电子科技大学格拉斯哥学院

Glasgow College UESTC

标准实验报告

Lab Report

（实验）课程名称：信号与系统

(LAB) Course Name：Signals and Systems

电子科技大学教务处制表

Lab Report 2

Student Name：Changgang Zheng 郑长刚
UESTC ID：2016200302027
UoG ID：2289258Z
Instructor：Huijuan Wu 吴慧娟
Location：UESTC, ChengDu, SiChuan Province, China
Date： March 29, 2018

1. **Laboratory name：** Signals and Systems
2. **Project name：** Represent signals using MATLAB
3. **Lab hours：** 2
4. **Theoretical background：**

   1. The basic concepts of signals and systems arise in a variety of contexts, from engineering design to financial analysis. In this lab1, you will learn how to represent, manipulate, and analyze basic signals and systems in MATLAB.

   2. Some basic MATLAB commands for representing signals include: zeros, ones, cos, sin, exp, real, imag, abs, angle, linspace, plot, stem, subplot, xlabel, ylabel, title.

   3. Some useful commands in Symbolic Math Toolbox are as: zeros, ones, cos, sin, exp, real, imag, abs, anglelinspace, plot, stem, subplot, xlabel, ylabel, title. Symbolic Math Toolbox: sym, subs, ezplot.
5. **Objective：**

   1. Familiar with some basic Matlab commands to represent and plot continus-time and discrete-time signals.

   2. Analyze the properties of system.
6. **Description：**

The following exercises are from the book," John R.Buck, Michael M. Daniel, Andrew C. Singer. Computer Exploration in Signals and Systems —— Using MATLAB."

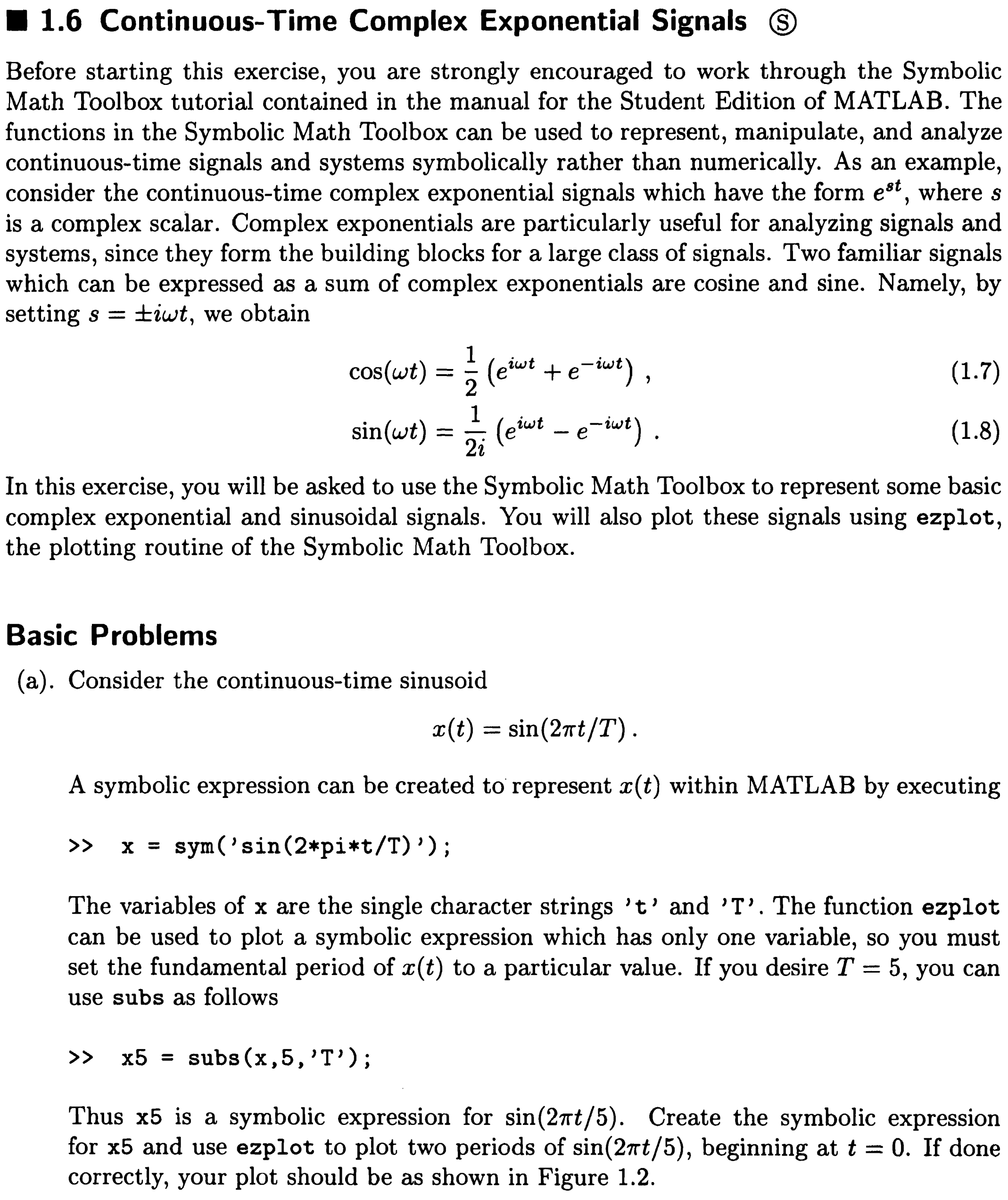
   1. Plot signals using Symbolic Math toolbox. 1.6(a)

   2. Analyze the properties of systems. 1.4(a)(b)(c)(f)

7. **Required instruments：**

   Computer, MATLAB
8. **Procedures, Analysis of Lab data & result and Conclusion：**

MATLAB codes and results for the exercises:



■ 1.6 Continuous-Time Complex Exponential Signals Ⓢ

Before starting this exercise, you are strongly encouraged to work through the Symbolic Math Toolbox tutorial contained in the manual for the Student Edition of MATLAB. The functions in the Symbolic Math Toolbox can be used to represent, manipulate, and analyze continuous-time signals and systems symbolically rather than numerically. As an example, consider the continuous-time complex exponential signals which have the form e^{st}, where s is a complex scalar. Complex exponentials are particularly useful for analyzing signals and systems, since they form the building blocks for a large class of signals. Two familiar signals which can be expressed as a sum of complex exponentials are cosine and sine. Namely, by setting $s = \pm i\omega t$, we obtain

$$\cos(\omega t) = \frac{1}{2}\left(e^{i\omega t} + e^{-i\omega t}\right) , \tag{1.7}$$

$$\sin(\omega t) = \frac{1}{2i}\left(e^{i\omega t} - e^{-i\omega t}\right) . \tag{1.8}$$

In this exercise, you will be asked to use the Symbolic Math Toolbox to represent some basic complex exponential and sinusoidal signals. You will also plot these signals using **`ezplot`**, the plotting routine of the Symbolic Math Toolbox.

Basic Problems

(a). Consider the continuous-time sinusoid

$$x(t) = \sin(2\pi t/T) .$$

A symbolic expression can be created to represent $x(t)$ within MATLAB by executing

```
>>  x = sym('sin(2*pi*t/T)');
```

The variables of `x` are the single character strings `'t'` and `'T'`. The function `ezplot` can be used to plot a symbolic expression which has only one variable, so you must set the fundamental period of $x(t)$ to a particular value. If you desire $T = 5$, you can use `subs` as follows

```
>>  x5 = subs(x,5,'T');
```

Thus `x5` is a symbolic expression for $\sin(2\pi t/5)$. Create the symbolic expression for `x5` and use `ezplot` to plot two periods of $\sin(2\pi t/5)$, beginning at $t = 0$. If done correctly, your plot should be as shown in Figure 1.2.

