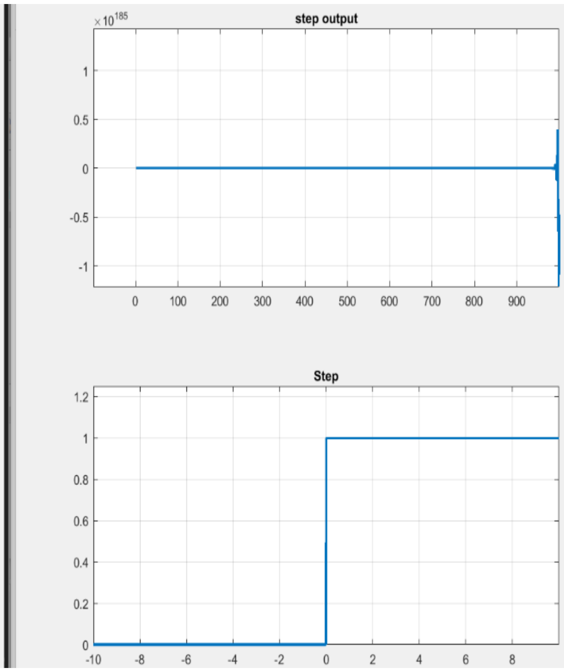
C

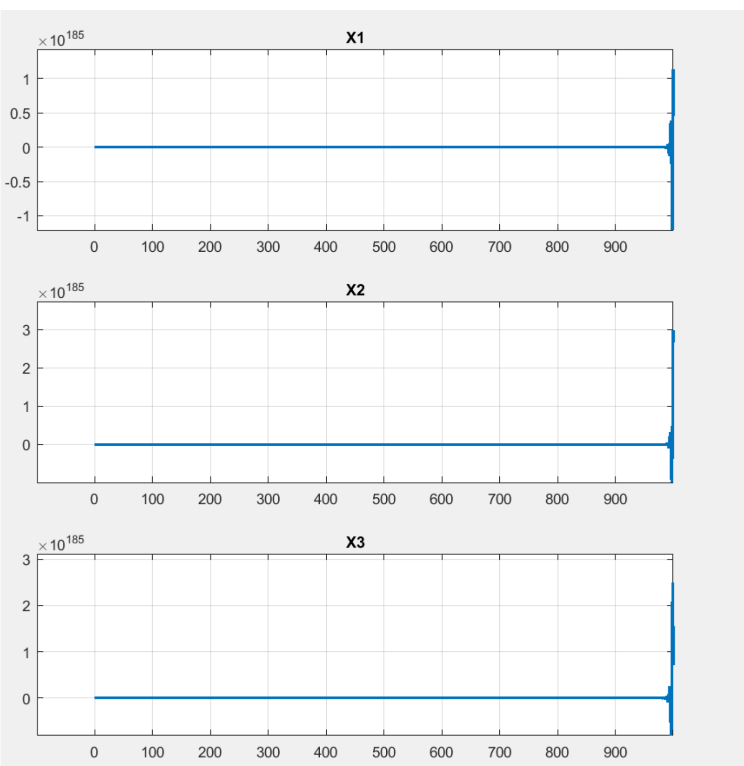
1	0	0

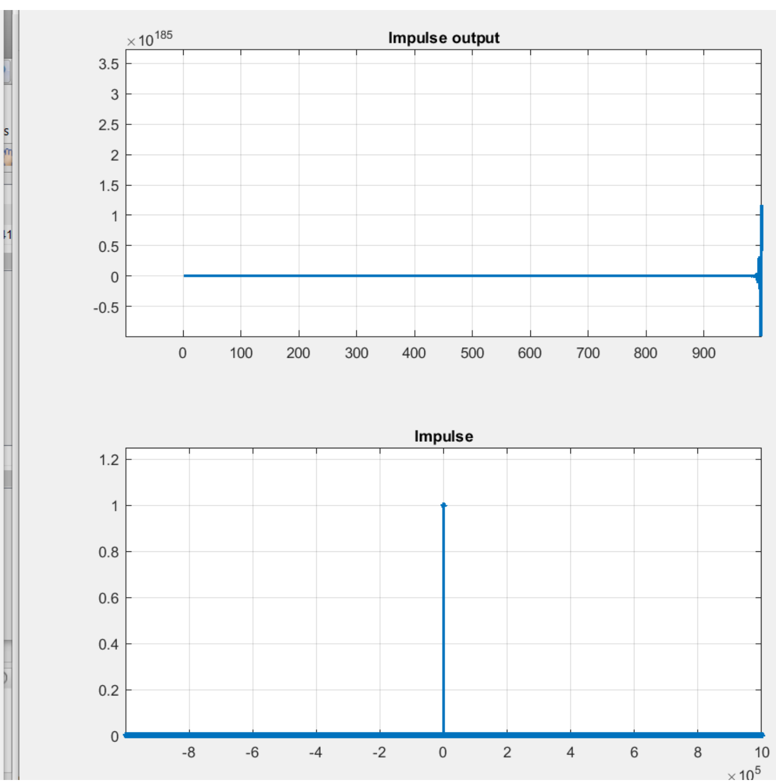
D

0









6) $84\dddot{y} - 35\ddot{y} + 14\dot{y} + 41y = -10u$

A

0	1	0
0	0	1
-0.4881	-0.1667	0.4167

B

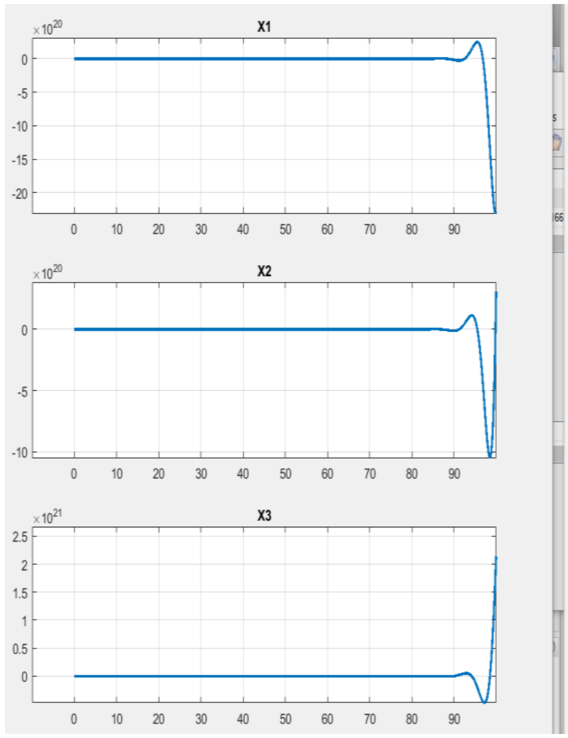
0
0
-0.1190

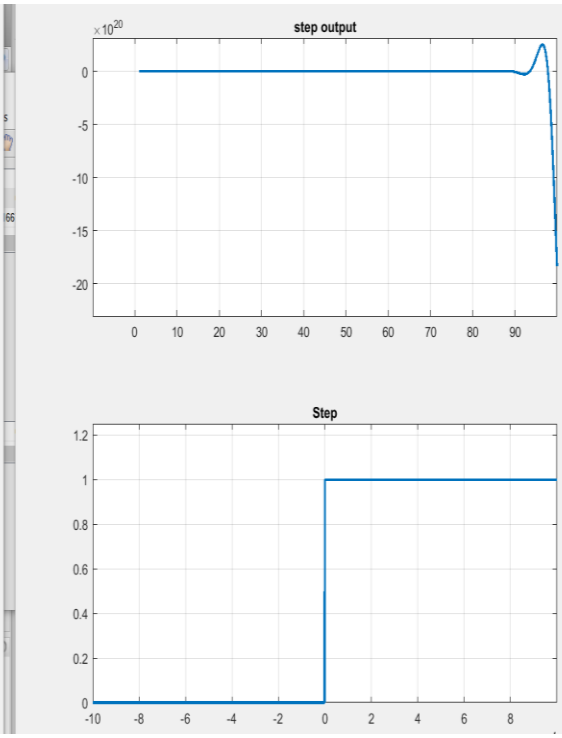
C

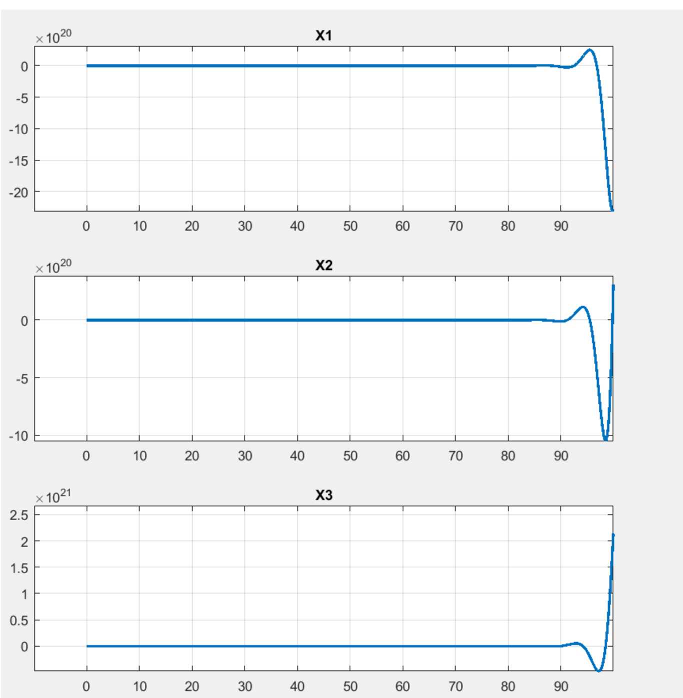
1	0	0

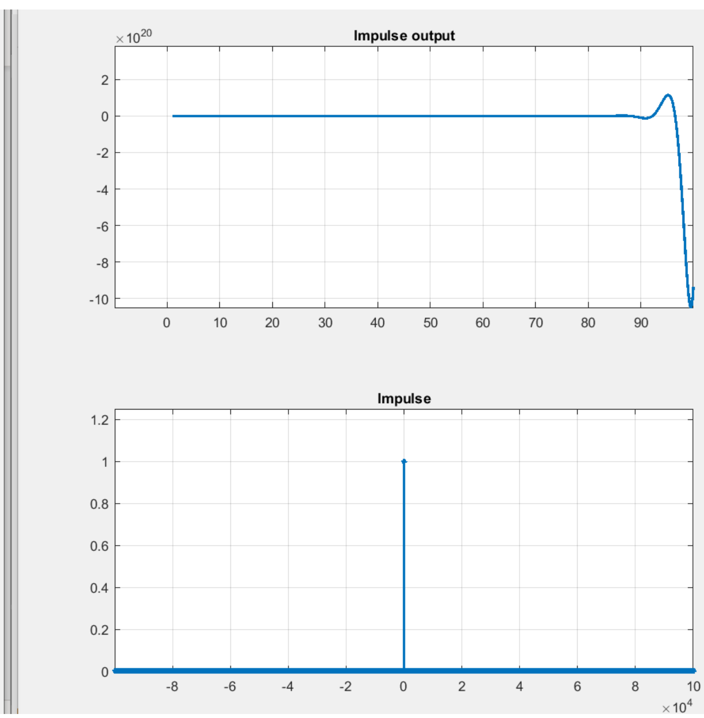
D

0









$$7)\ 35\dddot{y} - 94\ddot{y} - 79\dot{y} + 59y = 74\dddot{u} + 63\ddot{u} - 20\dot{u} + 18u$$

