ELEC 342 Lab 2 ----
Additional MATLAB features, Properties of Signals and Systems, Convolution and System Response

Name:	JUNPENG GAI
ID number:	40009896
Course number:	ELEC342-X Lab 2224
Date performed:	Monday, 13 February 2023
Due Date:	Monday, 27 February 2023
Lab Instructor Name:	Salameh, Ahmed

"I certify that this submission is my original work and meets the Faculty's Expectations of Originality"

Junpeng Gai
2023/02/26

盖俊鹏

Contents

1.Objectives

The objectives of this lab are about to get to know more about the implementation with signal and systems.

In order to do so we have to learn how to perform “conditions”, “for loops”, “proof linearly” and other system properties. Also, we can learn how to design an experimental test to verify the system is linear and time invariant.

The other side we will get to learn how to use the MATLAB to compute convolution.

2.Theory

2.1 For loops

For loops: in MATLAB when we are using for loop, we will use

```
For index in range
commands
end
```

Basically, this is a method of recursion, as long as we have a base case we can use for loop to do the following cases.

2.2 Condition statement

It’s similar in the C language, when there is a condition we can have 2 branches after it. For example, if the result is the same as expected we can have following statements:

```
If(condition=expected)
```

Do something

Else

Do something

2.3 Prove linearity

Linearity is when the linear combination of the input will give the same results of the combine of their outputs.

2.4 Even odd components

If there is function it can be a combination of even function and odd function. So, by using the graph method, we can plot x(n) and x(-n) separately and then by combine the graph we can get:

Odd component: $\frac{x(n)-x(-n)}{2}$

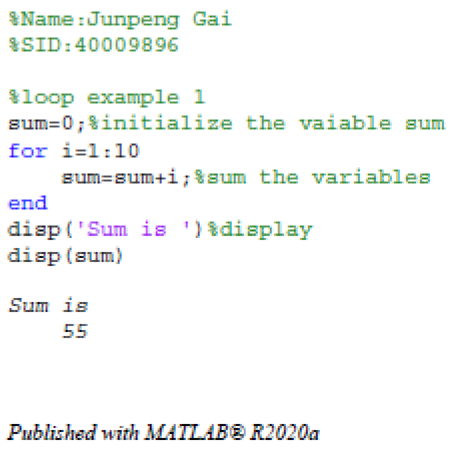
Even component: $\frac{x(n)+x(-n)}{2}$

3.Tasks/Results/Discussion

3.1 For loop

3.1.1 Example 1

Use the for loop to get the sum from 1 to 10:



```
%Name:Junpeng Gai
%SID:40009896

%loop example 1
sum=0;%initialize the vaiable sum
for i=1:10
    sum=sum+i;%sum the variables
end
disp('Sum is ')%display
disp(sum)

Sum is
    55
```

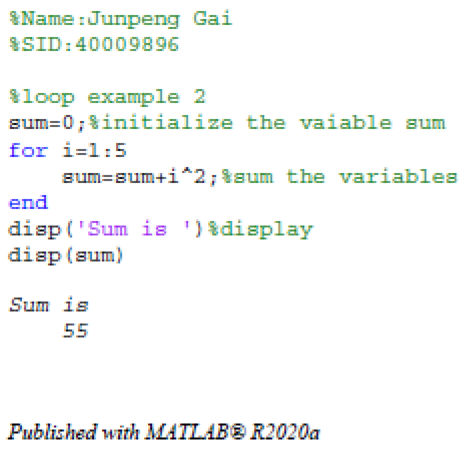
Published with MATLAB® R2020a

Figure 1 Code for example 1

3.1.2 Example 2

Example 2: The following loop finds the sum of the squares of the integers from 1 to 5, sum = $1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 1 + 4 + 9 + 16 + 25 = 55$.

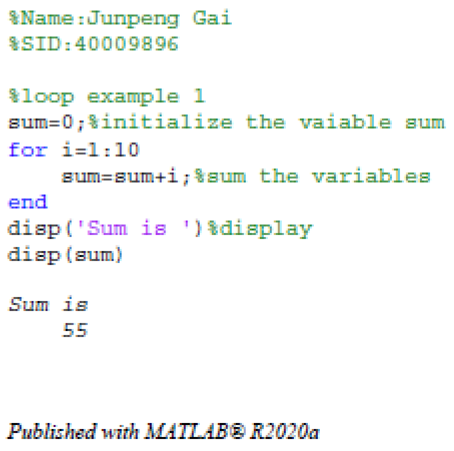
Figure 2 Example 2



```
%Name:Junpeng Gai
%SID:40009896

%loop example 2
sum=0;%initialize the vaiable sum
for i=1:5
    sum=sum+i^2;%sum the variables
end
disp('Sum is ')%display
disp(sum)

Sum is
    55


Published with MATLAB® R2020a
```

Figure 3 Code for example 2

3.1.2 Example 3 initialize array



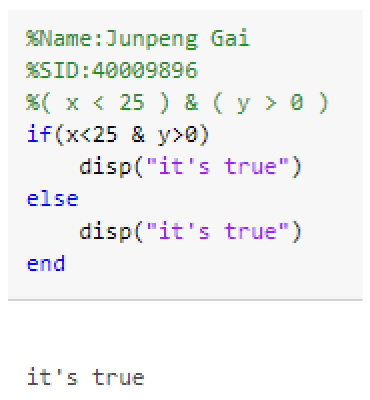
```
%Name:Junpeng Gai
%SID:40009896

%loop example 1
sum=0;%initialize the vaiable sum
for i=1:10
    sum=sum+i;%sum the variables
end
disp('Sum is ')%display
disp(sum)

Sum is
    55


Published with MATLAB® R2020a
```

Figure 4 Code for example 3

3.2 If statement

3.1.2 Example 1

Compare the condition and print it out:



```
%Name:Junpeng Gai
%SID:40009896
%( x < 25 ) & ( y > 0 )
if(x<25 & y>0)
    disp("it's true")
else
    disp("it's true")
end
```

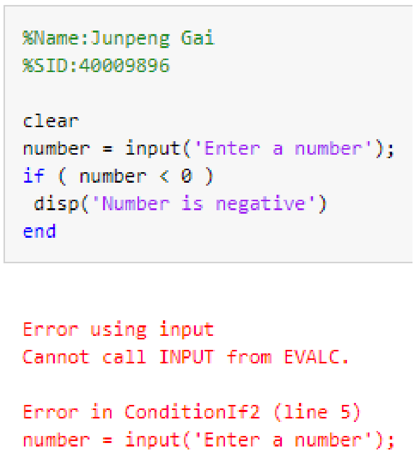
```
it's true
```

Figure 5 Code for example 1

3.1.2 Example 2

Enter a number and print it out if it's a negative number:

Because the publish can't show our input but this code works.

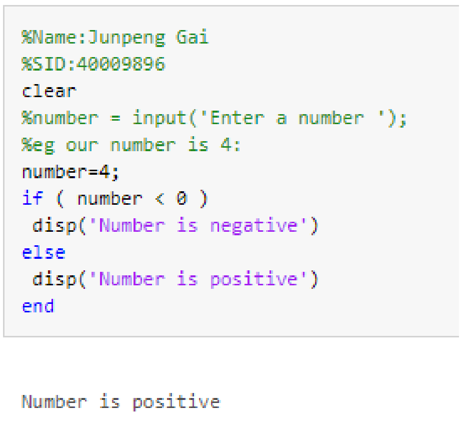


```
%Name:Junpeng Gai
%SID:40009896

clear
number = input('Enter a number');
if ( number < 0 )
 disp('Number is negative')
end
```

```
Error using input
Cannot call INPUT from EVALC.

Error in ConditionIf2 (line 5)
number = input('Enter a number');
```

Figure 6 Code for example 2

3.1.2 Example 3 If else statement:



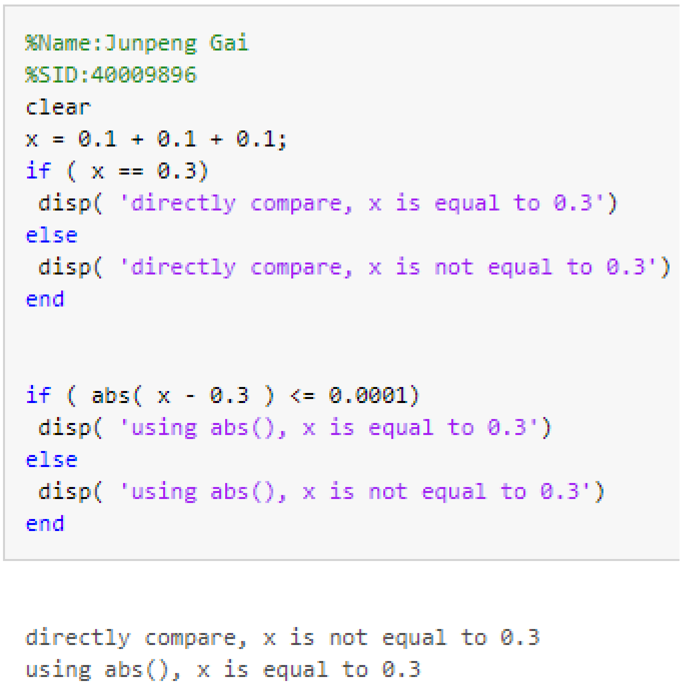
```
%Name:Junpeng Gai
%SID:40009896
clear
%number = input('Enter a number ');
%eg our number is 4:
number=4;
if ( number < 0 )
 disp('Number is negative')
else
 disp('Number is positive')
end
```

```
Number is positive
```

Figure 7 Code for example 3

We used 4 as an example, because 4 is smaller than 0, the second statement will be printed.

3.1.2 Example 4 compare 2 numbers



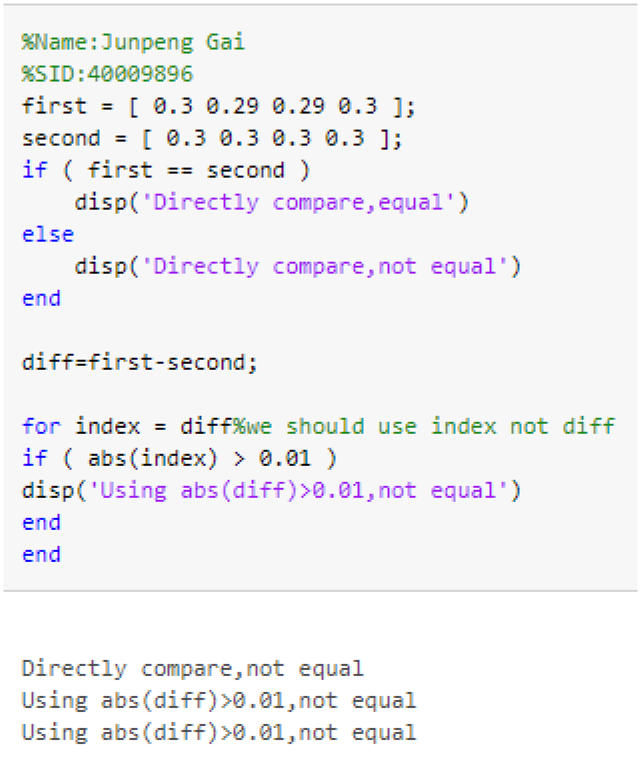
```
%Name:Junpeng Gai
%SID:40009896
clear
x = 0.1 + 0.1 + 0.1;
if ( x == 0.3)
 disp( 'directly compare, x is equal to 0.3')
else
 disp( 'directly compare, x is not equal to 0.3')
end


if ( abs( x - 0.3 ) <= 0.0001)
 disp( 'using abs(), x is equal to 0.3')
else
 disp( 'using abs(), x is not equal to 0.3')
end
```

```
directly compare, x is not equal to 0.3
using abs(), x is equal to 0.3
```

Figure 8 Code for example 4

After taking ENGR 391 we know that there are 2 kinds of error in the computer, truncation error and round off error. First error is because the computer save data in binary so it’s not possible for computer to save an exact recurring decima. Round error is because some umber will be round off in the computer. So, in the future we should use the absolute differences is within certain range instead of directly equals to the results.

3.1.2 Example 5



```
%Name:Junpeng Gai
%SID:40009896
first = [ 0.3 0.29 0.29 0.3 ];
second = [ 0.3 0.3 0.3 0.3 ];
if ( first == second )
    disp('Directly compare,equal')
else
    disp('Directly compare,not equal')
end

diff=first-second;

for index = diff%we should use index not diff
if ( abs(index) > 0.01 )
disp('Using abs(diff)>0.01,not equal')
end
end
```

```
Directly compare,not equal
Using abs(diff)>0.01,not equal
Using abs(diff)>0.01,not equal
```

Figure 9 Code for example 5

This example tells us that the comparison between 2 vectors which is the same methodology as we did before.

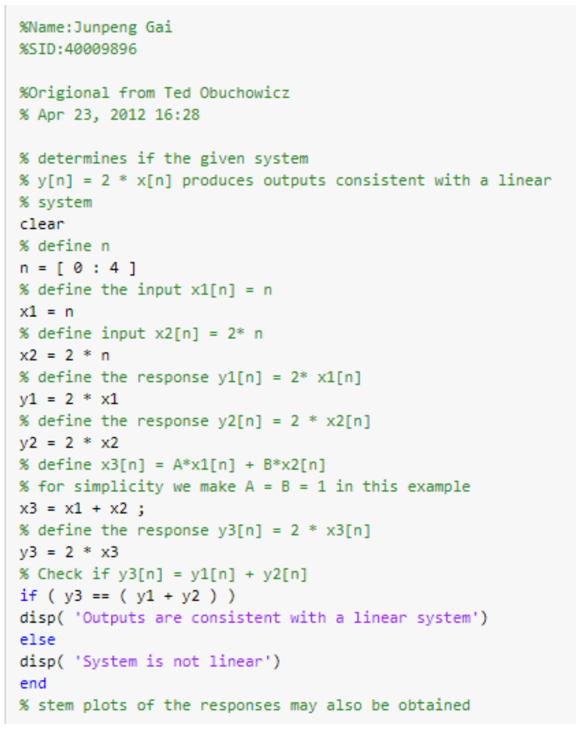
3.3 Linearity

3.1.2 Example 1

Prove y=2x is linear or not.