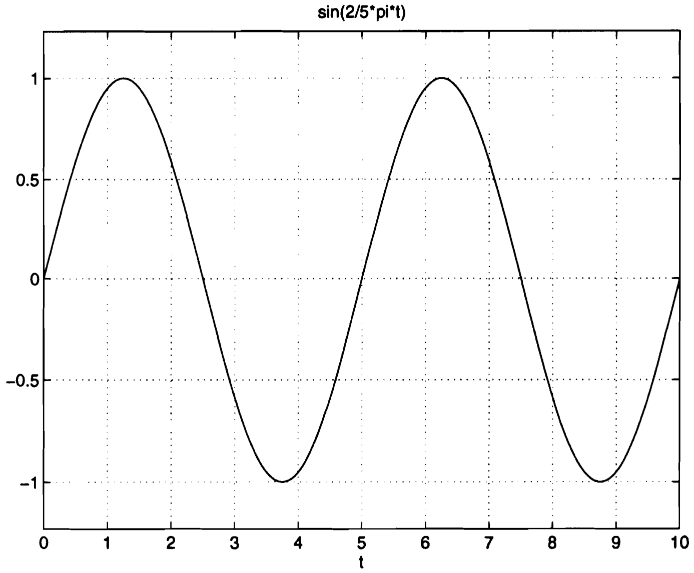


Figure 1.2. Two periods of the signal $\sin(2\pi t/5)$.

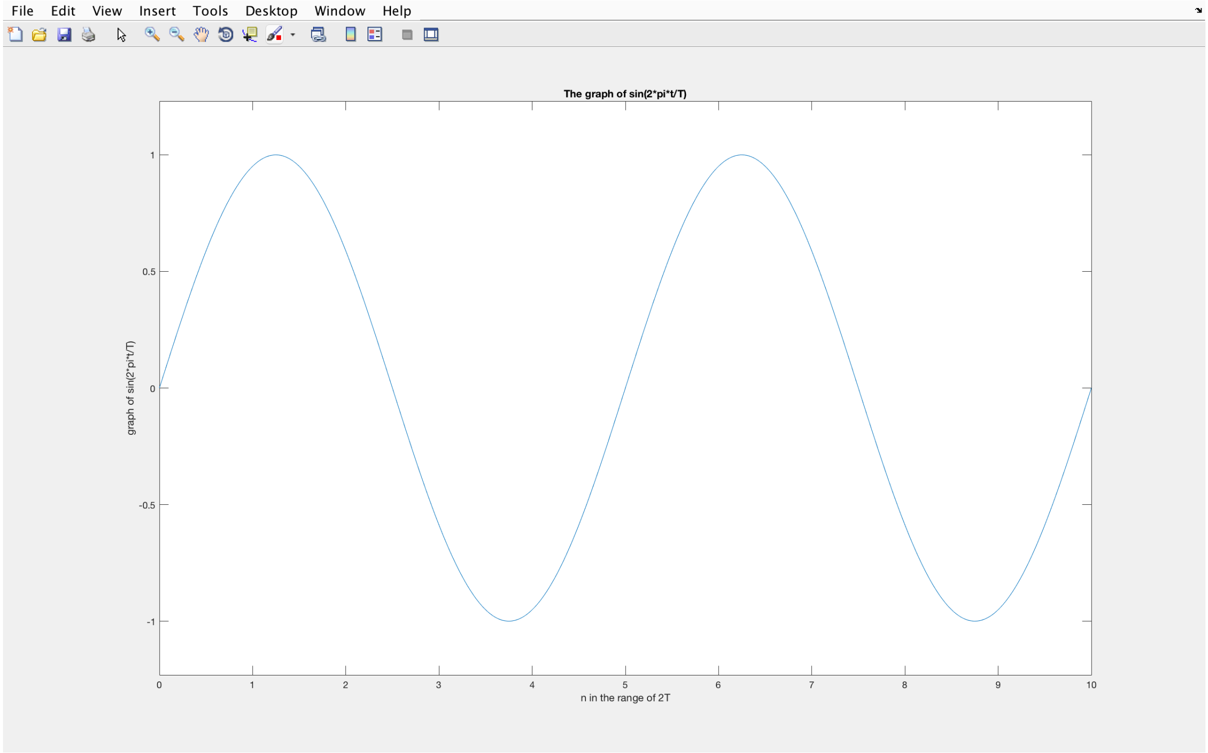
The code:

```
%Name: Matlab/CUDA: Signals and Systems Lab 2nd
%Auther: Changgang Zheng
%Student Number UESTC:2016200302027
%Student Number UoG:2289258z
%Institution: Glasgow College UESCT

function  problem_1st

    x=sym('sin(2*pi*t/T)');                    % creat the sybolic expression 
of the signal

    x5=subs(x,'T','5');                        %Use the subs function to 
replacee T by 5

    ezplot(x5,[0,10]);                         % plot the discrete signal
    title('The graph of sin(2*pi*t/T)');       % name the title of the figure 
as 'The graph of sin(2*pi*t/T)'
    xlabel('n in the range of 2T');            % name the label of x-axis as 
'n'
    ylabel('graph of sin(2*pi*t/T)')           % name the label of y-axis as 
'The graph of sin(2*pi*t/T)'
```

The result:

Answer for problem (a):

The result of the plot:

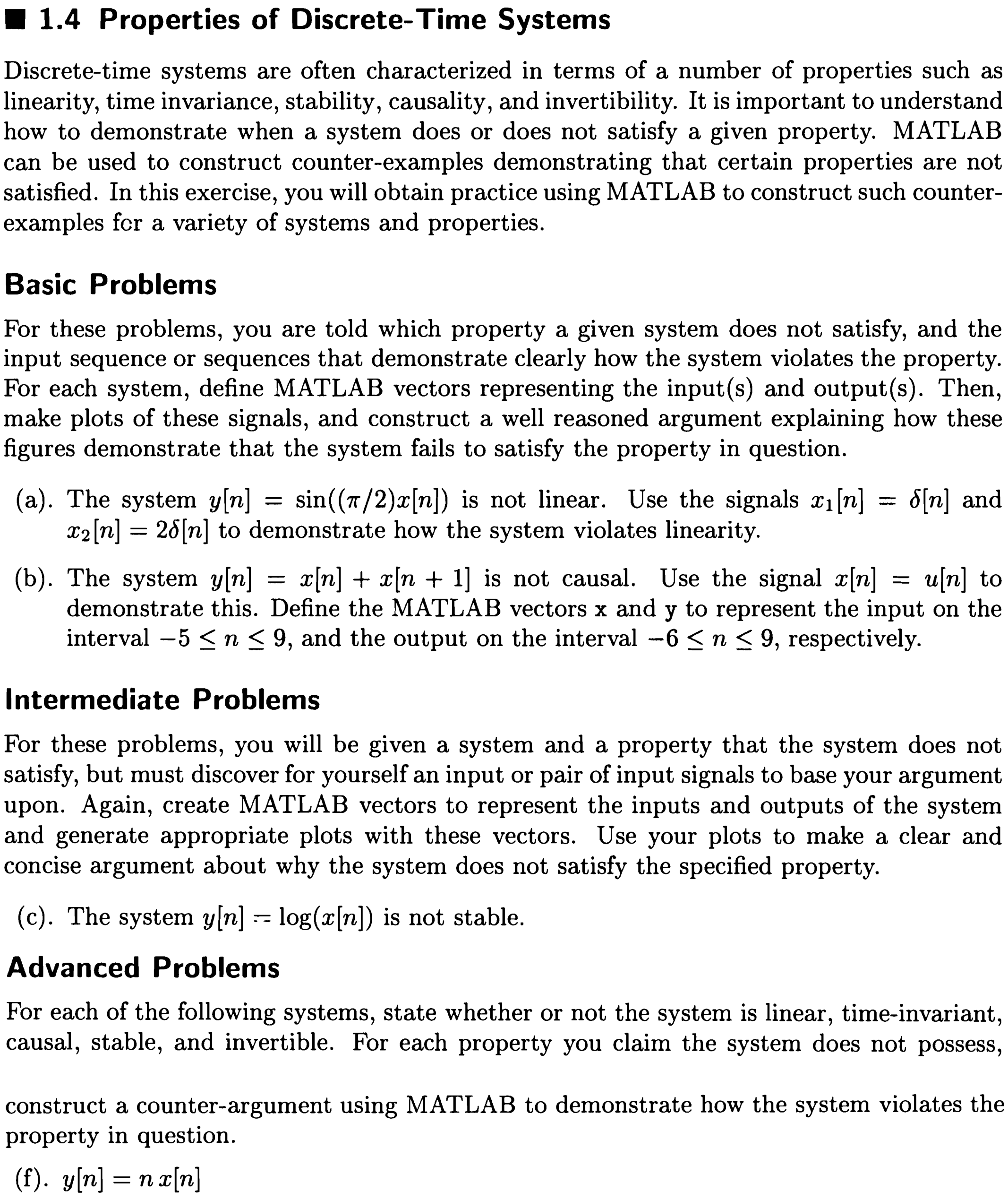


(Two period of the signal $\sin(2\pi t/5)$)

Summary and comments :

I use the Matlab to solve some problems such as to create and plot discrete time signal.

