Name: **SOLUTION (Havlicek)**

Section:

Laboratory Exercise 2

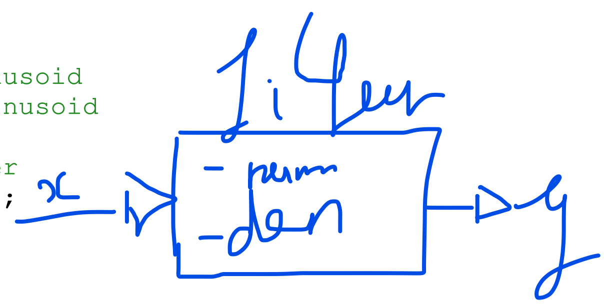
DISCRETE-TIME SYSTEMS: TIME-DOMAIN REPRESENTATION

2.1 SIMULATION OF DISCRETE-TIME SYSTEMS

Project 2.1 The Moving Average System

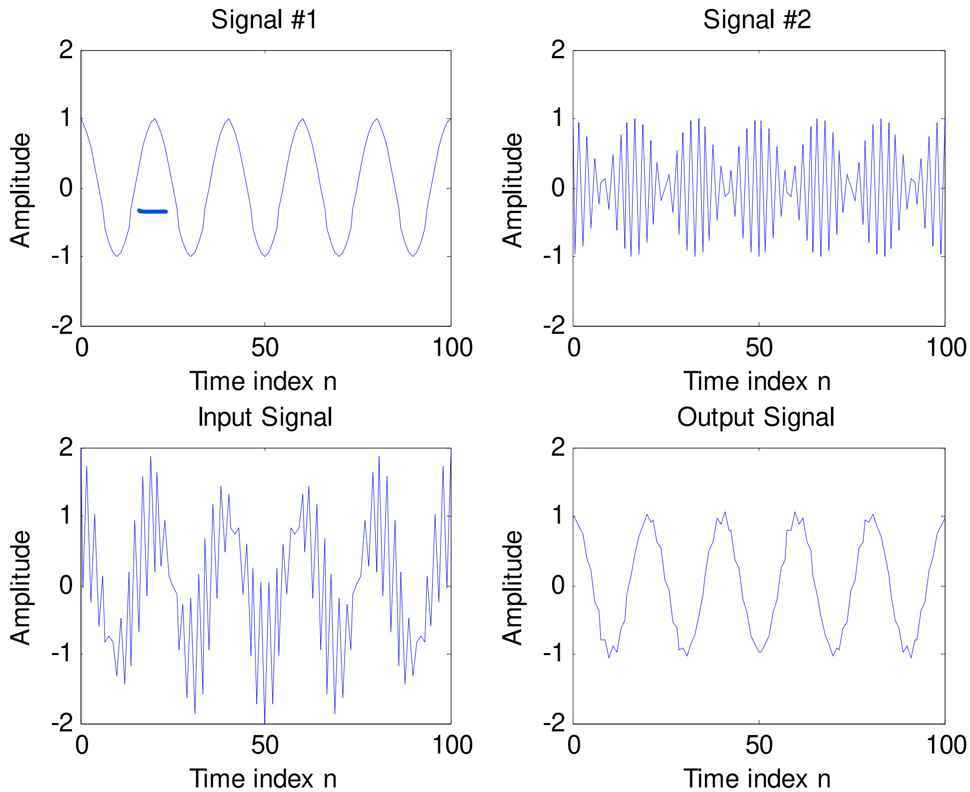
A copy of Program P2_1 is given below:

```
% Program P2_1
% Simulation of an M-point Moving Average Filter
% Generate the input signal
n = 0:100;
s1 = cos(2*pi*0.05*n); % A low-frequency sinusoid
s2 = cos(2*pi*0.47*n); % A high frequency sinusoid
x = s1+s2;
% Implementation of the moving average filter
M = input('Desired length of the filter = ');
num = ones(1,M);
y = filter(num,1,x)/M;
% Display the input and output signals
clf;
subplot(2,2,1);
plot(n, s1);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #1');
subplot(2,2,2);
plot(n, s2);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #2');
subplot(2,2,3);
plot(n, x);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Input Signal');
subplot(2,2,4);
plot(n, y);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Signal');
axis;
```



Answers:

Q2.1 The output sequence generated by running the above program for M = 2 with `x[n] = s1[n]+s2[n]` as the input is shown below.



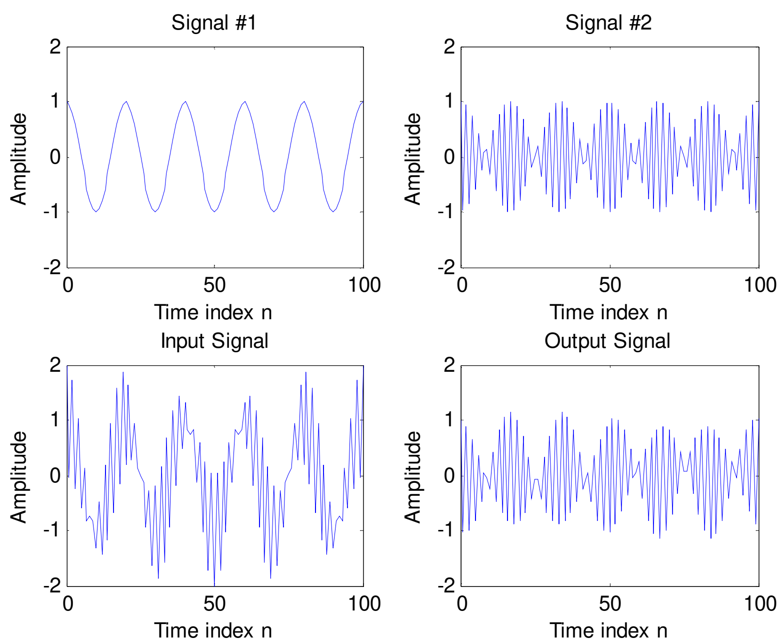
The component of the input `x[n]` suppressed by the discrete-time system simulated by this program is – Signal #2, the high frequency one (it is a low pass filter).

Q2.2 Program P2_1 is modified to simulate the LTI system `y[n] = 0.5(x[n]-x[n-1])` and process the input `x[n] = s1[n]+s2[n]` resulting in the output sequence shown below:

Note: the code is not required; however, it is included here to demonstrate a tricky way of making the modification to P2_1.

```
% Program Q2_2
% Modification of P2_1 to convert it to a high pass filter
% Generate the input signal
n = 0:100;
s1 = cos(2*pi*0.05*n); % A low-frequency sinusoid
s2 = cos(2*pi*0.47*n); % A high frequency sinusoid
x = s1+s2;
% Implementation of high pass filter
M = input('Desired length of the filter = ');
% By comparing eq. (2.13) to (2.3), you can see that "num"
% actually contains the impulse response (times the constant
```

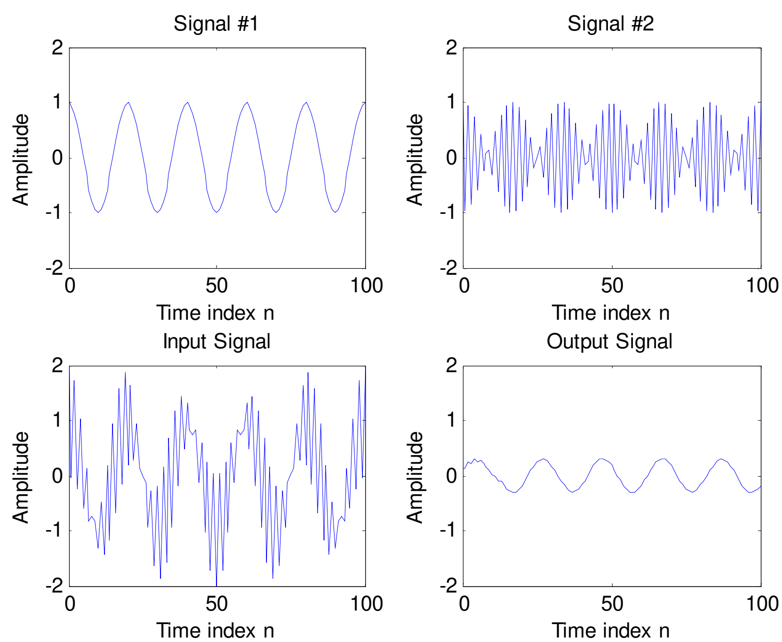
```
% M).  What we are actually doing in Q2.2 is multiplying the
% impulse response of the low pass filter in P2_1 by the
% sequency (-1)^n.  This shifts the low pass frequency
% response up to be centered at f=0.25, making it a high
% pass filter.
num = (-1).^[0:M-1];
y = filter(num,1,x)/M;
% Display the input and output signals
clf;
subplot(2,2,1);
plot(n, s1);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #1');
subplot(2,2,2);
plot(n, s2);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #2');
subplot(2,2,3);
plot(n, x);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Input Signal');
subplot(2,2,4);
plot(n, y);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Signal');
axis;
```



The effect of changing the LTI system on the input is – The system is now a high pass filter. It passes the high-frequency input component `s2` instead of the low frequency input component `s1`.

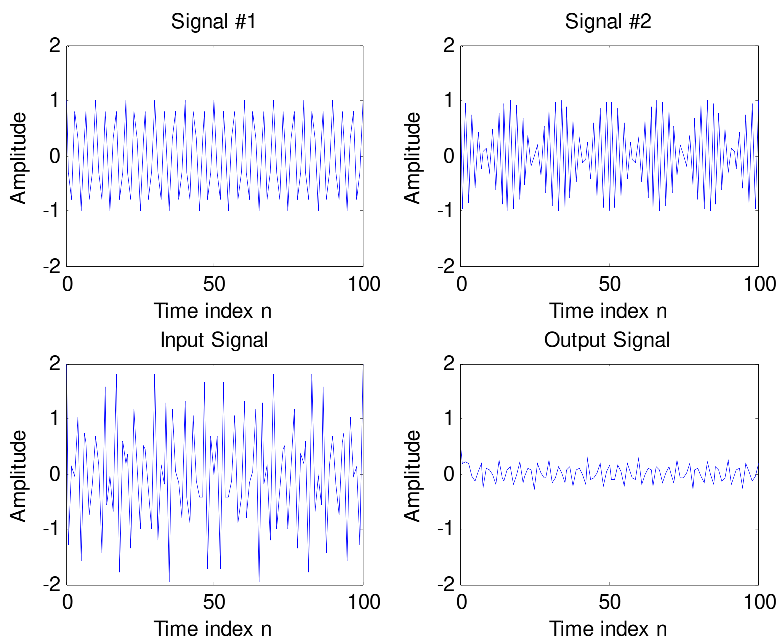
Q2.3 Program P2_1 is run for the following values of filter length M and following values of the frequencies of the sinusoidal signals `s1[n]` and `s2[n]`. The output generated for these different values of `M` and the frequencies are shown below.

`f1=0.05; f2=0.47; M=15`



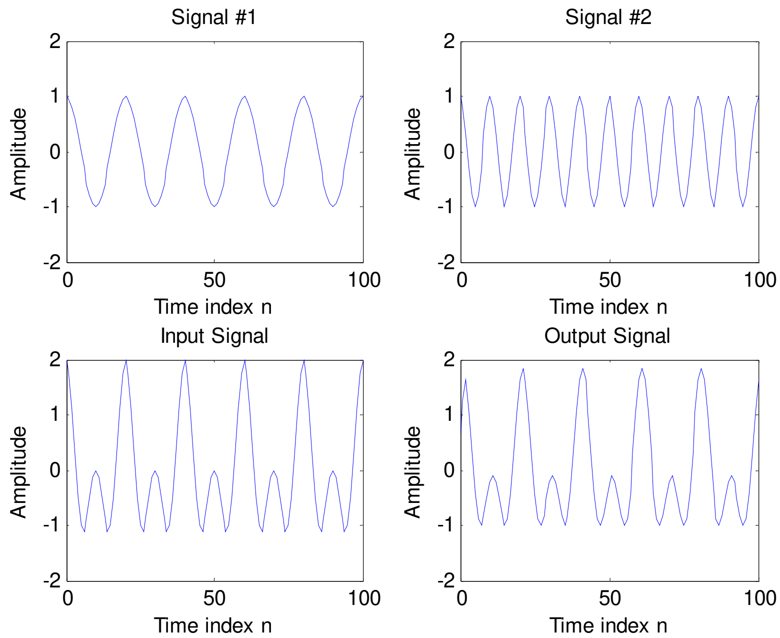
From these plots we make the following observations – with M=15, the low pass characteristic is much more pronounced (the passband is now very narrow). `s2` is still nearly eliminated in the output signal. `s1` is still passed, but at an attenuated level.

`f1=0.30; f2=0.47; M=4`



From these plots we make the following observations – with M=4, this filter performs more smoothing than in the case M=2. Both `s1` and `s2` are high frequency in this case, and they are both substantially attenuated in the output.

`f1=0.05; f2=0.10; M=3`



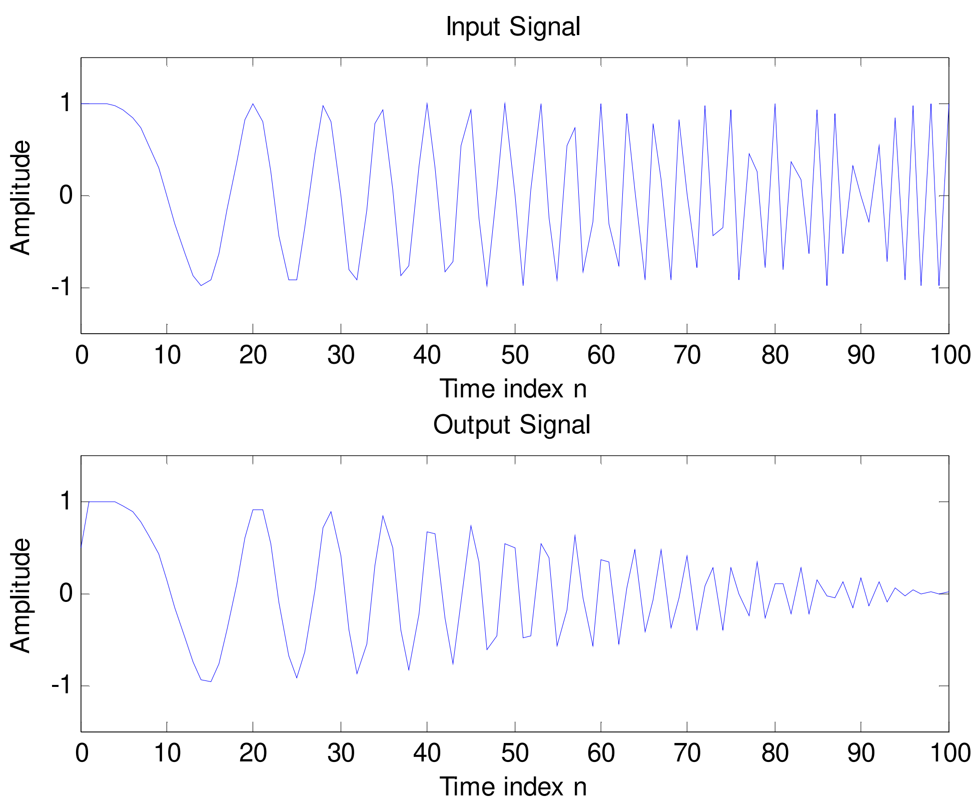
From these plots we make the following observations – here `s1` and `s2` are both low pass and they are both visible in the filter output. However, `s2`, the higher frequency input, is attenuated slightly more than `s1` in the system output.

Q2.4 The required modifications to Program P2_1 by changing the input sequence to a swept-frequency sinusoidal signal (length 101, minimum frequency 0, and a maximum frequency 0.5) as the input signal (see Program P1_7) are listed below:

```
% Program Q2_4
% Modify P2_1 to use a swept frequency chirp input
% Generate the input signal
n = 0:100;
a = pi/200;
b = 0;
arg = a*n.*n + b*n;
x = cos(arg);
% Implementation of the moving average filter
M = input('Desired length of the filter = ');
num = ones(1,M);
y = filter(num,1,x)/M;
% Display the input and output signals
clf;
subplot(2,1,1);
plot(n, x);
axis([0, 100, -1.5, 1.5]);
xlabel('Time index n'); ylabel('Amplitude');
title('Input Signal');
subplot(2,1,2);
plot(n, y);
axis([0, 100, -1.5, 1.5]);
```

```
xlabel('Time index n'); ylabel('Amplitude');
title('Output Signal');
axis;
```

The output signal generated by running this program is plotted below.



