Mamadou Kaba

27070179

ELEC 342 CN-X

11/07/2024

18/07/2024

Nadia Abdolkhani

Objective

The objectives of this experiment are to explore the properties of a discrete-time system, specifically its linearity and time invariance, using MATLAB. By designing and conducting experiments, we aim to determine if the system exhibits linear behavior, meaning it satisfies the properties of additivity and homogeneity, and if it is time-invariant, meaning its output response is consistent regardless of when the input signal is applied. These experiments will help us understand the system's fundamental characteristics and behavior under different conditions.

Theory

Linearity: A system is linear if it satisfies two key properties:

1. **Additivity:** For any two input signals x1[n] and x2[n]], the system's response to the sum of these inputs should be equal to the sum of the responses to each input individually. Mathematically, $Sys(a \cdot x1[n] + b \cdot x2[n]) = a \cdot Sys(x1[n]) + b \cdot Sys(x2[n])$, where a and b are scalars.

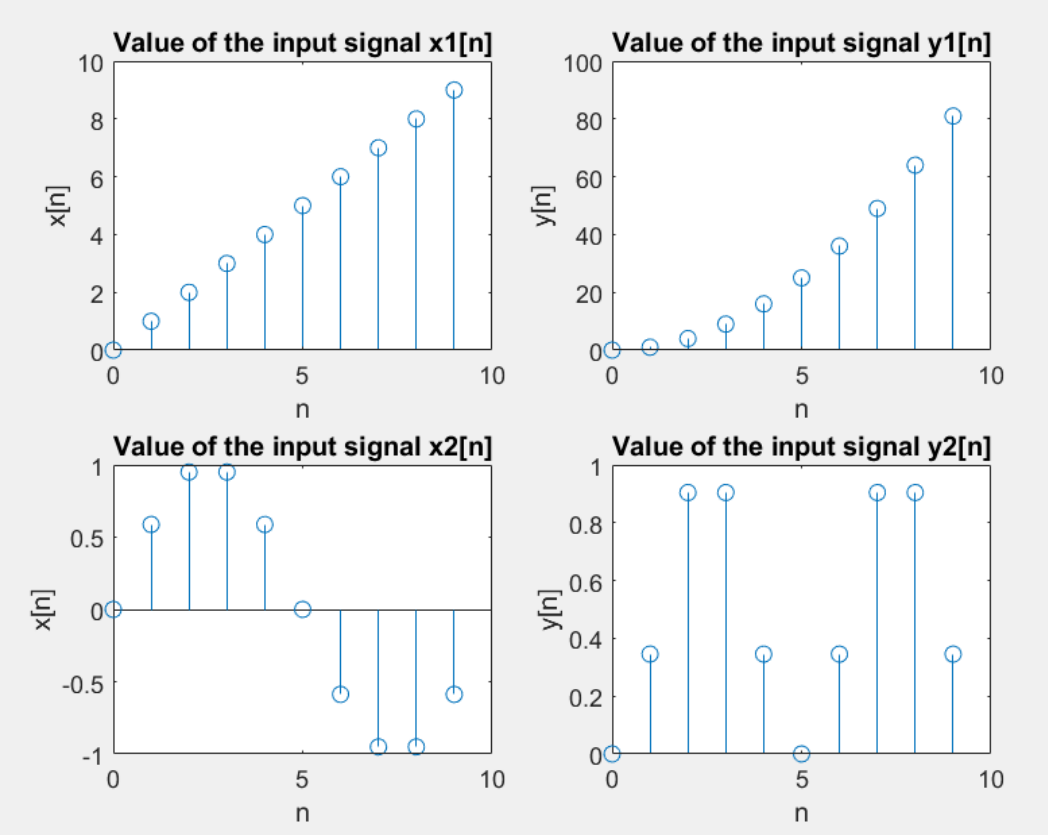
2. **Homogeneity:** For any input signal x[n] and scalar a, the system's response to the scaled input should be equal to the scaled response of the input.
   Mathematically, $Sys(a \cdot x[n]) = a \cdot Sys(x[n])$

Time Invariance: A system is time-invariant if its behavior does not change over time. This means that a time shift in the input signal results in an equivalent time shift in the output signal. Mathematically, if y[n] = Sys(x[n]) y[n] = Sys(x[n]) y[n] = Sys(x[n]), then $Sys(x[n-k]) = y[n-k]$ for any time shift k.