I gained more impression of the concept of the Matlab and how it could be used in the signals and systems, especially on the part of the discrete signals' display and analysis. For instance, how I could input the symbolic expression of the signals' function in the program by using 'x = sym(' sin (2*pi*t/T)');' and be more familiar with using the function 'sub' to replace some part in the signal. Besides, I know how to use the 'stem' to plot the signal and know how to make it to be readable like adding the title and add the notification of the x, y axis. Meanwhile, I am more familiar with the use of the Matlab and how I could organize my program (and using comment to make it readable).



■ 1.4 Properties of Discrete-Time Systems

Discrete-time systems are often characterized in terms of a number of properties such as linearity, time invariance, stability, causality, and invertibility. It is important to understand how to demonstrate when a system does or does not satisfy a given property. MATLAB can be used to construct counter-examples demonstrating that certain properties are not satisfied. In this exercise, you will obtain practice using MATLAB to construct such counter-examples fcr a variety of systems and properties.

Basic Problems

For these problems, you are told which property a given system does not satisfy, and the input sequence or sequences that demonstrate clearly how the system violates the property. For each system, define MATLAB vectors representing the input(s) and output(s). Then, make plots of these signals, and construct a well reasoned argument explaining how these figures demonstrate that the system fails to satisfy the property in question.

(a). The system $y[n] = \sin((\pi/2)x[n])$ is not linear. Use the signals $x_1[n] = \delta[n]$ and $x_2[n] = 2\delta[n]$ to demonstrate how the system violates linearity.

(b). The system $y[n] = x[n] + x[n+1]$ is not causal. Use the signal $x[n] = u[n]$ to demonstrate this. Define the MATLAB vectors **x** and **y** to represent the input on the interval $-5 \leq n \leq 9$, and the output on the interval $-6 \leq n \leq 9$, respectively.

Intermediate Problems

For these problems, you will be given a system and a property that the system does not satisfy, but must discover for yourself an input or pair of input signals to base your argument upon. Again, create MATLAB vectors to represent the inputs and outputs of the system and generate appropriate plots with these vectors. Use your plots to make a clear and concise argument about why the system does not satisfy the specified property.

(c). The system $y[n] = \log(x[n])$ is not stable.

Advanced Problems

For each of the following systems, state whether or not the system is linear, time-invariant, causal, stable, and invertible. For each property you claim the system does not possess,

construct a counter-argument using MATLAB to demonstrate how the system violates the property in question.