The results of Questions Q2.1 and Q2.2 from the response of this system to the swept-frequency signal can be explained as follows: we see again that this system is a low pass filter. At the left of the graphs, the input signal is a low frequency sinusoid that is passed to the output without attenuation. As n increases, the frequency of the input rises, and increasing attenuation is seen at the output. In Q2.1, the input was a sum of two sinusoids s1 and s2 with f1=0.05 and f2=0.47. The swept frequency input of Q2.4 reaches a frequency of 0.05 at n=10, where there is virtually no attenuation in the output shown above. This "explains" why s1 was passed by the system in Q2.1. The swept frequency input of Q2.4 reaches a frequency of 0.47 at approximately n=94, where the attenuation of the system is substantial. This "explains" why s2 was almost completely suppressed in the output in Q2.1.

There is no direct relationship between the result shown above for Q2.4 and the result obtained in Q2.2. However, using frequency domain concepts (Chapter 3) we can reason that, if the swept frequency signal was input to the system y[n] = 0.5(x[n] - x[n-1]), we would see a result opposite to what is shown above. Since the system would then be a high pass filter, there would be substantial attenuation of the output at the left side of the graph and virtually no attenuation at the right side of the graph. This "explains" why in Q2.2 the low frequency component s1 was suppressed in the system output, whereas the high frequency component s2 was passed.

Project 2.2 (Optional) A Simple Nonlinear Discrete-Time System

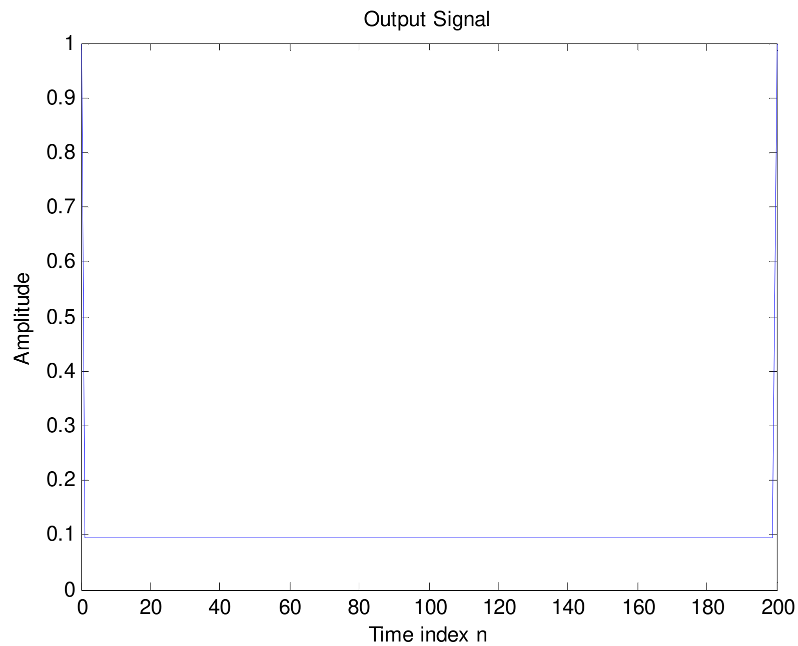
A copy of Program P2_2 is given below:

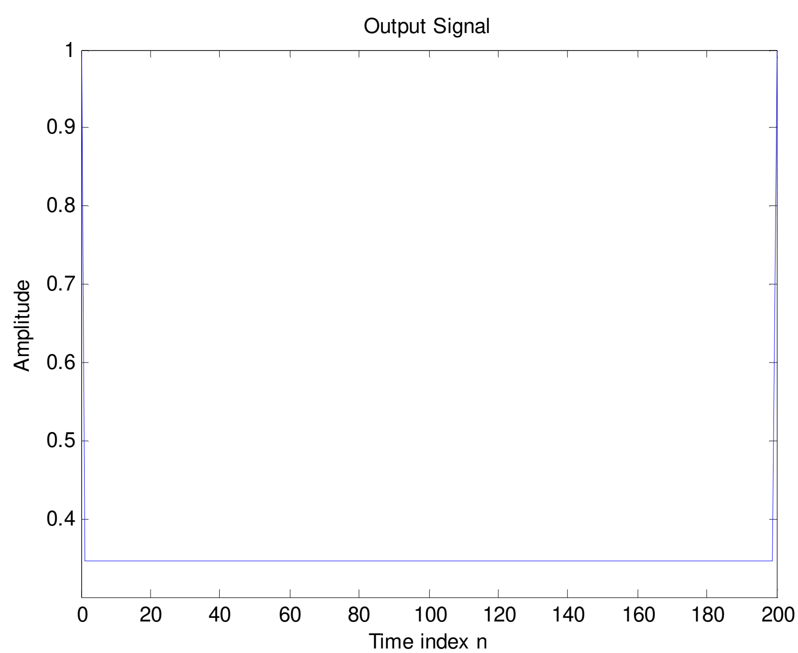
```
% Program P2_2
% Generate a sinusoidal input signal
clf;
n = 0:200;
x = cos(2*pi*0.05*n);
% Compute the output signal
x1 = [x 0 0];        % x1[n] = x[n+1]
x2 = [0 x 0];        % x2[n] = x[n]
x3 = [0 0 x];        % x3[n] = x[n-1]
y = x2.*x2-x1.*x3;
y = y(2:202);
% Plot the input and output signals
subplot(2,1,1)
plot(n, x)
xlabel('Time index n');ylabel('Amplitude');
title('Input Signal')
subplot(2,1,2)
plot(n,y)
xlabel('Time index n');ylabel('Amplitude');
title('Output signal');
```

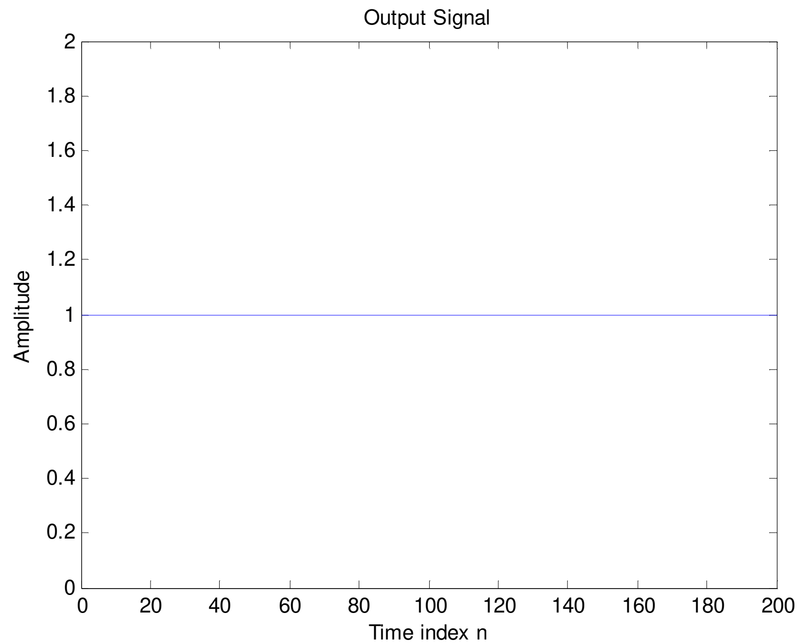
Answers:

Q2.5 The sinusoidal signals with the following frequencies as the input signals were used to generate the output signals: `f=0.05, f=0.1, f=0.25`

The output signals generated for each of the above input signals are displayed below:





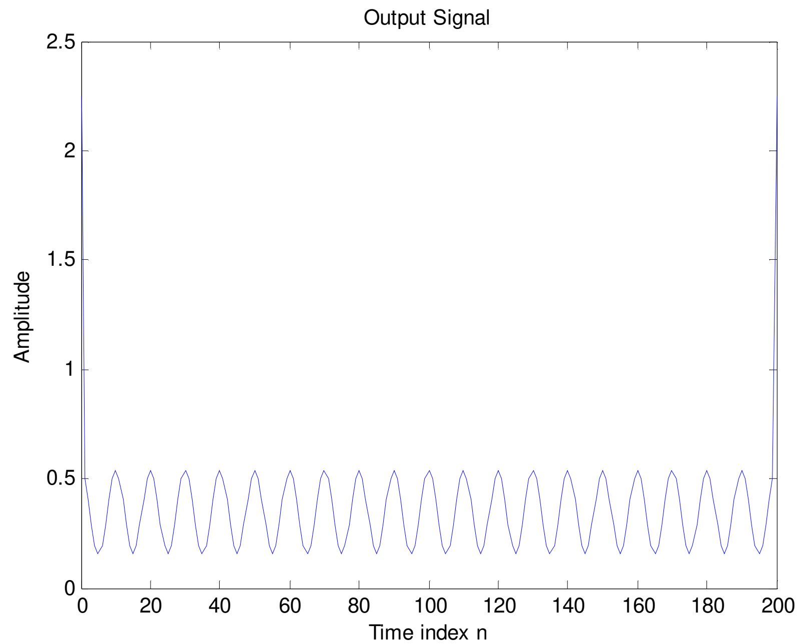


The output signals depend on the frequencies of the input signal according to the following rules: The answer to this question is omitted here because it will be part of a later homework assignment.

This observation can be explained mathematically as follows: The mathematical verification of the result is omitted here because it will be part of a later homework assignment.

Q2.6 The output signal generated by using sinusoidal signals of the form $x[n] = \cos(\omega_o n) + K$ as the input signal is shown below for the following values of ω_o and K -

$$\omega_o = 0.2\pi \ (f=0.1); \ K = 0.5$$



The dependence of the output signal $y[n]$ on the DC value K can be explained as –

The answer to this question is omitted here because it will be part of a later homework assignment.

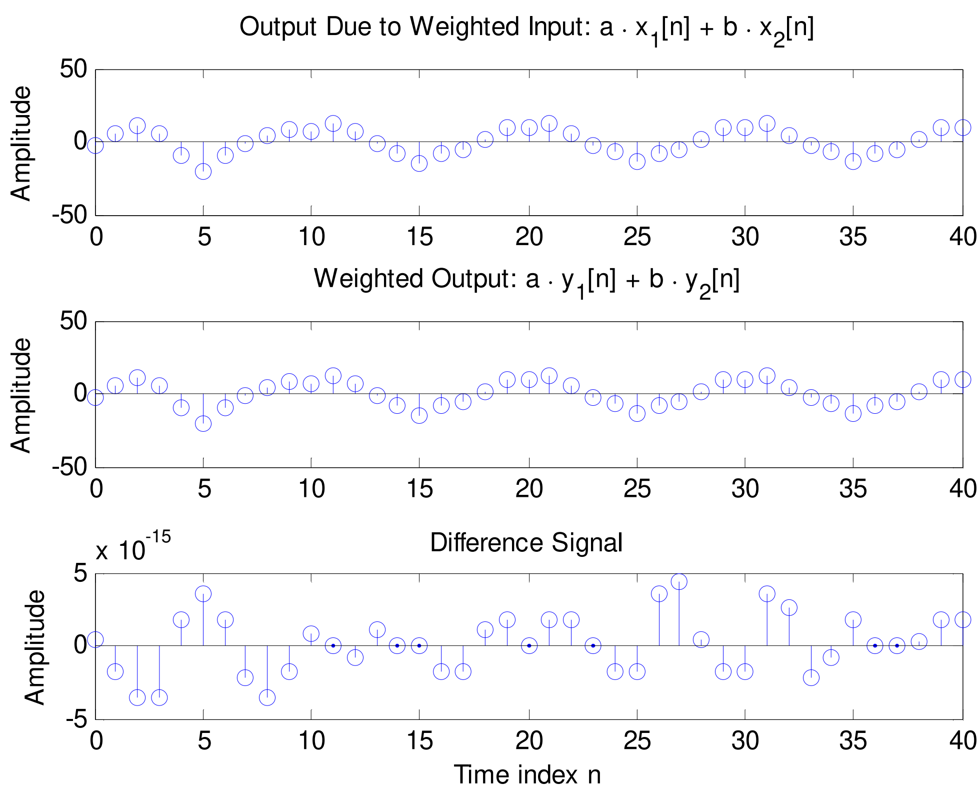
Project 2.3 Linear and Nonlinear Systems

A copy of Program P2_3 is given below:

```
% Program P2_3
% Generate the input sequences
clf;
n = 0:40;
a = 2;b = -3;
x1 = cos(2*pi*0.1*n);
x2 = cos(2*pi*0.4*n);
x = a*x1 + b*x2;
num = [2.2403 2.4908 2.2403];
den = [1 -0.4 0.75];
ic = [0 0]; % Set zero initial conditions
y1 = filter(num,den,x1,ic); % Compute the output y1[n]
y2 = filter(num,den,x2,ic); % Compute the output y2[n]
y = filter(num,den,x,ic); % Compute the output y[n]
yt = a*y1 + b*y2;
d = y - yt; % Compute the difference output d[n]
% Plot the outputs and the difference signal
subplot(3,1,1)
stem(n,y);
ylabel('Amplitude');
title('Output Due to Weighted Input: a \cdot x_{1}[n] + b \cdot
x_{2}[n]');
subplot(3,1,2)
stem(n,yt);
ylabel('Amplitude');
title('Weighted Output: a \cdot y_{1}[n] + b \cdot y_{2}[n]');
subplot(3,1,3)
stem(n,d);
xlabel('Time index n');ylabel('Amplitude');
title('Difference Signal');
```

Answers:

Q2.7 The outputs `y[n]`, obtained with weighted input, and `yt[n]`, obtained by combining the two outputs `y1[n]` and `y2[n]` with the same weights, are shown below along with the difference between the two signals:



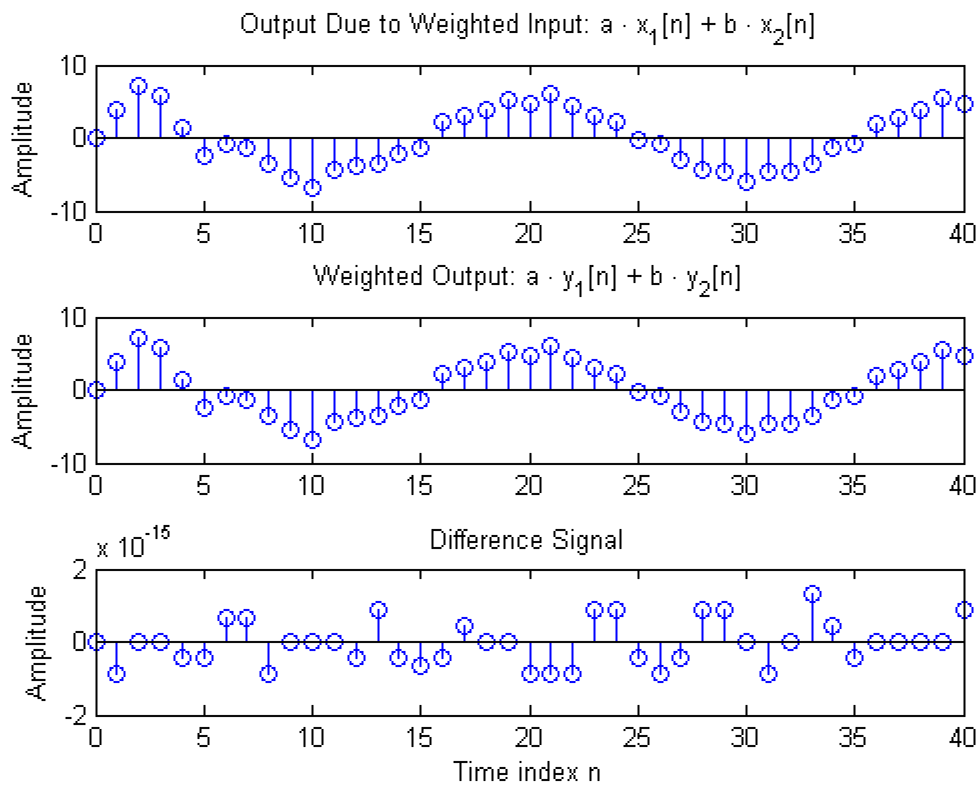
The two sequences are – the same up to numerical roundoff.