Results/Discussion

Part 1

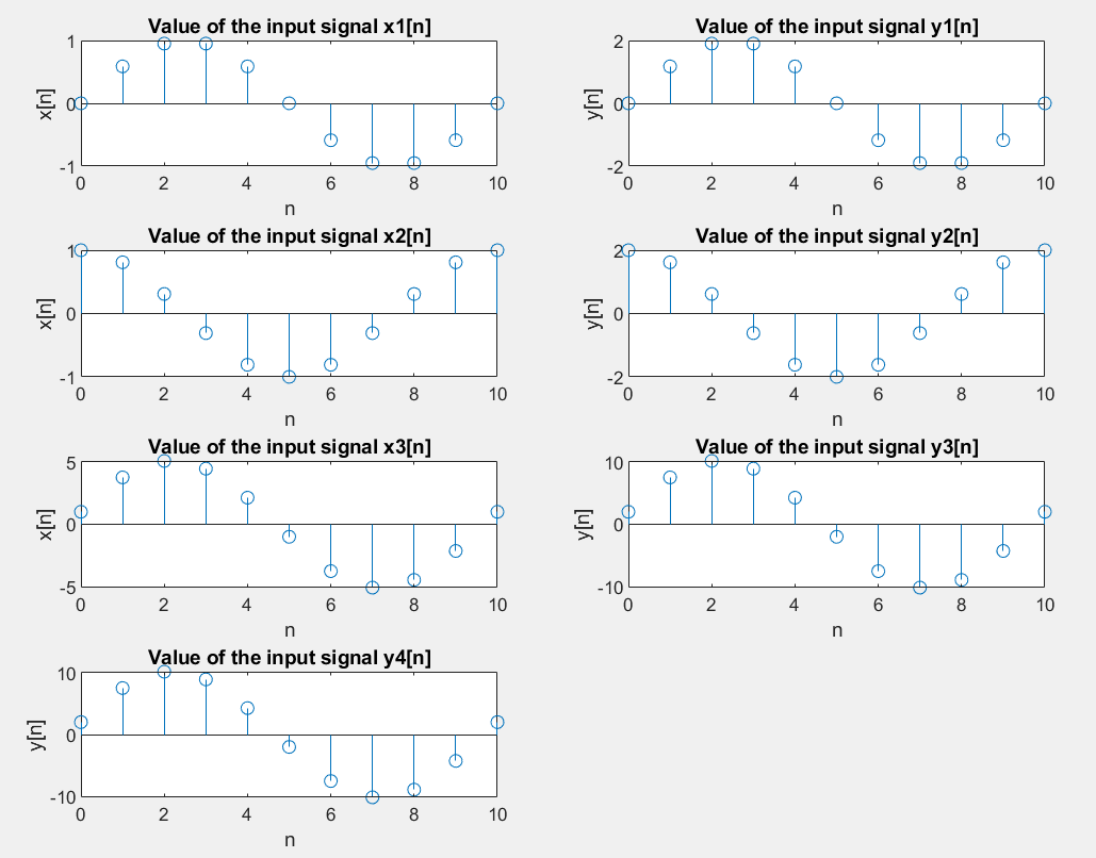
Question 1




```
The total energy in the signal x1[n] is:
   285

The total energy in the signal y1[n] is:
       15333

The total energy in the signal x2[n] is:
    5.0000

The total energy in the signal y2[n] is:
    3.7500
```

Question 2

a)




```
System is linear
>>
```

b)

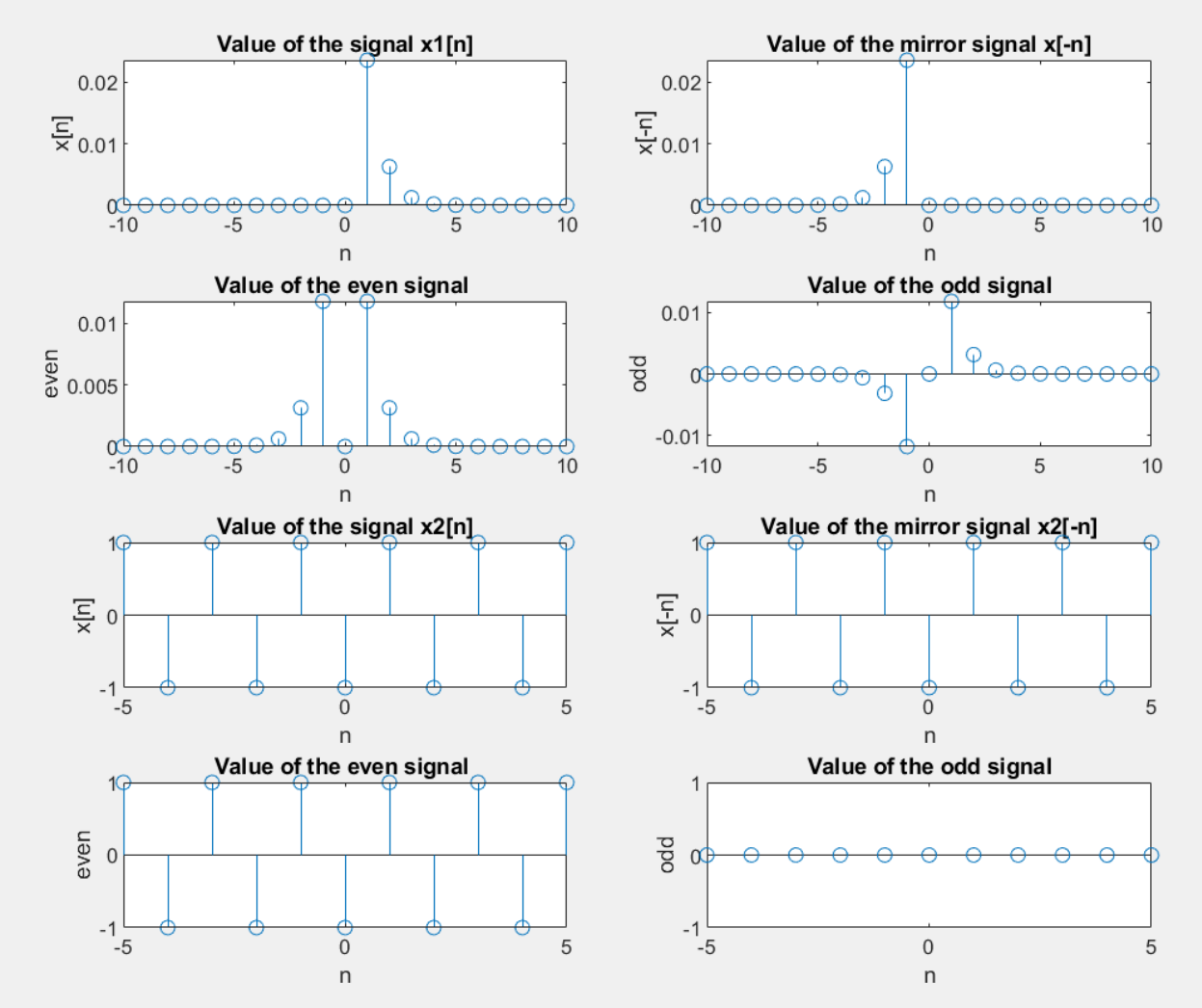
```
System (i) is not linear with x[n] = [0, 1]
System (i) is time-invariant with x[n] = [0, 1]
System (i) is not linear with larger set of values
System (i) is time-invariant with larger set of values
System (ii) is not linear with x[n] = [0, 1]
System (ii) is not time-invariant with x[n] = [0, 1]
System (ii) is not linear with larger set of values
System (ii) is not time-invariant with larger set of values
>>
```