(f). $y[n] = n\,x[n]$

The code:

```
%Name: Matlab/CUDA: Signals and Systems Lab 2nd
%Auther: Changgang Zheng
%Student Number UESTC:2016200302027
%Student Number UoG:2289258z
%Institution: Glasgow College UESCT

function  problem_2nd
    a=5; % set constant value
    b=7; % set constant value

    n_x=[-5:9];
    n_y=[-6:9];

    n=[-20:20];
    x_a1=zeros(1,41); % creat a all zero vector
    x_a1(1,21)=1;
    x_a2=zeros(1,41); % creat a all zero vector
    x_a2(1,21)=2;

    u=zeros(1,41); % creat a all zero vector

    for i=21:41
        u(i)=1;
    end

    x_b1=u;
    x_b2=zeros(1,41);

    y_input= sin((pi/2)*(a*x_a1+b*x_a2));
    y_output = a*sin((pi/2)*x_a1)+b*sin((pi/2)*x_a2);

    %% The plot of 1.4 a
    figure;             %create the first window for ploting
    subplot(2,1,1);     % set the place to display the plotting result
    stem(n,y_input);    %plot the figure
    title('Graph for the output for y[n] when
x[n]=a*x1[n]+b*x2[n]');   %name the title of the figure as 'Graph for the
output for y[n] when x[n]=a*x1[n]+b*x2[n]'
    xlabel('n');        %name the label of x-axis as 'n'
    ylabel('y[n] when x[n]=a*x1[n]+b*x2[n]');    %name the label of y-axis
as 'y[n]'

    subplot(2,1,2);     % set the place to display the plotting result
    stem(n,y_output);   %plot the figure
    title('Graph for the output for y[n]=a*y1[n]+b*y2[n]'); %name the title
of the figure as 'Graph for the output for y[n]=a*y1[n]+b*y2[n]'
    xlabel('n');        %name the label of x-axis as 'n'
    ylabel('y[n]=a*y1[n]+b*y2[n]')  %name the label of y-axis as
'y[n]=a*y1[n]+b*y2[n]'
    %%

    for i=1:40
        x_b2(i)=x_b1(i+1);
    end

    Y=x_b2+x_b1;

    Y_display=zeros(1,16); % set constant value
    X_display=zeros(1,15); % set constant value

    for i=1:16
        Y_display(i)=Y(i+14);
    end

    for i=1:15
        X_display(i)=x_b1(i+15);
    end

    %% The plot of 1.4 b
    figure;         %create a window for ploting
    subplot(2,1,1); %set the place to display the plotting result
    stem(n_x,X_display);    %plot the figure
    title('Graph for the output for x[n]');   %name the title of the figure
as 'Graph for the output for x[n]'
    xlabel('n');    %name the label of x-axis as 'n'
    ylabel('x[n]')  %name the label of y-axis as 'x[n]'

    subplot(2,1,2); %set the place to display the plotting result
    stem(n_y,Y_display);    %plot the figure
    title('Graph for the output for y[n]');   %name the title of the figure
as 'Graph for the output for y[n]'
    xlabel('n');    %name the label of x-axis as 'n'
    ylabel('y[n]=a*y1[n]+b*y2[n]') %name the label of y-axis as
'y[n]=a*y1[n]+b*y2[n]'
    %%

    %% The 1.4 c
    % x_c=zeros(1,41);  % set constant value
    % y_c=log(x_c);
    % y_c               % Force to plot the y_c, which is all infinity
however x_c finit which means unstable
    x_c = 2*cos(pi*n/10).^2;
    y_c=log(x_c);

    figure;             %create a window for ploting
    subplot(2,1,1);
    stem(n,x_c);        %plot the figure
    title('x_c[n]; Graph for checking the system is unstable');   % name
the title of the figure as 'x[n]; Graph for checking the system is
unstable'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('x_c[n]')    % name the label of y-axis as 'x[n]'

    subplot(2,1,2);
    stem(n,y_c);        %plot the figure
    ylim([-60,10]);
    title('y_c[n]; Graph for checking the system is unstable');   % name
the title of the figure as 'x[n]; Graph for checking the system is
unstable'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('y_c[n]')    % name the label of y-axis as 'x[n]'

    %%
    x_f_linear_1=[-20:20];
    x_f_linear_2=[-10:30];
    x_F_linear=a*x_f_linear_1+b*x_f_linear_2;
    y_F_input_linear=n.*x_F_linear;
    y_F_output_linear=a*n.*x_f_linear_1+b*n.*x_f_linear_2;

    %% The plot of 1.4 f linear
    figure;             % create a window for ploting

    subplot(4,1,1);     % set the place to display the plotting result
    stem(n,x_f_linear_1);   %plot the figure
    title('Graph for the x_f linear 1[n]');   %name the title of the figure
as 'Graph for the x_f_linear_1'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('x_f linear 1[n]');   %name the label of y-axis as
'x_f_linear_1'
```

```
    subplot(4,1,2);        % set the place to display the plotting result
    stem(n,x_f_linear_2);  % plot the figure
    title('Graph for the x_f linear 2[n]');   %name the title of the figure
as 'Graph for the x_f_linear_2'
    xlabel('n');           % name the label of x-axis as 'n'
    ylabel('x_f linear 2[n]') % name the label of y-axis as 'x_f_linear_2'

    subplot(4,1,3);         % set the place to display the plotting result
    stem(n,y_F_input_linear);   %plot the figure
    title('Graph for the output for y[n] when x[n]=a*x_f linear 1[n]+b*x_f
linear 2[n]');              % name the title of the figure as 'Graph for
the output for y[n] when x[n]=a*x1[n]+b*x2[n'
    xlabel('n');            % name the label of x-axis as 'n'
    ylabel('y[n] when x[n]=a*x_f linear 1[n]+b*x_f linear 2[n]');   %name
the label of y-axis as 'y[n]'

    subplot(4,1,4);         % set the place to display the plotting result
    stem(n,y_F_output_linear);  % plot the figure
    title('Graph for the output for y_F_output_linear=a*n.*x_f linear
1+b*n.*x_f linear 2');      % name the title of the figure as 'Graph for
the output for y_F_output_linear=a*n.*x_f_linear_1+b*n.*x_f_linear_2'
    xlabel('n');            % name the label of x-axis as 'n'
    ylabel('y_F output linear=a*n.*x_f linear 1+b*n.*x_f linear 2') % name
the label of y-axis as
'y_F_output_linear=a*n.*x_f_linear_1+b*n.*x_f_linear_2'
    %%

    x_f_invariant_1=zeros(1,41);
    x_f_invariant_2=zeros(1,41);

    for i=1:5
        x_f_invariant_1(i+18)=1;
    end
     for i=3:7
        x_f_invariant_2(i+18)=1;
    end

    y_f_invariant_1=n.*x_f_invariant_1;
    y_f_invariant_2=n.*x_f_invariant_2;

    %% The plot of 1.4 f invariant
    figure;     %create a window for ploting

    subplot(2,2,1);     % set the place to display the plotting result
    stem(n,y_f_invariant_1);%plot the figure
    title('y1[n]; Graph for checking the property of the time
invariant');            % name the title of the figure as 'y1[n];Graph for
checking the property of the time invariant'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('y1[n]')     % name the label of y-axis as 'y1[n]'

    subplot(2,2,2);     % set the place to display the plotting result
    stem(n,x_f_invariant_1);    % plot the figure
    title('x1[n]; Graph for checking the property of the time
invariant');            % name the title of the figure as 'x1[n];Graph for
checking the property of the time invariant'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('x1[n]')     % name the label of y-axis as 'x1[n]'

    subplot(2,2,3);      % set the place to display the plotting result
    stem(n,y_f_invariant_2);    % plot the figure
    title('y2[n]; Graph for checking the property of the time
invariant');        % name the title of the figure as 'y2[n]; Graph for
checking the property of the time invariant'
    xlabel('n');    % name the label of x-axis as 'n'
    ylabel('y2[n](get from the x2[n])') % name the label of y-axis as
'y2[n](get from the x2[n])'

    subplot(2,2,4); % set the place to display the plotting result
    stem(n,x_f_invariant_2);% plot the figure
    title('x2[n]; Graph for checking the property of the time
invariant');        % name the title of the figure as 'x2[n];Graph for
checking the property of the time invariant'
    xlabel('n');    % name the label of x-axis as 'n'
    ylabel('x2[n](get from the time shift of x1[n])') % name the label of
y-axis as 'x2[n](get from the time shift of x1[n])'
    %%

    x_f_causal=u;
    y_f_causal=n.*x_f_causal;

    %% The plot of 1.4 f casual
    figure;             % create a window for ploting

    subplot(2,1,1);     % set the place to display the plotting result
    stem(n,x_f_causal); % plot the figure
    title('x[n]; Graph for checking the property of casual');   %name the
title of the figure as 'x[n]; Graph for checking the property of casual'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('x[n]')      % name the label of y-axis as 'x[n]'

    subplot(2,1,2);     % set the place to display the plotting result
    stem(n,y_f_causal); % plot the figure
    title('y[n]; Graph for checking the property of casual');   %name the
title of the figure as 'y[n]; Graph for checking the property of casual'
    xlabel('n');        % name the label of x-axis as 'n'
    ylabel('y[n]')      % name the label of y-axis as 'y[n]'
    %%

    n_f_stable=[1:100];
    x_f_stable=ones(1,length(n_f_stable));
    y_f_stable=n_f_stable.*x_f_stable;

    %% The plot of 1.4 f stable
    figure; % create a window for ploting

    subplot(2,1,1);                 % set the place to display the plotting
result
    stem(n_f_stable,x_f_stable);    % plot the figure
    title('x[n]; Graph for checking the property of the stability');   %
name the title of the figure as 'x[n]; Graph for checking the property of
the stability'
    xlabel('n');                    % name the label of x-axis as 'n'
    ylabel('x[n]')                  % name the label of y-axis as 'x[n]'

    subplot(2,1,2); % set the place to display the plotting result
    stem(n_f_stable,y_f_stable);    % plot the figure
    title('y[n]; Graph for checking the property of the stability');   %
name the title of the figure as 'y[n]; Graph for checking the property of
the stability'
```

```
    xlabel('n');                    % name the label of x-axis as 'n'
    ylabel('y[n]')                  % name the label of y-axis as 'y[n]'
    %%

    inverse=1./n;
    x_f_invertible=ones(1,41);
    y_f_invertible=n.*x_f_invertible;
    y_f_invertible_to_x=y_f_invertible.*inverse;

    %% The plot of 1.4 f invertible
    figure;%create a window for ploting

    subplot(4,1,1);           % set the place to display the plotting 
result
    stem(n,x_f_invertible);   % plot the figure
    title('x[n]; Graph for checking the property of invertible');   %name 
the title of the figure as 'x[n]; Graph for checking the property of 
invertible'
    xlabel('n');              % name the label of x-axis as 'n'
    ylabel('x[n]')            % name the label of y-axis as 'x[n]'

    subplot(4,1,2);           % set the place to display the plotting 
result
    stem(n,y_f_invertible);   % plot the figure
    title('y[n]; Graph for checking the property of invertible');   % name 
the title of the figure as 'y[n]; Graph for checking the property of 
invertible'
    xlabel('n');              % name the label of x-axis as 'n'
    ylabel('y[n]')            % name the label of y-axis as 'y[n]'

    subplot(4,1,3);           % set the place to display the plotting 
result
    stem(n,inverse);          % plot the figure
    title('Inverse Function; Graph for checking the property of 
invertible');                 % name the title of the figure as 'Inverse 
Function; Graph for checking the property of invertible'
    xlabel('n');              % name the label of x-axis as 'n'
    ylabel('Inverse Function inverse=1/n') % name the label of y-axis as 
'x[n]'

    subplot(4,1,4);           % set the place to display the plotting 
result
    stem(n,y_f_invertible_to_x);    % plot the figure
    title('Function inversed back to x[n]; Graph for checking the property 
of invertible');              % name the title of the figure as 'Function 
inversed back to x[n]; Graph for checking the property of invertible'
    xlabel('n');              % name the label of x-axis as 'n'
    ylabel('Function inversed back to x[n]') % name the label of y-axis as 
'Function inversed back to x[n]'
    %%
```

The result:

Answer for problem (a):