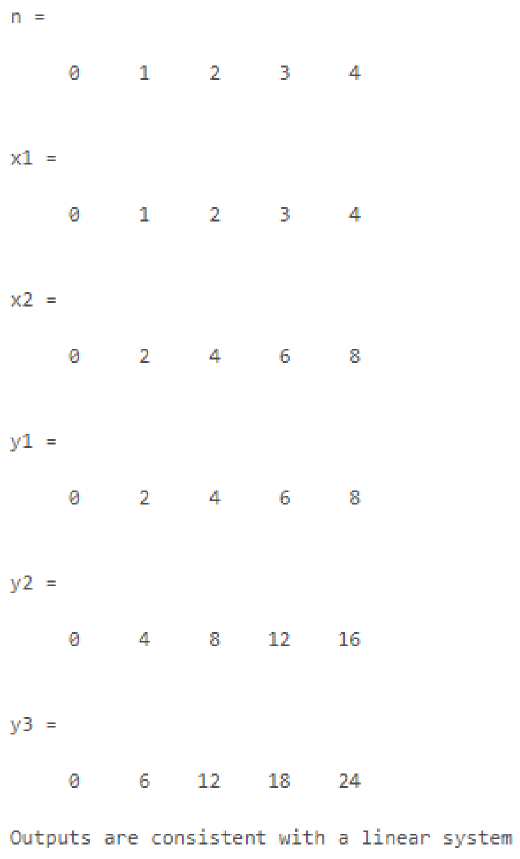
By running the example code given by Ted Obuchowicz, we can see the methodology beneath the code:

- Give 2 inputs and get their outputs
- Use input1+input2 as 3rd input to get output3
- Compare output1+output2=output3 or not.



```
%Name:Junpeng Gai
%SID:40009896

%Origional from Ted Obuchowicz
% Apr 23, 2012 16:28

% determines if the given system
% y[n] = 2 * x[n] produces outputs consistent with a linear
% system
clear
% define n
n = [ 0 : 4 ]
% define the input x1[n] = n
x1 = n
% define input x2[n] = 2* n
x2 = 2 * n
% define the response y1[n] = 2* x1[n]
y1 = 2 * x1
% define the response y2[n] = 2 * x2[n]
y2 = 2 * x2
% define x3[n] = A*x1[n] + B*x2[n]
% for simplicity we make A = B = 1 in this example
x3 = x1 + x2 ;
% define the response y3[n] = 2 * x3[n]
y3 = 2 * x3
% Check if y3[n] = y1[n] + y2[n]
if ( y3 == ( y1 + y2 ) )
disp( 'Outputs are consistent with a linear system')
else
disp( 'System is not linear')
end
% stem plots of the responses may also be obtained
```

Figure 10 Code for linearity



```
n =

     0     1     2     3     4


x1 =

     0     1     2     3     4


x2 =

     0     2     4     6     8


y1 =

     0     2     4     6     8


y2 =

     0     4     8    12    16


y3 =

     0     6    12    18    24

Outputs are consistent with a linear system
```

Figure 11 Output of linearity

3.3 Even Odd components

$$determines\ is\ this\ function\ even\ or\ odd : x = \sin\left(\frac{2pi}{10} * n\right) for\ n = 0,1,2,,9$$

3.1.2 Example1

```
%Name:Junpeng Gai
%SID:40009896
% determines if a signal x[n] = function(n)
% is even or odd
% x[n] is EVEN is x[n] = x[-n]
% x[n] is ODD if x[n] = -x[-n]
% define n
n = 0 : 9;
% define x1[n]
x1 = sin( 2*pi/10 * n );

subplot(1,2,1)
stem(n, x1)
% define x2 = x1[-n]
x2 = sin( 2*pi/10 * (-n) );
subplot(1,2,2)
stem(-n, x2)
if ( x2 == x1 )
 disp('EVEN')
elseif ( x2 == (-x1) )
 disp('ODD')
else
 disp('NEITHER EVEN NOR ODD')
end
```

```
ODD
```

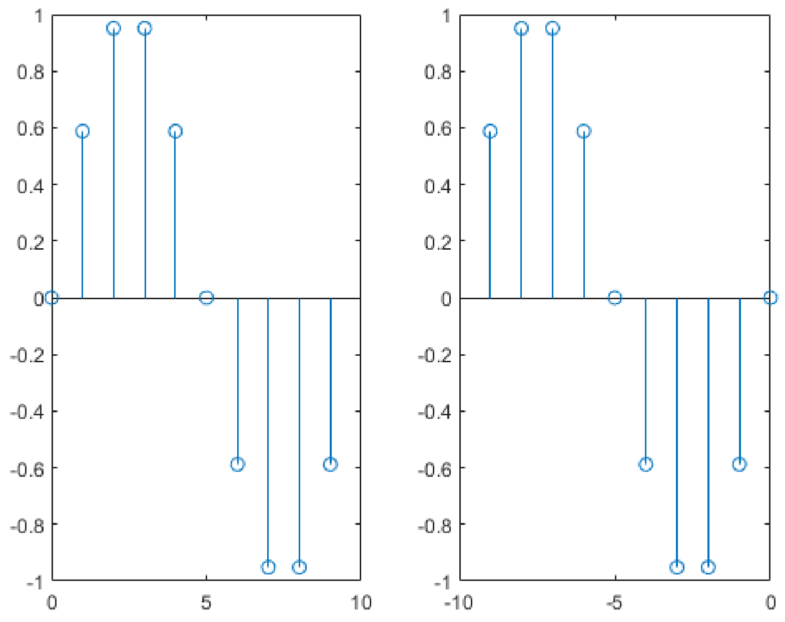
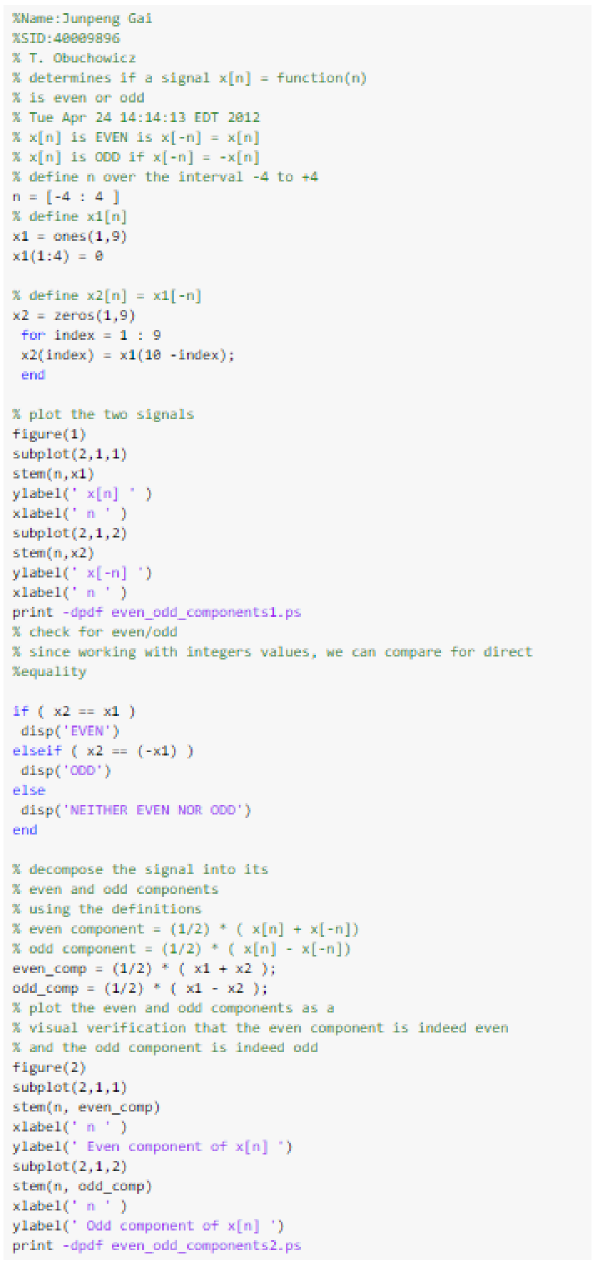


Figure 12 Example code for example 1

We can see that we started with domain from[-n,n], so we can plot the graph for x(n) and x(-n),by comparing we can see that this function is an odd function. On the other hand, it's a sin function so it an odd function.

3.1.2 Example 2



```
%Name:Junpeng Gai
%SID:40009896
% T. Obuchowicz
% determines if a signal x[n] = function(n)
% is even or odd
% Tue Apr 24 14:14:13 EDT 2012
% x[n] is EVEN is x[-n] = x[n]
% x[n] is ODD if x[-n] = -x[n]
% define n over the interval -4 to +4
n = [-4 : 4 ]
% define x1[n]
x1 = ones(1,9)
x1(1:4) = 0

% define x2[n] = x1[-n]
x2 = zeros(1,9)
 for index = 1 : 9
 x2(index) = x1(10 -index);
 end

% plot the two signals
figure(1)
subplot(2,1,1)
stem(n,x1)
ylabel(' x[n] ' )
xlabel(' n ' )
subplot(2,1,2)
stem(n,x2)
ylabel(' x[-n] ')
xlabel(' n ' )
print -dpdf even_odd_components1.ps
% check for even/odd
% since working with integers values, we can compare for direct
%equality

if ( x2 == x1 )
 disp('EVEN')
elseif ( x2 == (-x1) )
 disp('ODD')
else
 disp('NEITHER EVEN NOR ODD')
end

% decompose the signal into its
% even and odd components
% using the definitions
% even component = (1/2) * ( x[n] + x[-n])
% odd component = (1/2) * ( x[n] - x[-n])
even_comp = (1/2) * ( x1 + x2 );
odd_comp = (1/2) * ( x1 - x2 );
% plot the even and odd components as a
% visual verification that the even component is indeed even
% and the odd component is indeed odd
figure(2)
subplot(2,1,1)
stem(n, even_comp)
xlabel(' n ' )
ylabel(' Even component of x[n] ')
subplot(2,1,2)
stem(n, odd_comp)
xlabel(' n ' )
ylabel(' Odd component of x[n] ')
print -dpdf even_odd_components2.ps
```

Figure 13 Example code for example2

In this question we are required to get even and odd component of a unit step function, reading the code we realize we can initialize input with ones and zeros and then get x(-n). After that we can get expression for odd and even parts and plot them.

```
n =

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


x1 =

     1     1     1     1     1     1     1     1     1


x1 =

     0     0     0     0     1     1     1     1     1


x2 =

     0     0     0     0     0     0     0     0     0

NEITHER EVEN NOR ODD
```

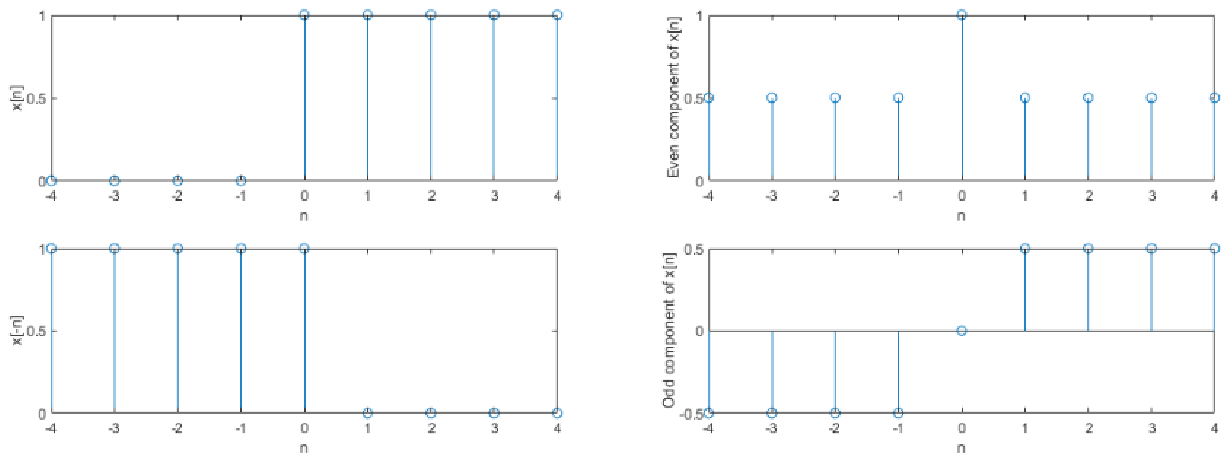
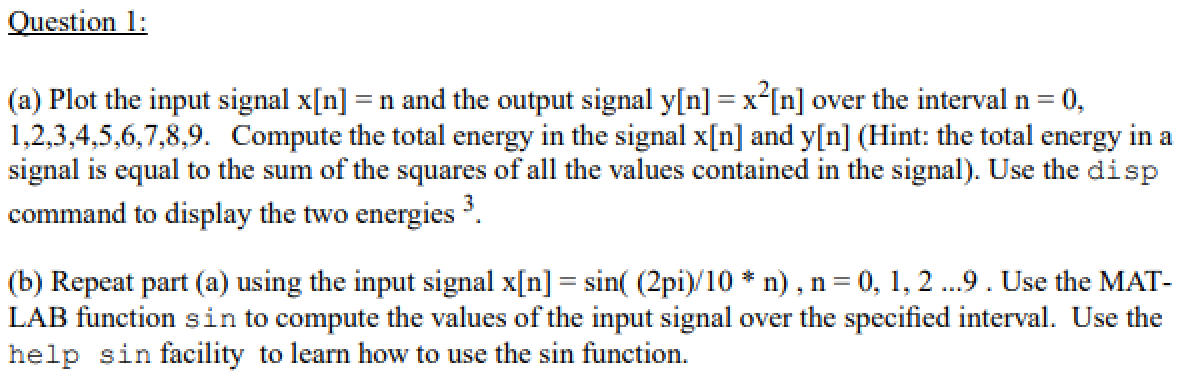


Figure 14 Output from the code above

After plotting them, we can see that there is an overlapping section when n=0, so (1+1)*0.5 gives 1 at origin.