A

0	1	0
0	0	1
-1.6857	2.2571	2.6857

B

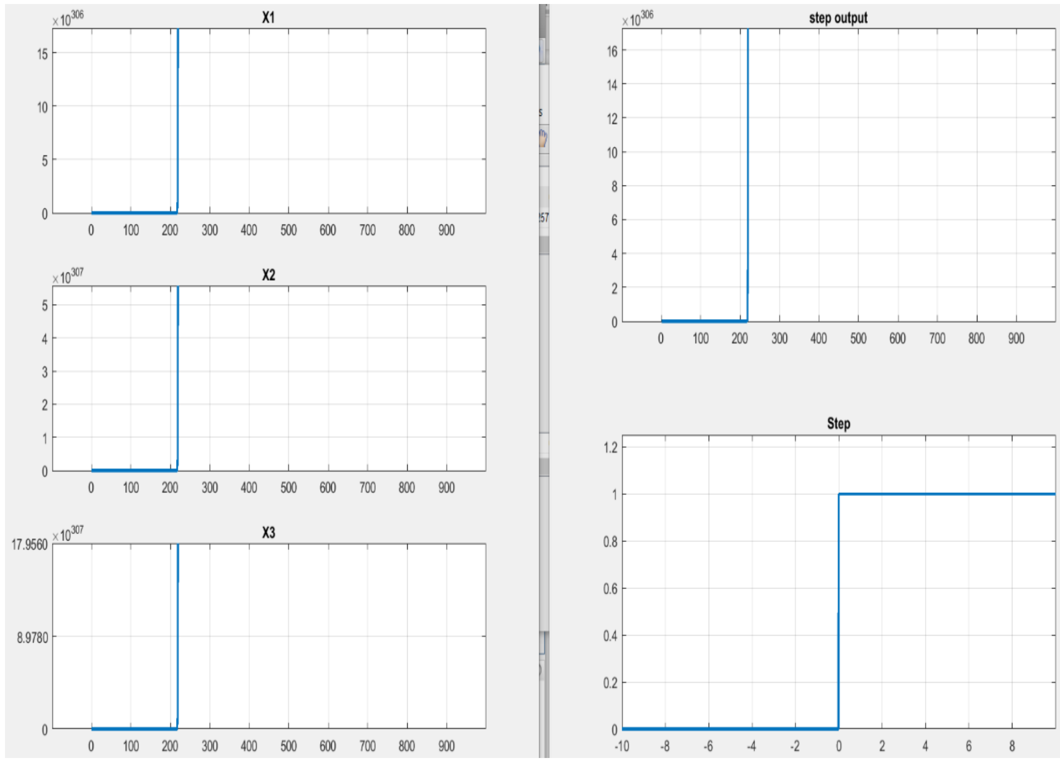
6.7069
22.2137
71.7485

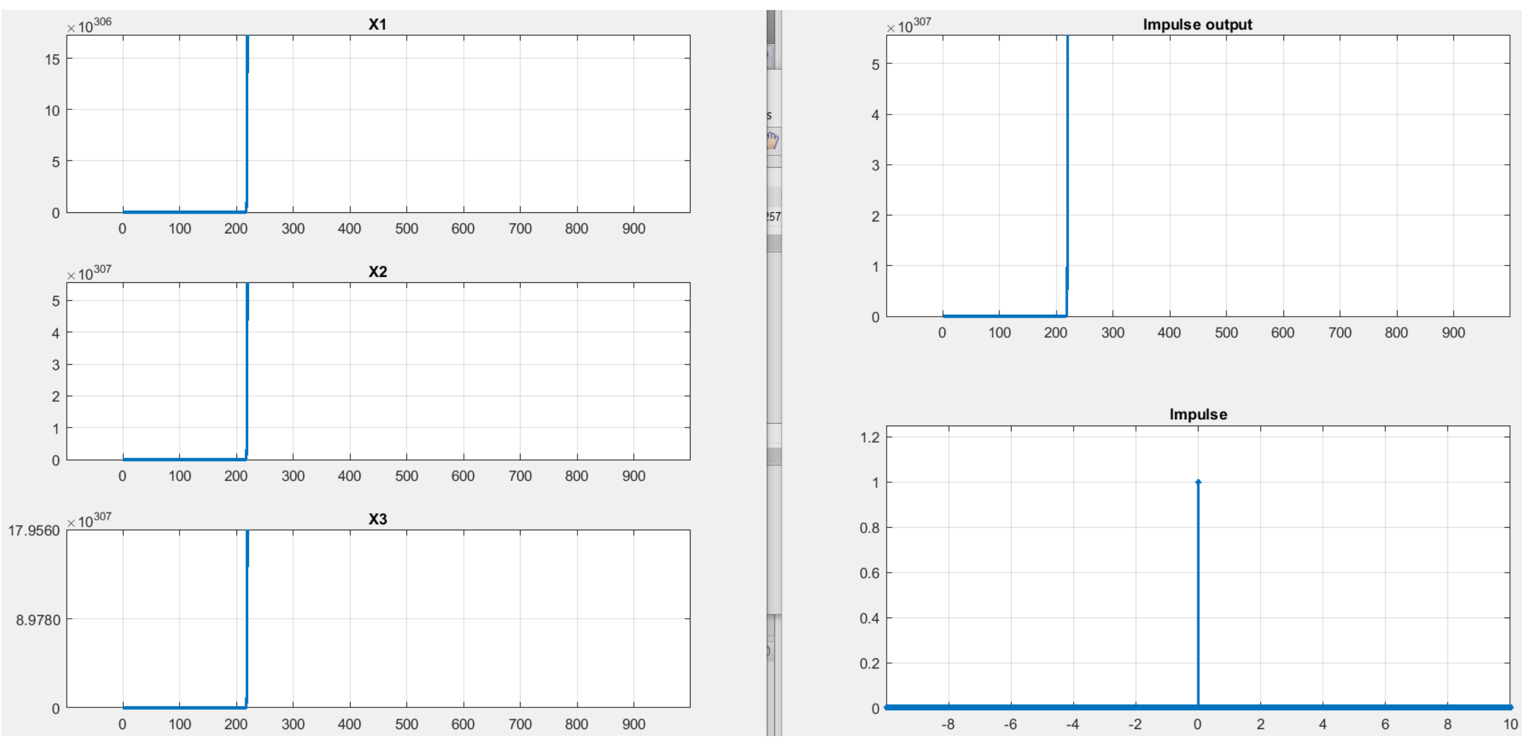
C

1	0	0

D

2.1143





8) $84\dddot{y} - 35\ddot{y} + 14\dot{y} + 41y = -10u$

A

0	1	0
0	0	1
0.0110	-0.0989	0.0110

B

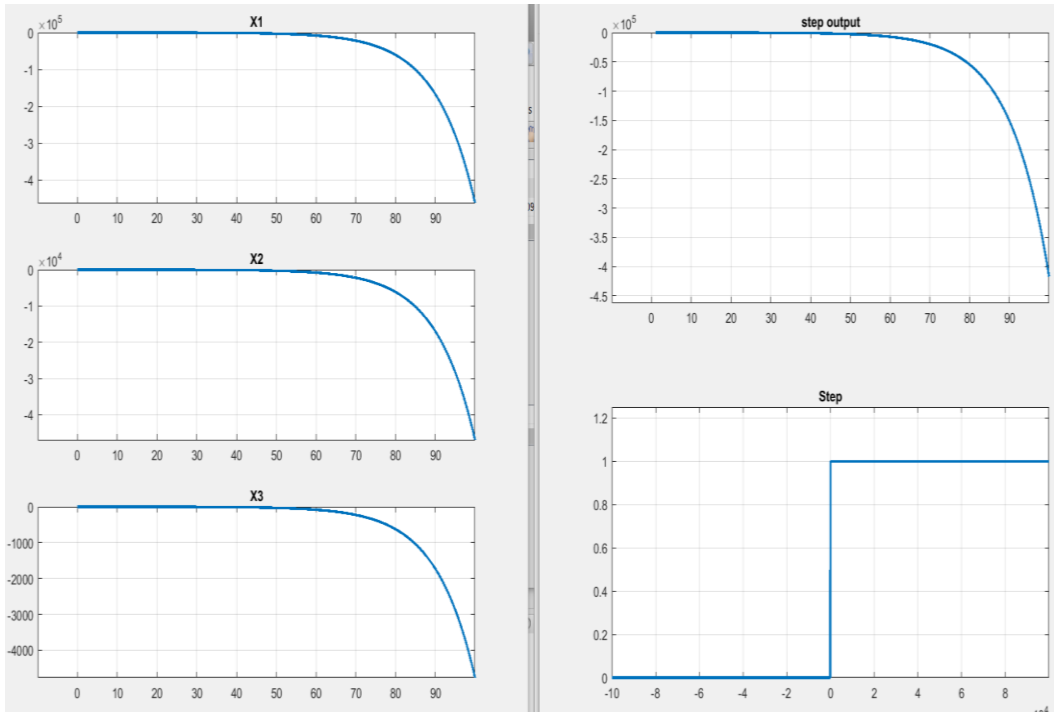
0
0
-0.2308

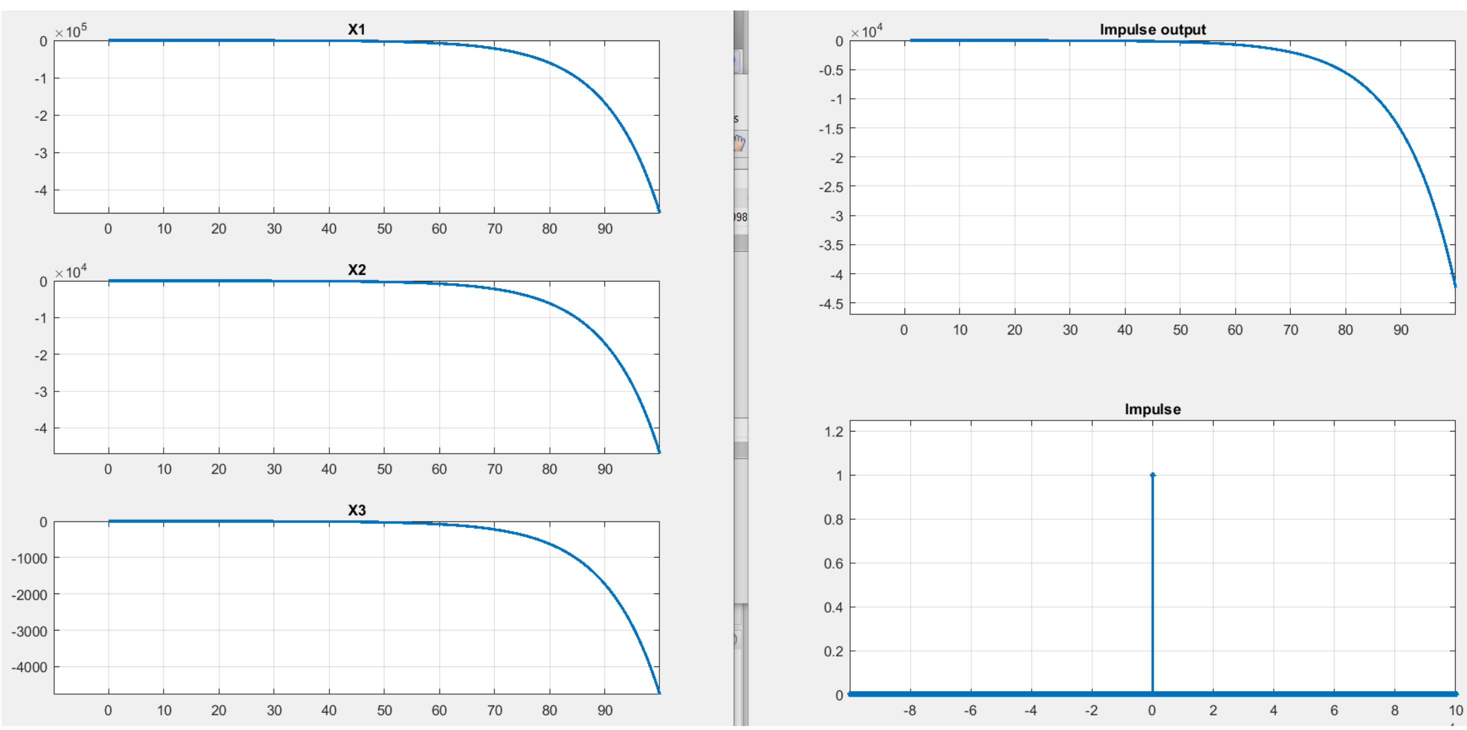
C

1	0	0

D

0





9) $-3\dddot{y} - 62\ddot{y} - 91\dot{y} + 19y = 47\dddot{u} - 2\ddot{u} - 20\dot{u} + 69u$

A

0	1	0
0	0	1
4.3333	-30.3333	-20.6667

B

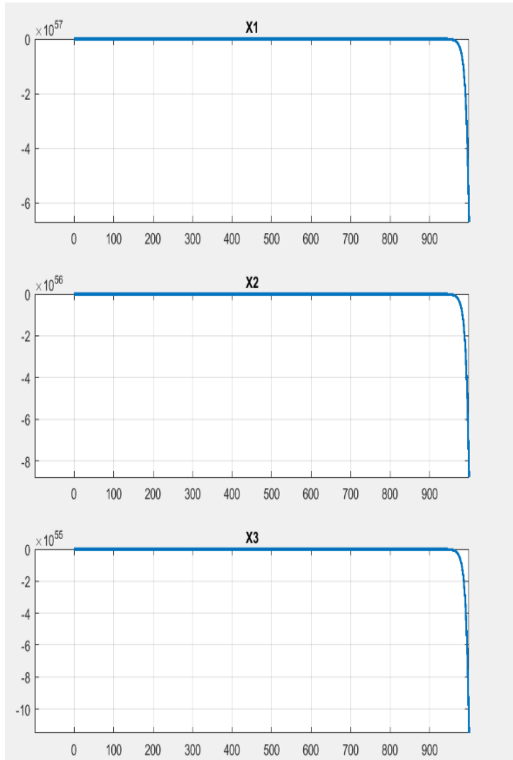
324.4444
-6.2233...
1.1868e...