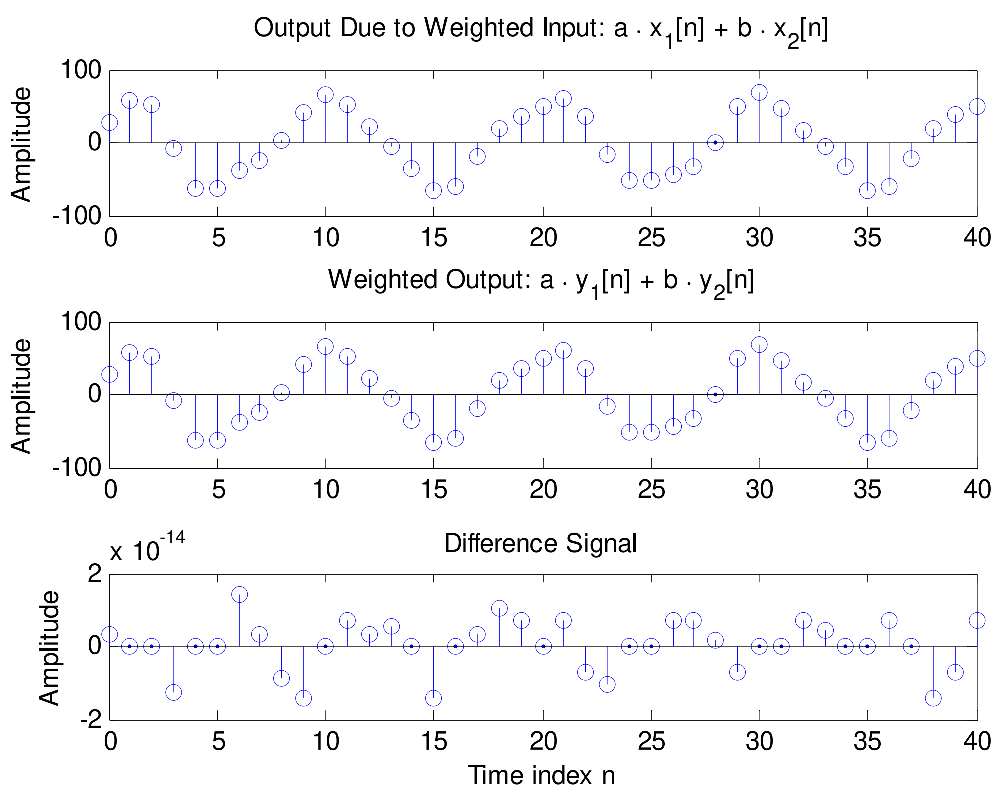
The system is – Linear.

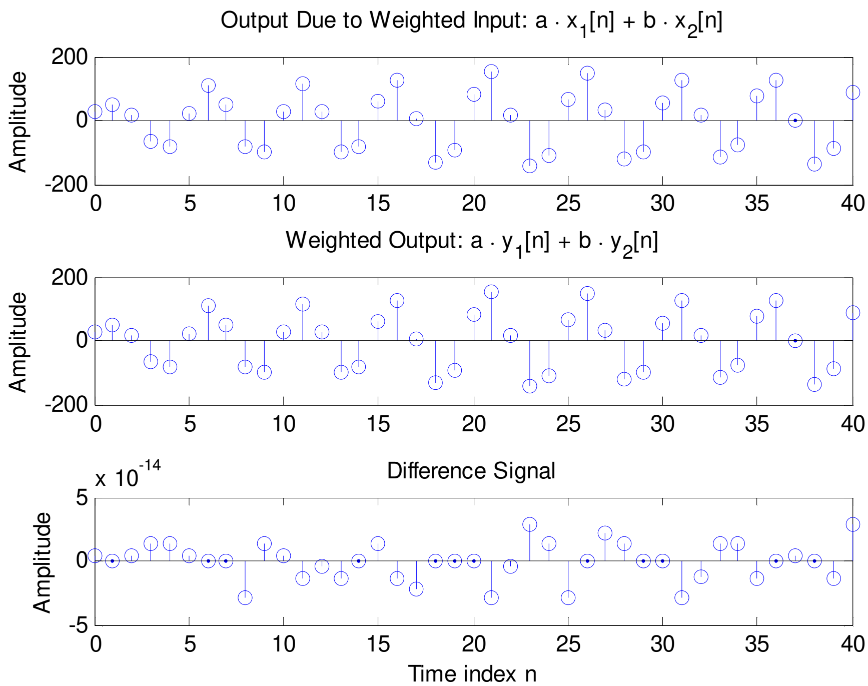
Q2.8 Program P2_3 was run for the following three different sets of values of the weighting constants, `a` and `b`, and the following three different sets of input frequencies:

1. `a=1; b=-1; f1=0.05; f2=0.4;`
2. `a=10; b=2; f1=0.10; f2=0.25;`
3. `a=2; b=10; f1=0.15; f2=0.20;`

The plots generated for each of the above three cases are shown below:


Output Due to Weighted Input: a · x1[n] + b · x2[n]
Amplitude
Weighted Output: a · y1[n] + b · y2[n]
Amplitude
x 10-15
Difference Signal
Amplitude
Time index n

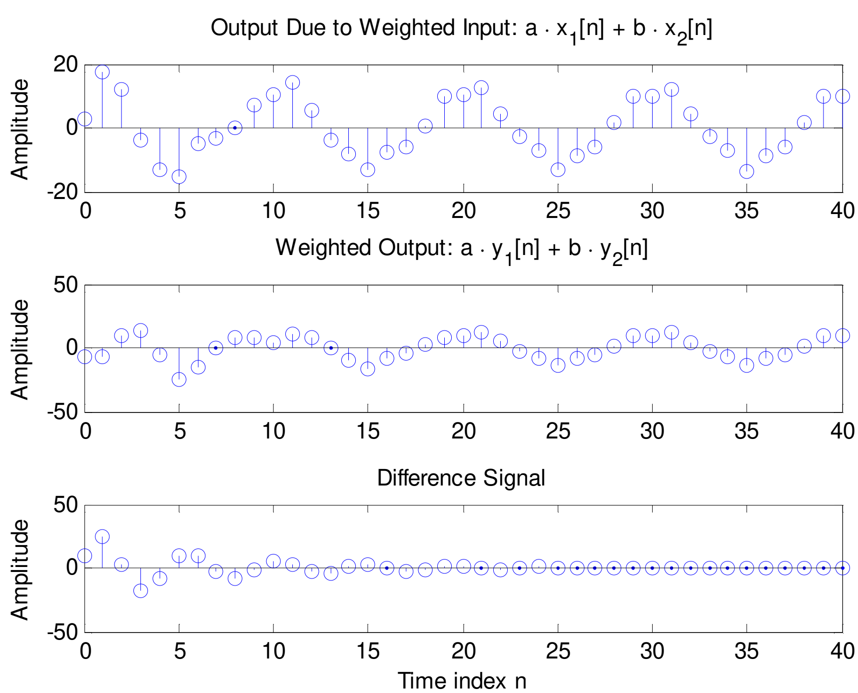

Output Due to Weighted Input: a · x1[n] + b · x2[n]
Amplitude
Weighted Output: a · y1[n] + b · y2[n]
Amplitude
x 10-14
Difference Signal
Amplitude
Time index n



Based on these plots we can conclude that the system with different weights is – Linear.

Q2.9 Program 2_3 was run with the following non-zero initial conditions – `ic = [5 10];`

The plots generated are shown below -

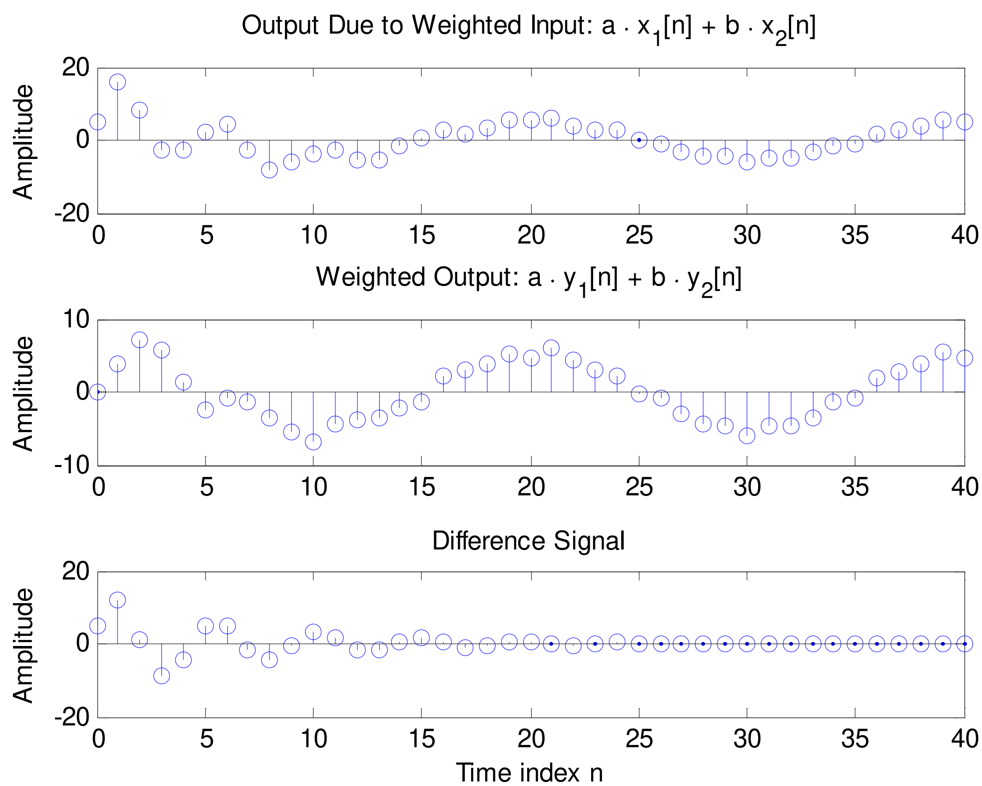


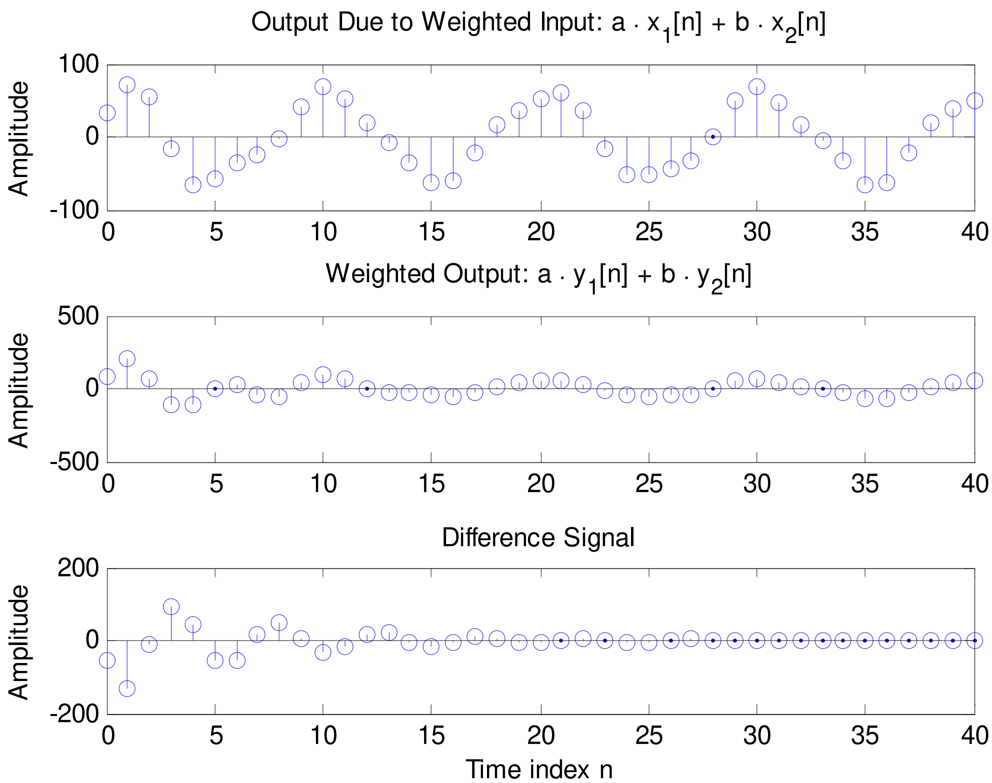
Based on these plots we can conclude that the system with nonzero initial conditions is – Nonlinear.

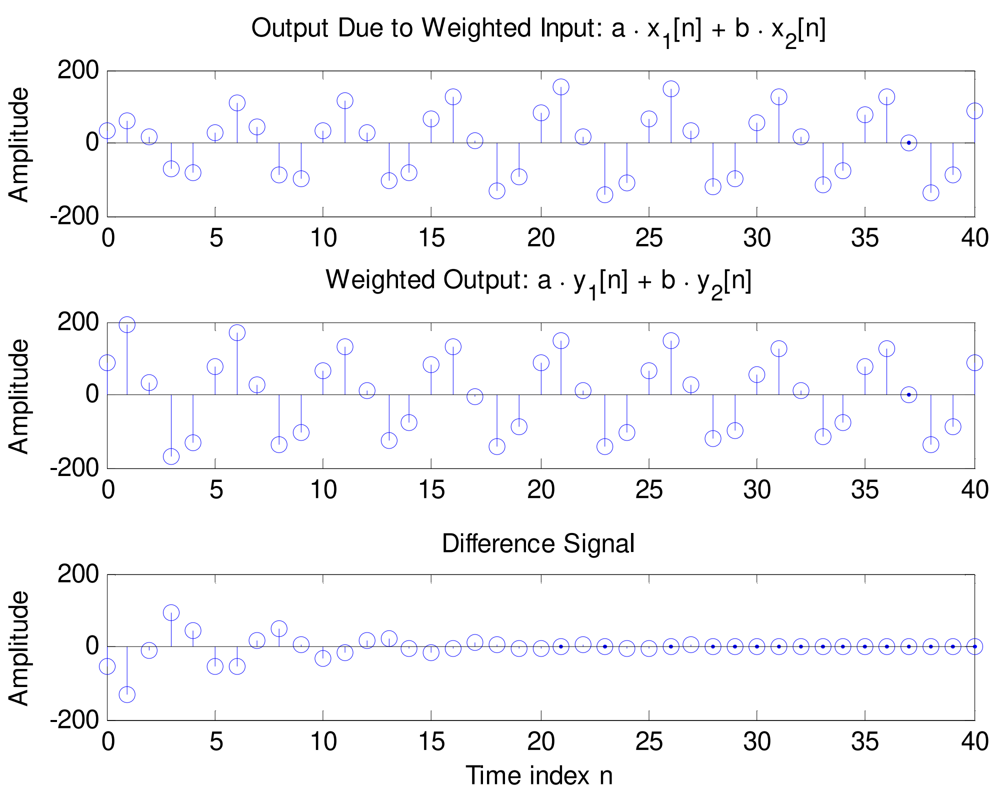
Q2.10 Program P2_3 was run with nonzero initial conditions and for the following three different sets of values of the weighting constants, `a` and `b`, and the following three different sets of input frequencies:

1. `a=1; b=-1; f1=0.05; f2=0.4;`
2. `a=10; b=2; f1=0.10; f2=0.25;`
3. `a=2; b=10; f1=0.15; f2=0.20;`

The plots generated for each of the above three cases are shown below:




Output Due to Weighted Input: a · x1[n] + b · x2[n]
Amplitude
Weighted Output: a · y1[n] + b · y2[n]
Difference Signal
Time index n

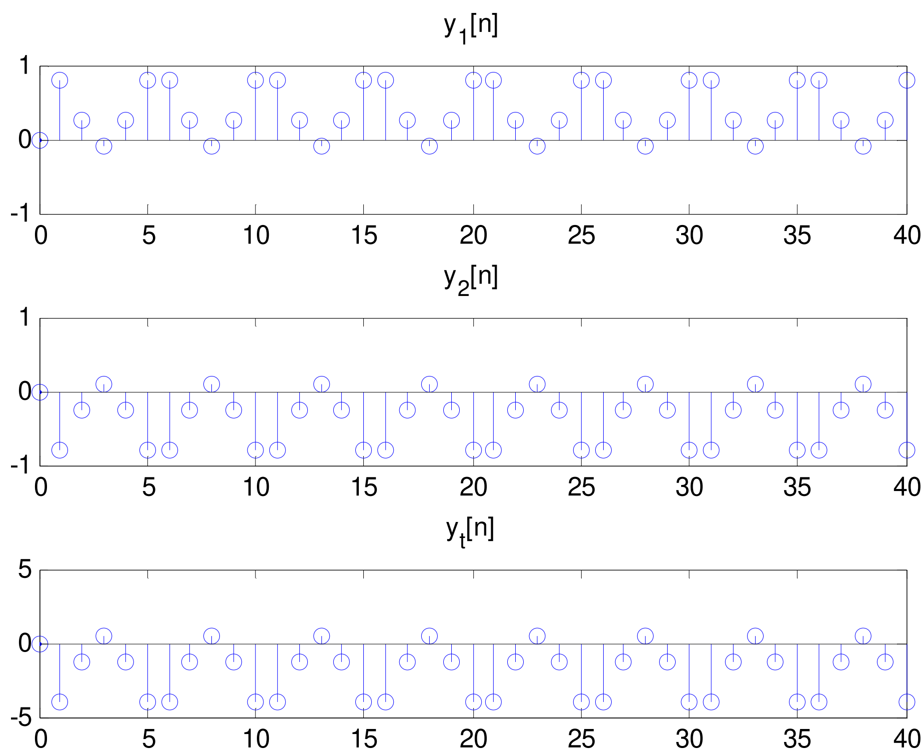

Output Due to Weighted Input: a · x1[n] + b · x2[n]
Amplitude
Weighted Output: a · y1[n] + b · y2[n]
Difference Signal
Time index n