4.Questions

4.1 Question 1



Question 1:

(a) Plot the input signal x[n] = n and the output signal y[n] = x^2[n] over the interval n = 0, 1,2,3,4,5,6,7,8,9. Compute the total energy in the signal x[n] and y[n] (Hint: the total energy in a signal is equal to the sum of the squares of all the values contained in the signal). Use the `disp` command to display the two energies [3].

(b) Repeat part (a) using the input signal x[n] = sin((2pi)/10 * n) , n = 0, 1, 2 ...9 . Use the MAT-LAB function `sin` to compute the values of the input signal over the specified interval. Use the `help sin` facility to learn how to use the sin function.

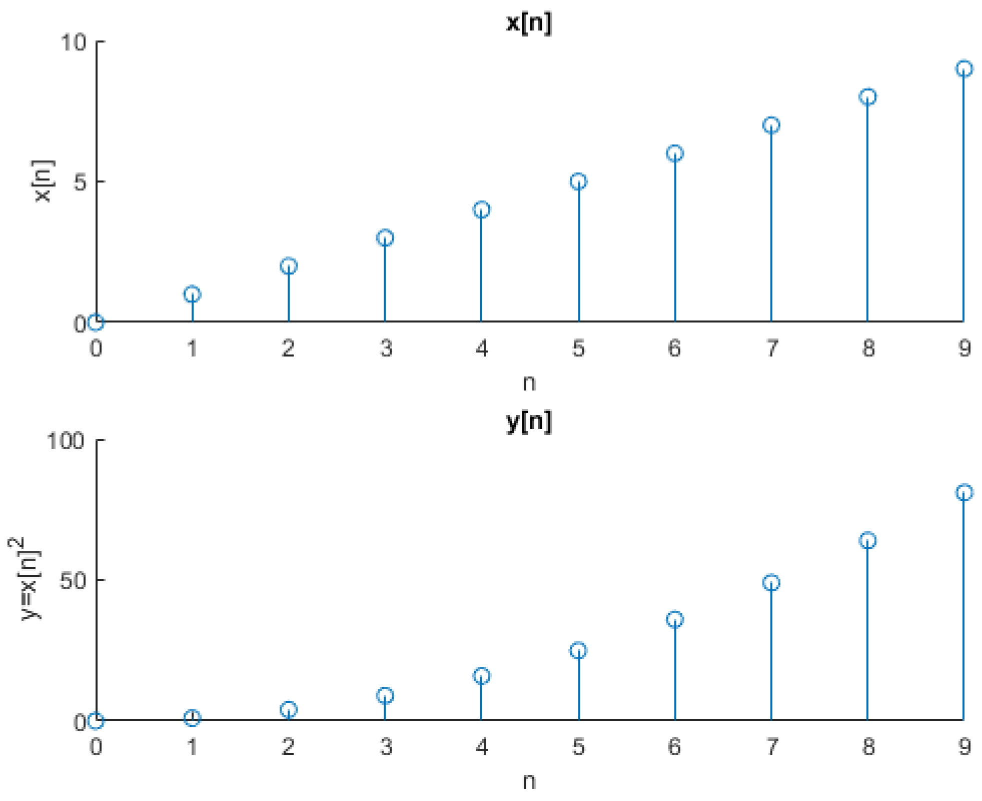
Figure 15 Question 1

4.1.1 Question1 -a

```
%Name:Junpeng Gai
%SID:40009896

n=(0:9);%defin the domain
X=n;    %defin x[n]
Y=X.^2; %defin y[n]
Energy_x=0; %initialize the energy for x
Energy_y=0; %initialize the energy for y
for index = X % loop in X array
 Energy_x = Energy_x + index.^2; % sum to Energy_x
end

disp('The Energy of x[n] is')
disp(Energy_x)  %calculate the total energy

for index = Y % % loop in Y array
 Energy_y = Energy_y + index.^2; % sum to Energy_y
end

disp('The Energy of y[n] is')
disp(Energy_y)  %calculate the total energy

subplot(2,1,1) %first graph
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
```

```
stem(n,X);
hold off
subplot(2,1,2)  %second graph
hold on
title('y[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('y=x[n]^2')%set label for y
stem(n,Y);
hold off
```

```
The Energy of x[n] is
   285

The Energy of y[n] is
       15333
```



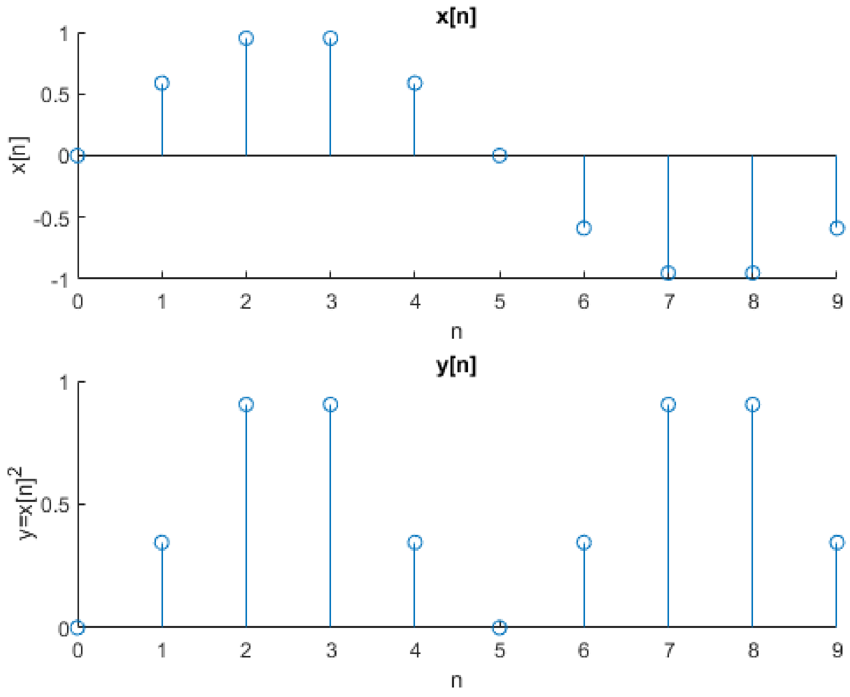
[Published with MATLAB® R2020a](#)

From the results we can see that the calculation is the same as expected value.

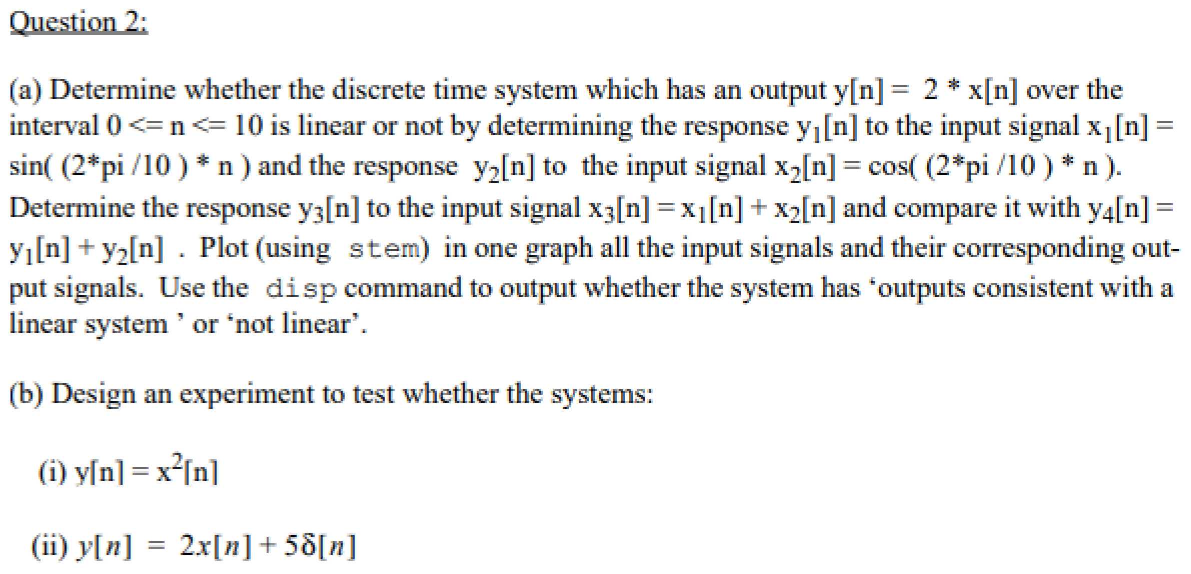
4.1.2 Question1 -b

```
%Name:Junpeng Gai
%SID:40009896

n=(0:9);%defin the domain
X= sin( (2*pi)/10 * n);    %defin x[n]
Y=X.^2; %defin y[n]
Energy_x=0; %initialize the energy for x
Energy_y=0; %initialize the energy for y
for index = X % loop in X array
 Energy_x = Energy_x + index.^2; % sum to Energy_x
end

disp('The Energy of x[n] is')
disp(Energy_x)  %calculate the total energy

for index = Y % % loop in Y array
 Energy_y = Energy_y + index.^2; % sum to Energy_y
end

disp('The Energy of y[n] is')
disp(Energy_y)  %calculate the total energy

subplot(2,1,1) %first graph
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,X);
hold off
subplot(2,1,2)  %second graph
hold on
title('y[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('y=x[n]^2')%set label for y
stem(n,Y);
hold off
```

```
The Energy of x[n] is
    5.0000

The Energy of y[n] is
    3.7500
```



4.2 Question 2



Question 2:

(a) Determine whether the discrete time system which has an output y[n] = 2 * x[n] over the interval 0 <= n <= 10 is linear or not by determining the response y_1[n] to the input signal x_1[n] = sin((2*pi /10) * n) and the response y_2[n] to the input signal x_2[n] = cos((2*pi /10) * n). Determine the response y_3[n] to the input signal x_3[n] = x_1[n] + x_2[n] and compare it with y_4[n] = y_1[n] + y_2[n] . Plot (using `stem`) in one graph all the input signals and their corresponding output signals. Use the `disp` command to output whether the system has 'outputs consistent with a linear system ' or 'not linear'.

(b) Design an experiment to test whether the systems:

(i) $y[n] = x^2[n]$

(ii) $y[n] = 2x[n] + 5\delta[n]$

Figure 16 Question2

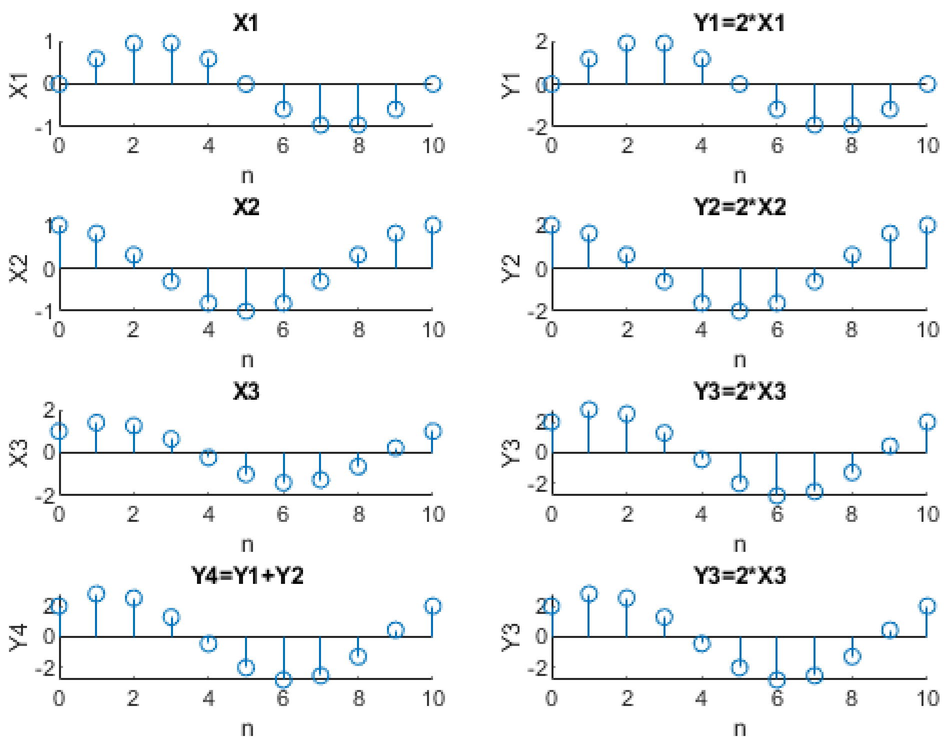
4.2.1 Question2 -a

```
%Name:Junpeng Gai
%SID:40009896

n=(0:10);%defin the domain
Y=2*X;    %defin system
```

```
X1=sin( (2*pi /10 ) * n ); %defin X1
X2=cos( (2*pi /10 ) * n ); %defin X2
X3=X1+X2;                  %defin X3 is the linear combination fo X1 and X2
Y1=2*X1;                   %defin OUTPUT for X1
Y2=2*X2;                   %defin OUTPUT for X2
Y3=2*X3;                    %defin OUTPUT for X3
Y4=Y1+Y2;                   %defin linear combination of Y1 and Y2

if ( Y4 == Y3 )   %condition,if success display it's a lineat sys
disp( 'Outputs are consistent with a linear system')
else              %else it isn't.
disp( 'System is not linear')
end

subplot(4,2,1);
hold on
title('X1')   %set the tittle
xlabel('n')     %set label for x
ylabel('X1')  %set label for y
stem(n,X1);     %plot input X1
hold off
subplot(4,2,2);
hold on
title('Y1=2*X1')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y1')  %set label for y
stem(n,Y1);     %plot output Y1
hold off
subplot(4,2,3);
hold on
title('X2')   %set the tittle
xlabel('n')     %set label for x
ylabel('X2')  %set label for y
stem(n,X2);     %plot input X2
hold off
subplot(4,2,4);
hold on
title('Y2=2*X2')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y2')  %set label for y
stem(n,Y2);     %plot output Y2
hold off
subplot(4,2,5);
hold on
title('X3')   %set the tittle
xlabel('n')     %set label for x
ylabel('X3')  %set label for y
stem(n,X3);     %plot input X3
hold off
```

```
subplot(4,2,6);
hold on
title('Y3=2*X3')   %set the tittle
xlabel('n')      %set label for x
ylabel('Y3')  %set label for y
stem(n,Y3);      %plot output Y3
hold off
subplot(4,2,7);
hold on
title('Y4=Y1+Y2')   %set the tittle
xlabel('n')      %set label for x
ylabel('Y4')  %set label for y
stem(n,Y4);      %plot output Y4,which is the left hand side
hold off
subplot(4,2,8);
hold on
title('Y3=2*X3')   %set the tittle
xlabel('n')      %set label for x
ylabel('Y3')  %set label for y
stem(n,Y3);      %plot output Y3,which is the right hand side
hold off
```