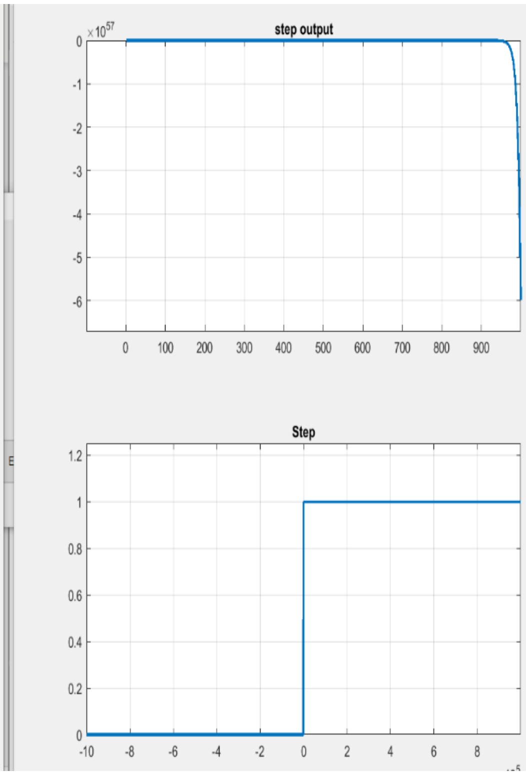
C

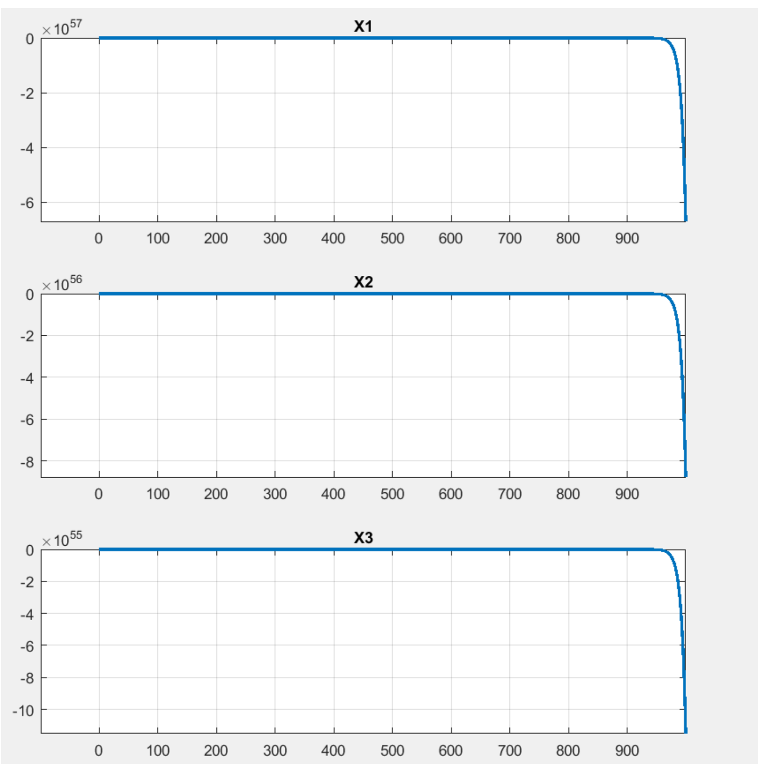
1	0	0

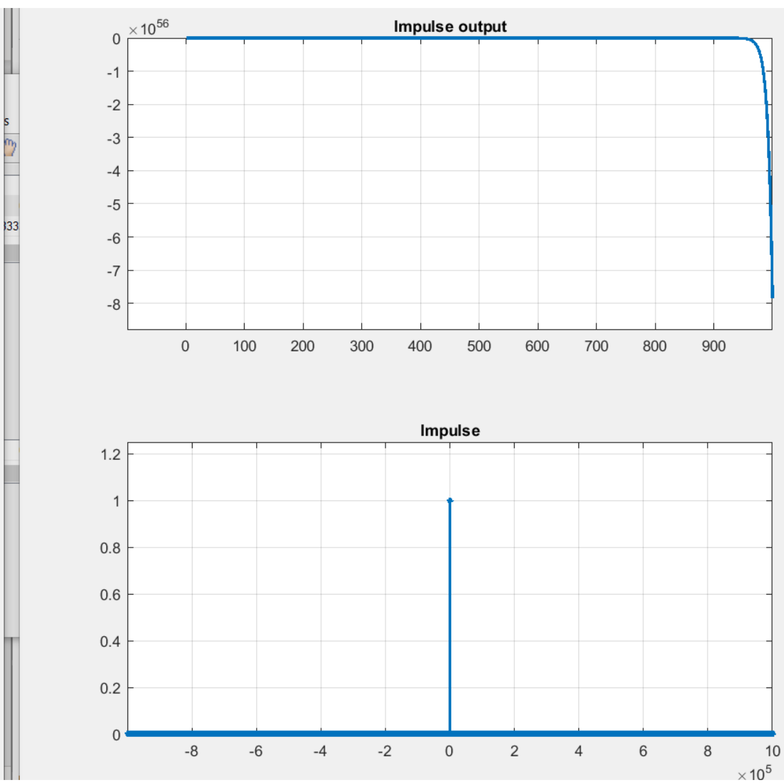
D

-15.6667









10) $\dddot{y} + 2\ddot{y} + 3\dot{y} + 4y = 5\dddot{u} + 6\ddot{u} + 7\dot{u} + 8u$