When using the input data x[n] = [0,1], the results indicated that System (i) is not linear but time-invariant, while System (ii) is neither linear nor time-invariant. To further validate these findings,

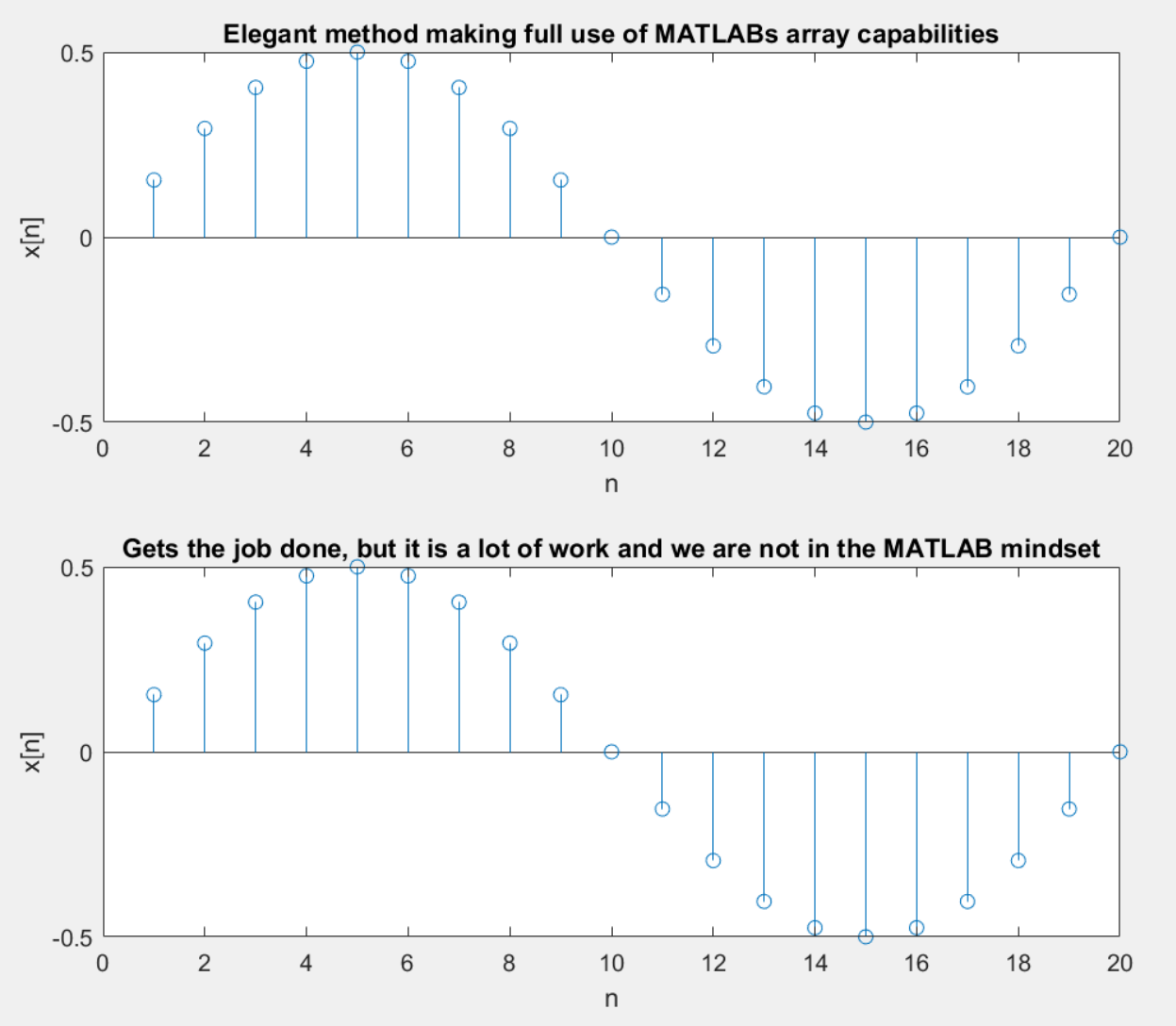
we repeated the experiments using a larger set of values x[n] = [0,1,2,…,10]. The results from the larger set confirmed that System (i) remains not linear but time-invariant, and System (ii) continues to be neither linear nor time-invariant. These results validate the original findings obtained with the smaller input data set. The choice of data impacts the results as it ensures that the system's properties are tested over a broader range of inputs, thereby providing more comprehensive evidence to support the initial conclusions. Using a more extensive input set helps in identifying consistent behavior or discrepancies that might not be evident with limited data, thus reinforcing the reliability of the conclusions.