Outputs are consistent with a linear system



Published with MATLAB® R2020a

4.2.2 Question2 -b
4.2.2.1 Question2 -b (i)

I would say the larger set of values will look like a continue time signal. But the results are the same no matter how larger number is as long its's within [1,0]

```
%Name:Junpeng Gai
%SID:40009896


delay = 1; %delay


x= [0:1];                %define input region[0:1]
y=x.^2;                  %define output
xa=3*x;                  %define input xa
ya=xa.^2;                %define output ya
xb=3*x;                  %define input xb
yb=xb.^2;                %define output yb
Yab=ya+yb;               %define output yab=ya+yb
xab=xa+xb;               %define input xab=xa+xb
yab=xab.^2;              %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:1]') %set the tittle
xlabel('n')              %set label for x
ylabel('y without delay')          %set label for y
stem(1:length(x),y)
hold off

subplot(4,1,2)
hold on
title('delay input(delay=1)')      %set the tittle
xlabel('n')              %set label for x
ylabel('delay input(delay=1)')      %set label for y
x1=[zeros(1:delay) x];             %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
hold on
title('delay ouput(delay=1)')      %set the tittle
xlabel('n')              %set label for x
ylabel('delay output(delay=1)')     %set label for y
y2=[0 x.^2];             %delay output by 1
stem(1:length(x)+delay,y2)
hold off
```

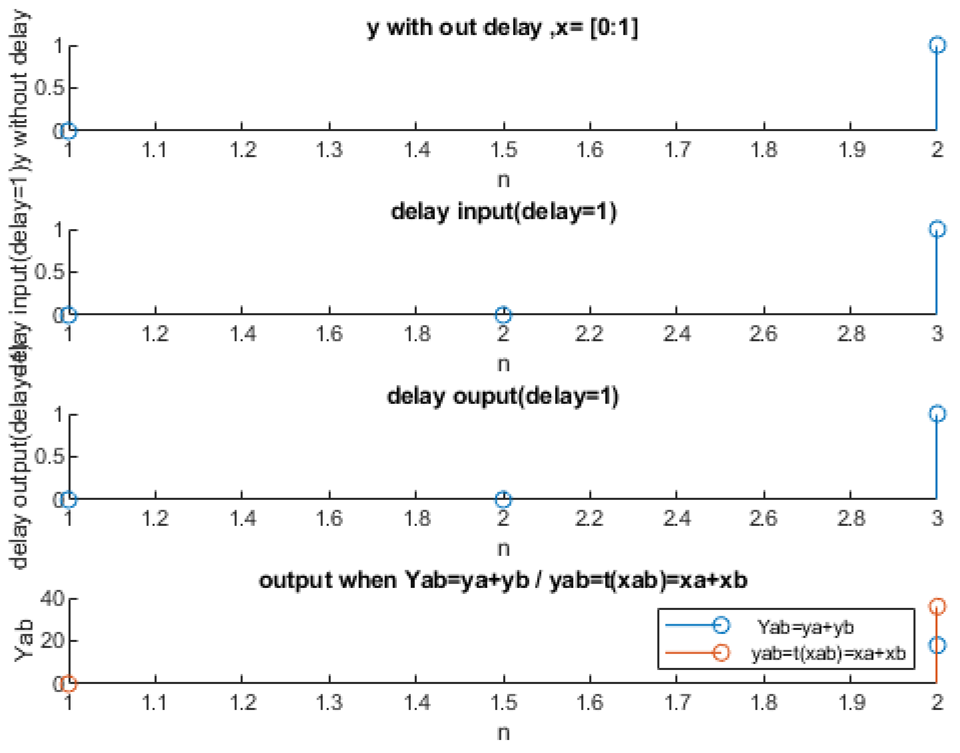
```
subplot(4,1,4)                    %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                       %set label for x
ylabel('Yab')                     %set label for y
p1=stem(1:length(x),Yab);         %plot output Yab when Yab=ya+yb
p2=stem(1:length(x),yab);         %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off



if(y1==y2)                        %compare the output for time invariant
    disp('with x= [0:1],time invariant ')
else

    disp('with x= [0:1],not time invariant ')
end


if(yab==Yab)                      %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end



figure %repeat the steps for x=[1:10];
x= [0:10];                        %define input region[0:10]
y=x.^2;                           %define output
xa=3*x;                           %define input xa
ya=xa.^2;                         %define output ya
xb=3*x;                           %define input xb
yb=xb.^2;                         %define output yb
Yab=ya+yb;                        %define output yab=ya+yb
xab=xa+xb;                        %define input xab=xa+xb
yab=xab.^2;                       %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:10]') %set the tittle
xlabel('n')                       %set label for x
ylabel('y without delay')         %set label for y
stem(1:length(x),y)
hold off
```

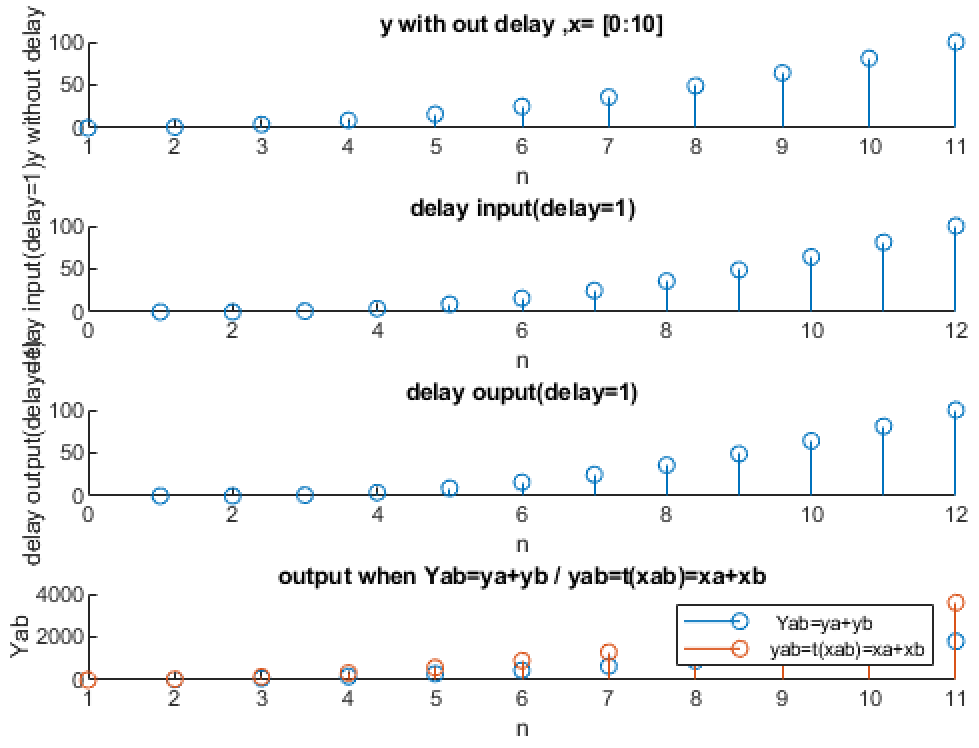
```
subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay input(delay=1)')      %set label for y
x1=[zeros(1:delay) x];              %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
hold on
title('delay ouput(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay output(delay=1)')     %set label for y
y2=[0 x.^2];                        %delay output by 1
stem(1:length(x)+delay,y2)
hold off


subplot(4,1,4)                      %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                         %set label for x
ylabel('Yab')                       %set label for y
p1=stem(1:length(x),Yab)  ;         %plot output Yab when Yab=ya+yb
p2=stem(1:length(x),yab)  ;         %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off



if(y1==y2)                          %compare the output for time invariant
   disp('with x= [0:10],time invariant ')
else

   disp('with x= [0:10],not time invariant ')
end


if(yab==Yab)                        %compare the output for linearty
   disp('Outputs are consistent with a linear system')
else

   disp('System is not linear ')
end
figure %repeat the steps for x==[1:100]
x= [0:100];                         %define input region[0:100]
```

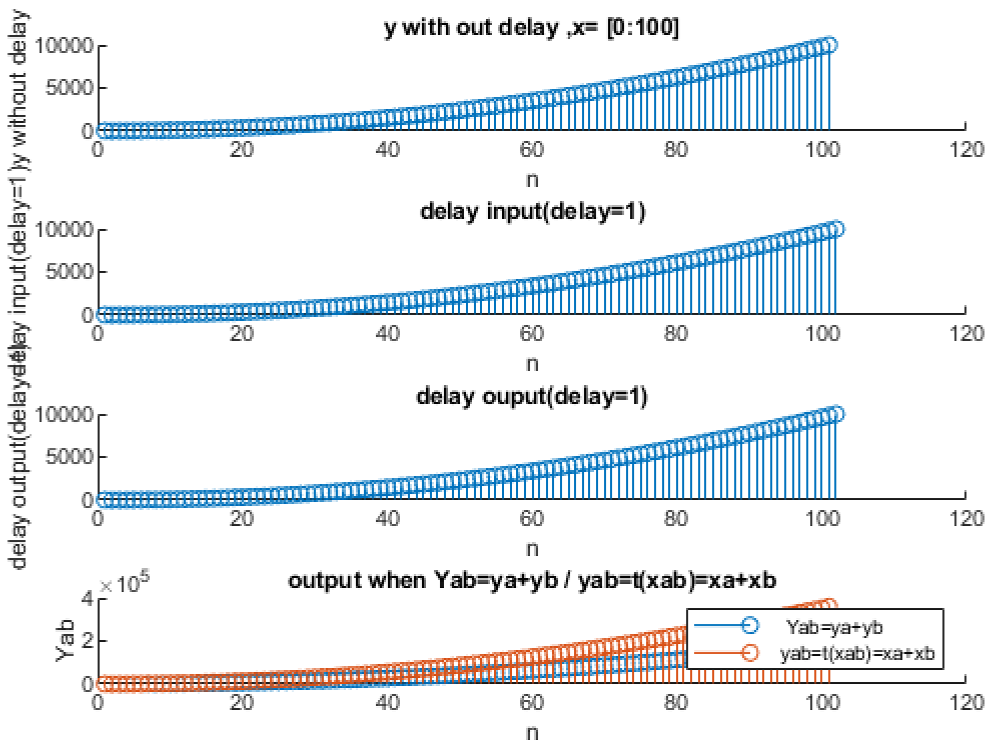
```
y=x.^2;                          %define output
xa=3*x;                          %define input xa
ya=xa.^2;                        %define output ya
xb=3*x;                          %define input xb
yb=xb.^2;                        %define output yb
Yab=ya+yb;                         %define output yab=ya+yb
xab=xa+xb;                        %define input xab=xa+xb
yab=xab.^2;                       %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:100]') %set the tittle
xlabel('n')                      %set label for x
ylabel('y without delay')        %set label for y
stem(1:length(x),y)
hold off

subplot(4,1,2)
hold on
title('delay input(delay=1)')      %set the tittle
xlabel('n')                      %set label for x
ylabel('delay input(delay=1)')     %set label for y
x1=[zeros(1:delay) x];            %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
hold on
title('delay ouput(delay=1)')      %set the tittle
xlabel('n')                      %set label for x
ylabel('delay output(delay=1)')    %set label for y
y2=[0 x.^2];                      %delay output by 1
stem(1:length(x)+delay,y2)
hold off


subplot(4,1,4)                    %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                      %set label for x
ylabel('Yab')                     %set label for y
p1=stem(1:length(x),Yab) ;          %stem output Yab when Yab=ya+yb
p2=stem(1:length(x),yab)  ;         %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off
```

```
if(y1==y2)                    %compare the output for time invariant
   disp('with x= [0:100],time invariant ')
else

   disp('with x= [0:100],not time invariant ')
end



if(yab==Yab)                  %compare the output for linearty
   disp('Outputs are consistent with a linear system')
else

   disp('System is not linear ')
end
```

```
with x= [0:1],time invariant
System is not linear
with x= [0:10],time invariant
System is not linear
with x= [0:100],time invariant
System is not linear
```