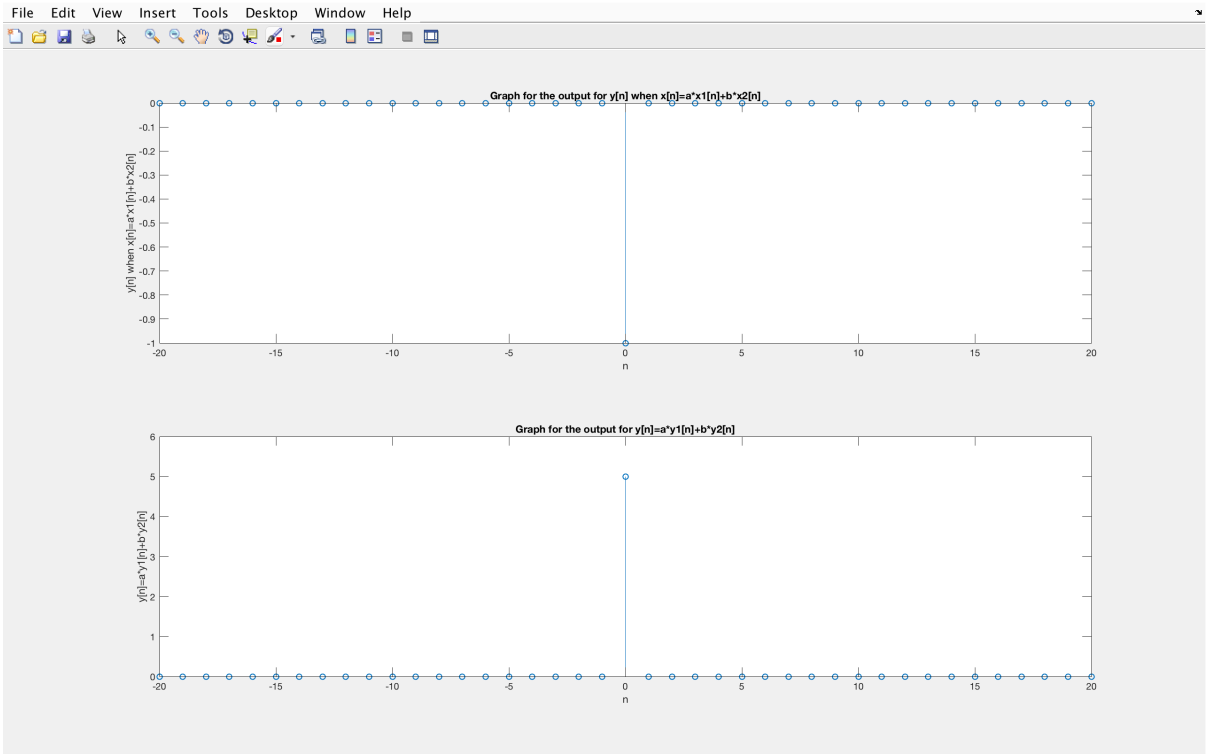
As these two signals (the output of ‘$x[n] = a\times x1[n] + \ b\times x2[n]$’ **and** ‘ $y[n] = a\times y1[n] + b\times y2[n]$)’ **are different, which means the system doesn’t satisfy the property of the linearity.**

I suppose ‘$x1[n] = \delta[n]$’ and ‘$x2[n] = 2\delta[n]$’. Variable ‘a’ and ‘b’ are two random numbers which I set it to be 5 and 7 respectively.

Core program:

```
a=5; % set constant value
b=7; % set constant value

n_x=[-5:9];
n_y=[-6:9];

n=[-20:20];
x_a1=zeros(1,41); % creat a all zero vector
x_a1(1,21)=1;
x_a2=zeros(1,41); % creat a all zero vector
x_a2(1,21)=2;

y_input= sin((pi/2)*(a*x_a1+b*x_a2));
y_output = a*sin((pi/2)*x_a1)+b*sin((pi/2)*x_a2);
```

The result of the plot:


File Edit View Insert Tools Desktop Window Help
Graph for the output for y[n] when x[n]=a*x1[n]+b*x2[n]
y[n] when x[n]=a*x1[n]+b*x2[n]
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
-0.8
-0.9
-1
-20
-15
-10
-5
0
5
10
15
20
n
Graph for the output for y[n]=a*y1[n]+b*y2[n]
y[n]=a*y1[n]+b*y2[n]
6
5
4
3
2
1
0
-20
-15
-10
-5
0
5
10
15
20
n

(Graph of the output of x[n]=a*x1[n]+ b*x2[n] and y[n]=a*y1[n]+b*y2[n] respectively) (a=5, b=7)

Answer for problem (b):

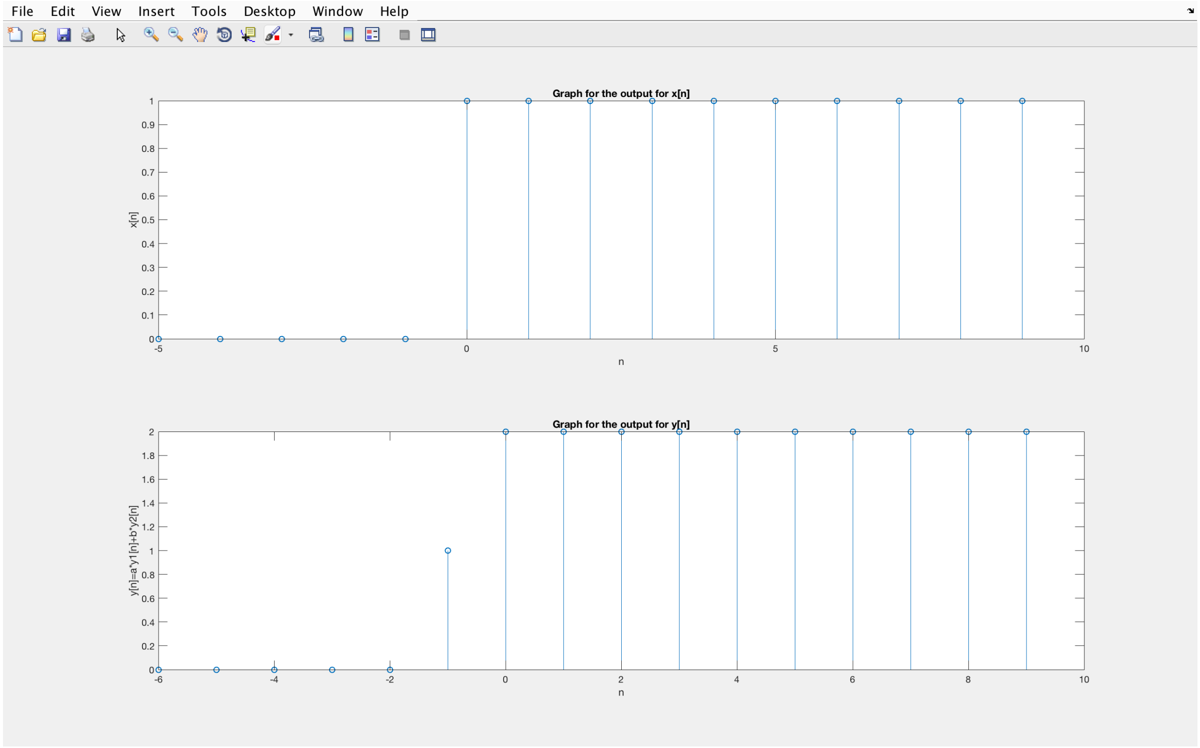
Before the input of the x[n], the outcome is already existing from the system, which do not satisfy the casual property as the output from the causal system can only related to the previous and now’s input. (For casual system, if x[t]=0 for t<t’, there must be y[n]=0 for t<t’) (non-anticipative system)

I suppose the input of the system is: ‘$x[n] = \mu[n]$’ . So the output of the system become ‘$y[n] = \mu[n] + \mu[n + 1]$’ .