Question 3

a and b)



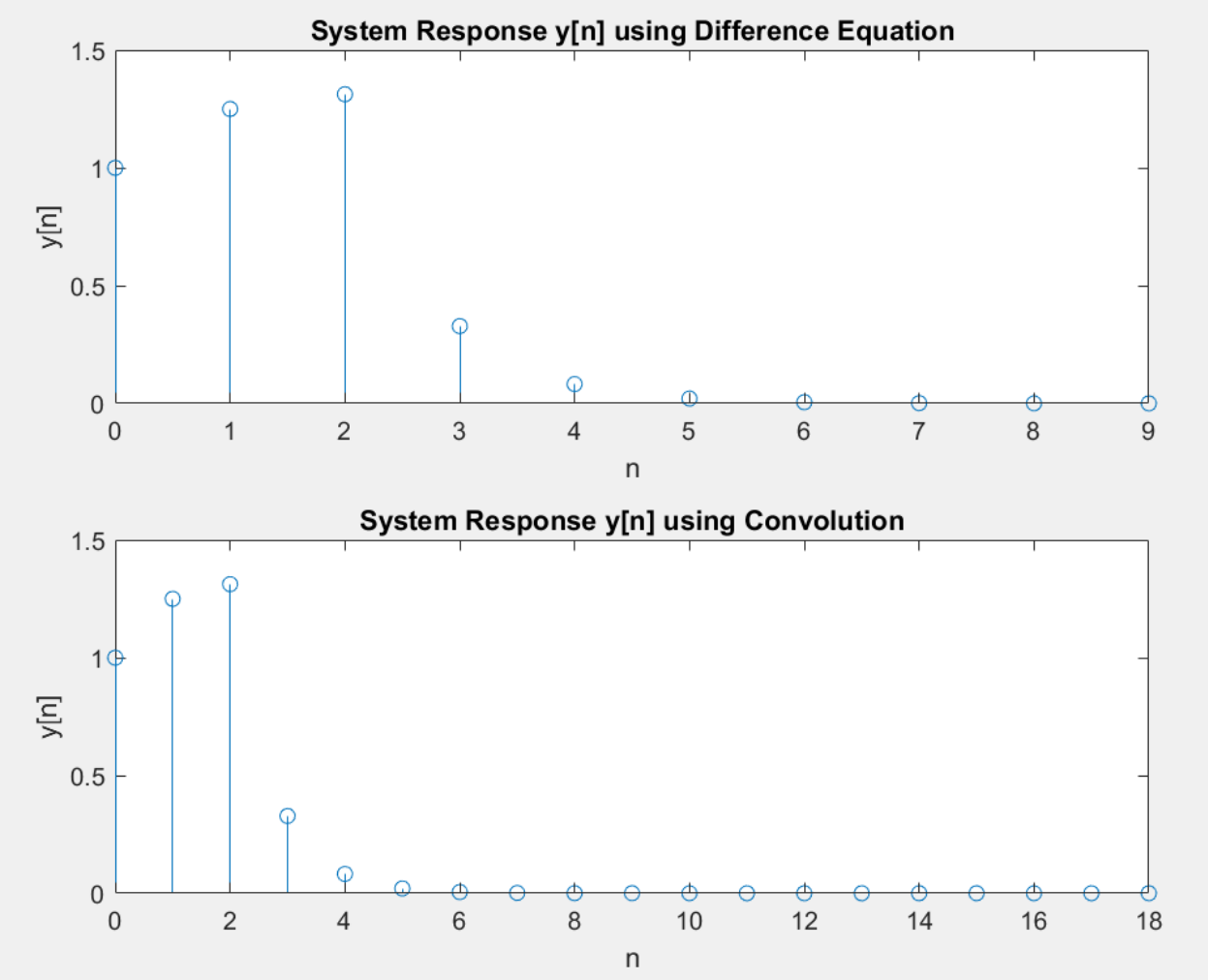
c)



The two methods for generating arrays in MATLAB highlight the contrast between vectorized operations and loop-based approaches. The vectorized method (x1 = sin((2*pi/40) * n) .* cos((2*pi/40) * n)) is more efficient and readable, taking full advantage of MATLAB's optimized array handling capabilities. It results in cleaner, more concise code, adhering to MATLAB's idiomatic style. Conversely, the loop-based method (for index = 1 : 20, x2(index) = sin((2*pi/40) * index) * cos((2*pi/40) * index); end) is less efficient, more verbose, and resembles procedural programming found in languages like C++. While the loop-based method is functional, it does not fully leverage MATLAB's strengths, making the vectorized approach generally preferable.