A

0	1	0
0	0	1
-4	-3	-2

B

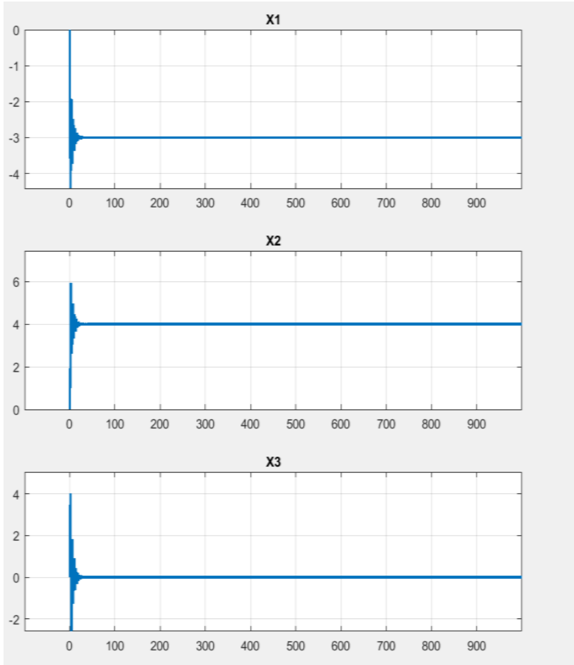
-4
0
0

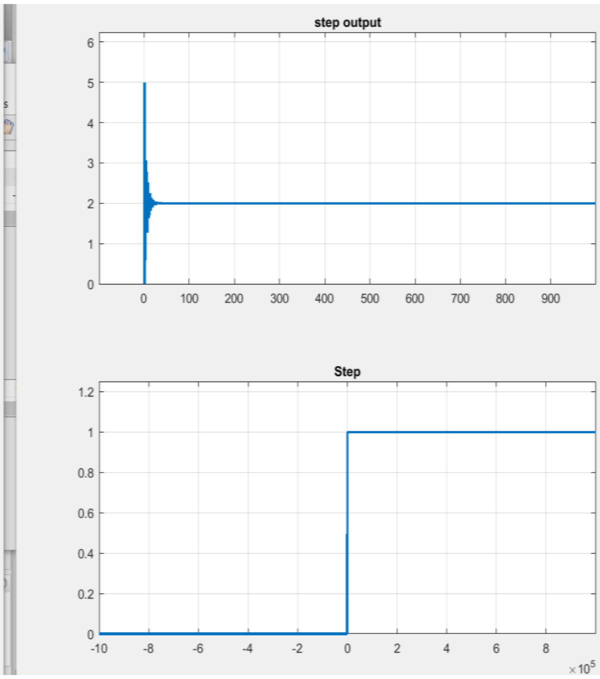
C

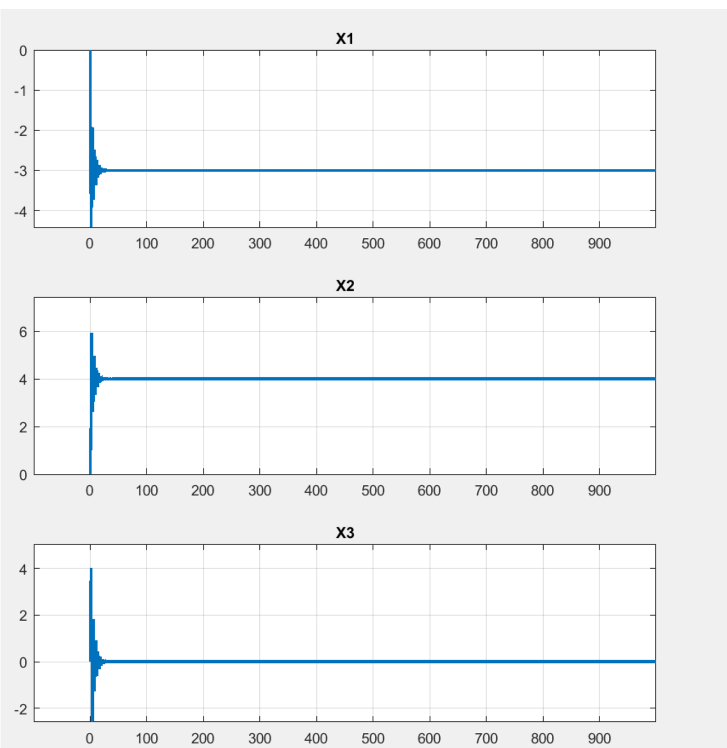
1	0	0

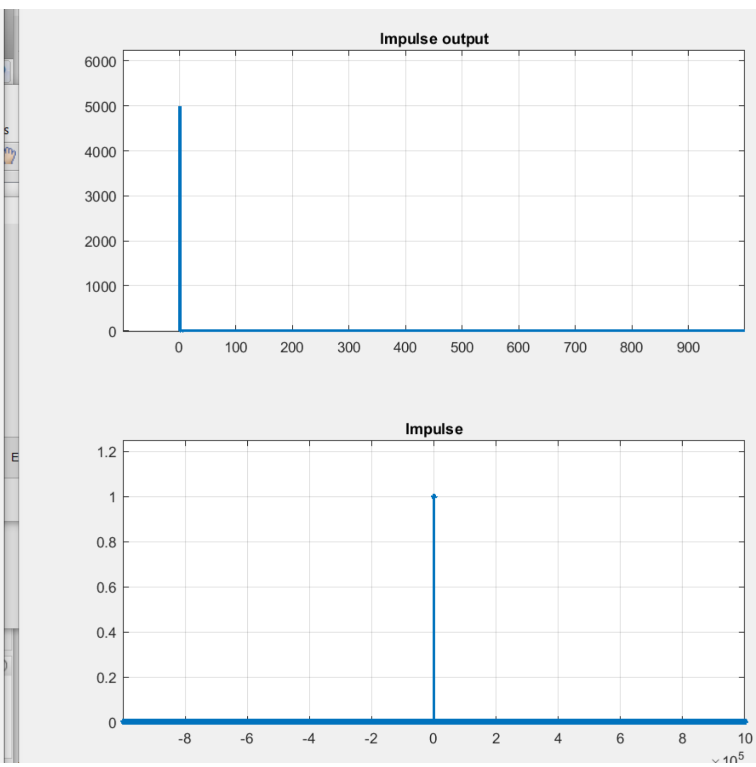
D

5



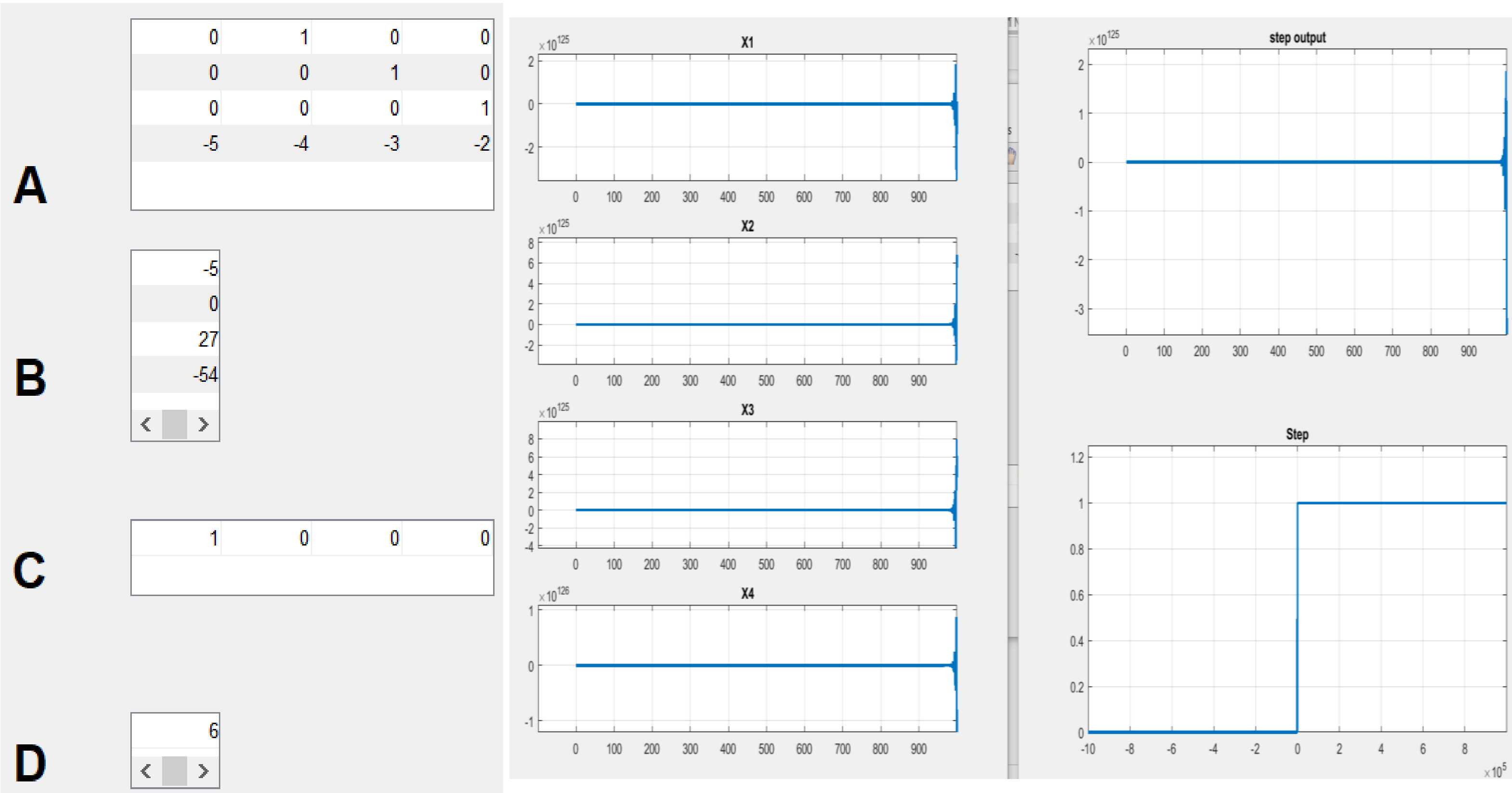


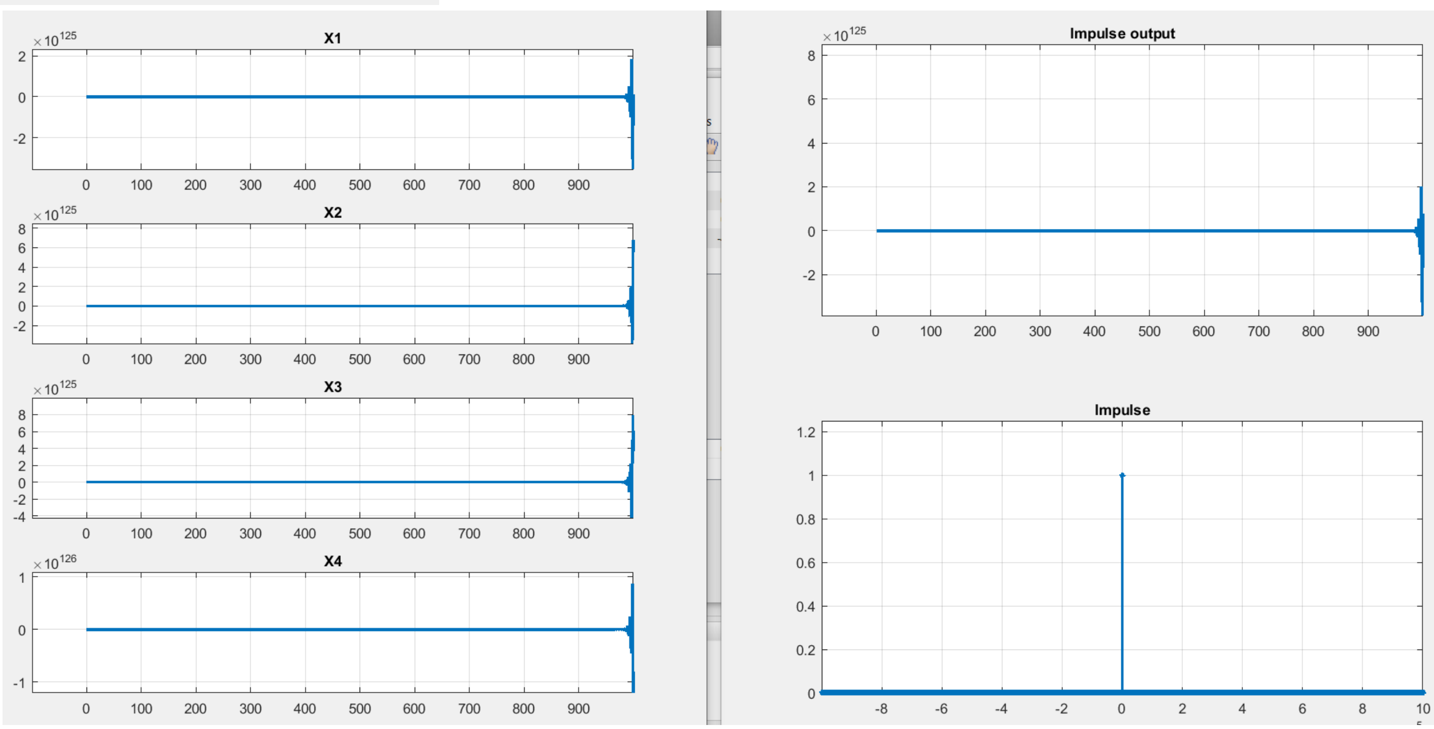




Fourth Order Systems:

1) $y^{(4)} + 2y^{(3)} + 3y^{(2)} + 4y^{(1)} + 5y = 6u^{(4)} + 7u^{(3)} + 8u^{(2)} + 9u^{(1)} + 10u$





2) $10y^{(4)} + 9y^{(3)} + 8y^{(2)} + 7y^{(1)} + 6y = 5u^{(4)} + 4u^{(3)} + 3u^{(2)} + 2u^{(1)} + u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-0.6000	-0.7000	-0.8000	-0.9000

B

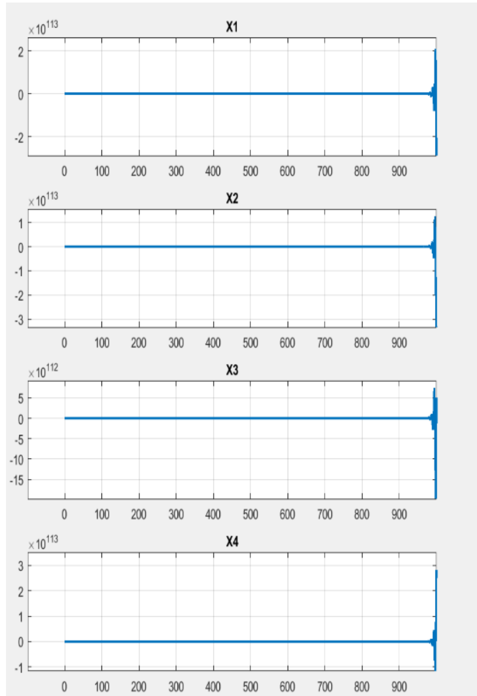
-0.0500
-0.0550
0.5285
-0.5967

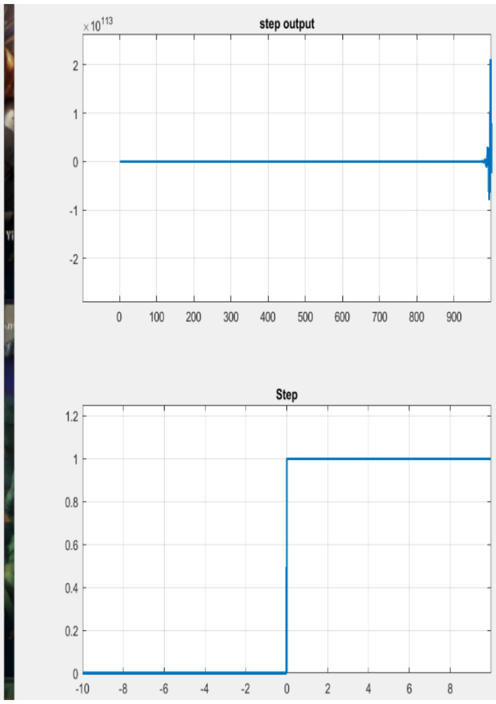
C

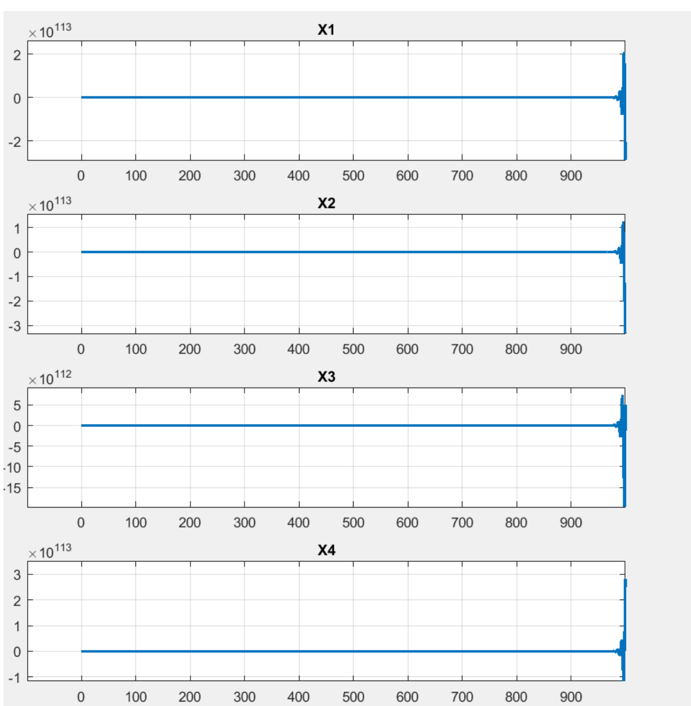
1	0	0	0

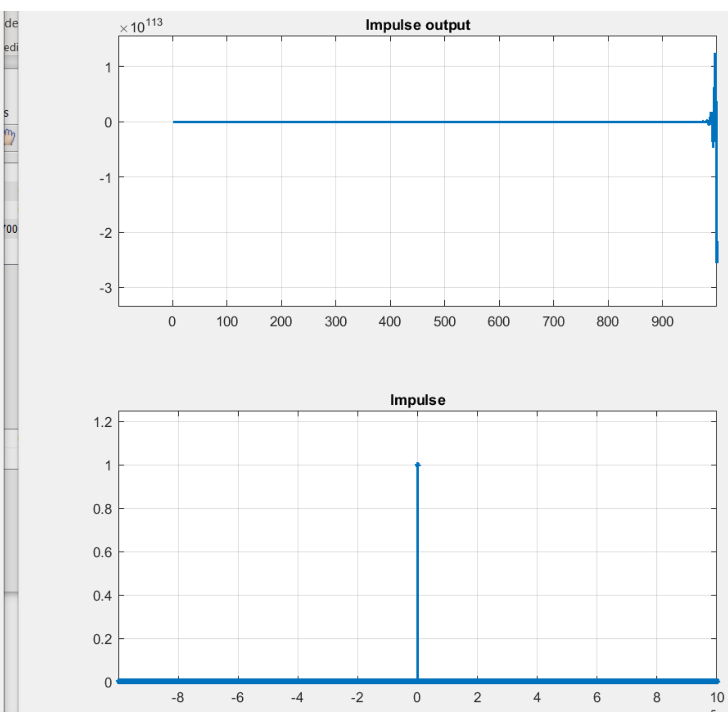
D

0.5000









3) $10y^{(4)} + 2y^{(3)} + 7y^{(2)} + 8y^{(1)} + 4y = 6u^{(4)} + 7u^{(3)} + 8u^{(2)} + 9u^{(1)} + 10u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-0.4000	-0.8000	-0.7000	-0.1000

B

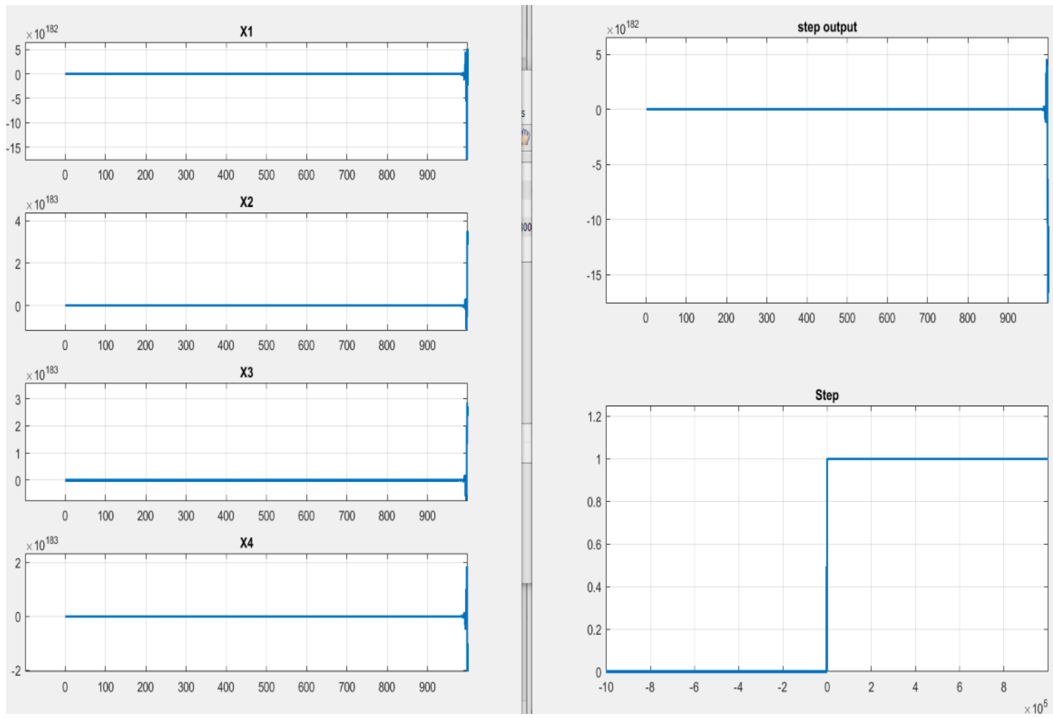
0.9700
0.0930
0.4826
-0.3094

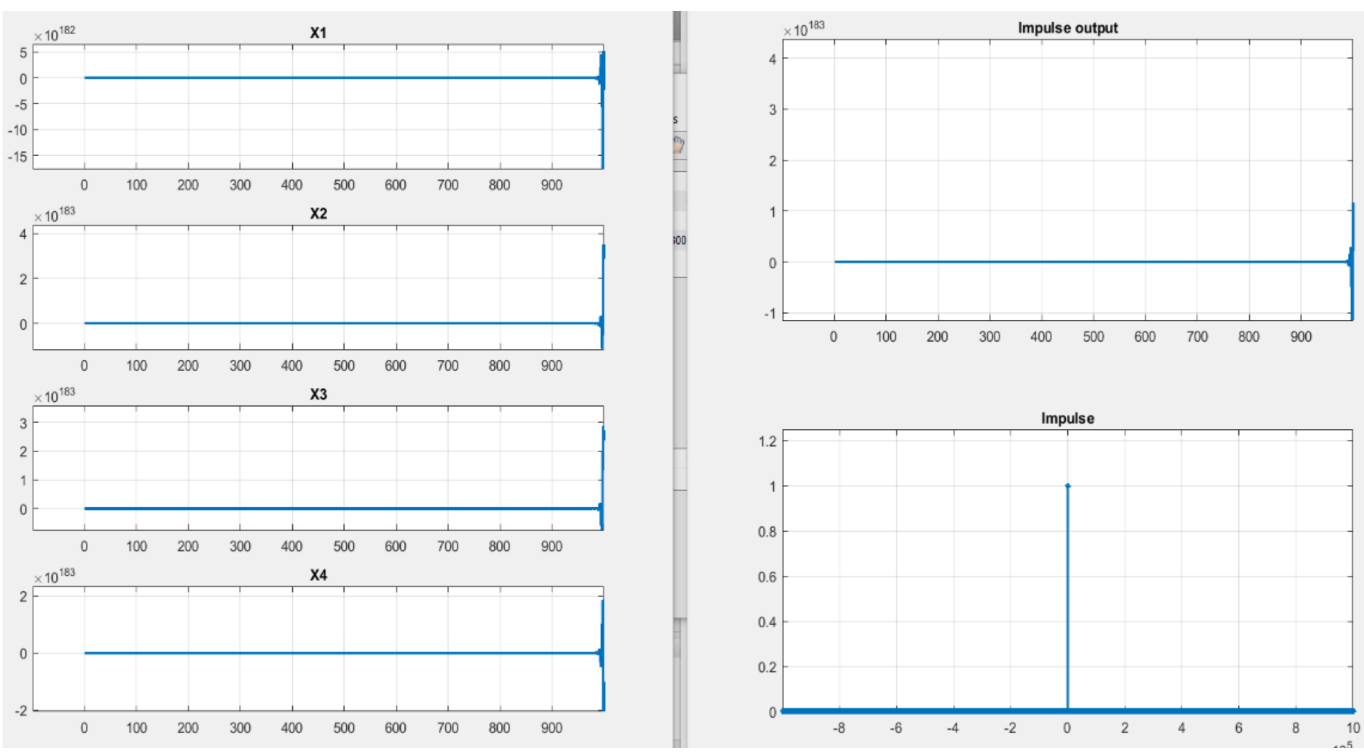
C

1	0	0	0

D

0.3000





$$4)\ 4y^{(4)} + 10y^{(3)} + 6y^{(2)} + 7y^{(1)} + 6y = 9u^{(4)} + 5u^{(3)} + 6u^{(2)} + 3u^{(1)} + 2u$$