Published with MATLAB® R2020a

4.2.2.2 Question2 -a (ii)

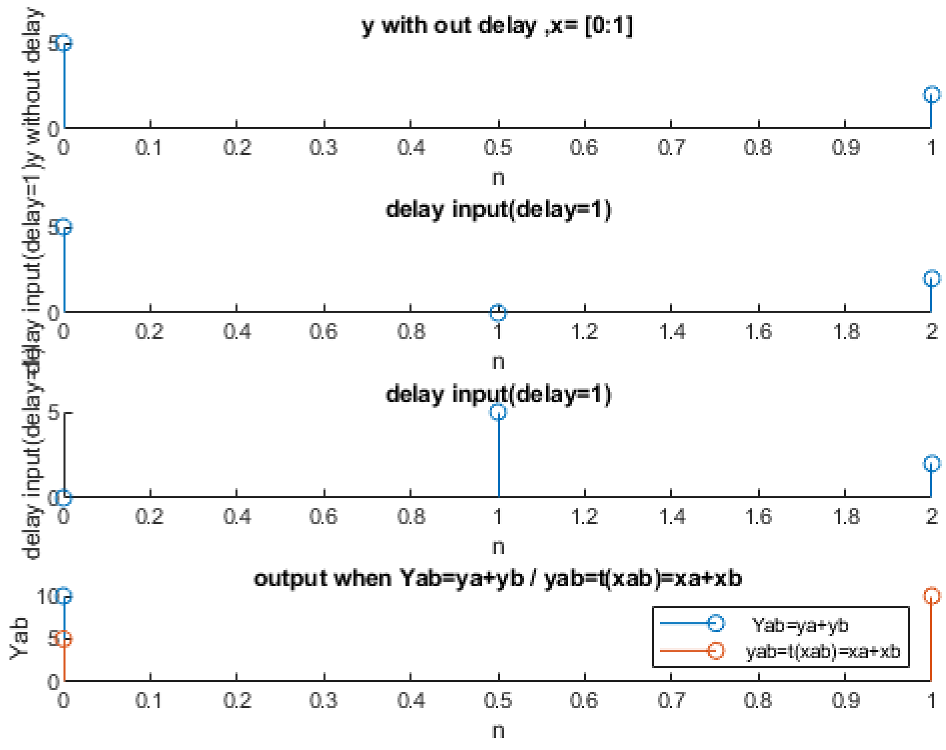
```
%Name:Junpeng Gai
%SID:40009896


delay = 1; %delay
n= [0:1];
impulse = n==0;
                        %define input region[0:1]
x=n;
y=2.*x+5.*impulse;

xa=3*x;                          %define input xa
ya=2.*xa+5.*impulse;;                         %define output ya
xb=2*x;                          %define input xb
yb=2.*xb+5.*impulse;                         %define output yb
Yab=ya+yb;                          %define output yab=ya+yb
xab=xa+xb;                          %define input xab=xa+xb
yab=2.*xab+5.*impulse;                         %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:1]') %set the tittle
xlabel('n')                        %set label for x
ylabel('y without delay')          %set label for y
stem(0:length(n)-1,y)
hold off


subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                        %set label for x
ylabel('delay input(delay=1)')      %set label for y
n1=[zeros(1:delay) n];             %delay =1 for input
x1=n1;
impulse1=[impulse 0];
y1=2.*x1+5.*impulse1;
stem(0:length(x)-1+delay,y1)
hold off



subplot(4,1,3)
hold on
```

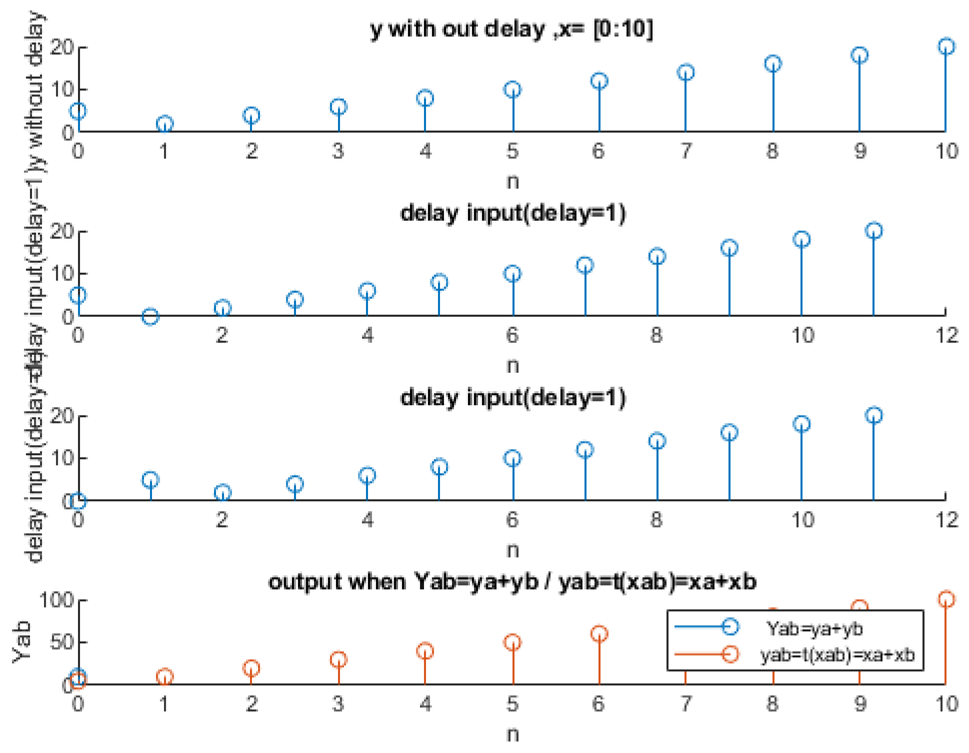
```
title('delay input(delay=1)')      %set the tittle
xlabel('n')                        %set label for x
ylabel('delay input(delay=1)')     %set label for y
y2=[zeros(1,delay) y]
stem(0:length(x)-1+delay,y2)
hold off



subplot(4,1,4)                     %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                        %set label for x
ylabel('Yab')                      %set label for y
p1=stem(0:length(x)-1,Yab);           %plot output Yab when Yab=ya+yb
p2=stem(0:length(x)-1,yab);           %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off

if(y1==y2)                         %compare the output for time invariant
    disp('with x= [0:1],time invariant ')
else

    disp('with x= [0:1],not time invariant ')
end



if(yab==Yab)                       %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end



figure
n= [0:10];
impulse = n==0;
                      %define input region[0:1]
x=n;
y=2.*x+5.*impulse;

xa=3*x;                            %define input xa
ya=2.*xa+5.*impulse;;                      %define output ya
xb=2*x;                            %define input xb
yb=2.*xb+5.*impulse;                       %define output yb
Yab=ya+yb;                         %define output yab=ya+yb
xab=xa+xb;                         %define input xab=xa+xb
yab=2.*xab+5.*impulse;                     %define output yab=T(xab)
```

```
subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:10]') %set the tittle
xlabel('n')                        %set label for x
ylabel('y without delay')          %set label for y
stem(0:length(n)-1,y)
hold off


subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                        %set label for x
ylabel('delay input(delay=1)')      %set label for y
n1=[zeros(1:delay) n];             %delay =1 for input
x1=n1;
impulse1=[impulse 0];
y1=2.*x1+5.*impulse1;
stem(0:length(x)-1+delay,y1)
hold off



subplot(4,1,3)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                        %set label for x
ylabel('delay input(delay=1)')      %set label for y
y2=[zeros(1,delay) y]
stem(0:length(x)-1+delay,y2)
hold off


subplot(4,1,4)                     %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                        %set label for x
ylabel('Yab')                      %set label for y
p1=stem(0:length(x)-1,Yab);           %plot output Yab when Yab=ya+yb
p2=stem(0:length(x)-1,yab);           %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off

if(y1==y2)                         %compare the output for time invariant
    disp('with x= [0:10],time invariant ')
else

    disp('with x= [0:10],not time invariant ')
end
```

```
if(yab==Yab)                          %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end
```

```
y2 =

     0     5     2

with x= [0:1],not time invariant
System is not linear

y2 =

     0     5     2     4     6     8    10    12    14    16    18    20

with x= [0:10],not time invariant
System is not linear
```





Published with MATLAB® R2020a

4.3 Question 3

Question 3:

(a) Plot the following signal x[n], it's mirror image x[-n], and it's even and odd components:

$x[n] = e^{-2|n|}\sin((2*pi/36)n)$, $0 <= n <= 10$

Use the MATLAB functions `exp` and `abs` .

(b) Repeat part (a) for the signal:

$x[n] = (-1)^{n-1}$, $-5 <= n <= 5$

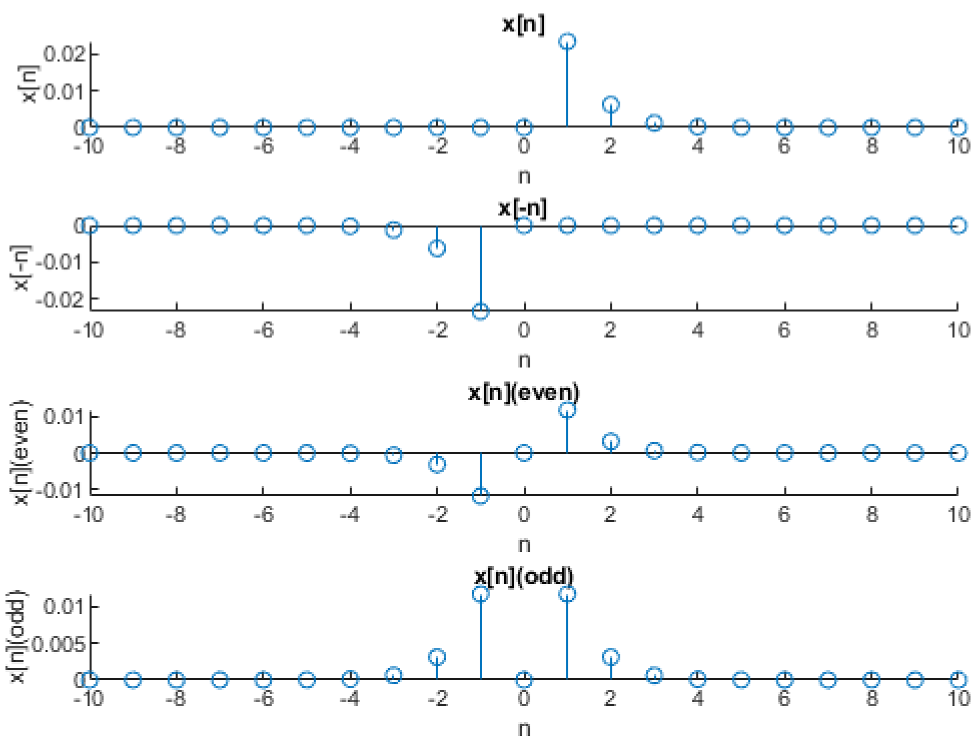
(c) Compare and contrast the two methods used to generate the two MATLAB arrays x1 and x2 in the following MATLAB code:

Figure 17 Question3

4.3.1 Question3 -a

```
%Name:Junpeng Gai
%SID:40009896
```

```
n=-10:10;
% initialize all elements to zero.
x1=zeros(1,21); % define length of x1 -10:10 all together are 21

x2=zeros(1,21);  % define length of x2 -10:10 all together are 21

for i=0:10
    x1(i+11)=exp(-2.*abs(i)).*sin( (2*pi/36).*i);
    %for 0:10 which are the 11th to 21st element in this array;
    x2(11-i)=exp(-2.*abs(-i)).*sin( (2*pi/36).*(-i));
    %for -10:0 which are the 1st to 11th element in this array;
end

x3=(x1+x2)/2; %get even component from above array=(1:21) correspoending to n=-10:10
x4=(x1-x2)/2 ;  %get even component from above array=(1:21) correspoending to n=-10:10
subplot(4,1,1)   %plot the x[n]
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,x1)
hold off
subplot(4,1,2)  %plot the x[-n]
hold on
title('x[-n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[-n]')  %set label for y
stem(n,x2)
hold off
subplot(4,1,3)  %plot the x[n](even)
hold on
title('x[n](even)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](even)')  %set label for y
stem(n,x3)
hold off
subplot(4,1,4)  %plot the x[n](odd)
hold on
title('x[n](odd)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](odd)')  %set label for y
stem(n,x4)
hold off
```