Part 2

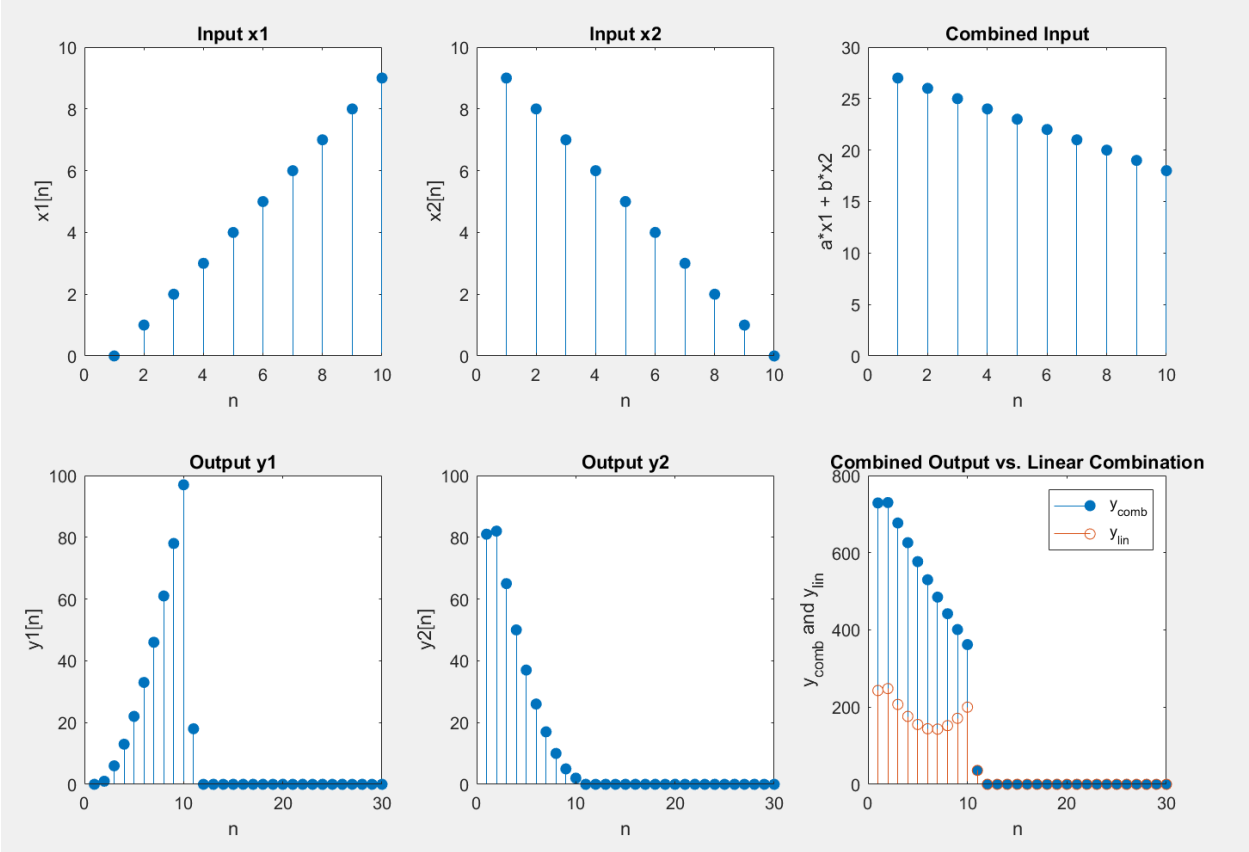
Question 1 and 2



The convolution method provides the entire system response considering the entire duration of the input and impulse response, whereas the difference equation method computes the response iteratively for each time step, inherently handling the initial condition at $y[-1] = 0$. The convolution output may have more values since it computes the full overlap of x[n] and H[n], resulting in a longer sequence.

Question 3

a)



(i)

The system is linear if it satisfies the properties of additivity and homogeneity. Specifically, for any two input signals x1[n] and x2[n], and scalars a and b, the system is linear if:
Sys1(a · x1 + b · x2) = a · Sys1(x1) + b ·Sys1(x2)

By testing this hypothesis, we can determine if the system behaves predictably under linear combinations of inputs. If the hypothesis is true, it confirms the system's linearity. If the hypothesis is false, we have a counter-example proving the system is not linear.

(ii)

To test the linearity of the system, we selected two distinct input signals and scalar multipliers to comprehensively assess the system's behavior. The input signals x1[n] = [0,1,2,…,9] and x2[n] = [9,8,7,…,0] were chosen because they represent simple yet

contrasting patterns: an ascending sequence and a descending sequence. These contrasting patterns help in observing how the system responds to both increasing and decreasing trends, which is crucial in identifying any non-linear behavior. Scalars a=2 and b=3 were used to test the system's response to scaled inputs, addressing the homogeneity aspect of linearity. By calculating the individual outputs y1 and y2, and then comparing the system's response to the combined input $a \cdot x1+b \cdot x2$ (denoted as ycomb) with the linear combination of the individual outputs $a \cdot y1+b \cdot y2$ (denoted as ylin), we can observe if the system satisfies the additivity property of linearity. If ycomb matches ylin , the system is linear; otherwise, it is not. This experimental design, with its diverse input patterns and scalar multipliers, aims to provide a robust and thorough test for the system's linearity by clearly revealing any discrepancies that would indicate non-linear behavior.

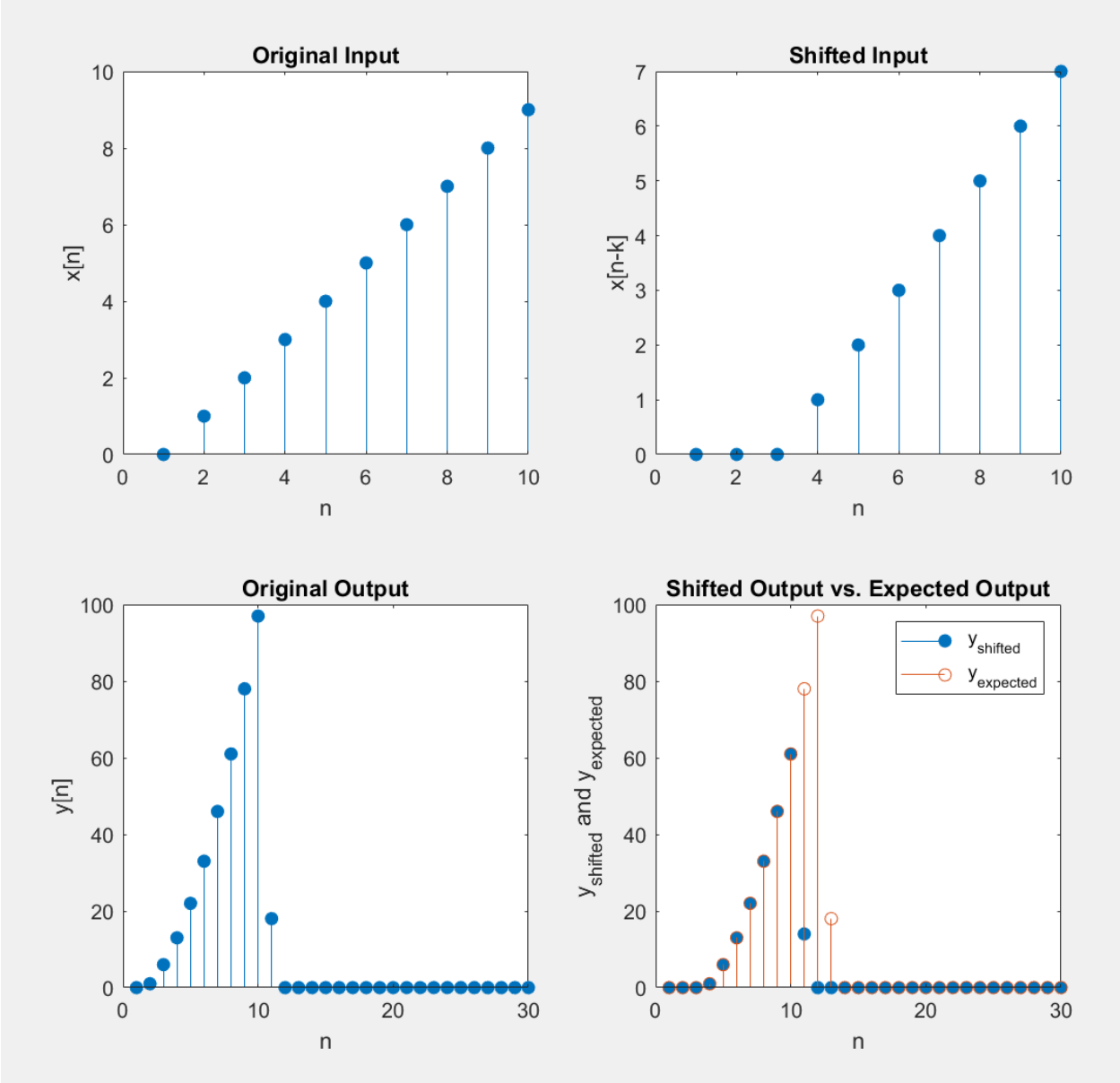
(iii)