A

0	1	0	0
0	0	1	0
0	0	0	1
-1.5000	-1.7500	-1.5000	-2.5000

B

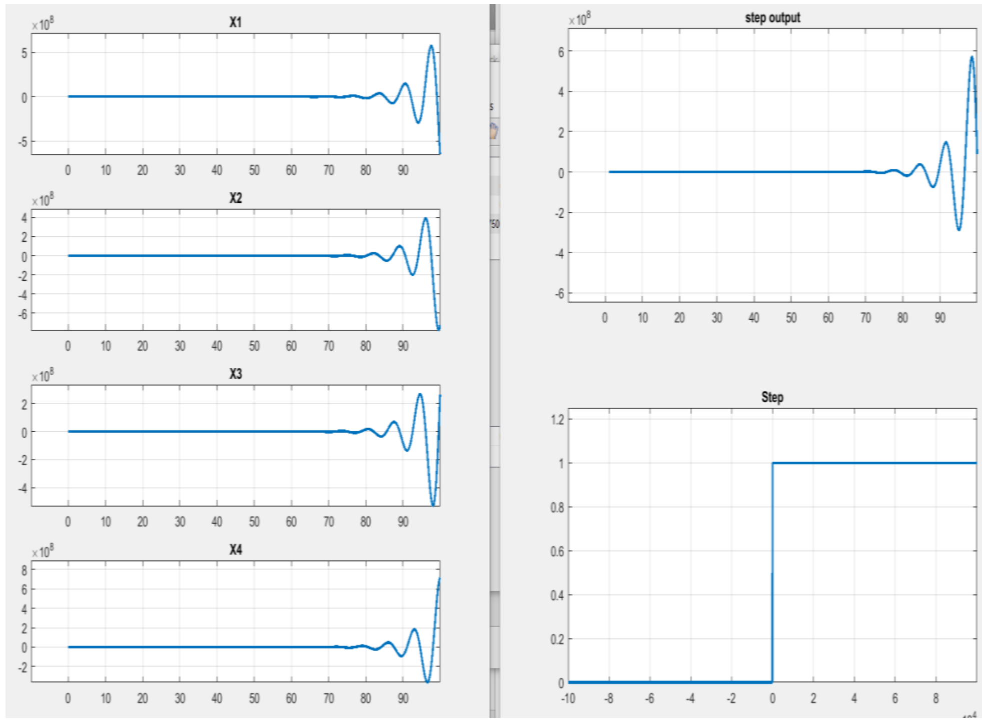
-4.3750
9.0625
-28.4688
62.3594

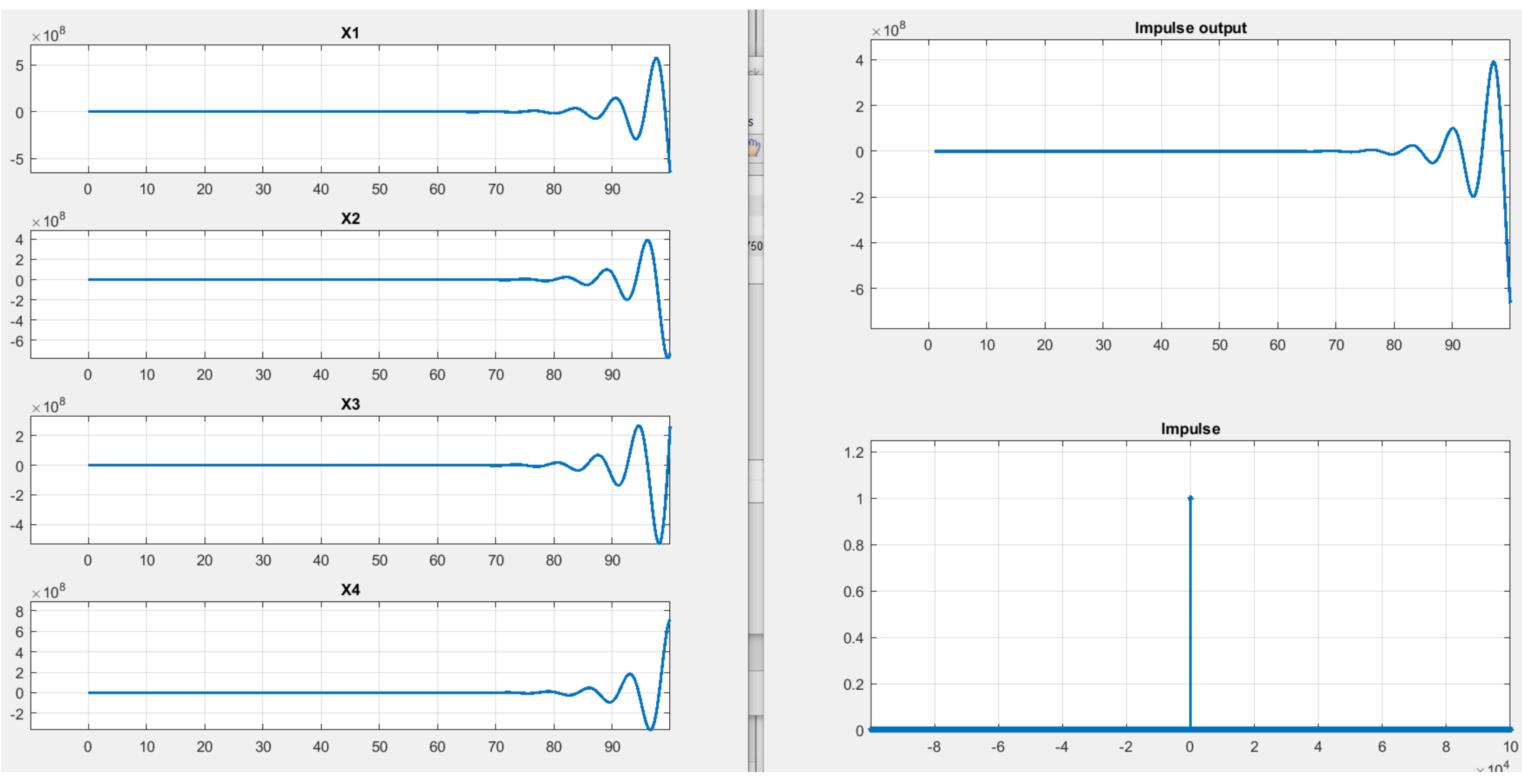
C

1	0	0	0

D

2.2500





5) $4y^{(4)} + 8\dddot{y} + 2\ddot{y} + 3\dot{y} + 4y = 3\dddot{u} + 6\ddot{u} + 8\dot{u} + u$

A

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & -0.7500 & -0.5000 & -2 \end{bmatrix}$$

B

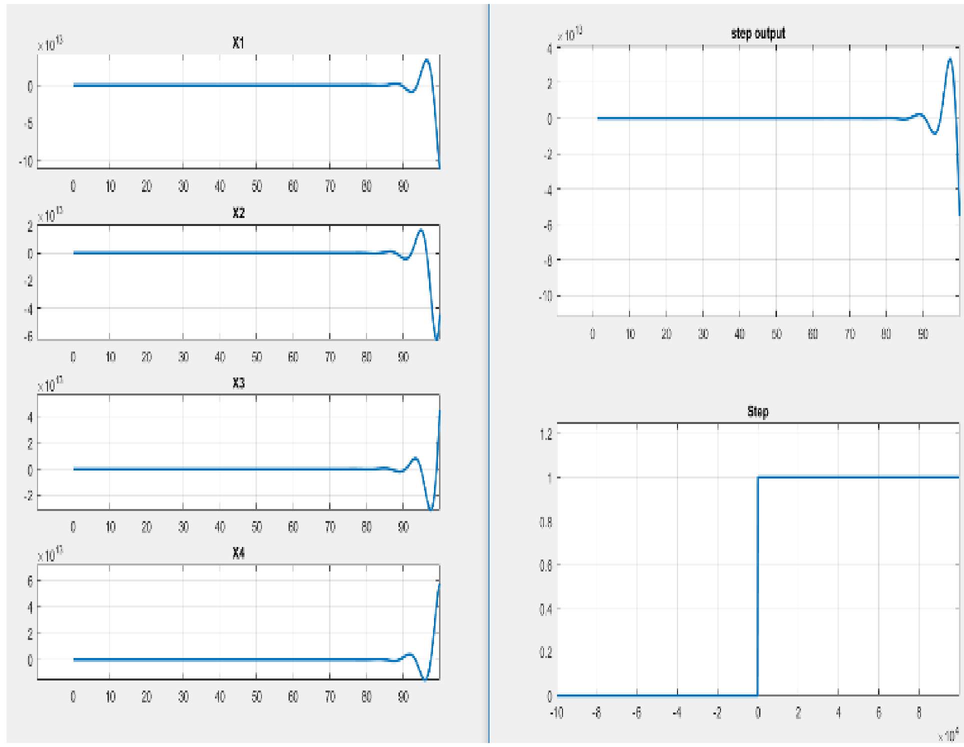
$$B = \begin{bmatrix} 0.7500 \\ 0 \\ 2.7500 \\ -5.8125 \end{bmatrix}$$

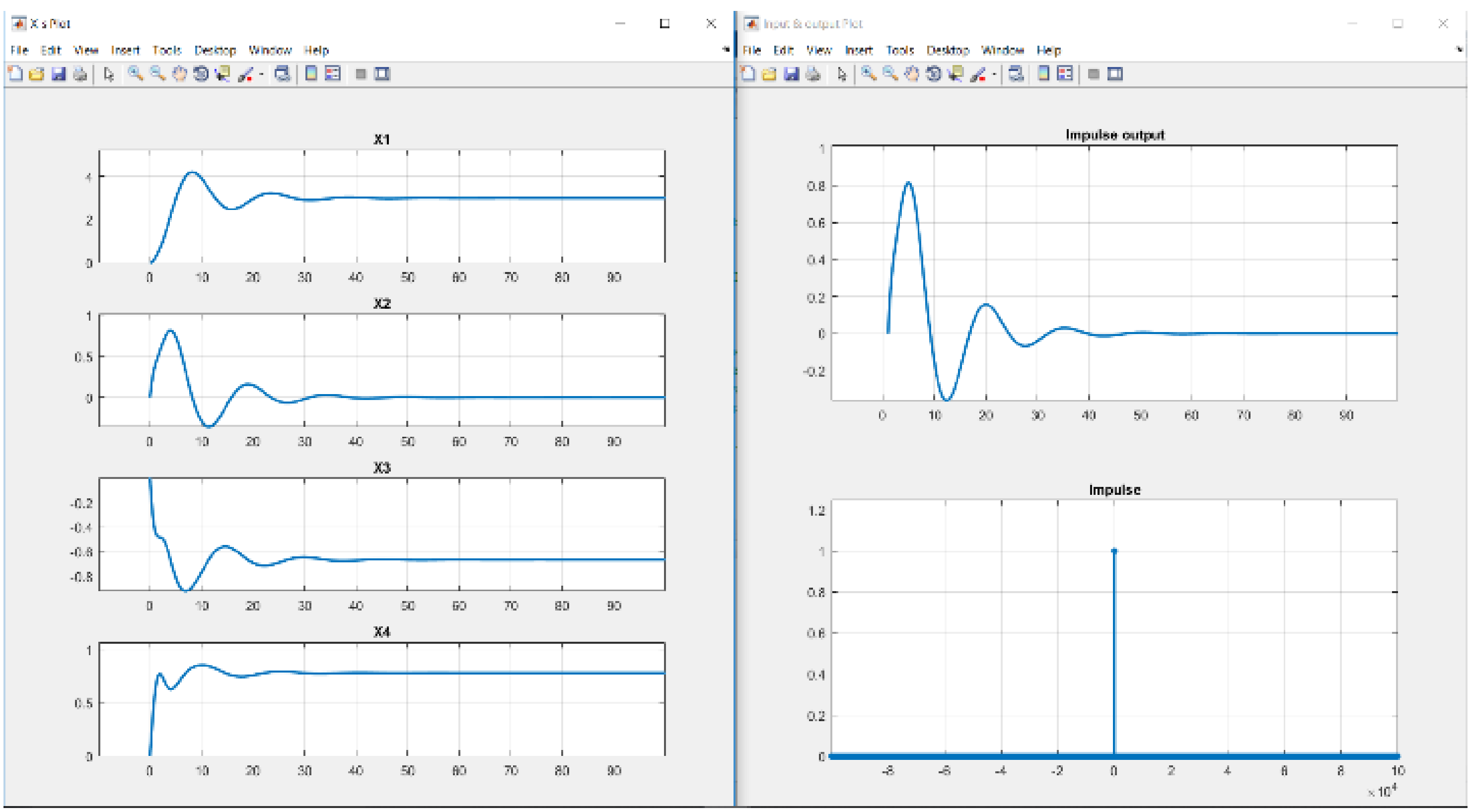
C

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

D

$$D = \begin{bmatrix} 0 \end{bmatrix}$$





6) $3y^{(4)} + 5\dddot{y} + 7\ddot{y} + 2\dot{y} + y = 2\ddot{u} + \dot{u} + 3u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-0.3333	-0.6667	-2.3333	-1.6667