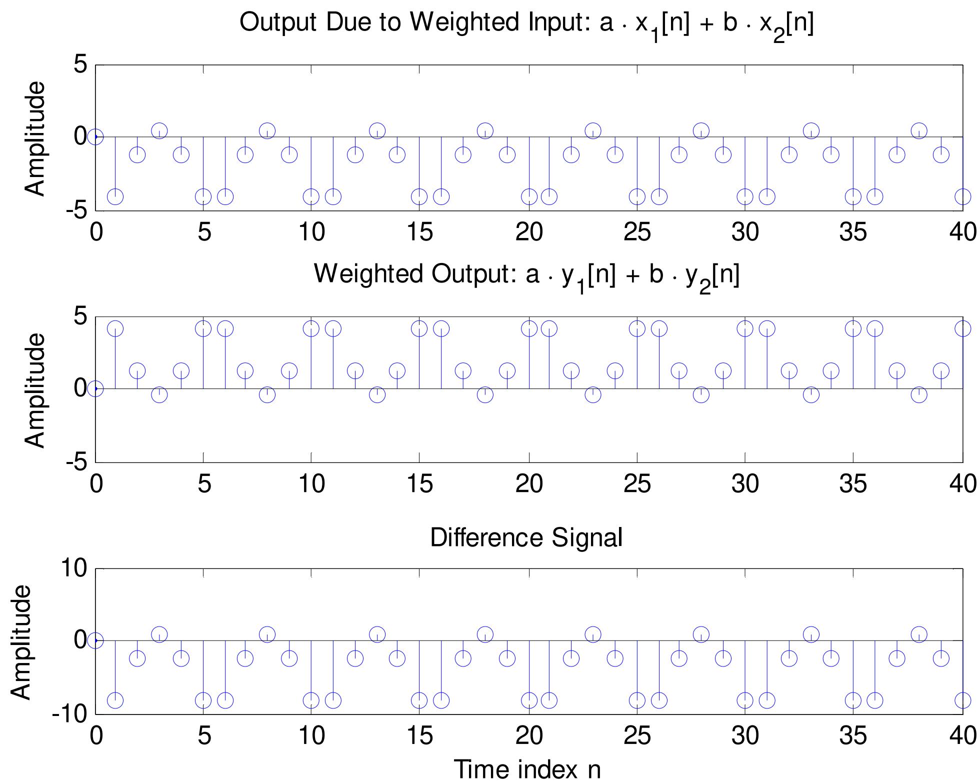
Based on these plots we can conclude that the system with nonzero initial conditions and different weights is – Nonlinear.

Q2.11 Program P2_3 was modified to simulate the system:

```
y[n] = x[n]x[n-1]
```

The output sequences `y1[n]`, `y2[n]`, and `y[n]` of the above system generated by running the modified program are shown below:





Comparing `y[n]` with `yt[n]` we conclude that the two sequences are – Not the Same.

This system is – Nonlinear.

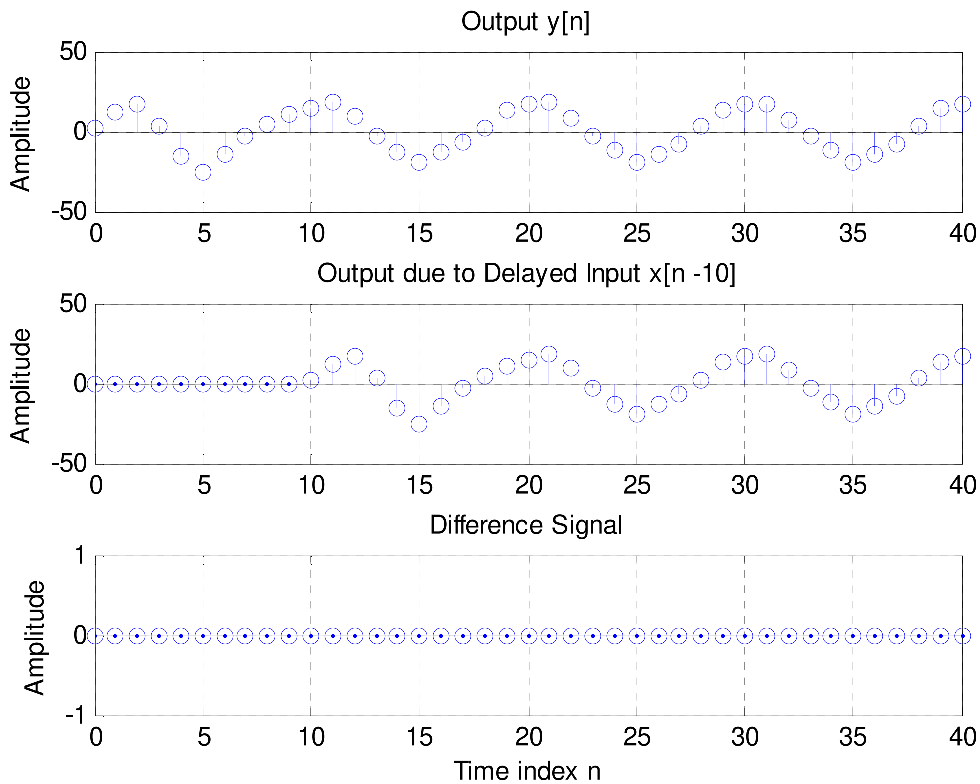
Project 2.4 Time-invariant and Time-varying Systems

A copy of Program P2_4 is given below:

```
% Program P2_4
% Generate the input sequences
clf;
n = 0:40; D = 10;a = 3.0;b = -2;
x = a*cos(2*pi*0.1*n) + b*cos(2*pi*0.4*n);
xd = [zeros(1,D) x];
num = [2.2403 2.4908 2.2403];
den = [1 -0.4 0.75];
ic = [0 0]; % Set initial conditions
% Compute the output y[n]
y = filter(num,den,x,ic);
% Compute the output yd[n]
yd = filter(num,den,xd,ic);
% Compute the difference output d[n]
d = y - yd(1+D:41+D);
% Plot the outputs
subplot(3,1,1)
stem(n,y);
ylabel('Amplitude');
title('Output y[n]'); grid;
subplot(3,1,2)
stem(n,yd(1:41));
ylabel('Amplitude');
title(['Output due to Delayed Input x[n Ð', num2str(D),']']); grid;
subplot(3,1,3)
stem(n,d);
xlabel('Time index n'); ylabel('Amplitude');
title('Difference Signal'); grid;
```

Answers:

Q2.12 The output sequences `y[n]` and `yd[n]` generated by running Program P2_4 are shown below -

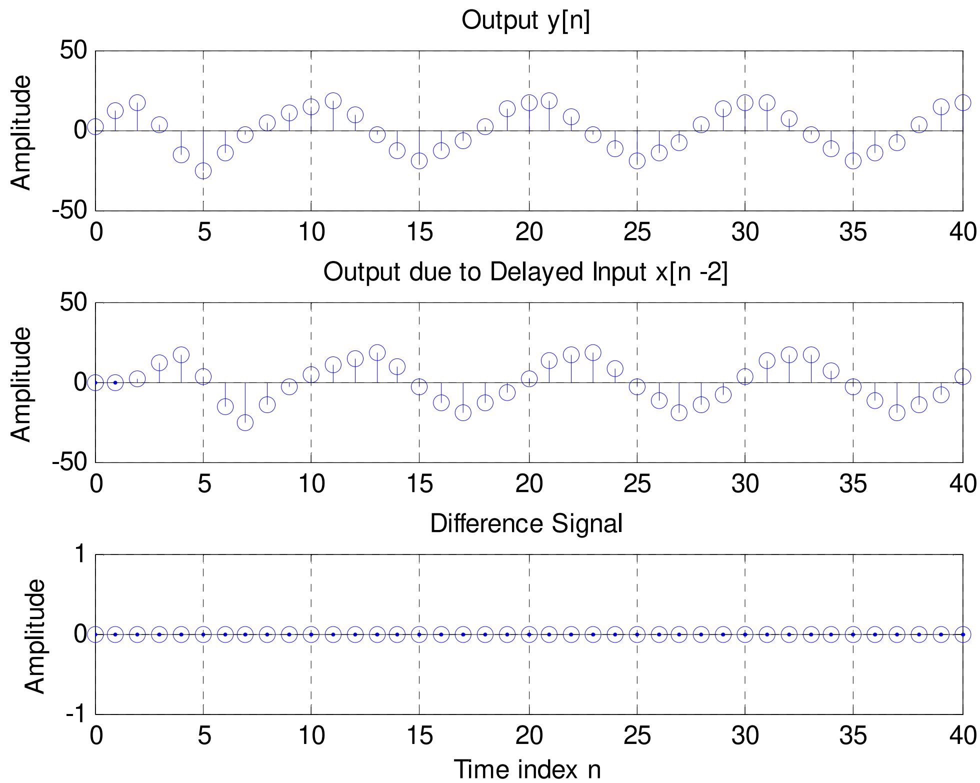


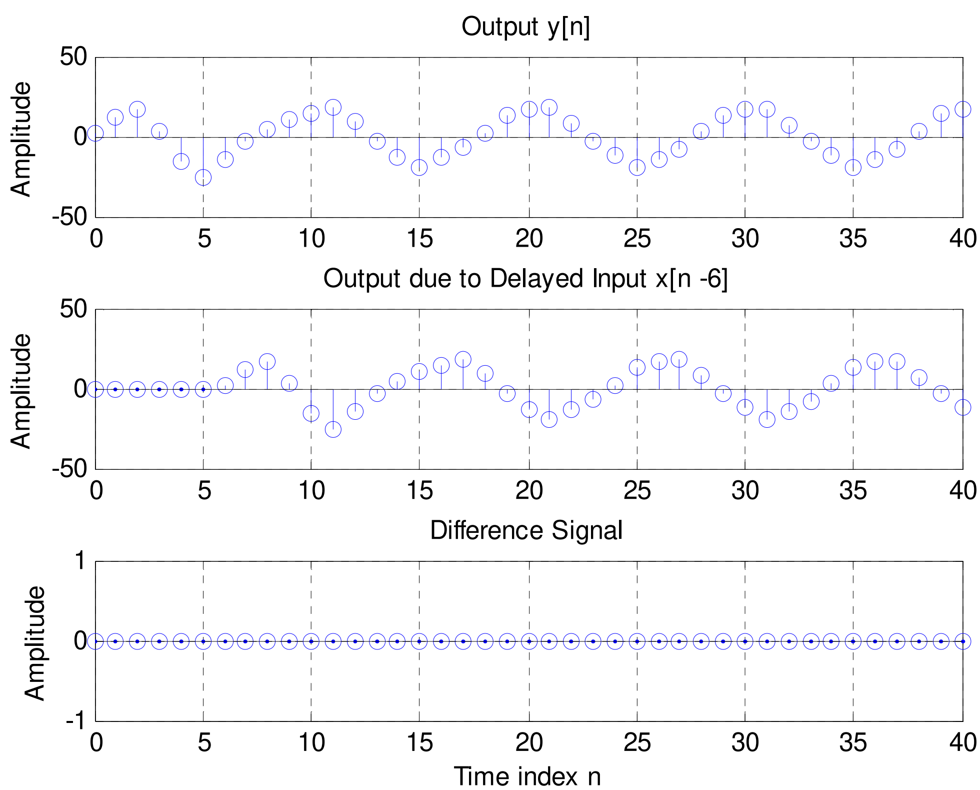
These two sequences are related as follows – `y[n-10] = yd[n].`

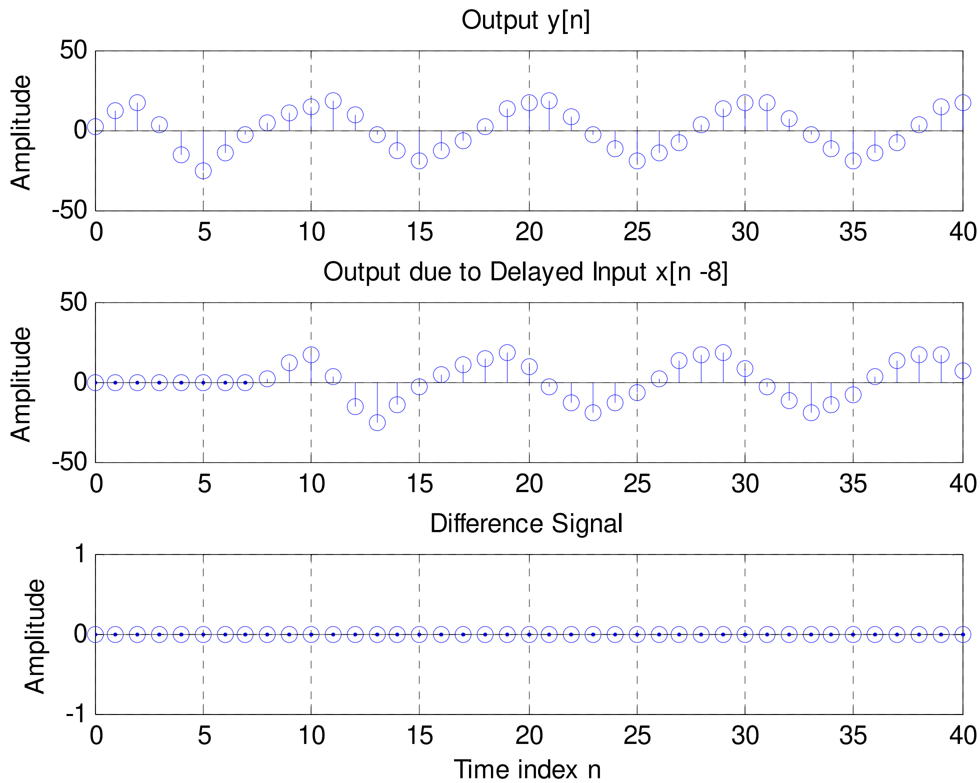
The system is – Time Invariant.

Q2.13 The output sequences `y[n]` and `yd[n]` generated by running Program P2_4 for the following values of the delay variable `D` – 2; 6; 8.

are shown below -


Output y[n]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Output due to Delayed Input x[n -2]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Difference Signal
Amplitude
1
0
-1
0 5 10 15 20 25 30 35 40
Time index n


Output y[n]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Output due to Delayed Input x[n -6]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Difference Signal
Amplitude
1
0
-1
0 5 10 15 20 25 30 35 40
Time index n



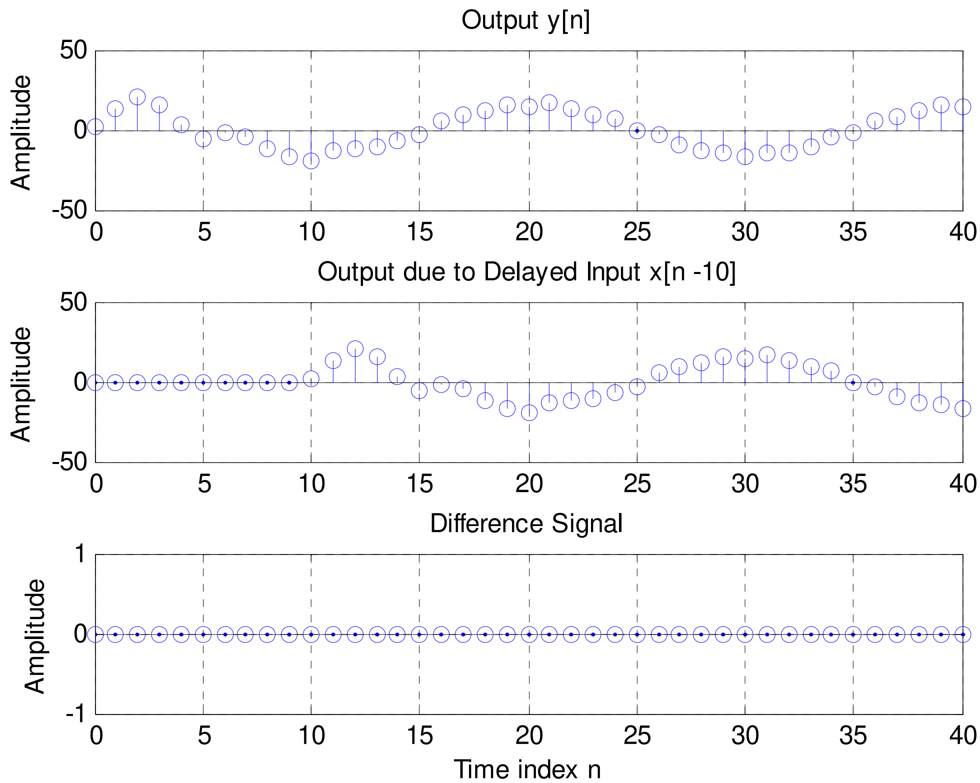
In each case, these two sequences are related as follows – `y[n-D] = yd[n]`.