When the inputs x1[n] and x2[n] were applied to the system, the outputs y1 and y2 exhibited distinct patterns as shown in the plots. The combined input $a \cdot x1+b \cdot x2$ resulted in a combined output ycomb that significantly deviated from the expected linear combination of the individual outputs ylin. Specifically, the plot for ycomb showed a much higher magnitude and different shape compared to ylin, indicating that the system's response to the combined input was not a simple linear combination of its responses to the individual inputs. This clear discrepancy demonstrates that the system does not exhibit linear behavior.

(iv)

From the experiment, we conclude that the system is not linear. This conclusion is based on the observation that the combined output ycom does not match the linear combination of the individual outputs ylin. The discrepancy between these two outputs provides a counter-example that disproves the hypothesis of linearity for this system. Therefore, we can state with certainty that the system is not linear based on the chosen inputs and observed outputs.

b)



(i)

The system is time-invariant if a time shift in the input signal results in an equivalent time shift in the output signal. Specifically, for an input signal x[n] and a time shift k, the system is time-invariant if: Sys1(x[n−k])=y[n−k] where y[n]=Sys1(x[n]).

By testing this hypothesis, we can determine if the system's response remains consistent regardless of when the input signal is applied. If the hypothesis is true, it confirms the

system's time invariance. If the hypothesis is false, we have a counter-example proving the system is time-varying.

(ii)

To test the time invariance of the system, we selected an input signal x[n] = [0,1,2,…,9] and applied a time shift of k = 2. This input sequence is simple yet effective in observing the system's response to a straightforward pattern. The ascending sequence allows us to clearly track how each element is processed by the system, and the chosen shift of k = 2 positions helps us determine if the system's output remains consistent when the input signal is delayed. By calculating the original output y = Sys1(x) and comparing the system's response to the shifted input x_shifted=[0,0,0,1,2,3,4,5,6,7] with the expected shifted output y_expected=[0,0,y(1:end−2)], we aim to observe if the system's output for the shifted input matches the shifted version of the original output. If the outputs align, it would indicate that the system is time-invariant. However, any discrepancy would demonstrate that the system's behavior changes with input timing, proving it is time-varying. This method provides a robust test for time invariance by using a clear and traceable input signal and a straightforward shift operation to evaluate the system's consistency over time.

(iii)

When the input x[n] = [0,1,2,…,9] was applied to the system, the original output y[n] showed an increasing trend followed by a sharp drop to zero. When the input was shifted by k=2 positions, the shifted output y_shifted did not match the expected shifted output y_expected. Specifically, the shifted output deviated significantly from the expected output, indicating that the system's response to the delayed input was not simply a shifted version of the original response. This discrepancy is clearly visible in the plots, demonstrating that the system's behavior changes with input timing.

(iv)

From the experiment, we conclude that the system is not time-invariant. This conclusion is based on the observation that the shifted output y_shifted does not match the expected shifted output y_expected. The discrepancy between these two outputs provides a counter-example

that disproves the hypothesis of time invariance for this system. Therefore, we can state with certainty that the system is time-varying based on the chosen inputs and observed outputs.

Conclusion

This lab provided a comprehensive exploration of various properties of discrete-time systems, including linearity and time-invariance, using MATLAB. Through a series of experiments, we tested multiple systems and analyzed their responses to different inputs. By examining the outputs, we determined which systems adhered to the principles of linearity and time-invariance and which did not. The use of MATLAB's powerful scripting and plotting capabilities facilitated precise analysis and visualization of system behaviors.

Appendix

Part1_question1

```
% Mamadou Kaba 27070179
% Question 1

clc;clear;close all;

n=0:9;

x1=n;
y1=x1.^2;

x2=sin(2*pi*n/10);
y2=x2.^2;

subplot(2,2,1)
stem(n,x1)
title('Value of the input signal x1[n]')
xlabel('n')
ylabel('x[n]')

subplot(2,2,2)
stem(n,y1)
title('Value of the input signal y1[n]')
xlabel('n')
ylabel('y[n]')

subplot(2,2,3)
stem(n,x2)
title('Value of the input signal x2[n]')
xlabel('n')
ylabel('x[n]')

subplot(2,2,4)
stem(n,y2)
title('Value of the input signal y2[n]')
xlabel('n')
ylabel('y[n]')

total_energy_x1=0;

for i=x1
    total_energy_x1 = total_energy_x1 + i.^2;
end

disp('The total energy in the signal x1[n] is: ')
disp (total_energy_x1)

total_energy_y1=0;

for i=y1
```

```
    total_energy_y1 = total_energy_y1 + i.^2;
end

disp('The total energy in the signal y1[n] is: ')
disp (total_energy_y1)

total_energy_x2=0;

for i=x2
    total_energy_x2 = total_energy_x2 + i.^2;
end

disp('The total energy in the signal x2[n] is: ')
disp (total_energy_x2)

total_energy_y2=0;

for i=y2
    total_energy_y2 = total_energy_y2 + i.^2;
end

disp('The total energy in the signal y2[n] is: ')
disp (total_energy_y2)
```