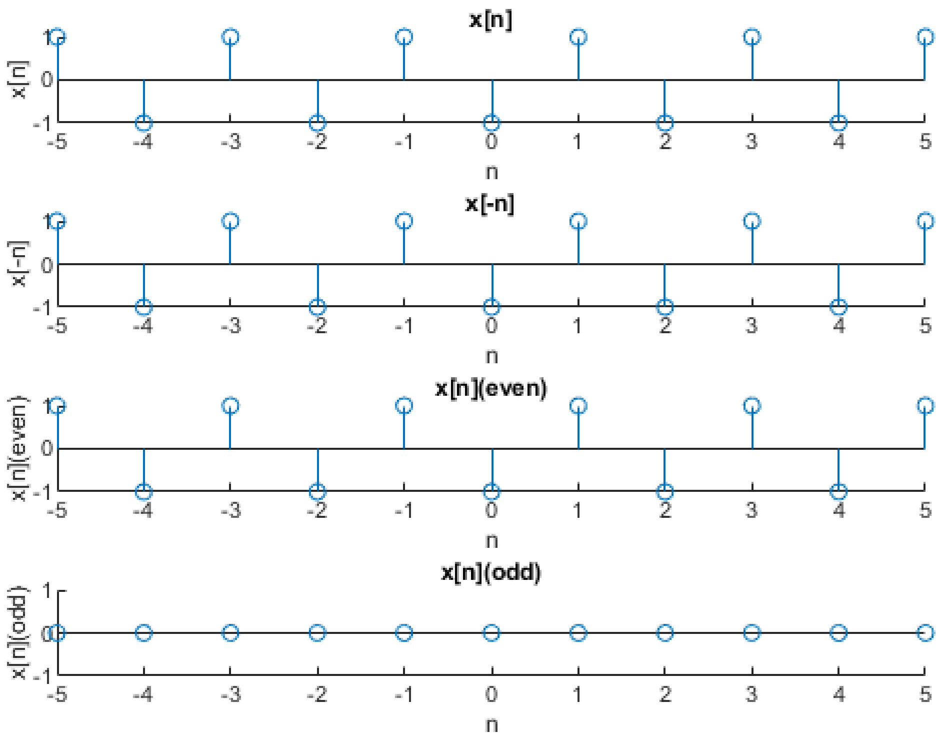


Published with MATLAB® R2020a

4.3.2 Question3 -b

```
%Name:Junpeng Gai
%SID:40009896

n=-5:5;

x1=(-1).^(n-1); % define x[n] for -5:5

x2=(-1).^(-n-1);  % define x[-n] for -5:5



x3=(x1+x2)/2; %get even component from above
x4=(x1-x2)/2 ;  %get even component from above
subplot(4,1,1)   %plot the x[n]
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,x1)
hold off
subplot(4,1,2)  %plot the x[-n]
hold on
```

```
title('x[-n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[-n]')  %set label for y
stem(n,x2)
hold off
subplot(4,1,3)  %plot the x[n](even)
hold on
title('x[n](even)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](even)')  %set label for y
stem(n,x3)
hold off
subplot(4,1,4)  %plot the x[n](odd)
hold on
title('x[n](odd)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](odd)')  %set label for y
stem(n,x4)
hold off
```



Published with MATLAB® R2020a

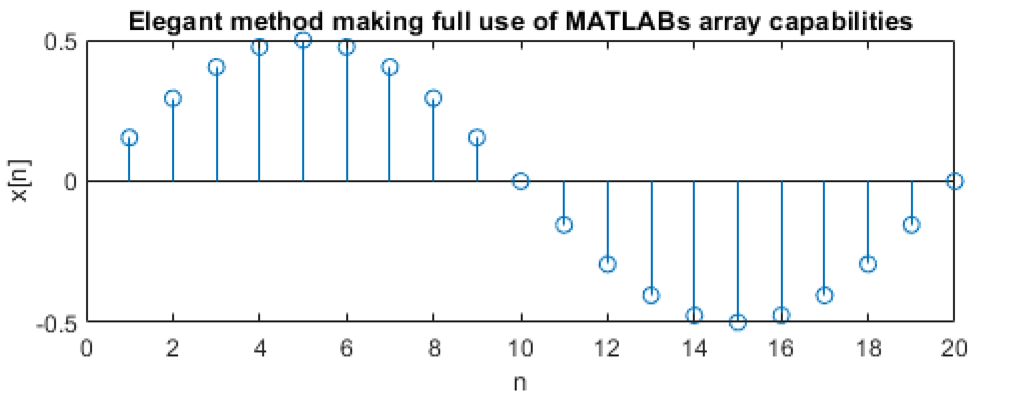
4.3.3 Question3 -c

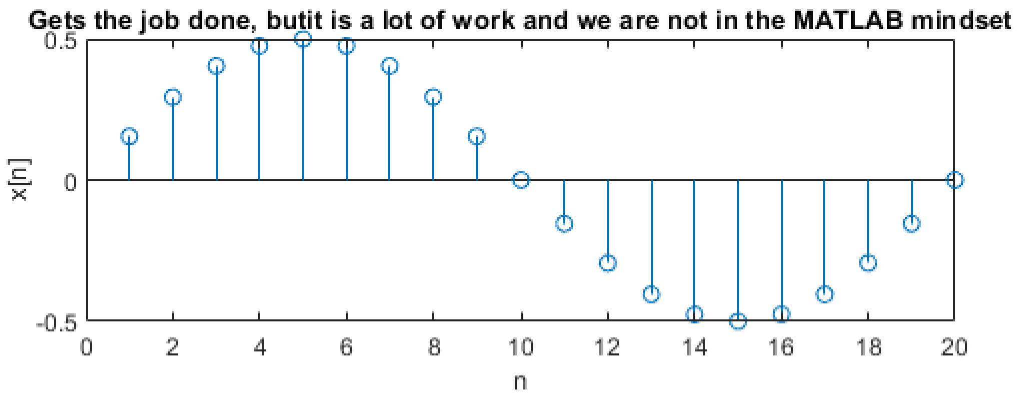
I am using the for loop to assign the value to the signal in question 3.a, because my domine is from [0,n] . This question ask us”compare and contrast”.

I used function to assign data in question3-b, But in question question3-a I used the array.

The answer both method will work , but the comcept in matlab is array based , so I can use y=f(x) to assign data to y instead of using for loop which is reduntent.

```
%Name:Junpeng Gai
%SID:40009896
% T. Obuchowicz
%Fri Apr 27 16:03:27 EDT 2012
clear
n = [1 : 20 ];
x1 = sin((2*pi/40) * n) .* cos((2*pi/40) * n);
for index = 1 : 20
% Note: In MATLAB, no need to pre-allocate the array,
% unlike C++ and other high-level programming languages.
 x2(index) = sin((2*pi/40) * index) * cos((2*pi/40) * index);
end
subplot(2,1,1)
stem(n, x1)
title('Elegant method making full use of MATLABs array capabilities')
xlabel('n')
ylabel('x[n]')
subplot(2,1,2)
stem(n, x2)
title('Gets the job done, butit is a lot of work and we are not in the MATLAB mindset')
xlabel('n')
ylabel('x[n]')
```

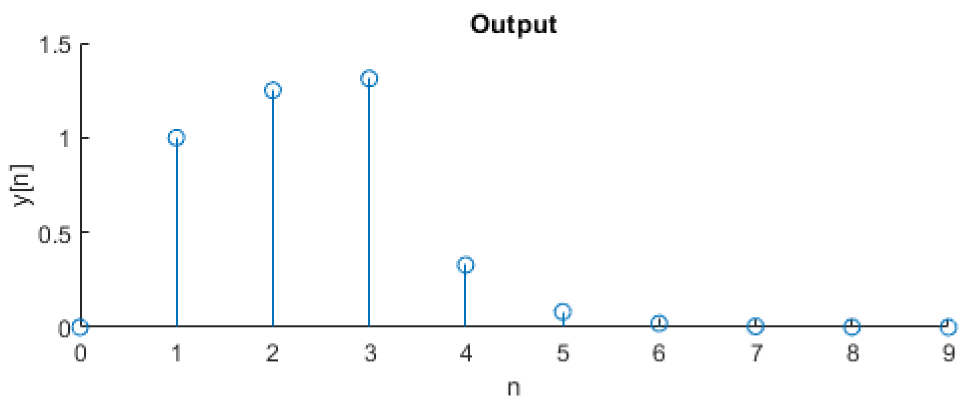




Published with MATLAB® R2020a

4.4 Part 2------Question 1

```
%Name:Junpeng Gai
%SID:40009896
n=0:9;
x=[0 1 1 1 0 0 0 0 0 0 ];
y=[0 0 0 0 0 0 0 0 0 0 ];
H=[1 0.25 0.25^2 0.25^3 0.25^4 0.25^5 0.25^6 0.25^7 0.25^8 0.25^9 ];
y(1)=x(1);%base case
for i =2:9
  y(i)=x(i)+0.25*((y(i-1))); %for loop to do the convolution
end
subplot(2,1,1)
hold on
title('Output')
xlabel('n')
ylabel('y[n]')
stem(0:length(y)-1,y);%plot the graph
hold off
```

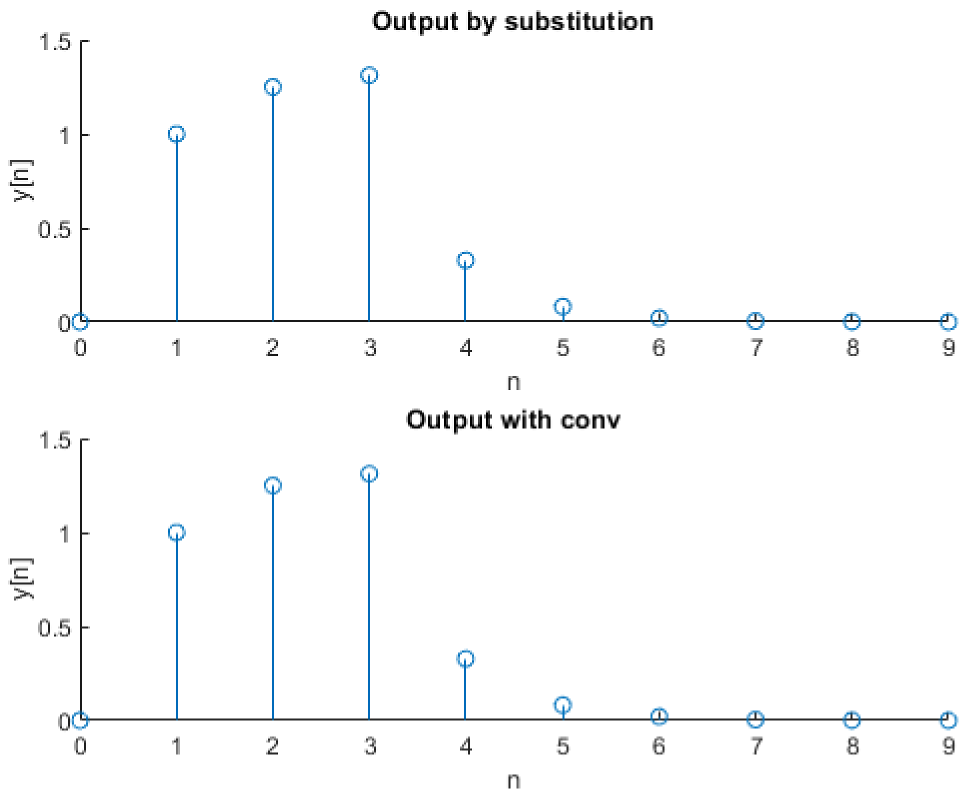


Published with MATLAB® R2020a

4.5 Part 2------Question 2

```
%Name:Junpeng Gai
%SID:40009896
n=0:9;
x=[0 1 1 1 0 0 0 0 0 0 ];
y=[0 0 0 0 0 0 0 0 0 0 ];
H=[1 0.25 0.25^2 0.25^3 0.25^4 0.25^5 0.25^6 0.25^7 0.25^8 0.25^9 ];
y(1)=x(1);%base case
for i =2:9
   y(i)=x(i)+0.25*((y(i-1))); %for loop to do the convolution
end
subplot(2,1,1)
hold on
title('Output by substitution')
xlabel('n')
ylabel('y[n]')
stem(0:length(y)-1,y);%plot the graph
hold off

c = conv(x, H);%convolute the x and h
C=[0 0 0 0 0 0 0 0 0 0 ];%cause we only need first 10 data
for i =1:10
   C(i)=c(i); %get the 1st 10 data
```

```
end
subplot(2,1,2)
hold on
title('Output with conv')
xlabel('n')
ylabel('y[n]')
stem(0:length(C)-1,C);
hold off
```



Published with MATLAB® R2020a

4.6 Part 2------Question 3

4.6.1 Is the system linear

4.6.1.1 i

I will choose signal like X[k]=k, because this kind of signal is not periodic and it’s linear itself.

Here is my design:

1. put the signal x1[k]=k and x2[k]=2k into the system.
2. Get the output y1[k]= $System$ (x1[k]) and y2[k]= $System$ (x2[k]).
3. Because input signal is linear so we can put x3[k] =3k into the system.
4. Get the output of y3[k]= $System$ (x3[k]).

5. Compare the results, if y3[k]=y1[k]+y2[k]?

Therefore, I will expect a linear output from the black box I was given.

$$x[k] = k$$

$$y = System(x)$$

If at the same time I add another input $x2[k] = 2k$ into the system I should observe the amplitude of the output becomes 3 times of the previous amplitude.

4.6.1.2 ii

The output should be a superposition of 2 outputs of 2 separate inputs:

Y3= $System$ (x1[k])+$System$ (x2[k]).

4.6.1.3 iii

Since I don’t have a system in my hand, I would say that

Like in question 2 in this lab, if Y3=Y4 then theoretically the system is linear.