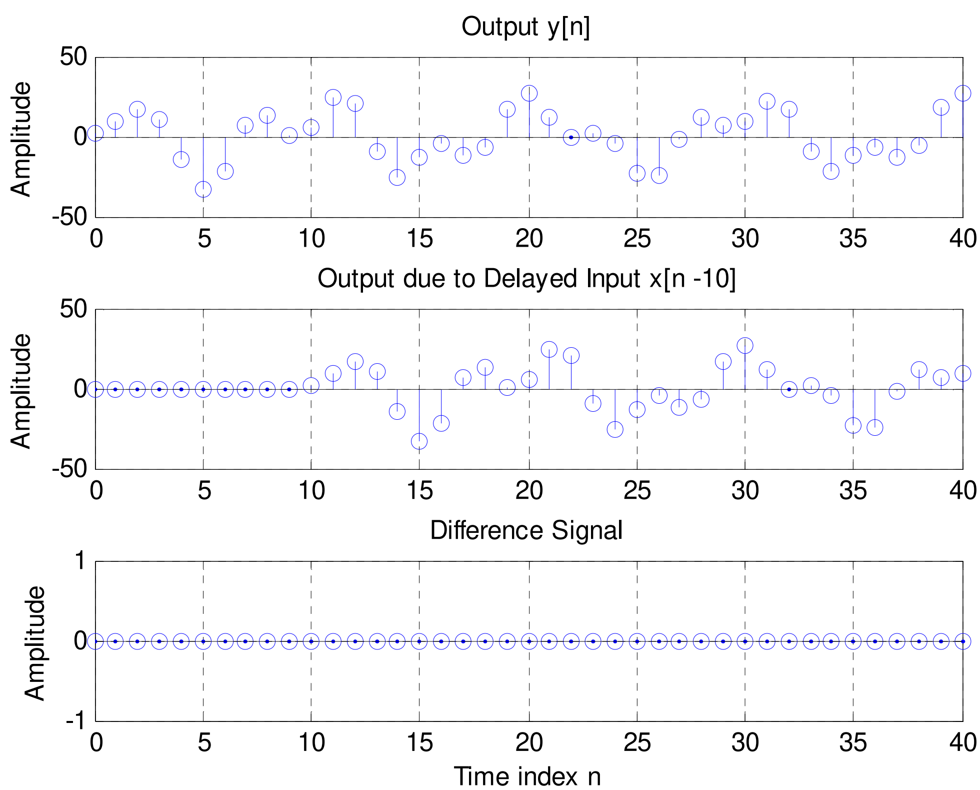
The system is – Time Invariant.

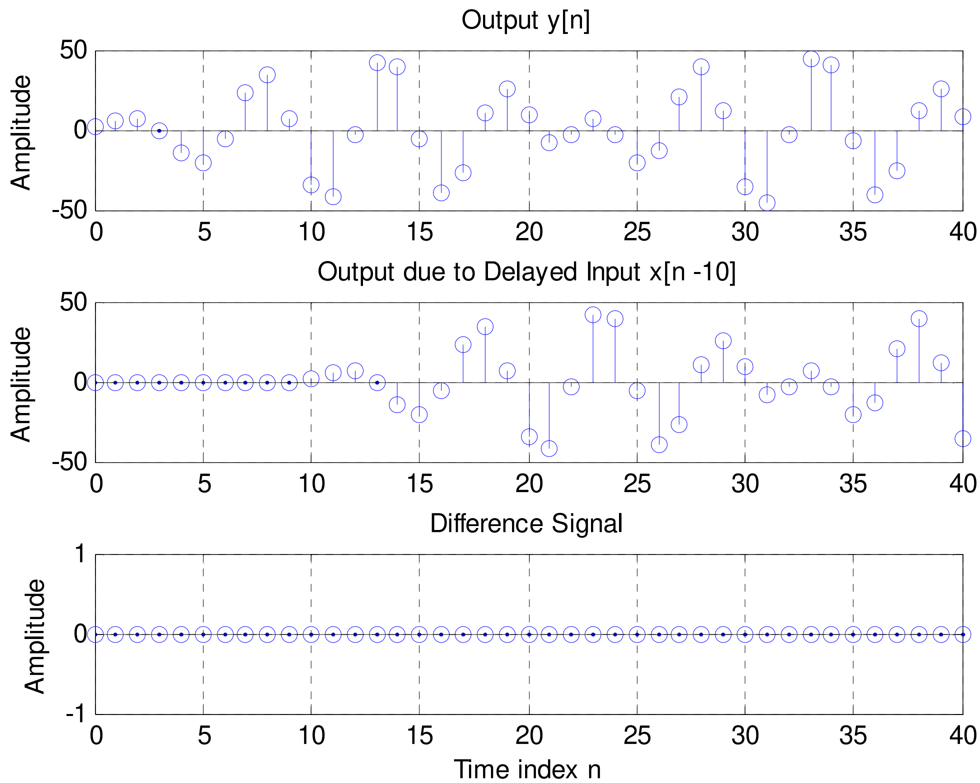
Q2.14 The output sequences `y[n]` and `yd[n]` generated by running Program P2_4 for the following values of the input frequencies –

1. `f1=0.05; f2=0.40;`
2. `f1=0.10; f2=0.25;`
3. `f1=0.15; f2=0.20;`

are shown below –


Output y[n]
50
0
-50
Amplitude
0
5
10
15
20
25
30
35
40
Output due to Delayed Input x[n -10]
50
0
-50
Amplitude
0
5
10
15
20
25
30
35
40
Difference Signal
1
0
-1
Amplitude
0
5
10
15
20
25
30
35
40
Time index n

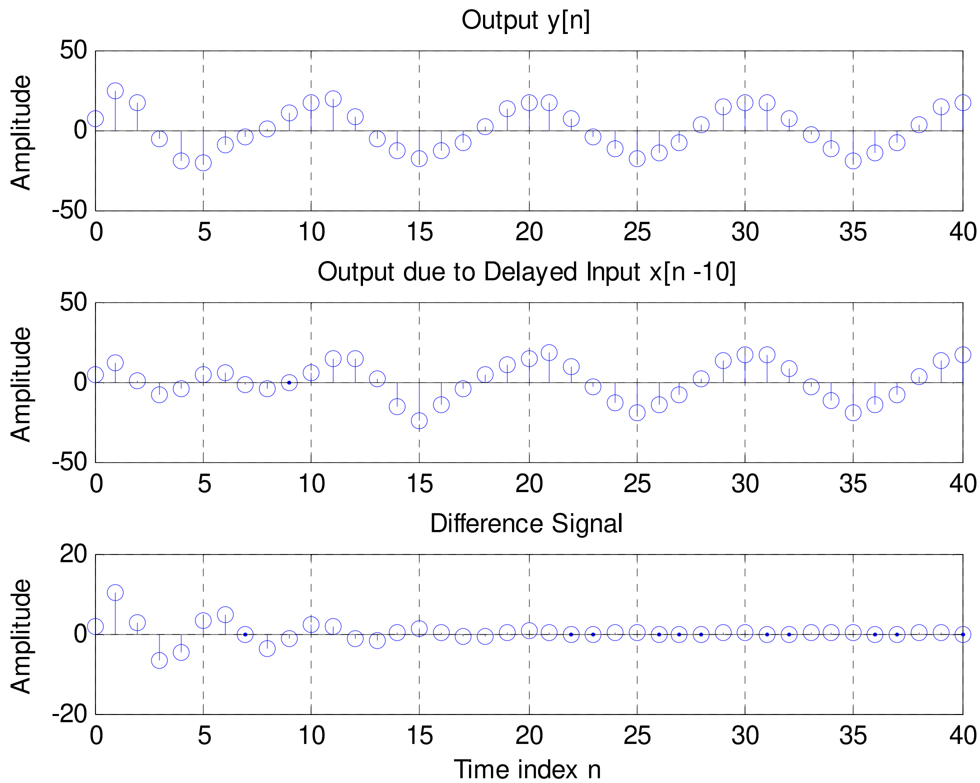

Output y[n]
50
0
-50
Amplitude
0
5
10
15
20
25
30
35
40
Output due to Delayed Input x[n -10]
50
0
-50
Amplitude
0
5
10
15
20
25
30
35
40
Difference Signal
1
0
-1
Amplitude
0
5
10
15
20
25
30
35
40
Time index n



In each case, these two sequences are related as follows – `y[n-10] = yd[n]`.

The system is – Time Invariant.

Q2.15 The output sequences `y[n]` and `yd[n]` generated by running Program P2_4 for non-zero initial conditions are shown below –



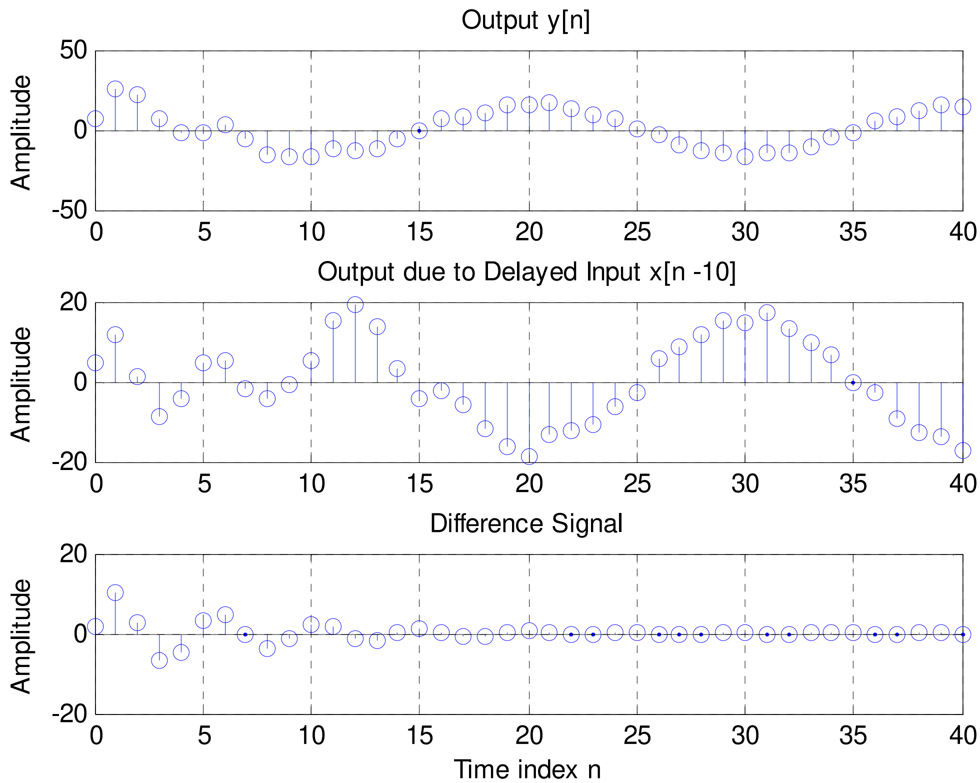
These two sequences are related as follows – `yd[n]` is NOT equal to the shift of `y[n]`.

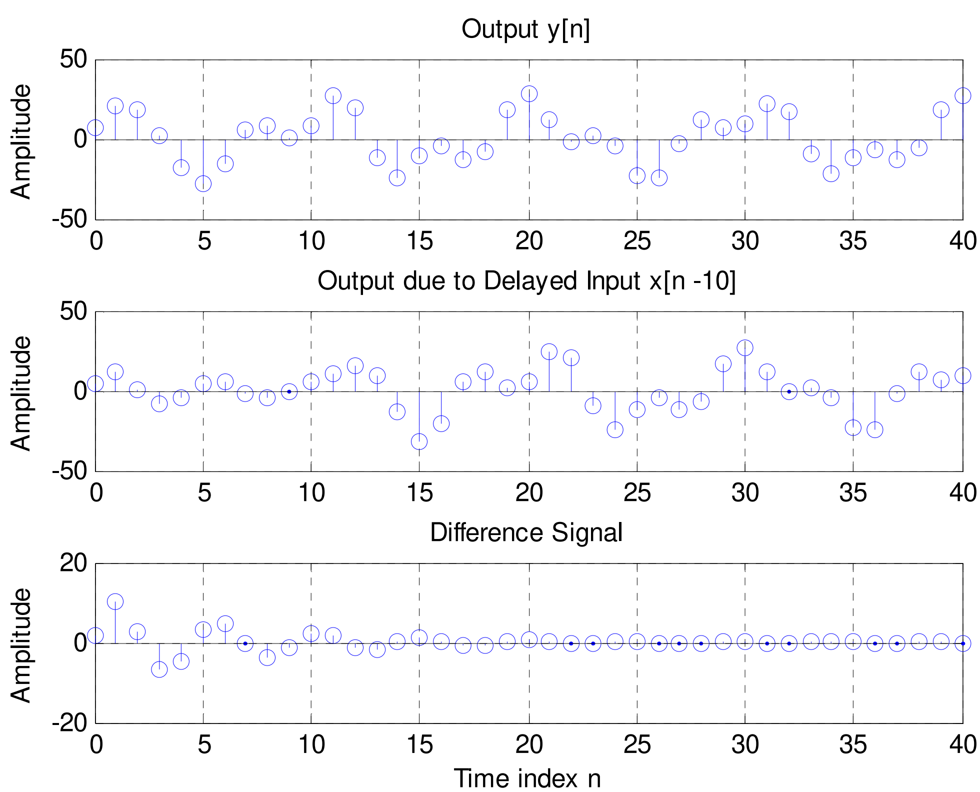
The system is – Time Varying.

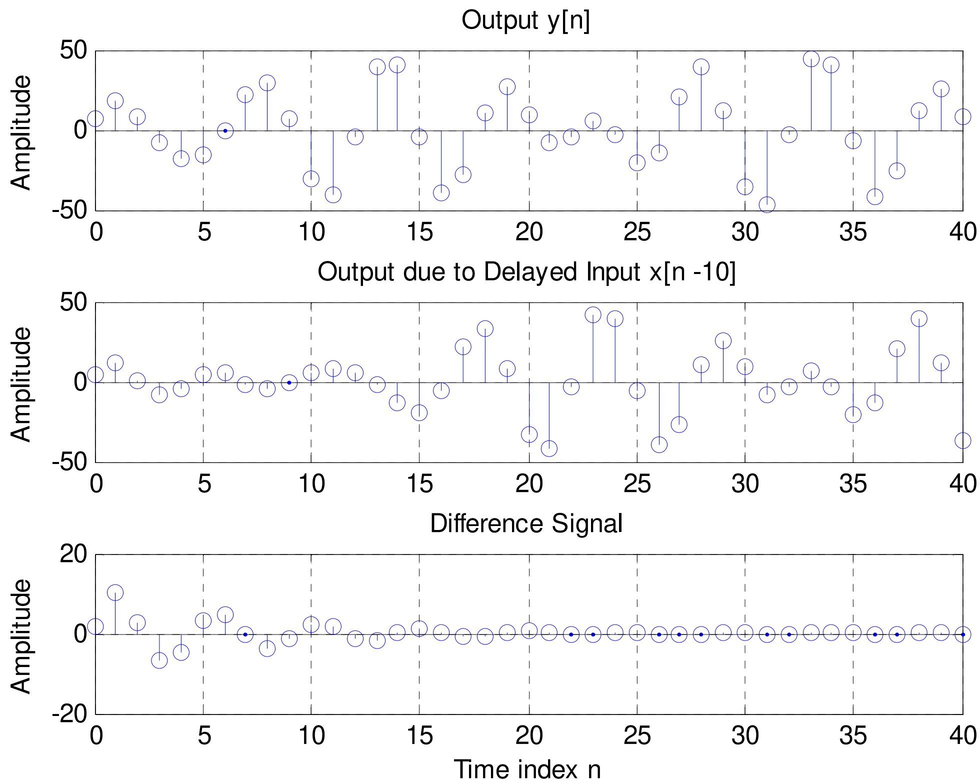
Q2.16 The output sequences `y[n]` and `yd[n]` generated by running Program P2_4 for non-zero initial conditions and following values of the input frequencies –

1. `f1=0.05; f2=0.40;`
2. `f1=0.10; f2=0.25;`
3. `f1=0.15; f2=0.20;`

are shown below -


Output y[n]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Output due to Delayed Input x[n -10]
Amplitude
20
0
-20
0 5 10 15 20 25 30 35 40
Difference Signal
Amplitude
20
0
-20
0 5 10 15 20 25 30 35 40
Time index n


Output y[n]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Output due to Delayed Input x[n -10]
Amplitude
50
0
-50
0 5 10 15 20 25 30 35 40
Difference Signal
Amplitude
20
0
-20
0 5 10 15 20 25 30 35 40
Time index n



In each case, these two sequences are related as follows – `yd[n]` is NOT given by the shift of `y[n]`.