Part1_question2a

```
% Mamadou Kaba 27070179
% Question 2 a)

clc;clear;close all;

n=0:10;

x1=sin(2*pi*n/10);
y1=2*x1;

x2=cos(2*pi*n/10);
y2=2*x2;

x3=5*x1+x2;
y3=2*x3;

y4=5*y1+y2;

if (y3==y4)
    disp('System is linear')
else
    disp('System is not linear')
end

subplot(4,2,1)
stem(n,x1)
title('Value of the input signal x1[n]')
```

```
xlabel('n')
ylabel('x[n]')

subplot(4,2,2)
stem(n,y1)
title('Value of the input signal y1[n]')
xlabel('n')
ylabel('y[n]')

subplot(4,2,3)
stem(n,x2)
title('Value of the input signal x2[n]')
xlabel('n')
ylabel('x[n]')

subplot(4,2,4)
stem(n,y2)
title('Value of the input signal y2[n]')
xlabel('n')
ylabel('y[n]')

subplot(4,2,5)
stem(n,x3)
title('Value of the input signal x3[n]')
xlabel('n')
ylabel('x[n]')

subplot(4,2,6)
stem(n,y3)
title('Value of the input signal y3[n]')
xlabel('n')
ylabel('y[n]')

subplot(4,2,7)
stem(n,y4)
title('Value of the input signal y4[n]')
xlabel('n')
ylabel('y[n]')
```

Part1_question2b

```
% Mamadou Kaba 27070179
% Question 2 b)

clc;clear;close all;

% i)
x1 = [0, 1];
y1 = x1.^2;

% Test for Linearity (i)
x2 = 3 * x1;
y2 = x2.^2;
```

```
x3 = 5 * x1 + x2;
y3 = x3.^2;

y4 = 5 * y1 + y2;

if all(y3 == y4)
    disp('System (i) is linear with x[n] = [0, 1]')
else
    disp('System (i) is not linear with x[n] = [0, 1]')
end

% Test for Time-Invariance (i)
x5 = [0, 0, x1];
y5 = x5.^2;

y6 = [0, 0, y1];

if all(y5 == y6)
    disp('System (i) is time-invariant with x[n] = [0, 1]')
else
    disp('System (i) is not time-invariant with x[n] = [0, 1]')
end

% Larger set of values for x[n] from 0 to 10
x1 = 0:10;
y1 = x1.^2;

% Test for Linearity with larger set (i)
x2 = 3 * x1;
y2 = x2.^2;

x3 = 5 * x1 + x2;
y3 = x3.^2;

y4 = 5 * y1 + y2;

if all(y3 == y4)
    disp('System (i) is linear with larger set of values')
else
    disp('System (i) is not linear with larger set of values')
end

% Test for Time-Invariance with larger set (i)
x5 = [0, 0, x1];
y5 = x5.^2;

y6 = [0, 0, y1];

if all(y5 == y6)
    disp('System (i) is time-invariant with larger set of values')
else
    disp('System (i) is not time-invariant with larger set of values')
end
```

```
% ii)
x1 = [0, 1];
y1 = 2*x1 + 5*[1, 0];

% Test for Linearity (ii)
x2 = 3 * x1;
y2 = 2*x2 + 5*[1, 0];

x3 = 5 * x1 + x2;
y3 = 2*x3 + 5*[1, 0];

y4 = 5 * y1 + y2;

if all(y3 == y4)
    disp('System (ii) is linear with x[n] = [0, 1]')
else
    disp('System (ii) is not linear with x[n] = [0, 1]')
end

% Test for Time-Invariance (ii)
x5 = [0, 0, x1];
y5 = 2*x5 + 5*[1, 0, 0, zeros(1, length(x5)-3)];

y6 = [0, 0, y1];

if all(y5 == y6)
    disp('System (ii) is time-invariant with x[n] = [0, 1]')
else
    disp('System (ii) is not time-invariant with x[n] = [0, 1]')
end

% Larger set of values for x[n] from 0 to 10 (ii)
x1 = 0:10;
y1 = 2*x1 + 5*[1, zeros(1, length(x1)-1)];

% Test for Linearity with larger set (ii)
x2 = 3 * x1;
y2 = 2*x2 + 5*[1, zeros(1, length(x1)-1)];

x3 = 5 * x1 + x2;
y3 = 2*x3 + 5*[1, zeros(1, length(x1)-1)];

y4 = 5 * y1 + y2;

if all(y3 == y4)
    disp('System (ii) is linear with larger set of values')
else
    disp('System (ii) is not linear with larger set of values')
end

% Test for Time-Invariance with larger set (ii)
x5 = [0, 0, x1];
y5 = 2*x5 + [5, zeros(1, length(x5)-1)];
```

```
y6 = [0, 0, y1];

if all(y5 == y6)
    disp('System (ii) is time-invariant with larger set of values')
else
    disp('System (ii) is not time-invariant with larger set of values')
end
```