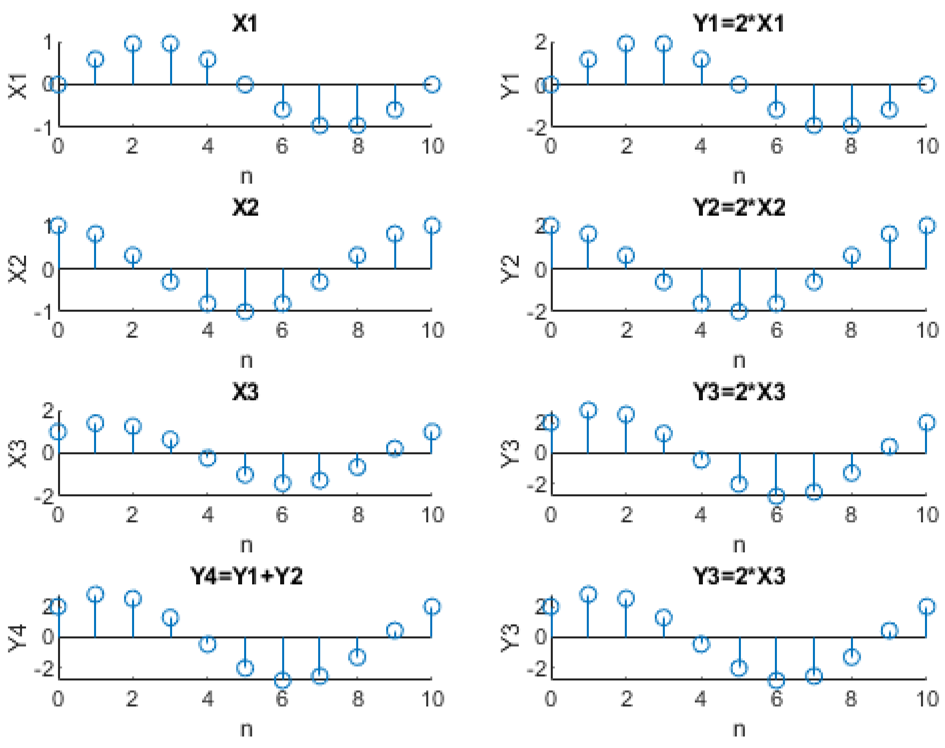


Figure 18 output from question 2

If it's like what we did in question 2.b

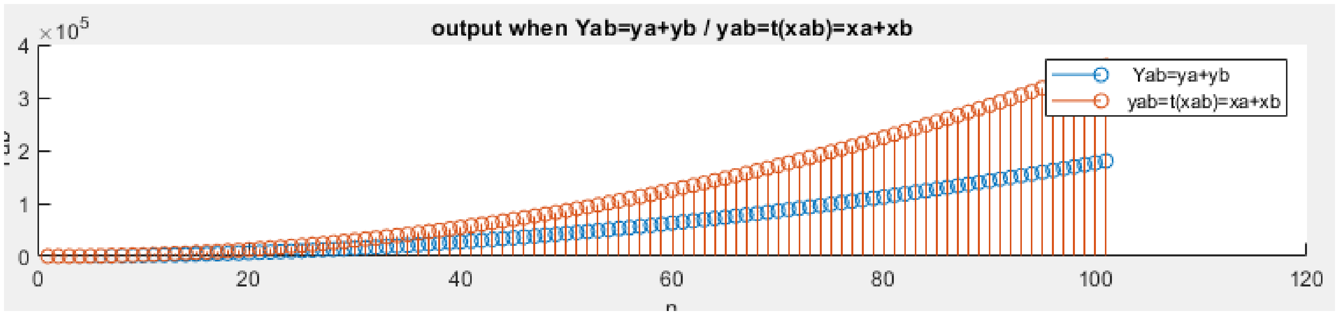


Figure 19 outputs from question2.b

So this system will not be linear since yab!=Yab;

4.6.2 Is the system time invariant

4.6.2.1 i

I will choose signal like X[k]=k, because this kind of signal is not periodic and it's linear itself.

Here is my design:

1. put the signal x1[k]=k into the system.

2. Get the output y[k]= $System$ (x[k]).

3. Delay the input by an amount named delay (let's say 1) x1[k].

4. Get the output of y1[k]= $System$ (x[k-delay]) = $System$ (x1[k]).

5. Delay the output y2[k]=y[k-delay];

6. Compare the 2 outputs if y2[k]= y1[k].

Therefore, I will expect 2 outputs are identical and they are shifted by delay from y[k].

4.6.2.2 ii

The output should be a shifted version of original output y[k].

4.6.2.3 iii

Since I don't have a system in my hand, I would say that

Like in question 2 in this lab, this can prof the system is time invariant, because the delay input and delay output are the same.

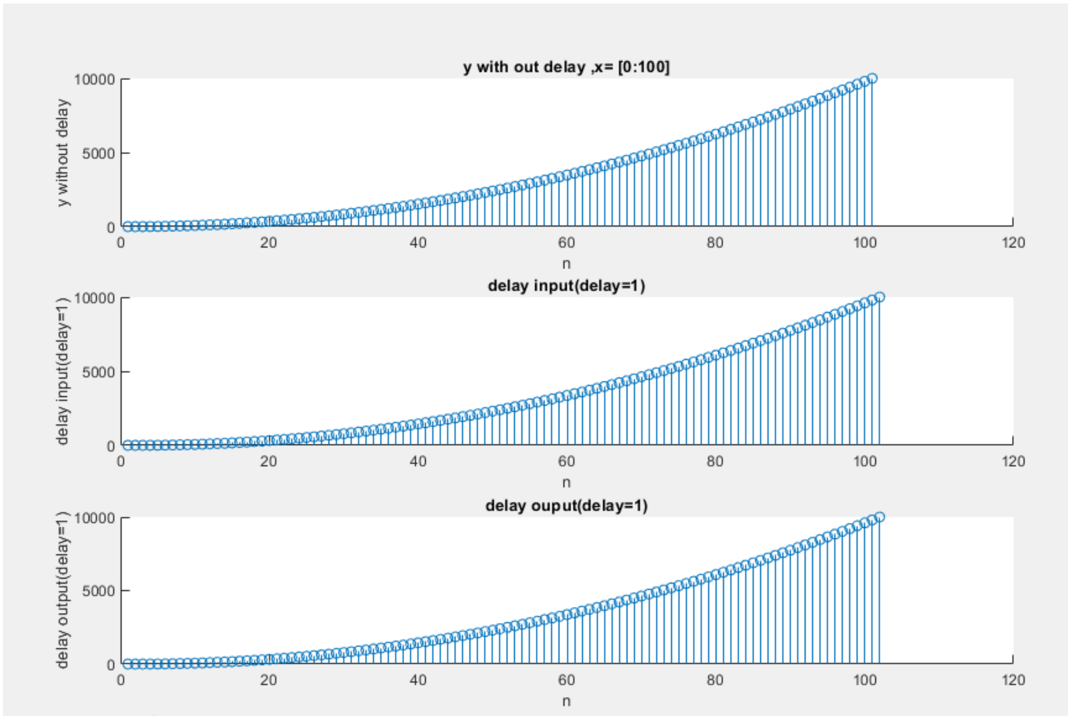


Figure 20 output from question 2

If it's like what we did in question 2.b

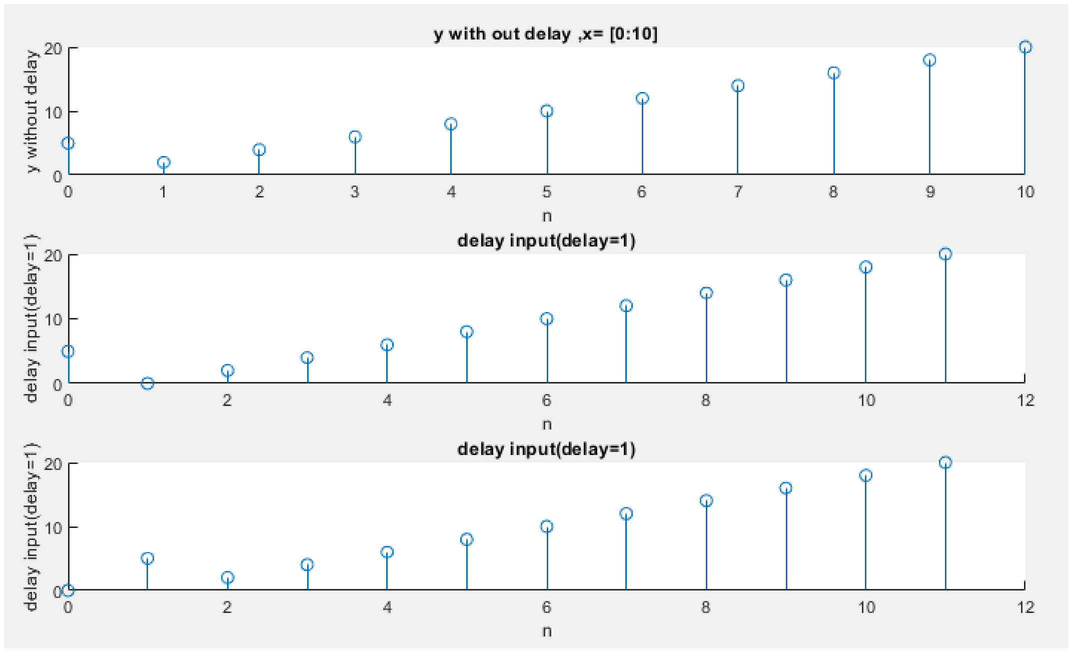


Figure 21 outputs from question2.b

So, this system will not be time invariant because the delay output and delay input are not the same.

5.Conclusions

Through the study of this chapter, we know how to use conditional statements. At the same time, we also learned how to prove that a signal system is linear, time invariant and how to calculate convolution.

In the question 1, we learned how to use MATLAB to prove that the system is a linear system after bringing in two different input signals.

In question 2, we learned how to use the even and odd component of the function signal that MATLAB. At the same time, in the second part of the second question, we also learned how to represent impulse in MATLAB and how to determine whether a signal system is time invariant.

In the third question, I learned that by using for loop to assignment value to output and using function to assign value to the output. We should prefer to use function to assign value, because this is the feature of MATLAB.

In the second part, we learned that y[n] = x[n] + (1/4) * y[n-1] For the given difference equation we can use for loop in MATLAB to calculate the output y[k]. But there is no initial condition in this question, so I default to y [-1] =0. Using recursion, I calculated the answer. At the same time, compared with the function given by the conv function, we found that these two values are the same.

For the second question in the second part, since we did not give a signal system, I wrote down the experiment I designed, and at the same time listed and analyzed the possible results separately.

6.Appendix

Question1 -a

```
%Name:Junpeng Gai
%SID:40009896

n=(0:9);%defin the domain
X=n;    %defin x[n]
Y=X.^2; %defin y[n]
Energy_x=0; %initialize the energy for x
Energy_y=0; %initialize the energy for y
for index = X % loop in X array
 Energy_x = Energy_x + index.^2; % sum to Energy_x
end

disp('The Energy of x[n] is')
disp(Energy_x)  %calculate the total energy

for index = Y % % loop in Y array
 Energy_y = Energy_y + index.^2; % sum to Energy_y
end

disp('The Energy of y[n] is')
disp(Energy_y)  %calculate the total energy

subplot(2,1,1) %first graph
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,X);
hold off
subplot(2,1,2)  %second graph
hold on
title('y[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('y=x[n]^2')%set label for y
stem(n,Y);
hold off
```

Question1 -b

```
%Name:Junpeng Gai
%SID:40009896

n=(0:9);%defin the domain
X= sin( (2*pi)/10 * n);    %defin x[n]
Y=X.^2; %defin y[n]
Energy_x=0; %initialize the energy for x
Energy_y=0; %initialize the energy for y
for index = X % loop in X array
 Energy_x = Energy_x + index.^2; % sum to Energy_x
end

disp('The Energy of x[n] is')
disp(Energy_x)  %calculate the total energy

for index = Y % % loop in Y array
 Energy_y = Energy_y + index.^2; % sum to Energy_y
end

disp('The Energy of y[n] is')
disp(Energy_y)  %calculate the total energy

subplot(2,1,1) %first graph
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,X);
hold off
subplot(2,1,2)  %second graph
hold on
title('y[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('y=x[n]^2')%set label for y
stem(n,Y);
hold off
```

Question2 -a

```
%Name:Junpeng Gai
%SID:40009896
```

```
n=(0:10);%defin the domain
X1=sin( (2*pi /10 ) * n ); %defin X1
X2=cos( (2*pi /10 ) * n ); %defin X2
X3=X1+X2;                %defin X3 is the linear combination fo X1 and X2
Y1=2*X1;                 %defin OUTPUT for X1
Y2=2*X2;                 %defin OUTPUT for X2
Y3=2*X3;                  %defin OUTPUT for X3
Y4=Y1+Y2;                 %defin linear combination of Y1 and Y2

if ( Y4 == Y3 )   %condition,if success display it's a lineat sys
disp( 'Outputs are consistent with a linear system')
else            %else it isn't.
disp( 'System is not linear')
end

subplot(4,2,1);
hold on
title('X1')   %set the tittle
xlabel('n')     %set label for x
ylabel('X1')  %set label for y
stem(n,X1);     %plot input X1
hold off
subplot(4,2,2);
hold on
title('Y1=2*X1')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y1')  %set label for y
stem(n,Y1);     %plot output Y1
hold off
subplot(4,2,3);
hold on
title('X2')   %set the tittle
xlabel('n')     %set label for x
ylabel('X2')  %set label for y
stem(n,X2);     %plot input X2
hold off
subplot(4,2,4);
hold on
title('Y2=2*X2')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y2')  %set label for y
stem(n,Y2);     %plot output Y2
hold off
subplot(4,2,5);
hold on
title('X3')   %set the tittle
xlabel('n')     %set label for x
ylabel('X3')  %set label for y
stem(n,X3);     %plot input X3
```

```
hold off
subplot(4,2,6);
hold on
title('Y3=2*X3')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y3')  %set label for y
stem(n,Y3);     %plot output Y3
hold off
subplot(4,2,7);
hold on
title('Y4=Y1+Y2')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y4')  %set label for y
stem(n,Y4);     %plot output Y4,which is the left hand side
hold off
subplot(4,2,8);
hold on
title('Y3=2*X3')   %set the tittle
xlabel('n')     %set label for x
ylabel('Y3')  %set label for y
stem(n,Y3);     %plot output Y3,which is the right hand side
hold off
```