The system is – Time Varying.

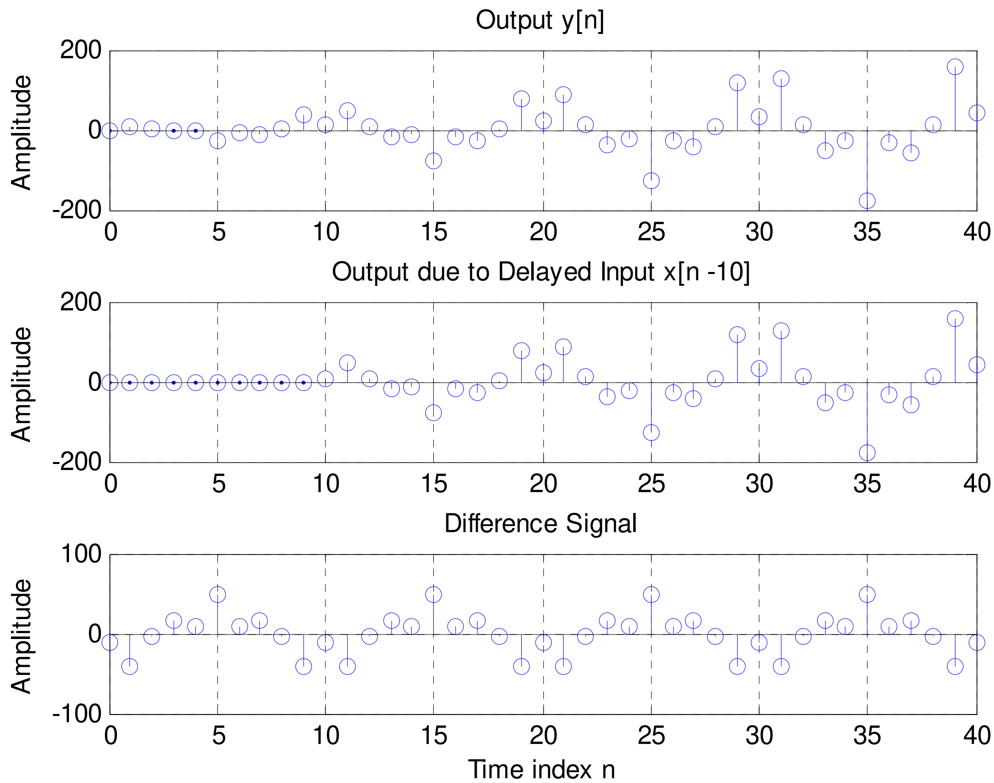
Q2.17 The modified Program 2_4 simulating the system

$$y[n] = n\ x[n] + x[n-1]$$

is given below:

```
% Program Q2_17
% Modification of P2_4 to implement the system
%    given by (2.16).
% Generate the input sequences
clf;
n = 0:40; D = 10;a = 3.0;b = -2;
x = a*cos(2*pi*0.1*n) + b*cos(2*pi*0.4*n);
xd = [zeros(1,D) x];
nd = 0:length(xd)-1;
% Compute the output y[n]
y = (n .* x) + [0 x(1:40)];
% Compute the output yd[n]
yd = (nd .* xd) + [0 xd(1:length(xd)-1)];
% Compute the difference output d[n]
d = y - yd(1+D:41+D);
% Plot the outputs
subplot(3,1,1)
stem(n,y);
ylabel('Amplitude');
title('Output y[n]'); grid;
subplot(3,1,2)
stem(n,yd(1:41));
ylabel('Amplitude');
title(['Output due to Delayed Input x[n -', num2str(D),']']); grid;
subplot(3,1,3)
stem(n,d);
xlabel('Time index n'); ylabel('Amplitude');
title('Difference Signal'); grid;
```

The output sequences `y[n]` and `yd[n]` generated by running modified Program P2_4 are shown below -



These two sequences are related as follows – `yd[n]` is NOT the shifted version of `y[n]`.

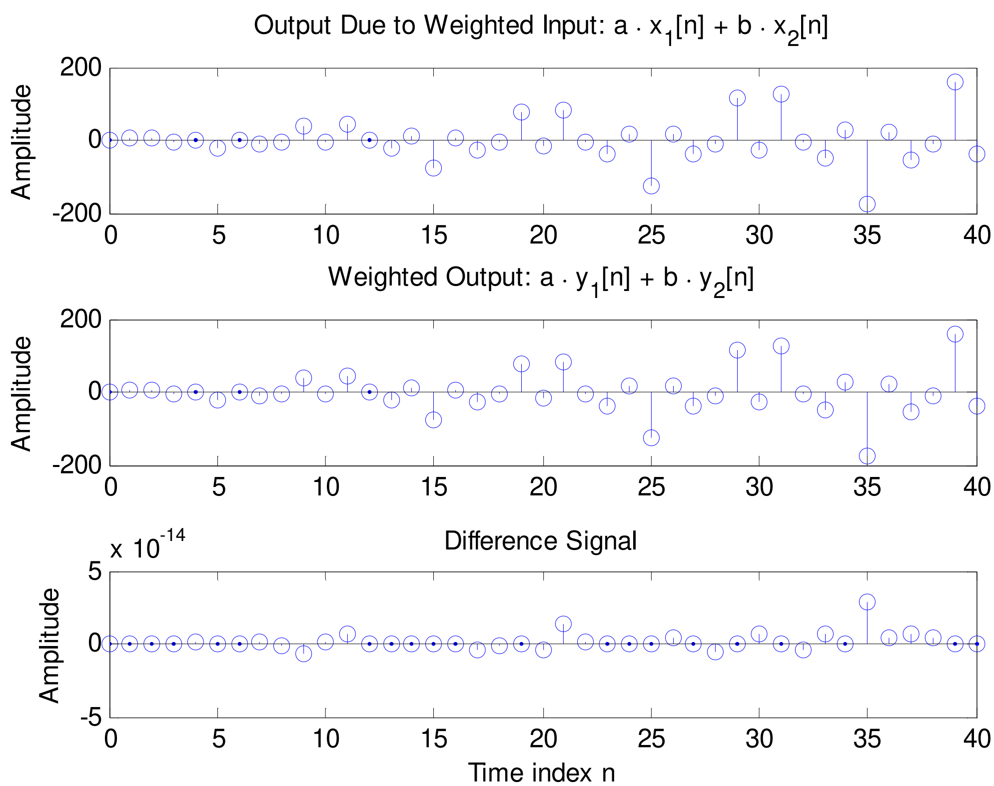
The system is – Time Varying.

Q2.18 (optional) The modified Program P2_3 to test the linearity of the system of (2.16) is shown below:

```
% Program Q2_18
% Modify P2_3 for Q2.18.
% Generate the input sequences
clf;
n = 0:40;
a = 2;b = -3;
x1 = cos(2*pi*0.1*n);
x2 = cos(2*pi*0.4*n);
x = a*x1 + b*x2;
y1 = (n .* x1) + [0 x1(1:40)]; % Compute the output y1[n]
y2 = (n .* x2) + [0 x2(1:40)]; % Compute the output y2[n]
y  = (n .* x)  + [0 x(1:40)];  % Compute the output y[n]
yt = a*y1 + b*y2;
d = y - yt; % Compute the difference output d[n]
% Plot the outputs and the difference signal
subplot(3,1,1)
stem(n,y);
ylabel('Amplitude');
title('Output Due to Weighted Input: a \cdot x_{1}[n] + b \cdot
x_{2}[n]');
```

```
subplot(3,1,2)
stem(n,yt);
ylabel('Amplitude');
title('Weighted Output: a \cdot y_{1}[n] + b \cdot y_{2}[n]');
subplot(3,1,3)
stem(n,d);
xlabel('Time index n');ylabel('Amplitude');
title('Difference Signal');
```

The outputs `y[n]` and `yt[n]` obtained by running the modified program P2_3 are shown below:



The two sequences are – The same up to numerical roundoff.

The system is – Linear.

2.5 LINEAR TIME-INVARIANT DISCRETE-TIME SYSTEMS

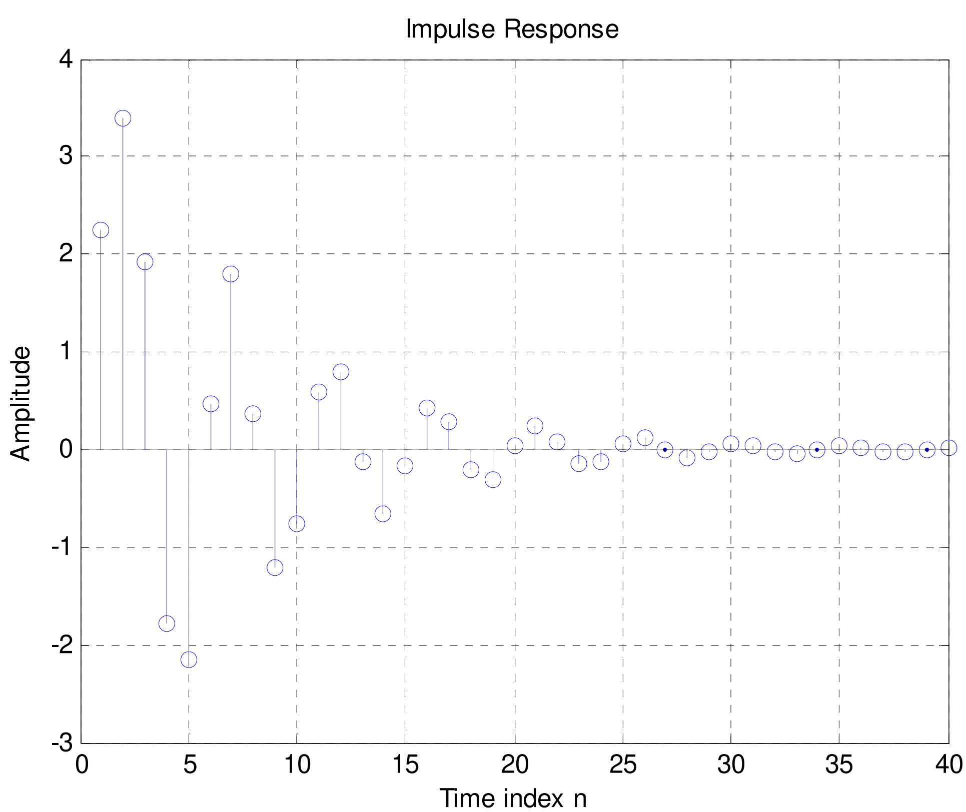
Project 2.5 Computation of Impulse Responses of LTI Systems

A copy of Program P2_5 is shown below:

```
% Program P2_5
% Compute the impulse response y
clf;
N = 40;
num = [2.2403 2.4908 2.2403];
den = [1 -0.4 0.75];
y = impz(num,den,N);
% Plot the impulse response
stem(y);
xlabel('Time index n'); ylabel('Amplitude');
title('Impulse Response'); grid;
```

Answers:

Q2.19 The first 40 samples of the impulse response of the discrete-time system of Project 2.3 generated by running Program P2_5 is given below:



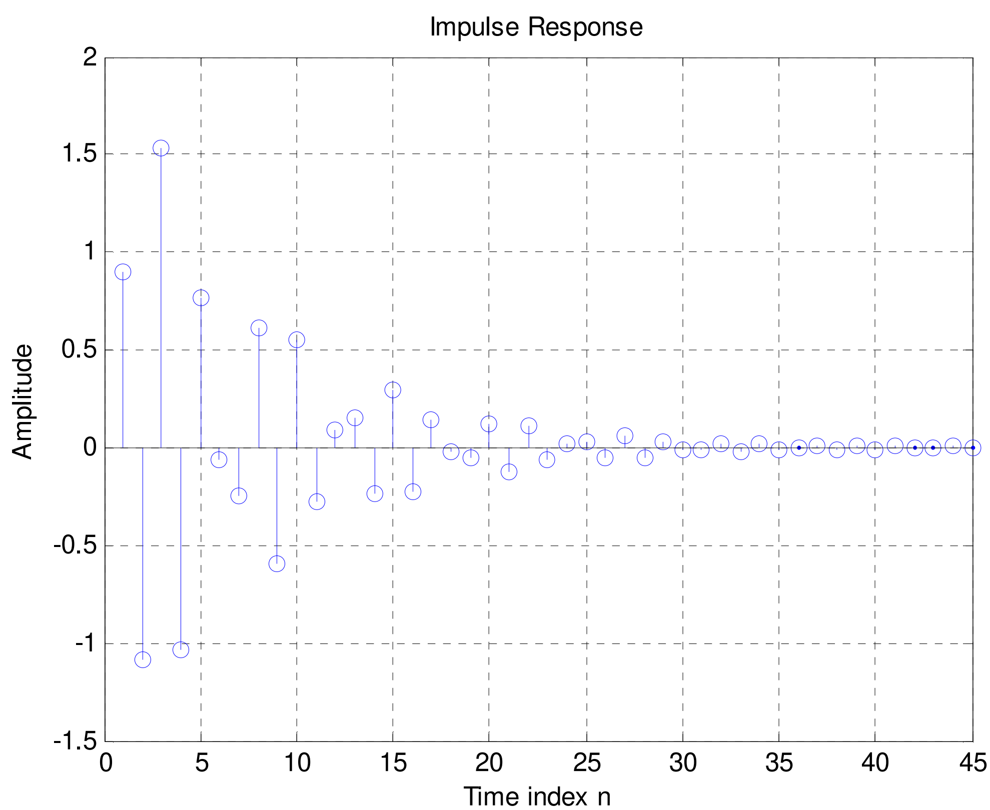
Q2.20 The required modifications to Program P2_5 to generate the impulse response of the following causal LTI system:

$$y[n] + 0.71y[n-1] - 0.46y[n-2] - 0.62y[n-3]$$
$$= 0.9x[n] - 0.45x[n-1] + 0.35x[n-2] + 0.002x[n-3]$$

are given below:

```
% Program Q2_20
% Compute the impulse response y
clf;
N = 45;
num = [0.9 -0.45  0.35  0.002];
den = [1.0  0.71 -0.46 -0.62];
y = impz(num,den,N);
% Plot the impulse response
stem(y);
xlabel('Time index n'); ylabel('Amplitude');
title('Impulse Response'); grid;
```

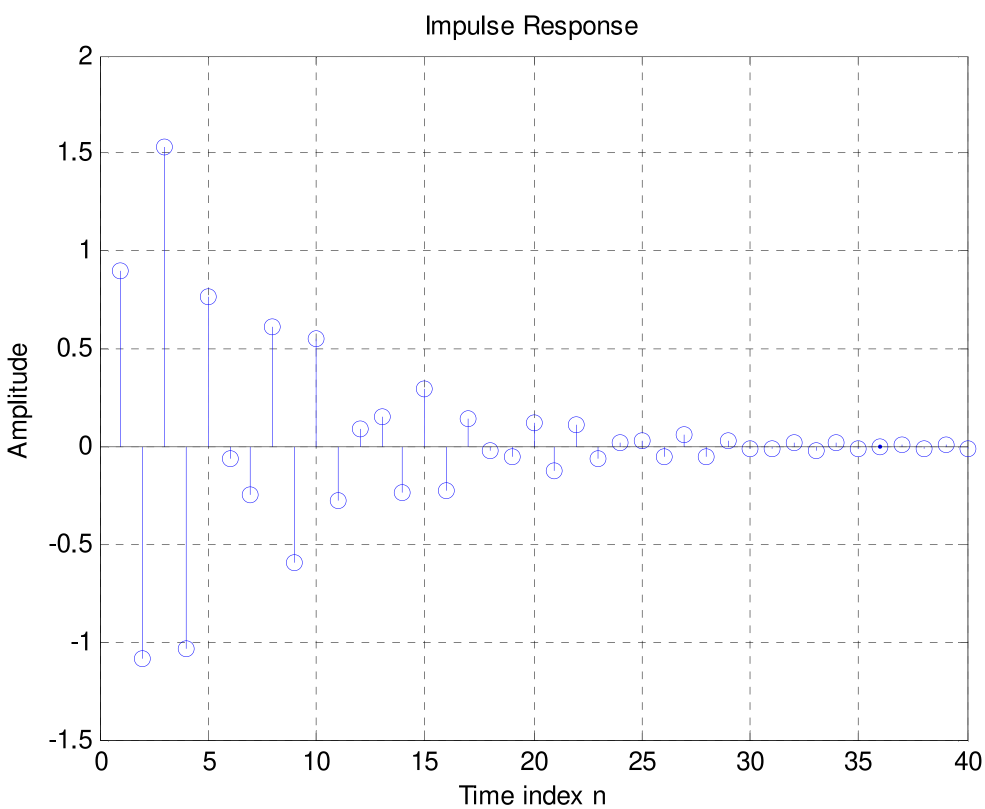
The first 45 samples of the impulse response of this discrete-time system generated by running the modified is given below:



Q2.21 The MATLAB program to generate the impulse response of a causal LTI system of Q2.20 using the `filter` command is indicated below:

```
% Program Q2_21
% Compute the impulse response y
clf;
N = 40;
num = [0.9 -0.45  0.35  0.002];
den = [1.0  0.71 -0.46 -0.62];
% input: unit pulse
x = [1 zeros(1,N-1)];
% output
y = filter(num,den,x);
% Plot the impulse response
% NOTE: the time axis will be WRONG; h[0] will
% be plotted at n=1; but this will agree with
% the INCORRECT plotting that was also done
% by program P2_5.
stem(y);
xlabel('Time index n'); ylabel('Amplitude');
title('Impulse Response'); grid;
```

The first 40 samples of the impulse response generated by this program are shown below:

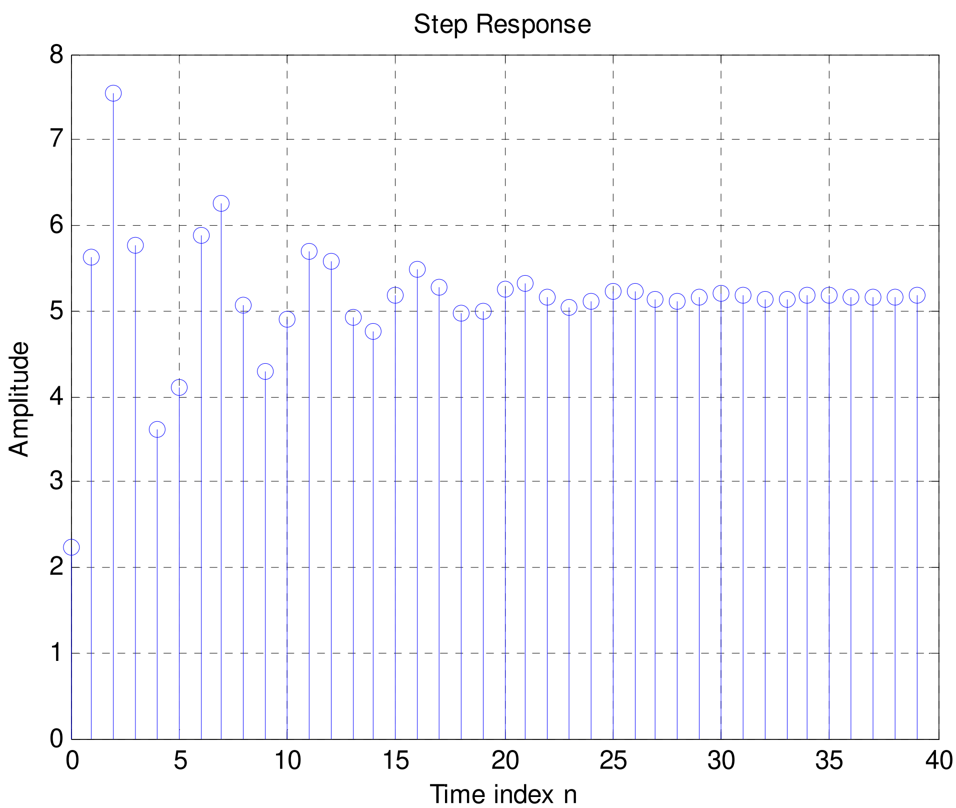


Comparing the above response with that obtained in Question Q2.20 we conclude - They are the SAME.

Q2.22 The MATLAB program to generate and plot the step response of a causal LTI system is indicated below:

```
% Program Q2_22
% Compute the step response s
clf;
N = 40;
n = 0:N-1;
num = [2.2403  2.4908  2.2403];
den = [1.0    -0.4     0.75];
% input: unit step
x = [ones(1,N)];
% output
y = filter(num,den,x);
% Plot the step response
stem(n,y);
xlabel('Time index n'); ylabel('Amplitude');
title('Step Response'); grid;
```

The first 40 samples of the step response of the LTI system of Project 2.3 are shown below:



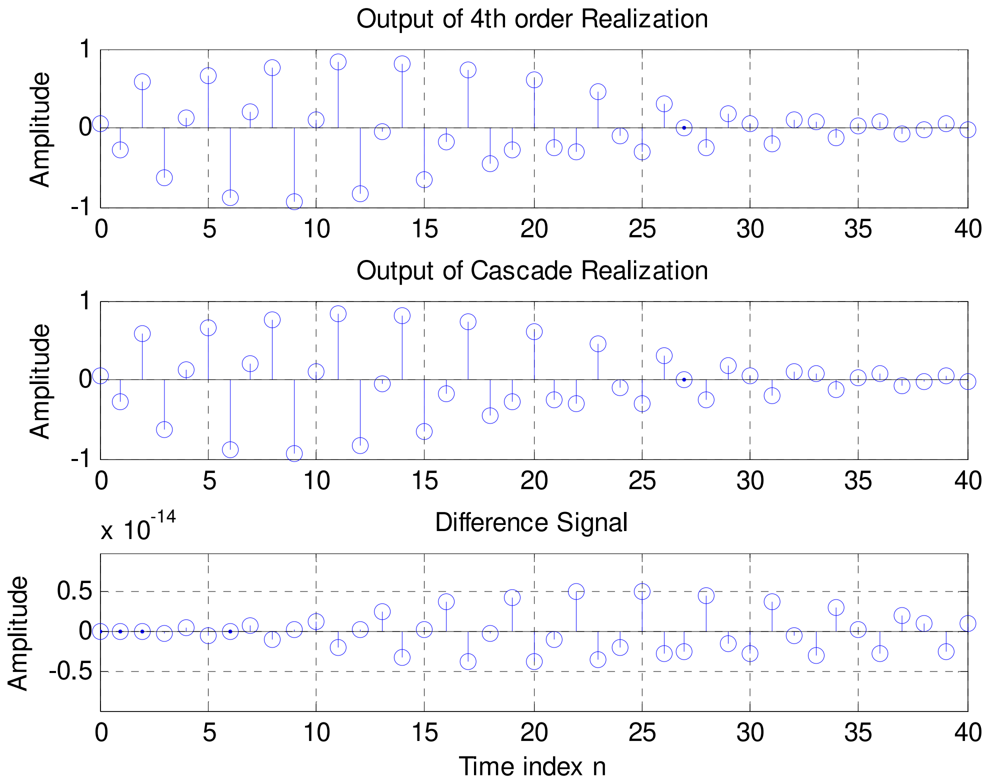
Project 2.6 Cascade of LTI Systems

A copy of Program P2_6 is given below:

```
% Program P2_6
% Cascade Realization
clf;
x = [1 zeros(1,40)]; % Generate the input
n = 0:40;
% Coefficients of 4th order system
den = [1 1.6 2.28 1.325 0.68];
num = [0.06 -0.19 0.27 -0.26 0.12];
% Compute the output of 4th order system
y = filter(num,den,x);
% Coefficients of the two 2nd order systems
num1 = [0.3 -0.2 0.4];den1 = [1 0.9 0.8];
num2 = [0.2 -0.5 0.3];den2 = [1 0.7 0.85];
% Output y1[n] of the first stage in the cascade
y1 = filter(num1,den1,x);
% Output y2[n] of the second stage in the cascade
y2 = filter(num2,den2,y1);
% Difference between y[n] and y2[n]
d = y - y2;
% Plot output and difference signals
subplot(3,1,1);
stem(n,y);
ylabel('Amplitude');
title('Output of 4th order Realization'); grid;
subplot(3,1,2);
stem(n,y2)
ylabel('Amplitude');
title('Output of Cascade Realization'); grid;
subplot(3,1,3);
stem(n,d)
xlabel('Time index n');ylabel('Amplitude');
title('Difference Signal'); grid;
```

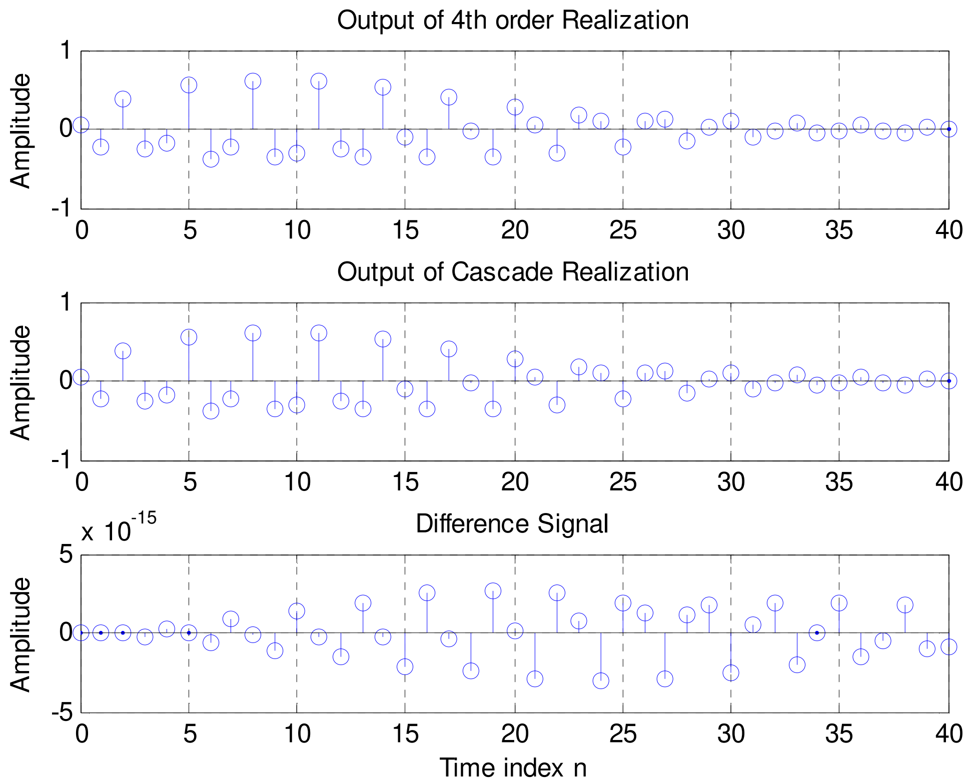
Answers:

Q2.23 The output sequences `y[n]`, `y2[n]`, and the difference signal `d[n]` generated by running Program P2_6 are indicated below:



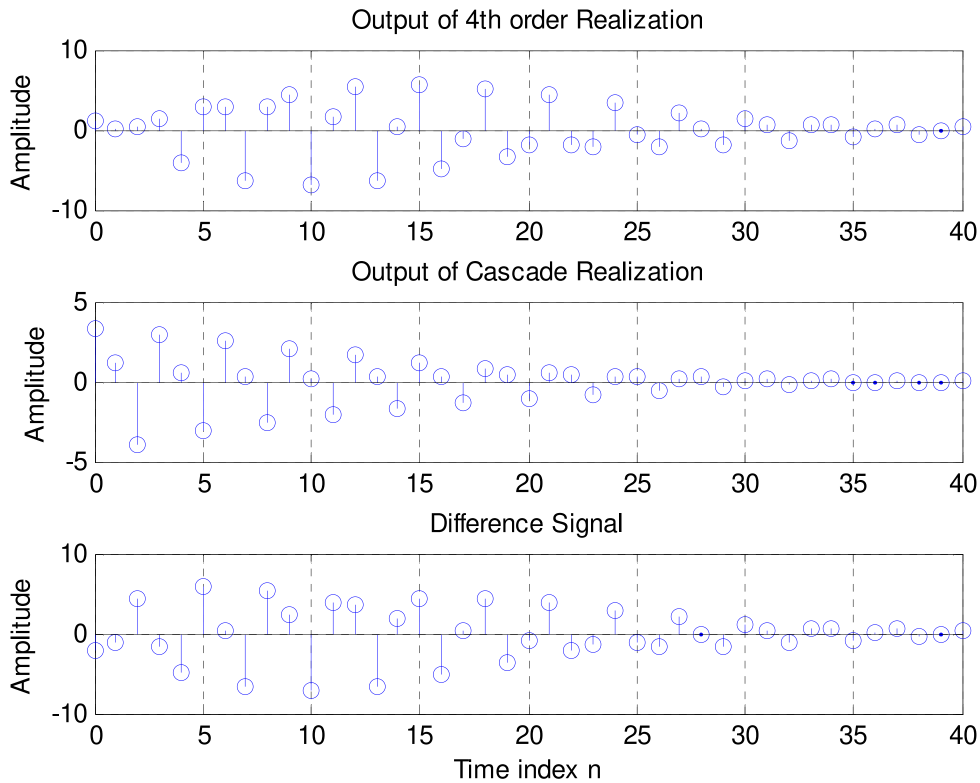
The relation between `y[n]` and `y2[n]` is – They are the SAME up to numerical roundoff.

Q2.24 The sequences generated by running Program P2_6 with the input changed to a sinusoidal sequence are as follows:



The relation between `y[n]` and `y2[n]` in this case is – The are the same up to numerical roundoff.

Q2.25 The sequences generated by running Program P2_6 with non-zero initial condition vectors are now as given below:



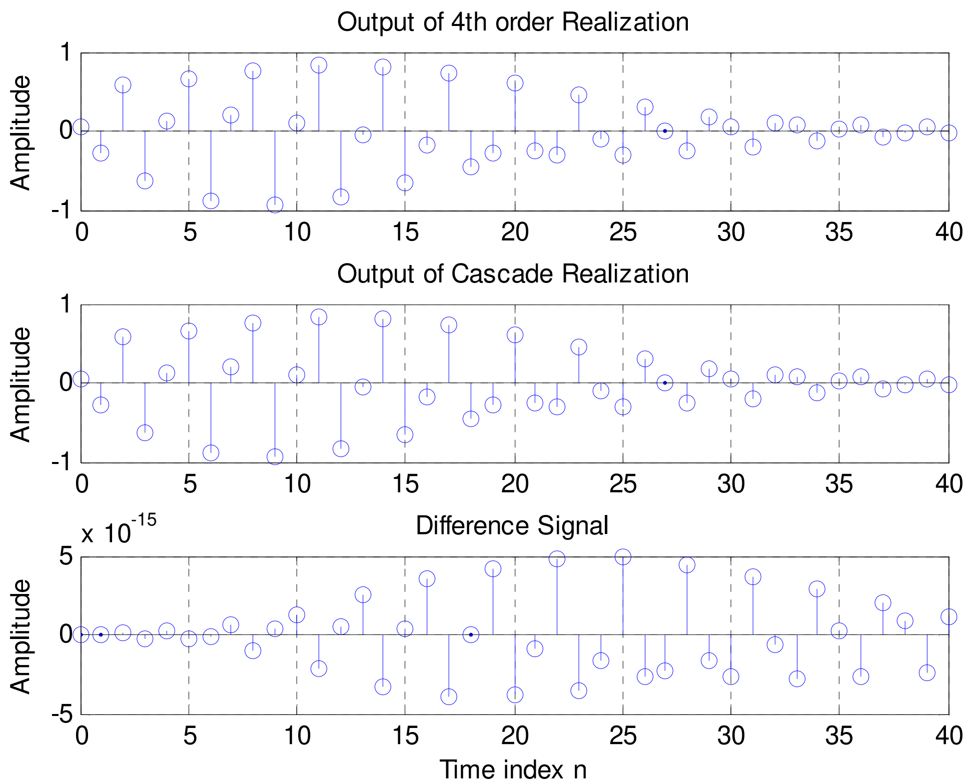
The relation between `y[n]` and `y2[n]` in this case is – They are NOT the same.