Core program:

```
 %% The 1.4 b
u=zeros(1,41); % creat a all zero vector
for i=21:41
    u(i)=1;
end
x_b1=u;
x_b2=zeros(1,41);
for i=1:40
    x_b2(i)=x_b1(i+1);
end
Y=x_b2+x_b1;
Y_display=zeros(1,16); % set constant value
X_display=zeros(1,15); % set constant value

for i=1:16
    Y_display(i)=Y(i+14);
end
for i=1:15
    X_display(i)=x_b1(i+15);
end
%%
```

The result of the plot:



(Graph of x[n] and y[n] respectively)

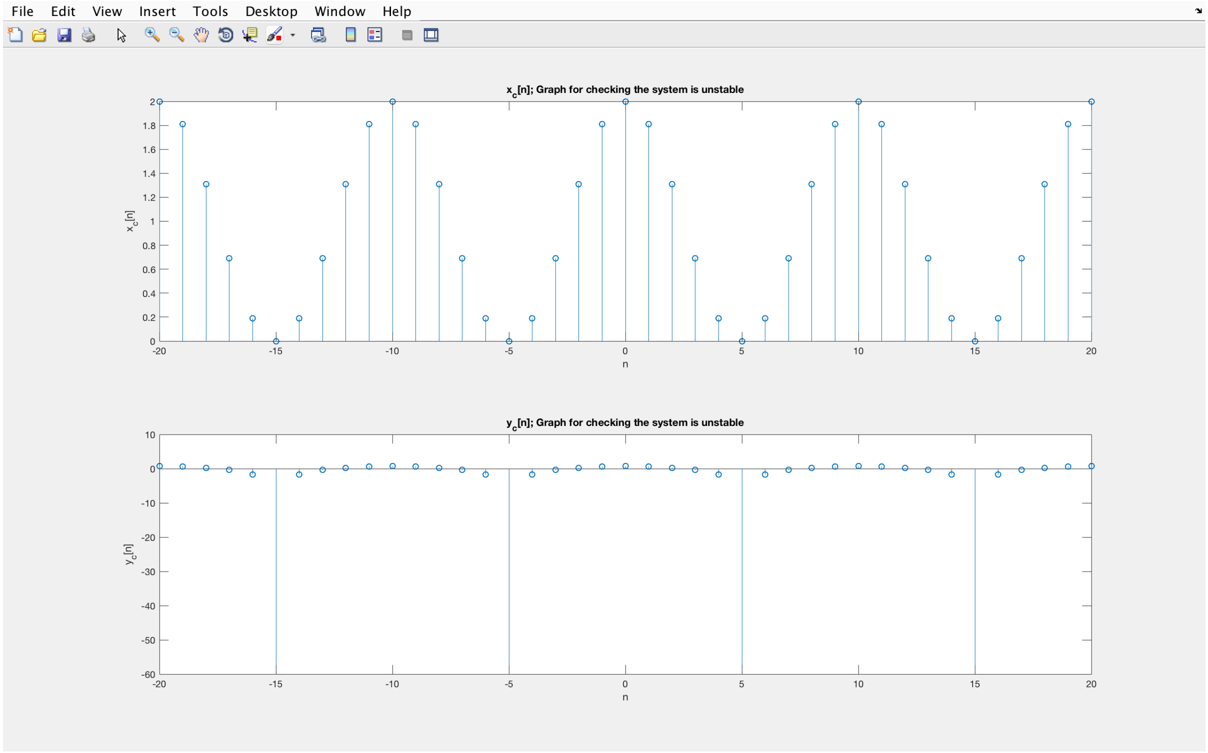
Answer for problem (c):

As the bounded input ($X_c[n] = 2cos(\pi n/10\ )^2$) cannot lead to the bounded output ($y[n] = log(x[n])$) when input of the signal is 0, the output would come to infinity. So the system is unstable.

Core program:

```
    n=[-20:20];

    %% The 1.4 c
    % x_c=zeros(1,41);     % set constant value
    % y_c=log(x_c);
    % y_c                  % Force to plot the y_c, which is all infinity
however x_c finit which means unstable

    x_c = 2*cos(pi*n/10).^2;
    y_c=log(x_c);
```

The result of the plot:



(Graph of the signals of Xc[n] (bounded input) and Yc[n] (unbounded output) respectively)

Answer for problem (f):

To prove the property of the Linearity:

This system satisfies the property of the linearity

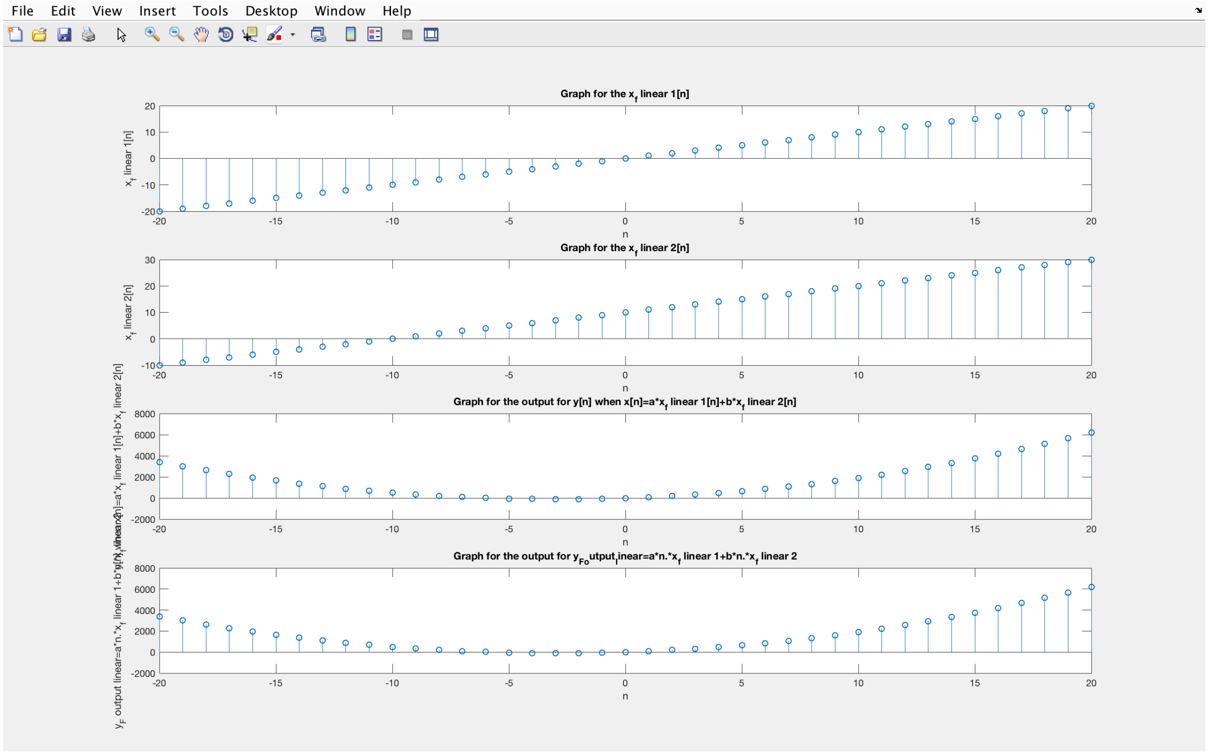
As these two signals (the output of $\mathrm{x}[\mathrm{n}] = \mathrm{a}\times x_f\mathrm{Linear1}[\mathrm{n}] + \mathrm{b}\times x_f\mathrm{Linear2}[\mathrm{n}]$ and $\mathrm{y}[\mathrm{n}] = \mathrm{a}\times y_f\mathrm{Linear1}[\mathrm{n}] + \mathrm{b}\times y_f\mathrm{Linear2}[\mathrm{n}]$) **are same, which means that it satisfies the additivity and scaling property. So the system satisfies the property of the linearity.**

The input is reconstructed by $'\mathrm{x\_f\_linear\_1}[\mathrm{n}] = \mathrm{n}'$ and $'\mathrm{x\_f\_linear\_2}[\mathrm{n}] = \mathrm{x\_f\_linear\_1}[\mathrm{n}+10]'$. The output of the system ‘y[n]=a* y_f_linear_1[n]+b* y_f_linear_2[n]’ should equal to the output of the ‘x[n] = a* x_f_linear_1[n] + b* x_f_linear_2[n]’.