B

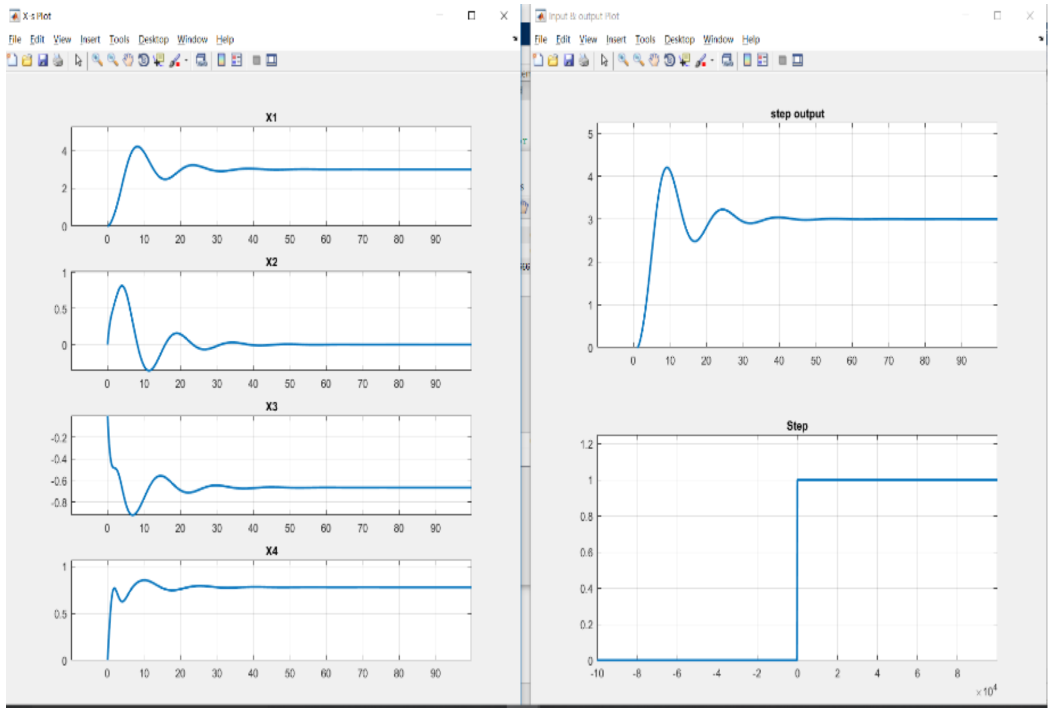
0
0.6667
-0.7778
0.7407

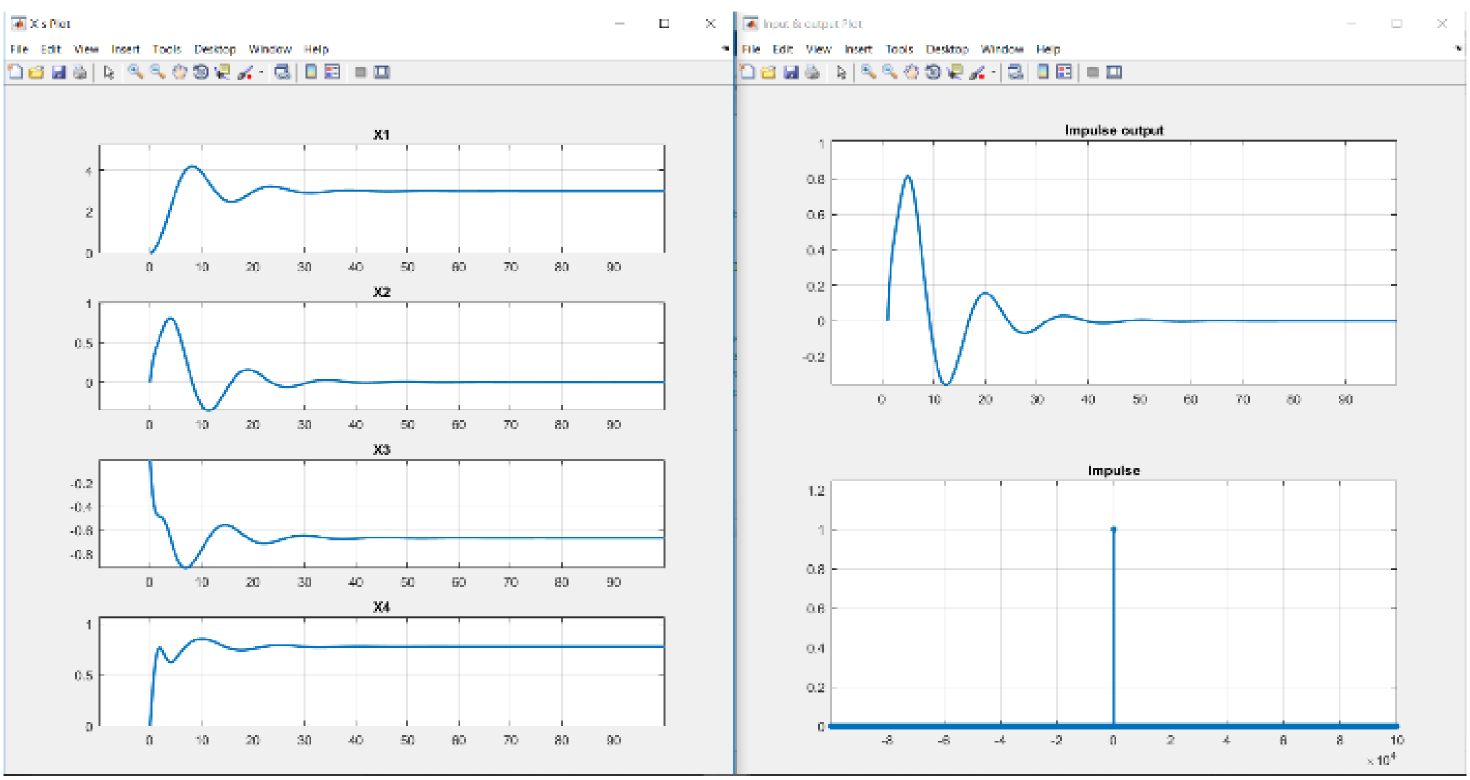
C

1	0	0	0

D

0





7) $6y^{(4)} + 2\dddot{y} + \ddot{y} + 8\dot{y} + 3y = 4\dot{u} + 2u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-0.5000	-1.3333	-0.1667	-0.3333

B

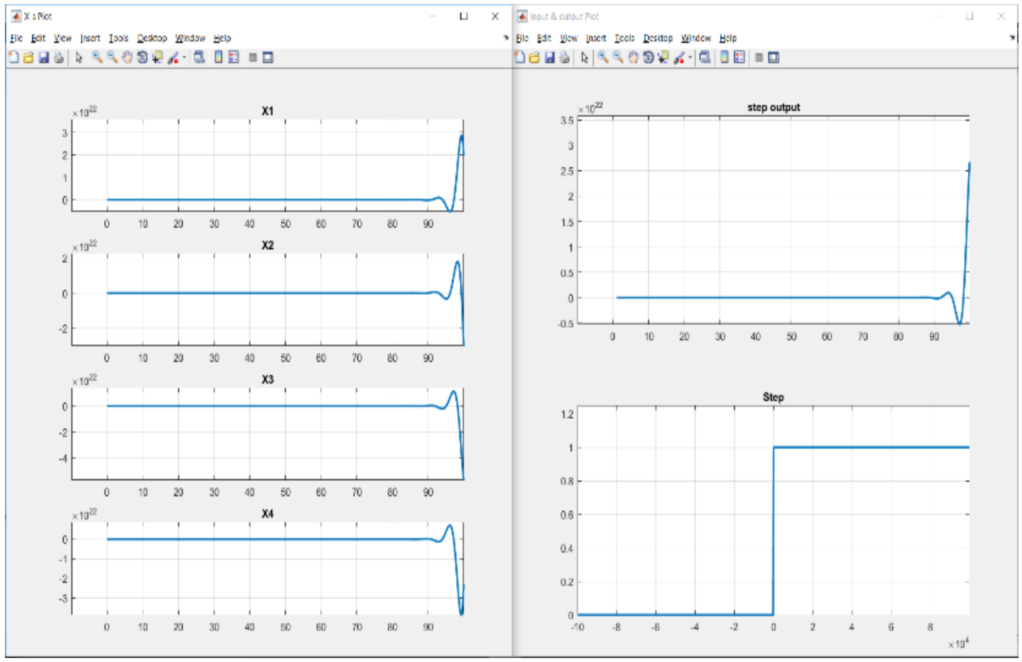
0
0
0.6667
0.1111

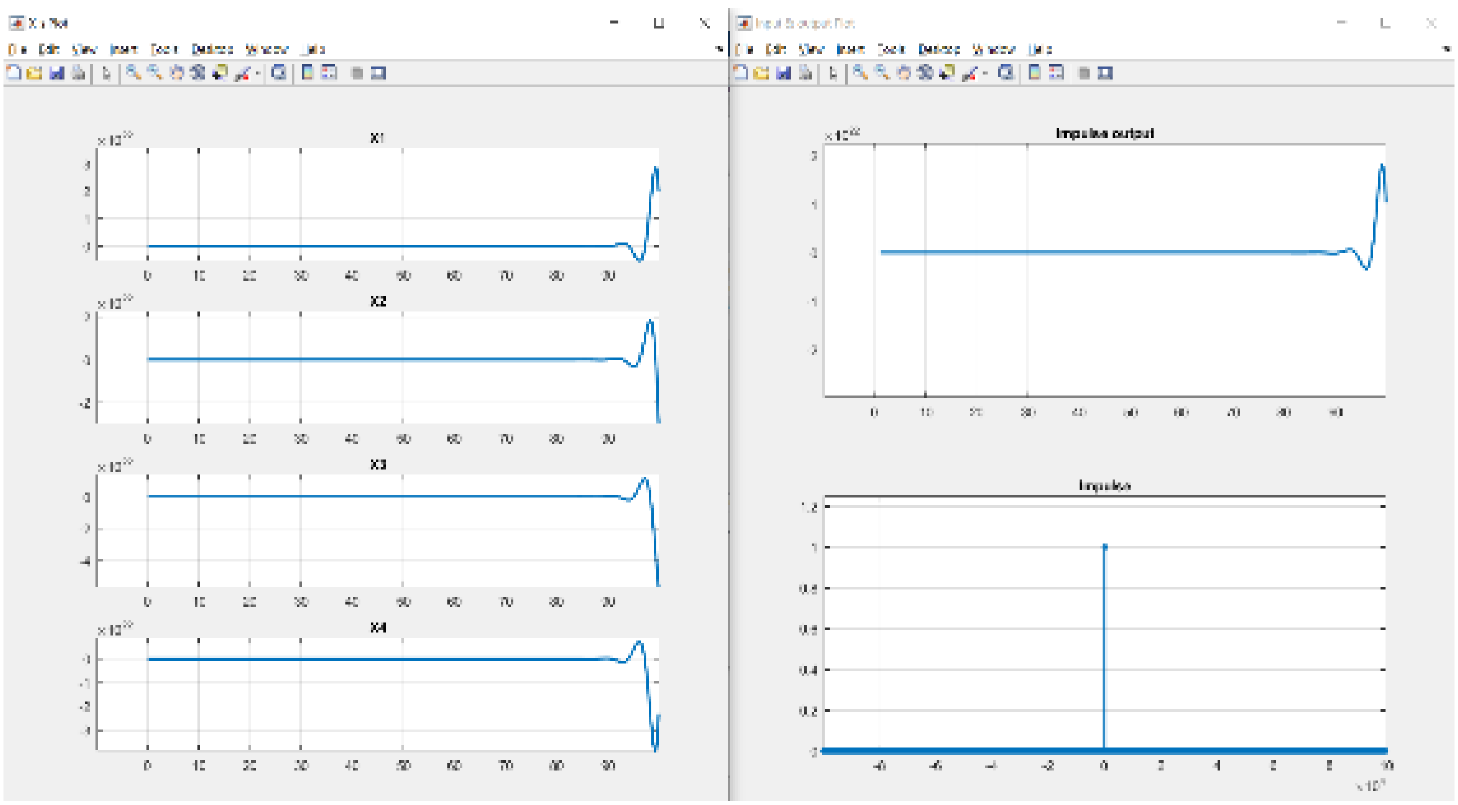
C

1	0	0	0

D

0





8) $2y^{(4)} + 4\dddot{y} + 4\ddot{y} + 3\dot{y} + 3y = 7u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-1.5000	-1.5000	-2	-2