Part1_question3ab

```
% Mamadou Kaba 27070179
% Question 3 a) and b)

clc;clear;close all;

% a)

n1= -10:10;
n_positive = 0:10;
x1_positive = exp(-2*abs(n_positive)) .* sin((2*pi/36) .* n_positive);
x1 = [zeros(1, 10), x1_positive];

x2 = zeros(1, 21);
for index = 1:21
    x2(index) = x1(22 - index);
end

even1 = 1/2 * (x1 + x2);
odd1 = 1/2 * (x1 - x2);

% b)

n2=-5:5;
x3= (-1).^(n2-1);
x4=(-1).^((-1.*n2)-1);
even2 = 1/2*(x3+x4);
odd2 = 1/2*(x3-x4);


subplot(4,2,1)
stem(n1,x1)
title('Value of the signal x1[n]')
xlabel('n')
ylabel('x[n]')

subplot(4,2,2)
stem(n1,x2)
title('Value of the mirror signal x[-n]')
xlabel('n')
ylabel('x[-n]')

subplot(4,2,3)
stem(n1,even1)
```

```
title('Value of the even signal ')
xlabel('n')
ylabel('even')

subplot(4,2,4)
stem(n1,odd1)
title('Value of the odd signal')
xlabel('n')
ylabel('odd')

subplot(4,2,5)
stem(n2,x3)
title('Value of the signal x2[n]')
xlabel('n')
ylabel('x[n]')

subplot(4,2,6)
stem(n2,x4)
title('Value of the mirror signal x2[-n]')
xlabel('n')
ylabel('x[-n]')

subplot(4,2,7)
stem(n2,even2)
title('Value of the even signal ')
xlabel('n')
ylabel('even')

subplot(4,2,8)
stem(n2,odd2)
title('Value of the odd signal')
xlabel('n')
ylabel('odd')
```

Part1_question3c

```
% 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, but it is a lot of work and we are not in the MATLAB
mindset')
xlabel('n')
ylabel('x[n]')
```

Part2_question1and2

```
% Mamadou Kaba 27070179
% Question 1 and 2

clc;clear;close all;

% Define the range for n
n = 0:9;

% Given input signal x[n]
x = [1, 1, 1, 0, 0, 0, 0, 0, 0, 0];

% Initialize y[n] with the same size as x[n]
y_diff_eq = zeros(1, length(n));

% Compute the response y[n] using the difference equation
for i = 1:length(n)
    if i == 1
        y_diff_eq(i) = x(i);
    else
        y_diff_eq(i) = x(i) + (1/4) * y_diff_eq(i-1);
    end
end

% Define the impulse response H[n]
H = (1/4).^n;

% Perform the convolution of x[n] and H[n]
y_conv = conv(x, H);


subplot(2,1,1)
stem(n, y_diff_eq)
title('System Response y[n] using Difference Equation')
xlabel('n')
ylabel('y[n]')

subplot(2,1,2)
n_conv = 0:(length(y_conv) - 1);
stem(n_conv, y_conv)
title('System Response y[n] using Convolution')
xlabel('n')
ylabel('y[n]')
```

Part2_question3

```
% Mamadou Kaba 27070179
% Question 3

clc;clear;close all;

% Testing Linearity of Sys1

x1 = 0:9;
x2 = [9, 8, 7, 6, 5, 4, 3, 2, 1, 0];
a = 2;
b = 3;


y1 = Sys1(x1);
y2 = Sys1(x2);

% Combined input and output
y_comb = Sys1(a * x1 + b * x2);
y_lin = a * y1 + b * y2;

% Check linearity
if all(y_comb == y_lin)
    disp('System (i) is linear with the chosen inputs')
else
    disp('System (i) is not linear with the chosen inputs')
end

% Plot the inputs and outputs for linearity test
figure;
subplot(2, 3, 1);
stem(x1, 'filled');
title('Input x1');
xlabel('n');
ylabel('x1[n]');

subplot(2, 3, 2);
stem(x2, 'filled');
title('Input x2');
xlabel('n');
ylabel('x2[n]');

subplot(2, 3, 3);
stem(a * x1 + b * x2, 'filled');
title('Combined Input');
xlabel('n');
ylabel('a*x1 + b*x2');

subplot(2, 3, 4);
stem(y1, 'filled');
title('Output y1');
xlabel('n');
ylabel('y1[n]');
```

```
subplot(2, 3, 5);
stem(y2, 'filled');
title('Output y2');
xlabel('n');
ylabel('y2[n]');

subplot(2, 3, 6);
stem(y_comb, 'filled');
hold on;
stem(y_lin, 'o');
title('Combined Output vs. Linear Combination');
xlabel('n');
ylabel('y_{comb} and y_{lin}');
legend('y_{comb}', 'y_{lin}');

% Initial input data for time-invariance test
x = 0:9;
k = 2;  % shift amount

% Original output
y = Sys1(x);

% Shifted input
x_shifted = [zeros(1, k), x(1:end-k)];

% Output for shifted input
y_shifted = Sys1(x_shifted);

% Expected output by shifting original output
y_expected = [zeros(1, k), y(1:end-k)];

% Check time-invariance
if all(y_shifted == y_expected)
    disp('System (i) is time-invariant with the chosen inputs')
else
    disp('System (i) is not time-invariant with the chosen inputs')
end

% Plot the inputs and outputs for time-invariance test
figure;
subplot(2, 2, 1);
stem(x, 'filled');
title('Original Input');
xlabel('n');
ylabel('x[n]');

subplot(2, 2, 2);
stem(x_shifted, 'filled');
title('Shifted Input');
xlabel('n');
ylabel('x[n-k]');

subplot(2, 2, 3);
stem(y, 'filled');
title('Original Output');
```

```
xlabel('n');
ylabel('y[n]');

subplot(2, 2, 4);
stem(y_shifted, 'filled');
hold on;
stem(y_expected, 'o');
title('Shifted Output vs. Expected Output');
xlabel('n');
ylabel('y_{shifted} and y_{expected}');
legend('y_{shifted}', 'y_{expected}');
```