Core program:

```
%% The 1.4 f linear
x_f_linear_1=[-20:20];
x_f_linear_2=[-10:30];
x_F_linear=a*x_f_linear_1+b*x_f_linear_2;
y_F_input_linear=n.*x_F_linear;
y_F_output_linear=a*n.*x_f_linear_1+b*n.*x_f_linear_2;
%%
```

The result of the plot:



(Graph of x1[n], x2[n], the output of x[n]=a*x1[n]+ b*x2[n] and y[n]=a*y1[n]+b*y2[n] respectively) (a=5, b=7)

To prove the property of the time-invariant:

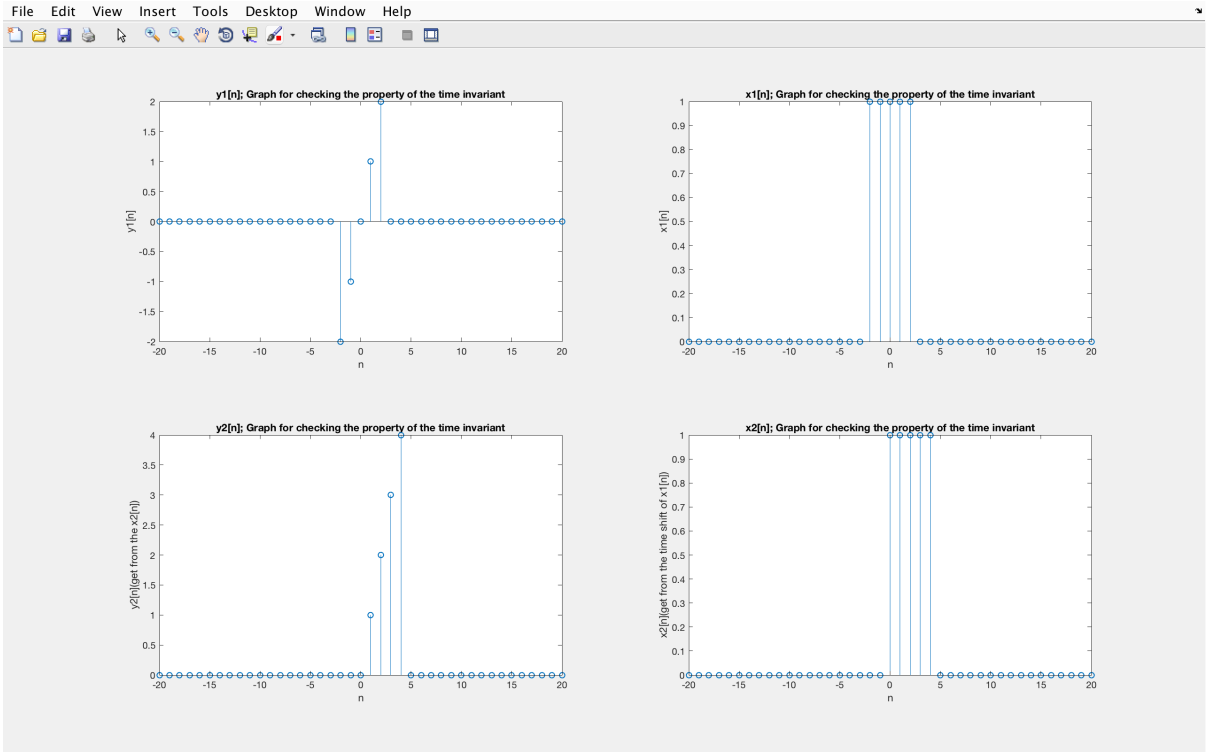
This system does not satisfy the property of time-invariant.

The characteristics of the system should be fixed over time if the system are time-invariant. However, from the graphs, the output varies with the shift of the input. So this system is not time-invariant.

Core program:

```
%% The 1.4 f time-invariant

x_f_invariant_1=zeros(1,41);
x_f_invariant_2=zeros(1,41);

for i=1:5
    x_f_invariant_1(i+18)=1;
end
 for i=3:7
    x_f_invariant_2(i+18)=1;
end

y_f_invariant_1=n.*x_f_invariant_1;
y_f_invariant_2=n.*x_f_invariant_2;

%%
```

The result of the plot:


File Edit View Insert Tools Desktop Window Help
y1[n]; Graph for checking the property of the time invariant
y1[n]
n
x1[n]; Graph for checking the property of the time invariant
x1[n]
n
y2[n]; Graph for checking the property of the time invariant
y2[n](get from the x2[n])
n
x2[n]; Graph for checking the property of the time invariant
x2[n](get from the time shift of x1[n])
n

(Graph of x1[n], x2[n], y1[n] and y2[n])

To prove the property of Casual:

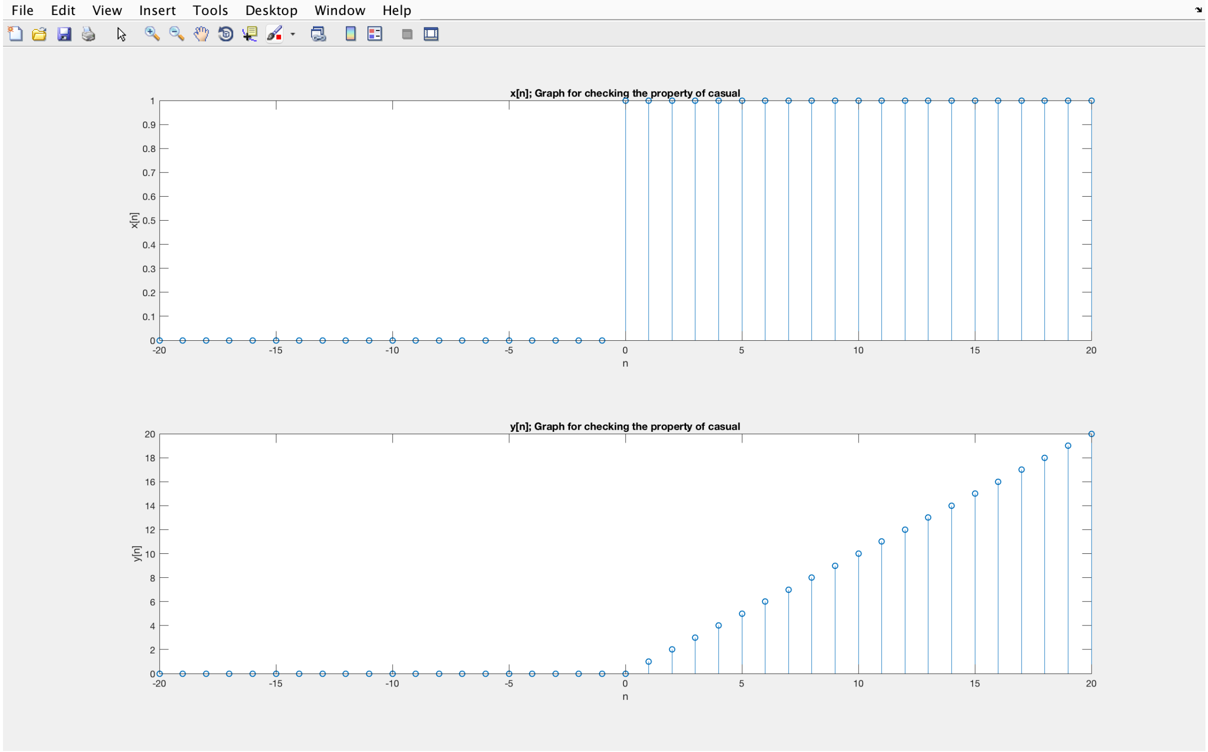
This system satisfies the casual property.