B

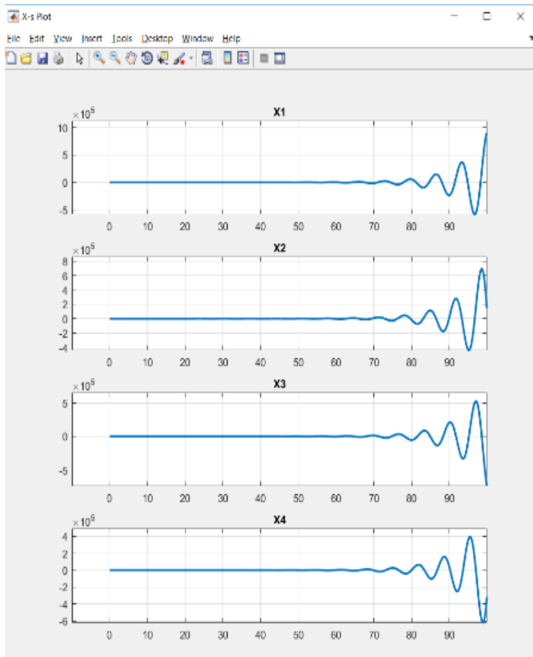
0
0
0
3.5000

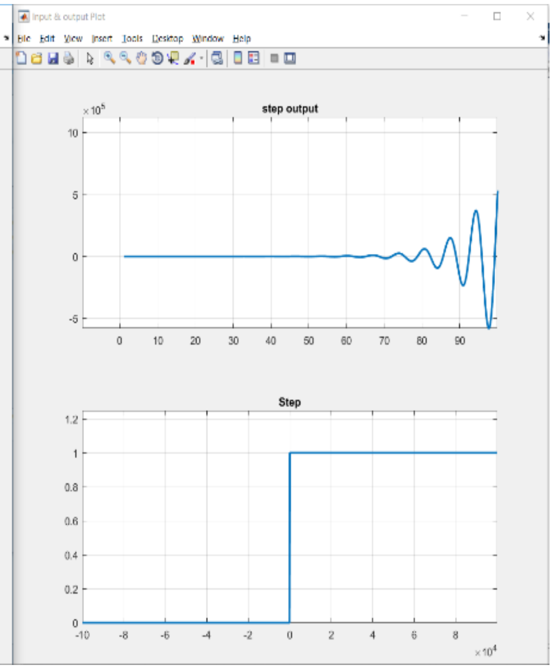
C

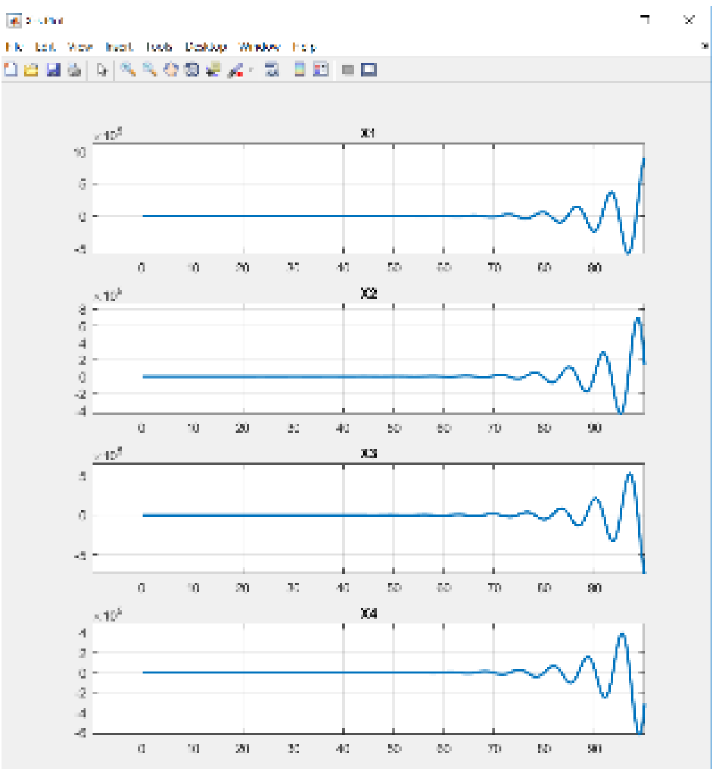
1	0	0	0

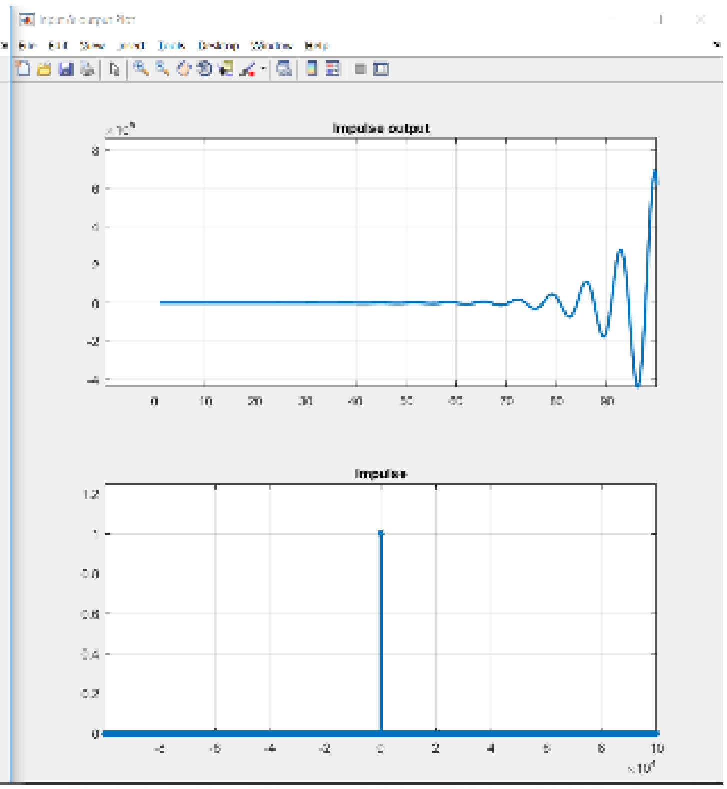
D

0









9) $8y^{(4)} + 9\dddot{y} + \ddot{y} + 4\dot{y} + 9y = 2u^{(4)} + 6\dddot{u} + 8\ddot{u} + 10\dot{u} + 5u$

A

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1.1250 & -0.5000 & -0.1250 & -1.1250 \end{bmatrix}$$

B

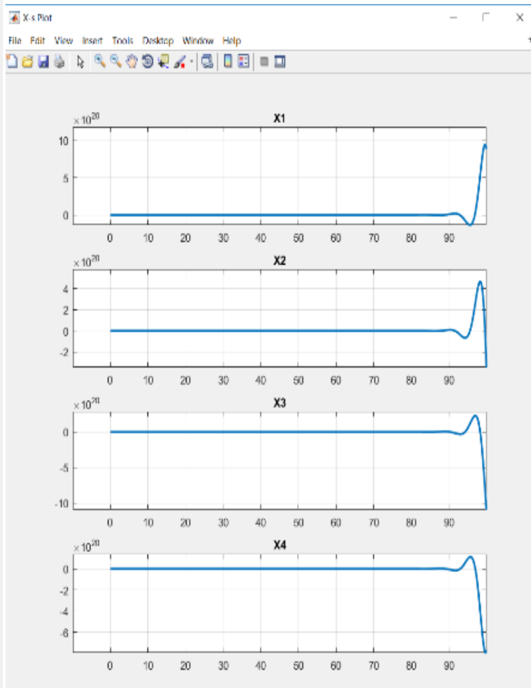
$$\begin{bmatrix} 0.4688 \\ 0.4414 \\ 0.9600 \\ -1.0258 \end{bmatrix}$$

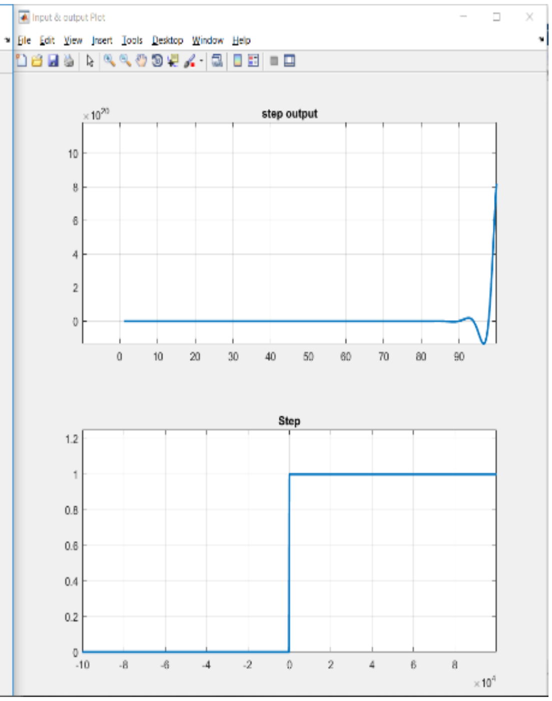
C

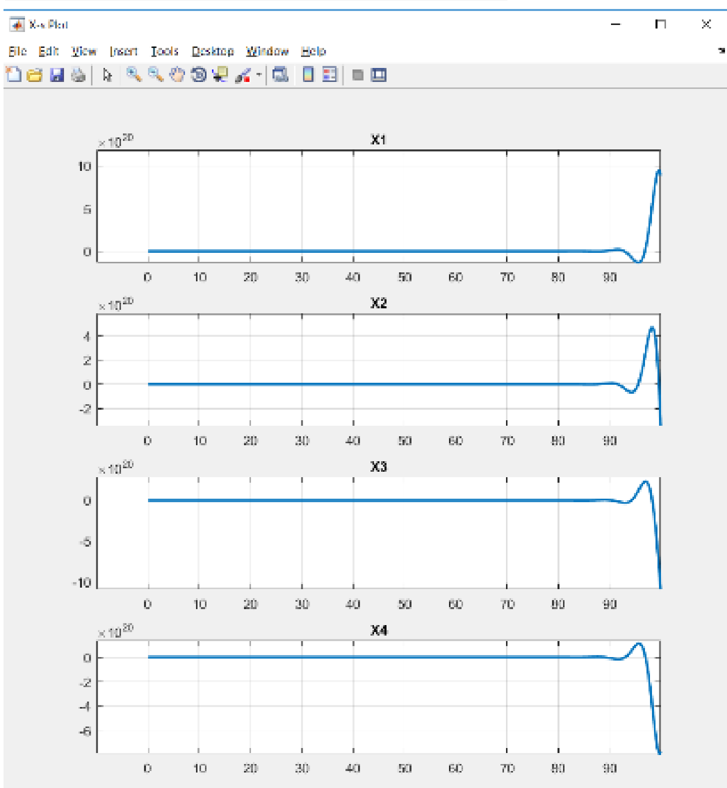
$$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