Outputs are consistent with a linear system

Question2 -b (i)

```
%Name:Junpeng Gai
%SID:40009896



delay = 1; %delay



x= [0:1];                        %define input region[0:1]
y=x.^2;                          %define output
xa=3*x;                           %define input xa
ya=xa.^2;                          %define output ya
xb=3*x;                           %define input xb
yb=xb.^2;                          %define output yb
Yab=ya+yb;                          %define output yab=ya+yb
xab=xa+xb;                         %define input xab=xa+xb
yab=xab.^2;                        %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:1]') %set the tittle
xlabel('n')                      %set label for x
ylabel('y without delay')          %set label for y
```

```
stem(1:length(x),y)
hold off

subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay input(delay=1)')      %set label for y
x1=[zeros(1:delay) x];              %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
hold on
title('delay ouput(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay output(delay=1)')     %set label for y
y2=[0 x.^2];                        %delay output by 1
stem(1:length(x)+delay,y2)
hold off


subplot(4,1,4)                      %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                         %set label for x
ylabel('Yab')                       %set label for y
p1=stem(1:length(x),Yab);           %plot output Yab when Yab=ya+yb
p2=stem(1:length(x),yab);           %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off



if(y1==y2)                          %compare the output for time invariant
    disp('with x= [0:1],time invariant ')
else

    disp('with x= [0:1],not time invariant ')
end


if(yab==Yab)                        %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end
```

```
figure %repeat the steps for x=[1:10];
x= [0:10];                          %define input region[0:10]
y=x.^2;                             %define output
xa=3*x;                             %define input xa
ya=xa.^2;                           %define output ya
xb=3*x;                             %define input xb
yb=xb.^2;                           %define output yb
Yab=ya+yb;                          %define output yab=ya+yb
xab=xa+xb;                          %define input xab=xa+xb
yab=xab.^2;                         %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:10]') %set the tittle
xlabel('n')                         %set label for x
ylabel('y without delay')           %set label for y
stem(1:length(x),y)
hold off

subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay input(delay=1)')      %set label for y
x1=[zeros(1:delay) x];              %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
hold on
title('delay ouput(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay output(delay=1)')     %set label for y
y2=[0 x.^2];                        %delay output by 1
stem(1:length(x)+delay,y2)
hold off


subplot(4,1,4)                      %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                         %set label for x
ylabel('Yab')                       %set label for y
p1=stem(1:length(x),Yab)  ;         %plot output Yab when Yab=ya+yb
p2=stem(1:length(x),yab)  ;         %plot output yab when yab=t(xab)=xa+xb
```

```
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off




if(y1==y2)                          %compare the output for time invariant
    disp('with x= [0:10],time invariant ')
else

    disp('with x= [0:10],not time invariant ')
end



if(yab==Yab)                          %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end
figure %repeat the steps for x==[1:100]
x= [0:100];                          %define input region[0:100]
y=x.^2;                              %define output
xa=3*x;                              %define input xa
ya=xa.^2;                            %define output ya
xb=3*x;                              %define input xb
yb=xb.^2;                            %define output yb
Yab=ya+yb;                           %define output yab=ya+yb
xab=xa+xb;                           %define input xab=xa+xb
yab=xab.^2;                          %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:100]') %set the tittle
xlabel('n')                          %set label for x
ylabel('y without delay')            %set label for y
stem(1:length(x),y)
hold off

subplot(4,1,2)
hold on
title('delay input(delay=1)')        %set the tittle
xlabel('n')                          %set label for x
ylabel('delay input(delay=1)')       %set label for y
x1=[zeros(1:delay) x];               %delay =1 for input
y1=x1.^2;
stem(1:length(x)+delay,y1)
hold off

subplot(4,1,3)
```

```
hold on
title('delay ouput(delay=1)')       %set the tittle
xlabel('n')                         %set label for x
ylabel('delay output(delay=1)')     %set label for y
y2=[0 x.^2];                        %delay output by 1
stem(1:length(x)+delay,y2)
hold off


subplot(4,1,4)                      %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                         %set label for x
ylabel('Yab')                       %set label for y
p1=stem(1:length(x),Yab) ;          %stem output Yab when Yab=ya+yb
p2=stem(1:length(x),yab)  ;         %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off



if(y1==y2)                          %compare the output for time invariant
    disp('with x= [0:100],time invariant ')
else

    disp('with x= [0:100],not time invariant ')
end


if(yab==Yab)                        %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end
```

Question2 -a (ii)

```
%Name:Junpeng Gai
%SID:40009896


delay = 1; %delay
n= [0:1];
impulse = n==0;
```

```
                    %define input region[0:1]
x=n;
y=2.*x+5.*impulse;

xa=3*x;                    %define input xa
ya=2.*xa+5.*impulse;;                %define output ya
xb=2*x;                    %define input xb
yb=2.*xb+5.*impulse;               %define output yb
Yab=ya+yb;                 %define output yab=ya+yb
xab=xa+xb;                 %define input xab=xa+xb
yab=2.*xab+5.*impulse;               %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:1]') %set the tittle
xlabel('n')                %set label for x
ylabel('y without delay')      %set label for y
stem(0:length(n)-1,y)
hold off


subplot(4,1,2)
hold on
title('delay input(delay=1)')    %set the tittle
xlabel('n')                %set label for x
ylabel('delay input(delay=1)')    %set label for y
n1=[zeros(1:delay) n];        %delay =1 for input
x1=n1;
impulse1=[impulse 0];
y1=2.*x1+5.*impulse1;
stem(0:length(x)-1+delay,y1)
hold off



subplot(4,1,3)
hold on
title('delay input(delay=1)')    %set the tittle
xlabel('n')                %set label for x
ylabel('delay input(delay=1)')    %set label for y
y2=[zeros(1,delay) y]
stem(0:length(x)-1+delay,y2)
hold off


subplot(4,1,4)               %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                %set label for x
```

```
ylabel('Yab')                      %set label for y
p1=stem(0:length(x)-1,Yab);           %plot output Yab when Yab=ya+yb
p2=stem(0:length(x)-1,yab);           %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off

if(y1==y2)                         %compare the output for time invariant
    disp('with x= [0:1],time invariant ')
else

    disp('with x= [0:1],not time invariant ')
end



if(yab==Yab)                        %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end



figure
n= [0:10];
impulse = n==0;
                       %define input region[0:1]
x=n;
y=2.*x+5.*impulse;

xa=3*x;                            %define input xa
ya=2.*xa+5.*impulse;;                         %define output ya
xb=2*x;                            %define input xb
yb=2.*xb+5.*impulse;                        %define output yb
Yab=ya+yb;                         %define output yab=ya+yb
xab=xa+xb;                         %define input xab=xa+xb
yab=2.*xab+5.*impulse;                       %define output yab=T(xab)

subplot(4,1,1)  %origional plot
hold on
title('y with out delay ,x= [0:10]') %set the tittle
xlabel('n')                        %set label for x
ylabel('y without delay')          %set label for y
stem(0:length(n)-1,y)
hold off



subplot(4,1,2)
hold on
title('delay input(delay=1)')       %set the tittle
```

```
xlabel('n')                  %set label for x
ylabel('delay input(delay=1)')     %set label for y
n1=[zeros(1:delay) n];          %delay =1 for input
x1=n1;
impulse1=[impulse 0];
y1=2.*x1+5.*impulse1;
stem(0:length(x)-1+delay,y1)
hold off




subplot(4,1,3)
hold on
title('delay input(delay=1)')      %set the tittle
xlabel('n')                  %set label for x
ylabel('delay input(delay=1)')     %set label for y
y2=[zeros(1,delay) y]
stem(0:length(x)-1+delay,y2)
hold off


subplot(4,1,4)                 %origional plot
hold on
title('output when Yab=ya+yb / yab=t(xab)=xa+xb')   %set the tittle
xlabel('n')                  %set label for x
ylabel('Yab')                  %set label for y
p1=stem(0:length(x)-1,Yab);          %plot output Yab when Yab=ya+yb
p2=stem(0:length(x)-1,yab);          %plot output yab when yab=t(xab)=xa+xb
legend([p1 p2],' Yab=ya+yb','yab=t(xab)=xa+xb')
hold off

if(y1==y2)                     %compare the output for time invariant
    disp('with x= [0:10],time invariant ')
else

    disp('with x= [0:10],not time invariant ')
end



if(yab==Yab)                     %compare the output for linearty
    disp('Outputs are consistent with a linear system')
else

    disp('System is not linear ')
end
```

Question3 -a

```
%Name:Junpeng Gai
%SID:40009896

n=-10:10;
% initialize all elements to zero.
x1=zeros(1,21); % define length of x1 -10:10 all together are 21

x2=zeros(1,21);  % define length of x2 -10:10 all together are 21

for i=0:10
    x1(i+11)=exp(-2.*abs(i)).*sin( (2*pi/36).*i);
    %for 0:10 which are the 11th to 21st element in this array;
    x2(11-i)=exp(-2.*abs(-i)).*sin( (2*pi/36).*(-i));
    %for -10:0 which are the 1st to 11th element in this array;
end

x3=(x1+x2)/2; %get even component from above array=(1:21) correspoending to n=-10:10
x4=(x1-x2)/2 ;  %get even component from above array=(1:21) correspoending to n=-10:10
subplot(4,1,1)   %plot the x[n]
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,x1)
hold off
subplot(4,1,2)  %plot the x[-n]
hold on
title('x[-n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[-n]')  %set label for y
stem(n,x2)
hold off
subplot(4,1,3)  %plot the x[n](even)
hold on
title('x[n](even)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](even)')  %set label for y
stem(n,x3)
hold off
subplot(4,1,4)  %plot the x[n](odd)
hold on
title('x[n](odd)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](odd)')  %set label for y
```

```
stem(n,x4)
hold off
```

Question3 -b

```
%Name:Junpeng Gai
%SID:40009896

n=-5:5;

x1=(-1).^(n-1); % define x[n] for -5:5

x2=(-1).^(-n-1);  % define x[-n] for -5:5



x3=(x1+x2)/2; %get even component from above
x4=(x1-x2)/2 ;  %get even component from above
subplot(4,1,1)   %plot the x[n]
hold on
title('x[n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n]')  %set label for y
stem(n,x1)
hold off
subplot(4,1,2)  %plot the x[-n]
hold on
title('x[-n]')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[-n]')  %set label for y
stem(n,x2)
hold off
subplot(4,1,3)  %plot the x[n](even)
hold on
title('x[n](even)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](even)')  %set label for y
stem(n,x3)
hold off
subplot(4,1,4)  %plot the x[n](odd)
hold on
title('x[n](odd)')   %set the tittle
xlabel('n')     %set label for x
ylabel('x[n](odd)')  %set label for y
stem(n,x4)
hold off
```

Question3 -c

```
%Name:Junpeng Gai
%SID:40009896
% T. Obuchowicz
%Fri Apr 27 16:03:27 EDT 2012
clear
n = [1 : 20 ];
x1 = sin((2*pi/40) * n) .* cos((2*pi/40) * n);
for index = 1 : 20
% Note: In MATLAB, no need to pre-allocate the array,
% unlike C++ and other high-level programming languages.
 x2(index) = sin((2*pi/40) * index) * cos((2*pi/40) * index);
end
subplot(2,1,1)
stem(n, x1)
title('Elegant method making full use of MATLABs array capabilities')
xlabel('n')
ylabel('x[n]')
subplot(2,1,2)
stem(n, x2)
title('Gets the job done, butit is a lot of work and we are not in the MATLAB mindset')
xlabel('n')
ylabel('x[n]')
```

Part 2------Question 1

```
%Name:Junpeng Gai
%SID:40009896
n=0:9;
x=[0 1 1 1 0 0 0 0 0 0 ];
y=[0 0 0 0 0 0 0 0 0 0 ];
H=[1 0.25 0.25^2 0.25^3 0.25^4 0.25^5 0.25^6 0.25^7 0.25^8 0.25^9 ];
y(1)=x(1);%base case
for i =2:9
  y(i)=x(i)+0.25*((y(i-1))); %for loop to do the convolution
end
subplot(2,1,1)
hold on
title('Output')
xlabel('n')
ylabel('y[n]')
stem(0:length(y)-1,y);%plot the graph
hold off
```

Part 2------Question 2

```
%Name:Junpeng Gai
%SID:40009896
n=0:9;
x=[0 1 1 1 0 0 0 0 0 0 ];
y=[0 0 0 0 0 0 0 0 0 0 ];
H=[1 0.25 0.25^2 0.25^3 0.25^4 0.25^5 0.25^6 0.25^7 0.25^8 0.25^9 ];
y(1)=x(1);%base case
for i =2:9
  y(i)=x(i)+0.25*((y(i-1))); %for loop to do the convolution
end
subplot(2,1,1)
hold on
title('Output by substitution')
xlabel('n')
ylabel('y[n]')
stem(0:length(y)-1,y);%plot the graph
hold off

c = conv(x, H);%convolute the x and h
C=[0 0 0 0 0 0 0 0 0 0 ];%cause we only need first 10 data
for i =1:10
  C(i)=c(i); %get the 1st 10 data
end
subplot(2,1,2)
```

```
hold on
title('Output with conv')
xlabel('n')
ylabel('y[n]')
stem(0:length(C)-1,C);
hold off
```