Before the input of the x[n], this system does not have out comes, which satisfy the casual property that the output only can only related to the previous and now's input. (For casual system, if x[t]=0 for t<t', there must be y[n]=0 for t<t') (non-anticipative system)

Core program:

```
%% The 1.4 f casual
u=zeros(1,41); % creat a all zero vector

for i=21:41
    u(i)=1;
end

x_f_causal=u;
y_f_causal=n.*x_f_causal;
%%
```

The result of the plot:



(Graph of x[n] and y[n])

To prove the property of stable:

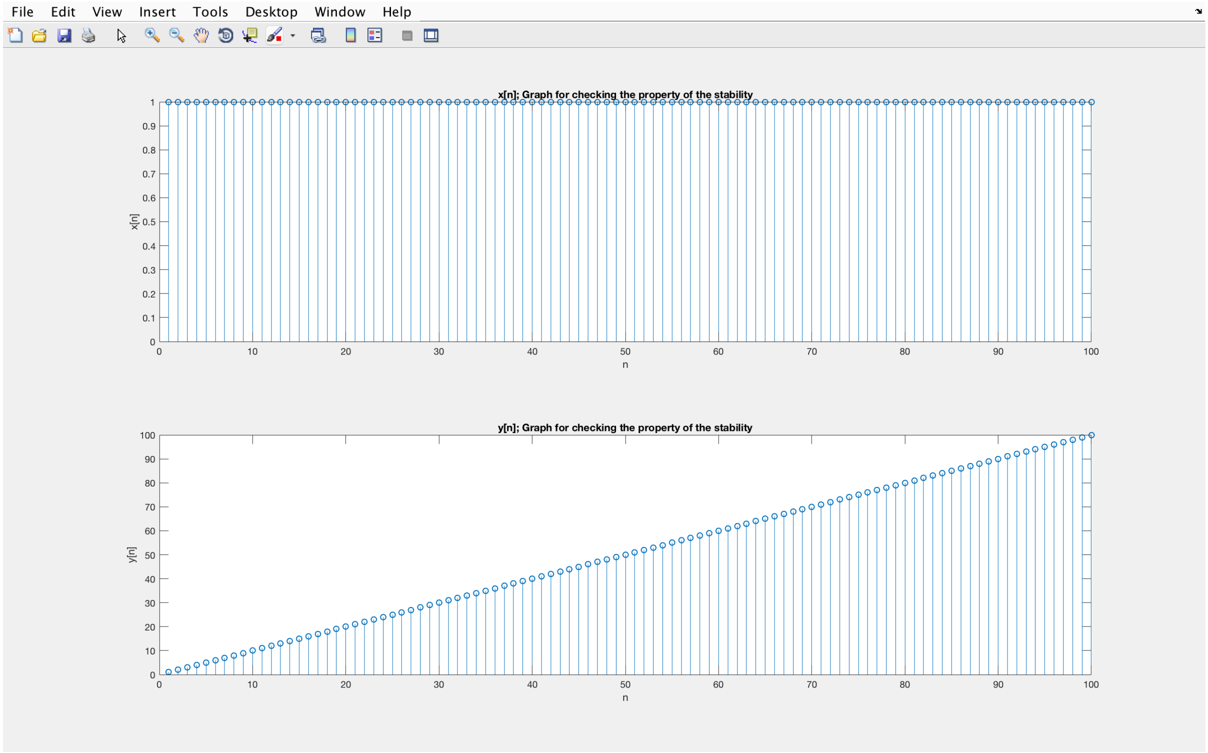
This system is unstable.

For the stable system, the bounded inputs lead to bounded outputs. However, it can be easily known that the bounded input (x[n]=u[n]) **leads to an unbounded output** (The output would increase continuously to infinity). **So this system is unstable.**

Core program:

```
%% The 1.4 f stable

n_f_stable=[1:100];
x_f_stable=ones(1,length(n_f_stable));
y_f_stable=n_f_stable.*x_f_stable;

%%
```

The result of the plot:



(Graph of x[n] y[n])

To prove the property of the invertible:

This system is not invertible.

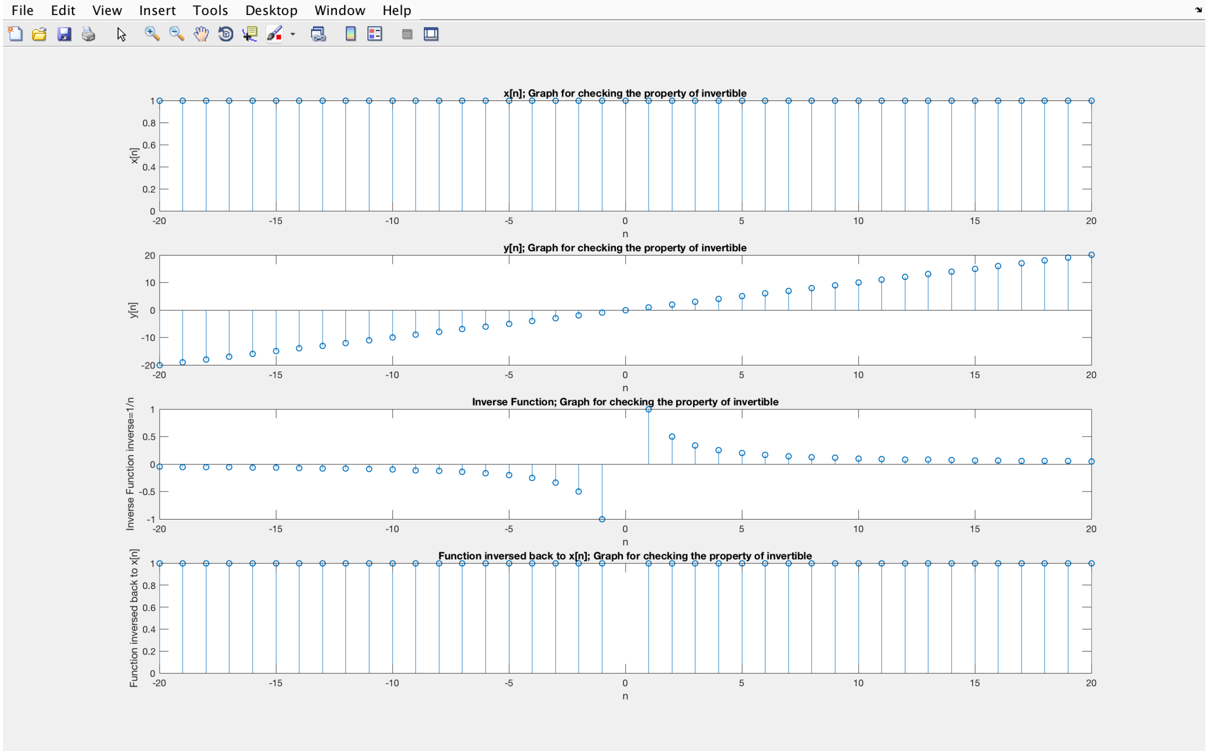
According to the graph I plotted below, if the 'X[n]' is '1' from '-20' to '20', we can see that only '1/n' can help the system (Y[n]=nX[n]) **to be invertible. However, from the graph, we can realize that the point 0 does not exist compare to the X[n]. So this system is not invertible.** (Inverse system does not exist or there is no inverse system cascade with original system)

If the domain does not include the point n=0, the system would be invertible.

Core program:

```
%% The 1.4 f invertible
n=[-20:20];
inverse=1./n;
x_f_invertible=ones(1,41);
y_f_invertible=n.*x_f_invertible;
y_f_invertible_to_x=y_f_invertible.*inverse;

%%
```

The result of the plot:


File
Edit
View
Insert
Tools
Desktop
Window
Help
x[n]; Graph for checking the property of invertible
x[n]
1
0.8
0.6
0.4
0.2
0
-20
-15
-10
-5
0
5
10
15
20
n
y[n]; Graph for checking the property of invertible
y[n]
20
10
0
-10
-20
Inverse Function; Graph for checking the property of invertible
Inverse Function inverse=1/n
1
0.5
0
-0.5
-1
Function inversed back to x[n]; Graph for checking the property of invertible
Function inversed back to x[n]

(Graph of x[n], y[n], Inverse function and Function inversed back to x[n])

Summary and comments：

This section requires us to examine a system's attributes such as causal, linearity, time-invariant, stable… During this experiment, I could only try to find if there exist some condition that have conflicts with the definition. However, it would be more grounded if we check it by definition directly.

During this experiment, I use the Matlab to plot each conditions. Using each graph to check if they have each property.

From this part, I have learned loads of techniques on using Matlab to plot the graph to falsify some properties of the system, as Matlab cannot directly prove the property. Moreover, I understand more on verifying the property of the system and how I could prove it or how they can be shown by plotting of the input-output signal on Matlab.

Suggestion for this lab：

It would be better if we can get the answers of the lab to check if our work is staying on track. Thanks!

Score：

Instructor：