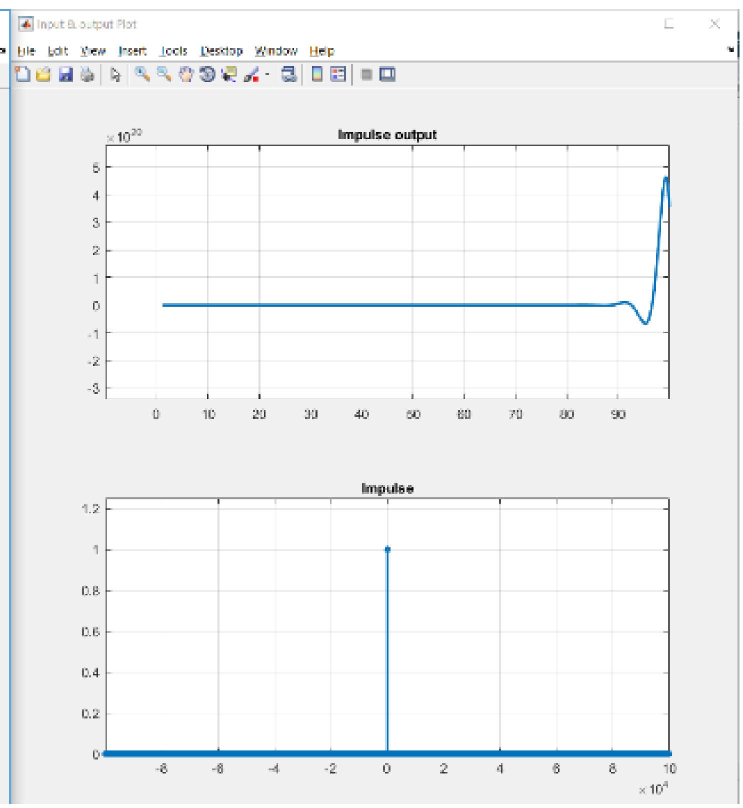
D

$$\begin{bmatrix} 0.2500 \end{bmatrix}$$









10) $10y^{(4)} + 3\dddot{y} + 10\ddot{y} + 2\dot{y} + 4y = 9\dot{u} + 6u$

A

0	1	0	0
0	0	1	0
0	0	0	1
-0.4000	-0.2000	-1	-0.3000

B

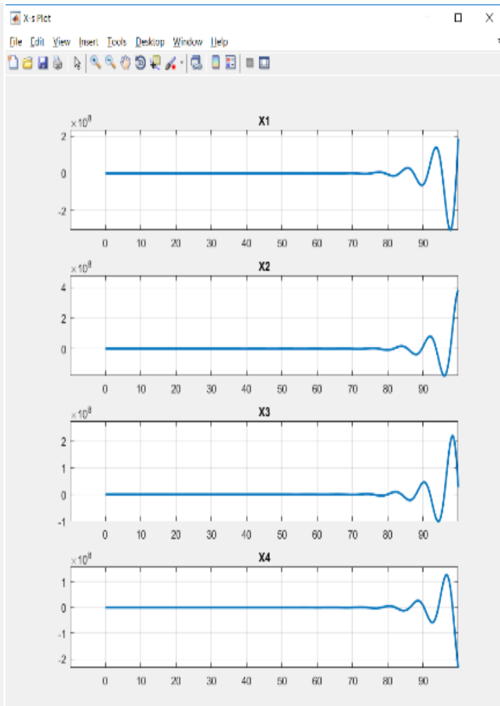
0
0
0.9000
0.3300

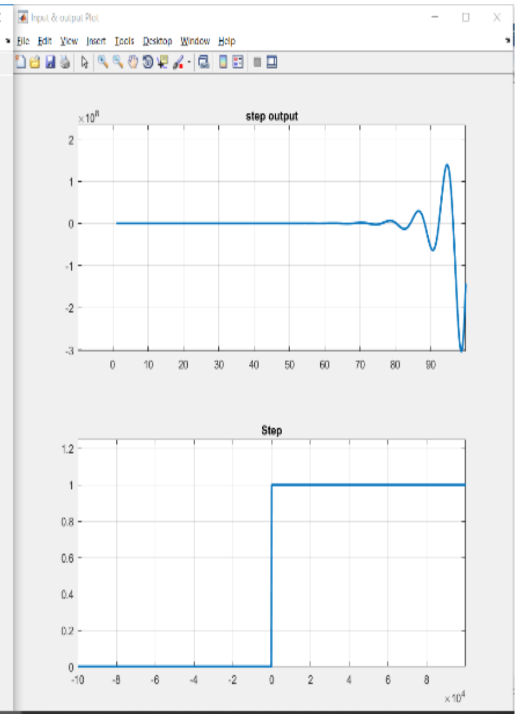
C

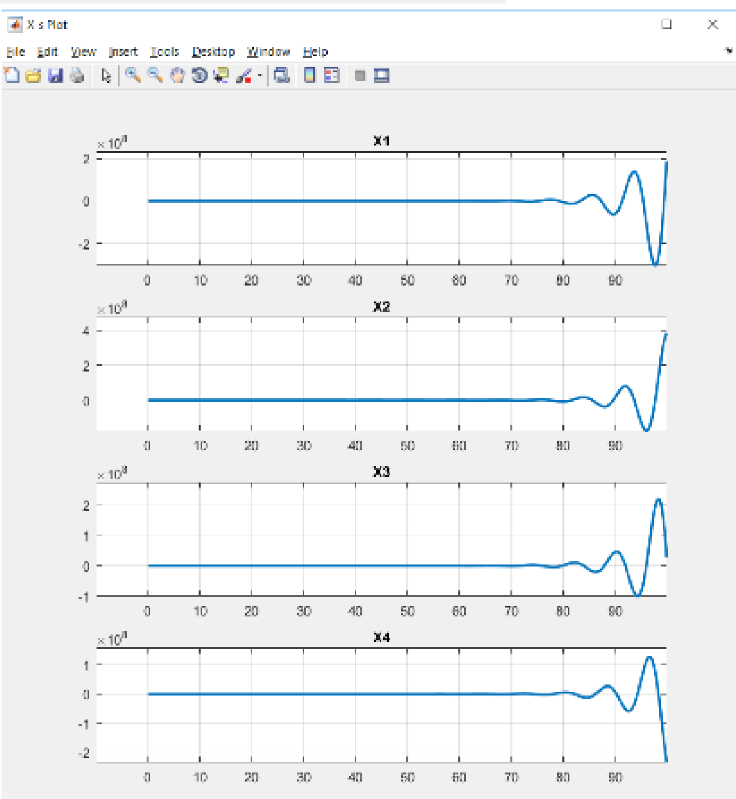
1	0	0	0

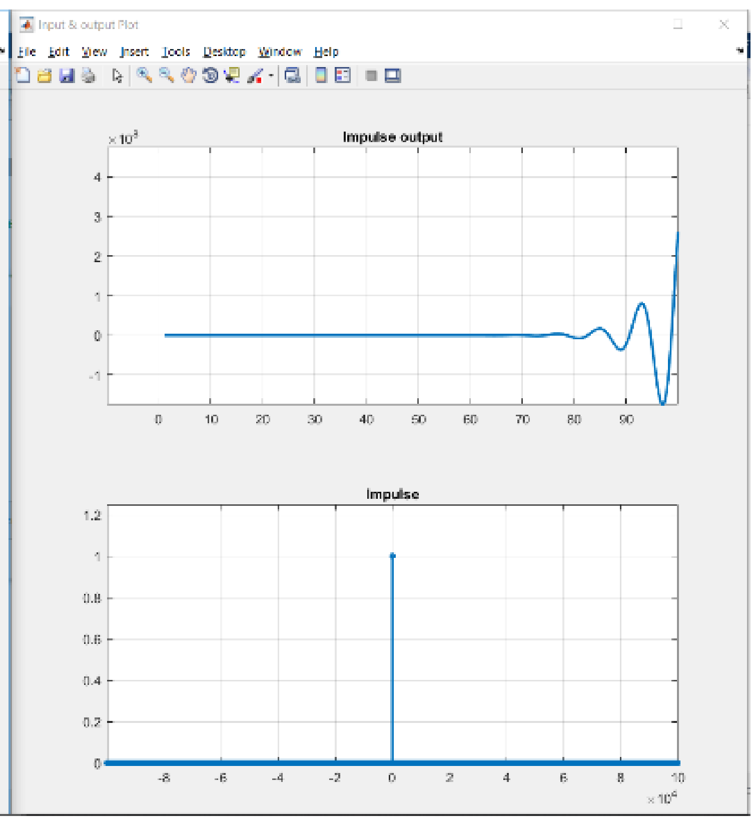
D

0









05 *References*

1) MATLAB Documentation
2) Lecture Notes.

06 *Table of Contribution*

All work was done through online meetings on zoom:

Name	Role
Loay Abdalla Youssef	• MATLAB Coding • Numerical Approximation
Amr Ehab Abdelaziz	• MATLAB Coding • System Modelling
Amr Ahmed Mohamed Fathy	• MATLAB Coding • Examples & Testing
Loay Anwar Abdelrazek Hegazy	• Hand Analysis • Numerical Approximation
Loay Khaled Mohamed Abdo	• Hand Analysis • Examples & Testing

07 *Appendix*

MATLAB Code

```
function varargout = Test(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Test_OpeningFcn, ...
                   'gui_OutputFcn',  @Test_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function Test_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;

opengl('save', 'software');
setGlobalx(1);
set(gcf,'name','System Response');
movegui('center');
guidata(hObject, handles);
clc
function varargout = Test_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
function nValue_Callback(hObject, eventdata, handles)
function nValue_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function mValue_Callback(hObject, eventdata, handles)
function mValue_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

%setting a global variable for the enter button conditions
function setGlobalx(val)
global x
x = val;

function r = getGlobalx
global x
r = x;


function enterTag_Callback(hObject, eventdata, handles)
r = getGlobalx;
if r == 1

%getting the n & m from the editable text field

    handles.n = str2num(get(handles.nValue, 'string'));
    handles.m = str2num(get(handles.mValue, 'string'));

%error message for n > m

    if handles.m > handles.n
        msgbox('Error, m should be less than or equal n');

    else
        setGlobalx(2);
        r=2;

%hiding the input fields

        set(handles.nValue, 'Visible', 'Off');
        set(handles.mValue, 'Visible', 'Off');
        set(handles.text2, 'Visible', 'Off');
        set(handles.text3, 'Visible', 'Off');
```

```
%creating a text and editable field for the user to enter
the a-s and b-s values

        for i= handles.n+1 : -1 : 1
            handles.textA{i} =
uicontrol('Style','text','String',sprintf('%s%d','a',i-
1),'Position',[-70+(120*(handles.n-i+2))-50 191 60 20]);
            handles.A{i} =
uicontrol('style','edit','Position',[-70+(120*(handles.n-
i+2)) 191 60 20]);
        end
        for i= handles.m+1 : -1 : 1
            handles.textB{i} =
uicontrol('Style','text','String',sprintf('%s%d','b',i-
1),'Position',[-70+(120*(handles.m-i+2))-50 136 60 20]);
            handles.B{i} =
uicontrol('style','edit','Position',[-70+(120*(handles.m-
i+2)) 136 60 20]);
        end
    end
elseif r == 2
   setGlobalx(3);
   r = 3;

%initializing the a-s and b-s array to zero

   handles.a=zeros(1,10);
   handles.b=zeros(1,10);

%creating a variable to carry the highest y coefficient

   handles.nthcoeff = str2double(get(handles.A{handles.n+1}
, 'String'));

%store the a-s and b-s coefficient and delete the created
fields

    for i= handles.n+1 : -1 : 1
        handles.a(i) = str2double(get(handles.A{i} ,
'String'));
        delete(handles.A{i});
        delete(handles.textA{i});
    end
    for i= handles.m+1 : -1 : 1
```

```
        handles.b(i) = str2double( get(handles.B{i} ,
'String'));
        delete(handles.B{i});
        delete(handles.textB{i});
    end

%show selection field of input

    set(handles.inputType, 'Visible', 'on');

    elseif r == 3
    setGlobalx(4);
    r=4;

%initializing the beta array to zeros

   handles.beta = zeros(1,4);

%calculating the beta and store it in the array

   if handles.n > 0
   handles.beta(1) = -
handles.b(handles.n+1)/handles.a(handles.n+1);
   end

   if handles.n > 1
   handles.beta(2) =(1/handles.a(handles.n+1))*(-
handles.b(handles.n)+(handles.a(handles.n)/handles.a(handle
s.n+1)*handles.b(handles.n+1)));
   end

   if handles.n > 2
   handles.beta(3) = (1/handles.a(handles.n+1))*(-
handles.b(handles.n-1)-
(handles.a(handles.n)/handles.a(handles.n+1))*(-
handles.b(handles.n)+(handles.a(handles.n)/handles.a(handle
s.n+1)*handles.b(handles.n+1)))+(handles.a(handles.n-
1)/handles.a(handles.n+1))*handles.b(handles.n+1));
   end

   if handles.n > 3
   handles.beta(4)=(1/handles.a(handles.n+1))*(-
handles.b(handles.n-2)-
(handles.a(handles.n)/handles.a(handles.n+1))*(-
```

```
handles.b(handles.n-1)-
(handles.a(handles.n)/handles.a(handles.n+1))*(-
handles.b(handles.n)+(handles.a(handles.n)/handles.a(handle
s.n+1))*handles.b(handles.n+1))+(handles.a(handles.n-
1)/handles.a(handles.n+1))*handles.b(handles.n+1)-
(handles.a(handles.n-1)/handles.a(handles.n+1))*(-
handles.b(handles.n)+(handles.a(handles.n)/handles.a(handle
s.n+1))*handles.b(handles.n+1))+(handles.a(handles.n-
2)/handles.a(handles.n+1))*handles.b(handles.n+1)));
   end

%initializing the Matrices and calculating it

   A = zeros(handles.n);
    for r = 1 : handles.n
        for c = 1 : handles.n
         if r == handles.n
            A(handles.n,c) = -
handles.a(c)/handles.nthcoeff;
         end
         if r == c-1
             A(r,c)=1;
         end
        end
    end


   B = zeros(handles.n,1);

   B(handles.n,1) =
(handles.b(1)+handles.a(handles.n)*handles.beta(handles.n))
/handles.nthcoeff;

   for r= 1:handles.n-1
       B(r,1)=-handles.beta(r+1);
       B(handles.n,1) = B(handles.n,1) +
(handles.a(r)*handles.beta(r))/handles.nthcoeff;
   end


    C = zeros(1,handles.n);
    C(1,1) = 1;

    D(1,1) = -handles.beta(1);
```

```
%displaying the matrices

    ff=figure ;
    UA=uicontrol('Style','text','String','A','Position',[30
320 30 30],'FontSize',20,'FontWeight','bold');
    TA=uitable(ff,'Data',A,'Position',[100 320 50*handles.n
20*handles.n+20]);

set(TA,'ColumnName','','RowName','','ColumnWidth',{50});

    UB=uicontrol('Style','text','String','B','Position',[30
220 30 30],'FontSize',20,'FontWeight','bold');
    TB=uitable(ff,'Data',B,'Position',[100 200 50
20*handles.n+20]);

set(TB,'ColumnName','','RowName','','ColumnWidth',{50});

    UC=uicontrol('Style','text','String','C','Position',[30
120 30 30],'FontSize',20,'FontWeight','bold');
    TC=uitable(ff,'Data',C,'Position',[100 120 50*handles.n
20+20]);

set(TC,'ColumnName','','RowName','','ColumnWidth',{50});

    UD=uicontrol('Style','text','String','D','Position',[30
20 30 30],'FontSize',20,'FontWeight','bold');
    TD=uitable(ff,'Data',D,'Position',[100 20 50 20+20]);

set(TD,'ColumnName','','RowName','','ColumnWidth',{50});

    set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0
0.25 0.6]);
    movegui(ff,'center')
    set(gcf,'name','Matrices','numbertitle','off');

%setting the number of samples and the step between every
sample

    N=100000;
    T=0.001;

%initializing the x-s array
```

```
    x1 = zeros(1,N);
    x2 = zeros(1,N);
    x3 = zeros(1,N);
    x4 = zeros(1,N);

%initializing the y array

    handles.y = zeros(1,N);

    x = {x1,x2,x3,x4};

%initializing the first element of x

    for i=1:4
        x{i}(1) = 0;
    end

%initializing the time array

    t = (0:N-1)*T;

%initializing the y and I first element of the array

    handles.y(1) = 0;
    handles.I(1)= 0;

%update loop

    for k= 2:N
        for i= 1:handles.n
            if i == handles.n
            sum = B(handles.n,1);
                for j = 1 : handles.n-1
                sum = sum + A(handles.n,j)*x{j}(k);
                end
            x{handles.n}(k) = (1/(1-
T*A(handles.n,handles.n)))*(T*sum+x{handles.n}(k-1));
            else
                x{i}(k) = T*(x{i+1}(k-1)+B(i,1))+x{i}(k-1);
            end
        end
        handles.y(k) = x{1}(k) + D(1,1);
        handles.I(k) = (handles.y(k)-handles.y(k-1))/T;
    end
```

```
    handles.imp = zeros(1,2*N);
    handles.imp(1,N) = 1;
    timp =(-N+1:N);
    handles.step = zeros(1,N);
    for i =N:2*N
    handles.step(1,i) = 1;
    end

    f1=figure ;
    set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0
0.5 1]);
    movegui(f1,'east')
    set(gcf,'name','Input & output
Plot','numbertitle','off');
    subplot(2,1,1);



    if handles.stepInput.Value == 1

%plots y

    plot (t+1,handles.y,'markersize',3,'linewidth',2);
    title('step output')
    axis([min(t)-max(t)*0.1 max(t) min(handles.y)
max(handles.y)*1.25])
    grid on;

%plots the step function

    subplot(2,1,2);
    tstep =(-N:N-1);
    plot (tstep,handles.step,'markersize',3,'linewidth',2);
    title('Step')
    axis([min(tstep) max(tstep) 0 1.25])
    grid on;

    elseif handles.impulseInput.Value == 1

%plots y

    plot (t+1,handles.I,'markersize',3,'linewidth',2);
    title('Impulse output')
```

```
    axis([min(t)-max(t)*0.1 max(t) min(handles.I)
max(handles.I)*1.25])
    grid on;
%plots the impulse function
    subplot(2,1,2);
    stem (timp,handles.imp,'markersize',3,'linewidth',2);
    title('Impulse')
    axis([min(timp) max(timp) min(handles.imp)
max(handles.imp)*1.25])
    grid on;
    end

    f2=figure;
    set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0
0.5 1]);
    movegui(f2,'west');
    set(gcf,'name','X-s Plot','numbertitle','off')

%plots x-s

    for i=1:handles.n
        subplot(handles.n,1,i)
        plot(t,x{i},'linewidth',2)
        XText = strcat('X',num2str(i));
        title(XText)
        axis([min(t)-max(t)*0.1 max(t) min(x{i})
max(x{i})*1.25])
        grid on;
    end

%resetting the app

    set(handles.inputType, 'Visible', 'off');
    setGlobalx(1);
    r=1;
    set(handles.nValue, 'Visible', 'on','String','');
    set(handles.mValue, 'Visible', 'on','String','');
    set(handles.text2, 'Visible', 'on');
    set(handles.text3, 'Visible', 'on');
    grid ON

end
%updating the handles function
guidata(hObject, handles);
```