Q2.26 The modified Program P2_6 with the two 2nd-order systems in reverse order and with zero initial conditions is displayed below:

```
% Program Q2_26
% Cascade Realization
clf;
x = [1 zeros(1,40)]; % Generate the input
n = 0:40;
% Coefficients of 4th order system
den = [1 1.6 2.28 1.325 0.68];
num = [0.06 -0.19 0.27 -0.26 0.12];
% Compute the output of 4th order system
y = filter(num,den,x);
% Coefficients of the two 2nd order systems
num1 = [0.3 -0.2 0.4];den1 = [1 0.9 0.8];
num2 = [0.2 -0.5 0.3];den2 = [1 0.7 0.85];
% Output y1[n] of the first stage in the cascade
y1 = filter(num2,den2,x);
% Output y2[n] of the second stage in the cascade
y2 = filter(num1,den1,y1);
% Difference between y[n] and y2[n]
d = y - y2;
% Plot output and difference signals
subplot(3,1,1);
stem(n,y);
ylabel('Amplitude');
```

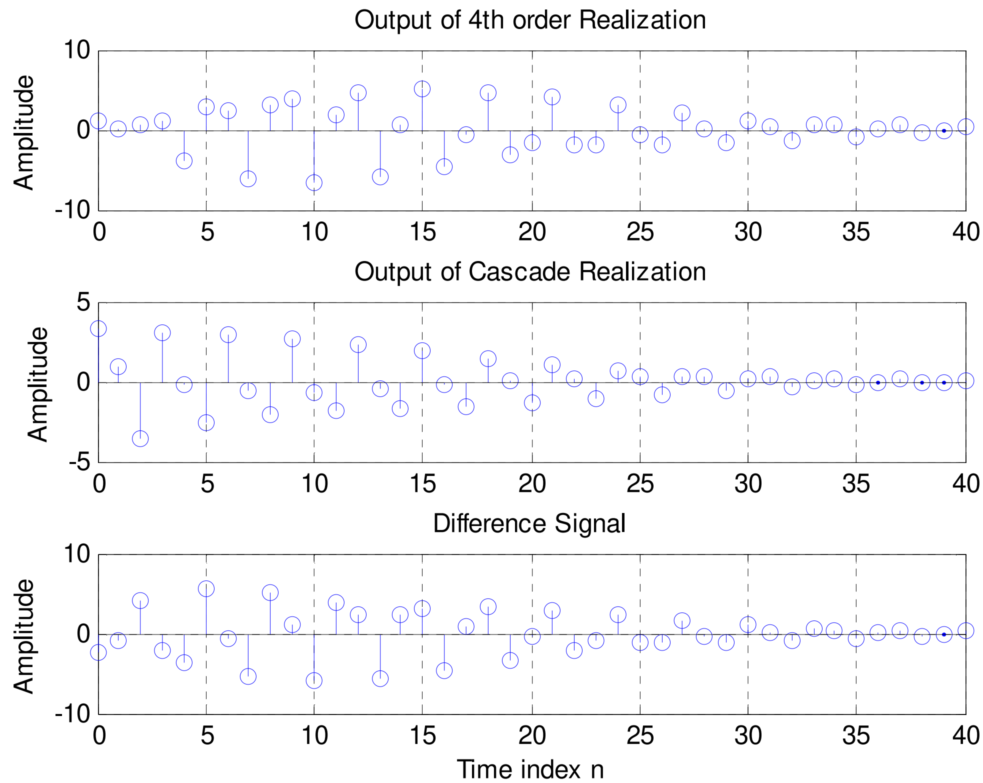
```
title('Output of 4th order Realization'); grid;
subplot(3,1,2);
stem(n,y2)
ylabel('Amplitude');
title('Output of Cascade Realization'); grid;
subplot(3,1,3);
stem(n,d)
xlabel('Time index n');ylabel('Amplitude');
title('Difference Signal'); grid;
```

The sequences generated by running the modified program are sketched below:



The relation between `y[n]` and `y2[n]` in this case is – They are the SAME up to numerical roundoff.

Q2.27 The sequences generated by running the modified Program P2_6 with the two 2nd-order systems in reverse order and with non-zero initial conditions are displayed below:



The relation between `y[n]` and `y2[n]` in this case is – They are NOT the same.

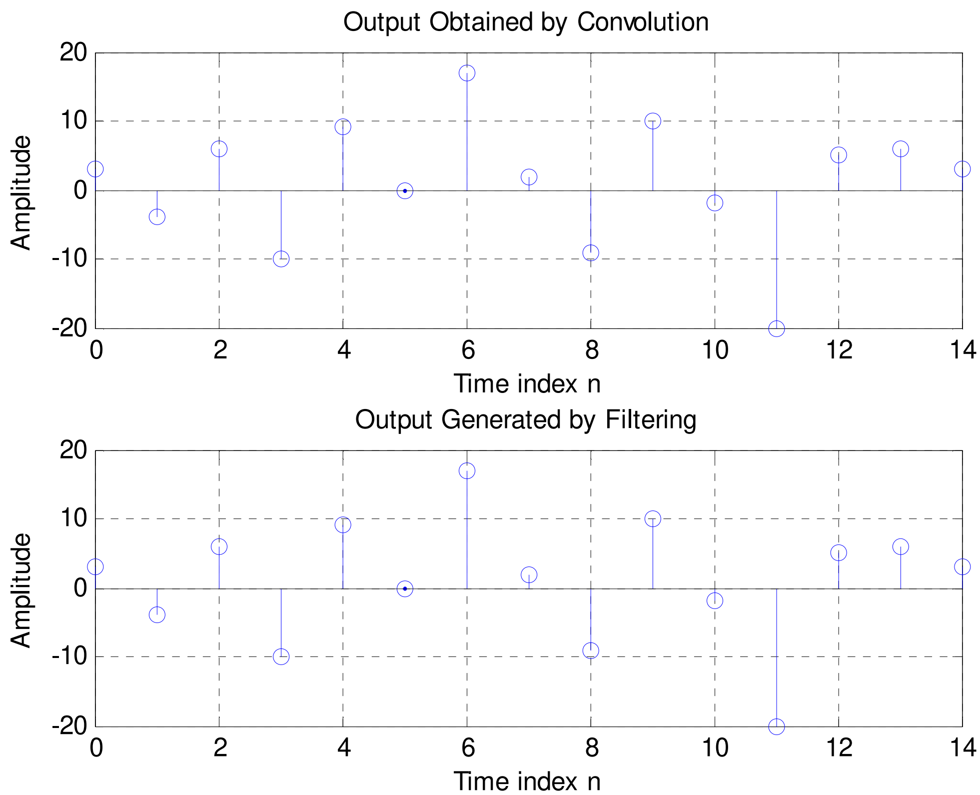
Project 2.7 Convolution

A copy of Program P2_7 is reproduced below:

```
% Program P2_7
clf;
h = [3 2 1 -2 1 0 -4 0 3];   % impulse response
x = [1 -2 3 -4 3 2 1];       % input sequence
y = conv(h,x);
n = 0:14;
subplot(2,1,1);
stem(n,y);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Obtained by Convolution'); grid;
x1 = [x zeros(1,8)];
y1 = filter(h,1,x1);
subplot(2,1,2);
stem(n,y1);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Generated by Filtering'); grid;
```

Answers:

Q2.28 The sequences `y[n]` and `y1[n]` generated by running Program P2_7 are shown below:



The difference between `y[n]` and `y1[n]` is - They are the SAME.

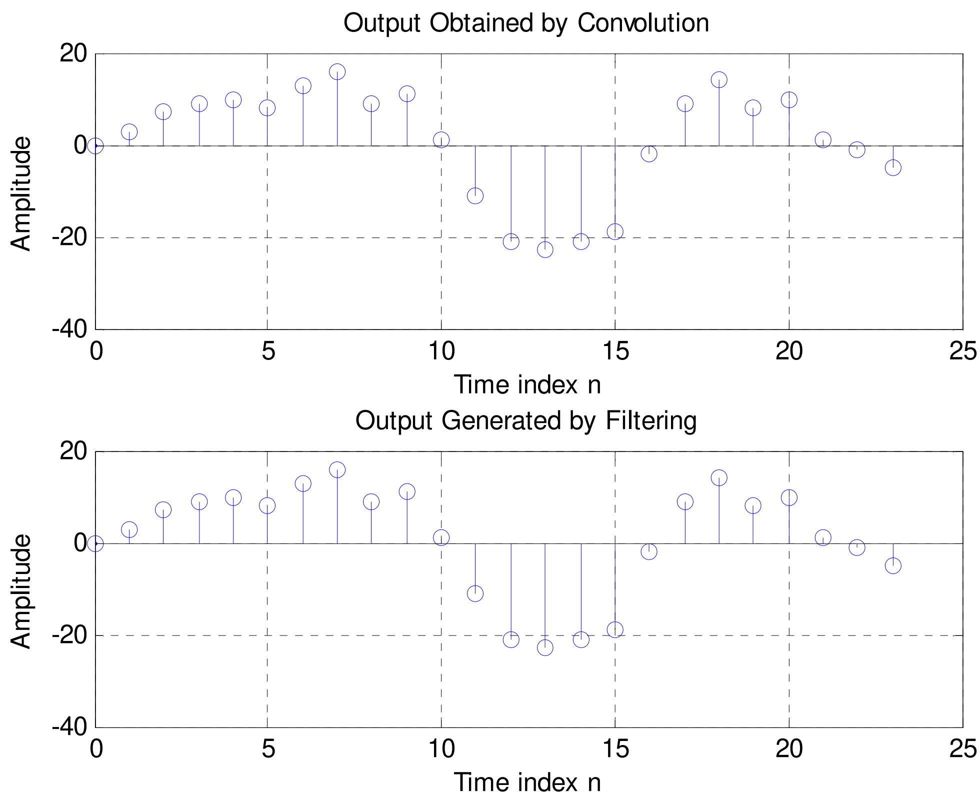
The reason for using `x1[n]` as the input, obtained by zero-padding `x[n]`, for generating `y1[n]` is – For two sequences of length N1 and N2, `conv` returns the resulting sequence of length N1+N2-1. By contrast, `filter` accepts an input signal and a system specification. The returned result is the same length as the input signal. Therefore, to obtain directly comparable results from `conv` and `filt`, it is necessary to supply `filt` with an input that has been zero padded out to length `length(x)+length(h)-1`.

Q2.29 The modified Program P2_7 to develop the convolution of a length-15 sequence `h[n]` with a length-10 sequence `x[n]` is indicated below:

```
% Program Q2_29
clf;
h = [3 1 4 1 5 9 2 6 5 4 -3 -1 -4 -1 -5];   % impulse response
x = [0 1 2 1 0 -1 -2 -1 0 1];          % input sequence
y = conv(h,x);
n = 0:length(h)+length(x)-2;
```

```
subplot(2,1,1);
stem(n,y);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Obtained by Convolution'); grid;
x1 = [x zeros(1,length(h)-1)];
y1 = filter(h,1,x1);
subplot(2,1,2);
stem(n,y1);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Generated by Filtering'); grid;
```

The sequences `y[n]` and `y1[n]` generated by running modified Program P2_7 are shown below:



The difference between `y[n]` and `y1[n]` is - They are the SAME.

Project 2.8 Stability of LTI Systems

A copy of Program P2_8 is given below:

```
Program P2_8
% Stability test based on the sum of the absolute
% values of the impulse response samples
clf;
num = [1 -0.8]; den = [1 1.5  0.9];
N = 200;
h = impz(num,den,N+1);
parsum = 0;
for k = 1:N+1;
    parsum = parsum + abs(h(k));
    if abs(h(k)) < 10^(-6), break, end
end
% Plot the impulse response
n = 0:N;
stem(n,h)
xlabel('Time index n'); ylabel('Amplitude');
% Print the value of abs(h(k))
disp('Value =');disp(abs(h(k)));
```

Answers:

Q2.30 The purpose of the `for` command is – to cause a block of Matlab statements to be repeated for a specified number of times; this implements a "for" or "do" loop.

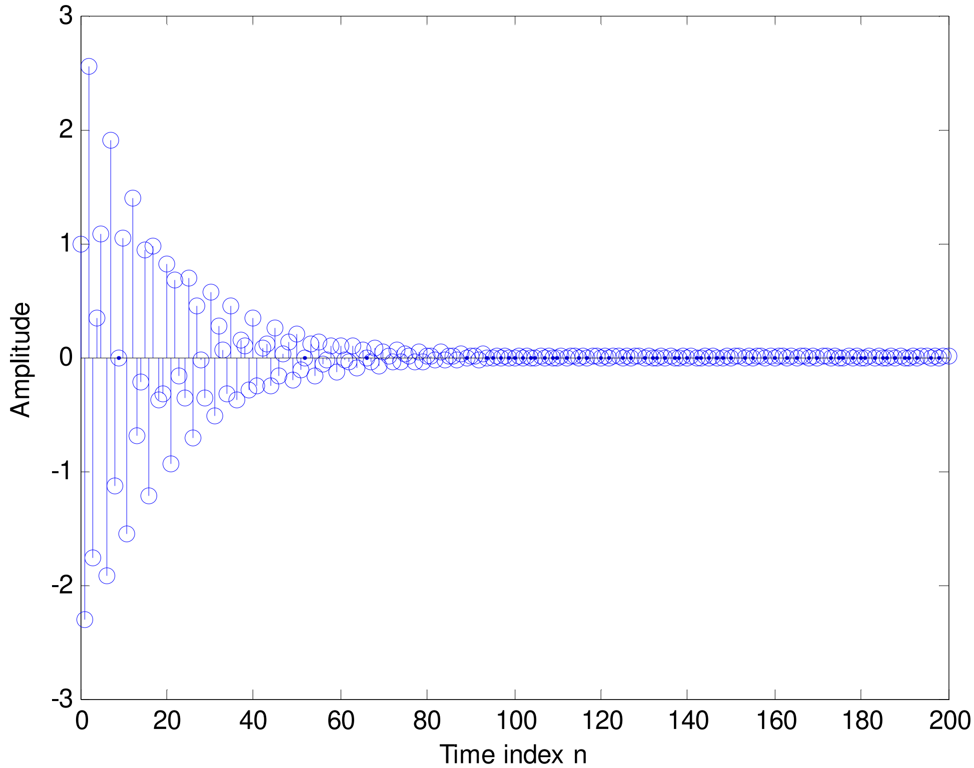
The purpose of the `end` command is – to mark the end of the block of statements that is to be repeated.

Q2.31 The purpose of the `break` command is – to terminate the execution of a "for" or "while" loop.

Q2.32 The discrete-time system of Program P2_8 is -

$$y[n] + 1.5y[n-1] + 0.9y[n-2] = x[n] - 0.8x[n-1]$$

The impulse response generated by running Program P2_8 is shown below:



The value of |`h(K)`| here is - 1.6761e-005

From this value and the shape of the impulse response we can conclude that the system is – very LIKELY to be stable.

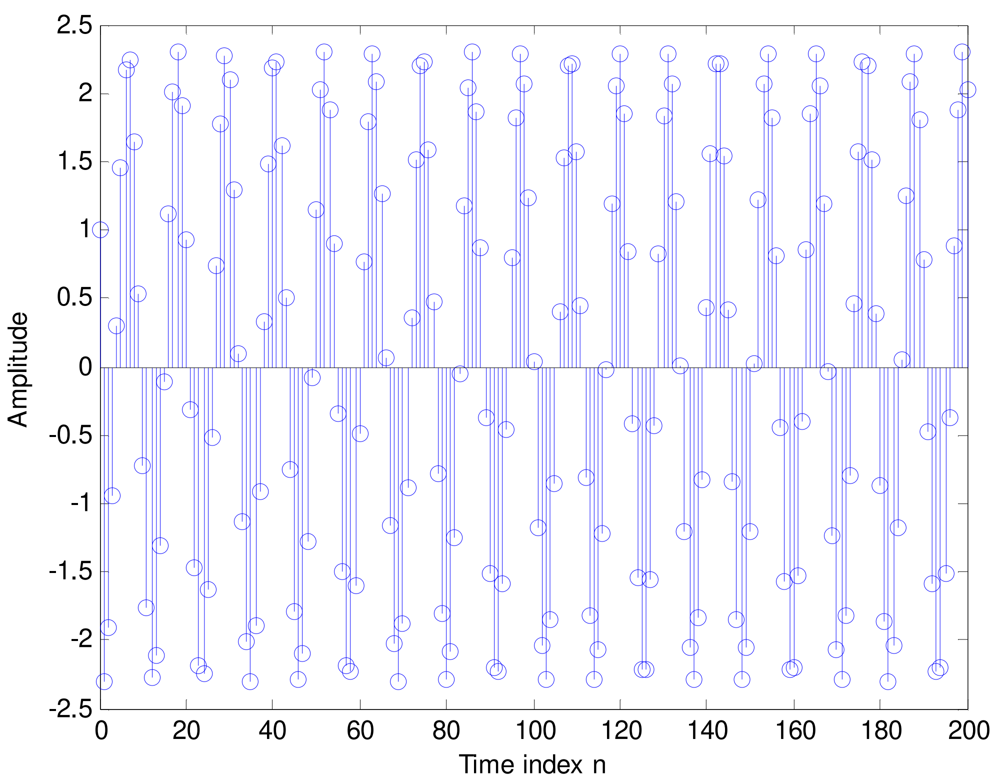
By running Program P2_8 with a larger value of N the new value of |`h(K)`| is - 9.1752e-007

From this value we can conclude that the system is - very LIKELY to be stable.

Q2.33 The modified Program P2_8 to simulate the discrete-time system of Q2.33 is given below:

```
% Program Q2_33
% Stability test based on the sum of the absolute
% values of the impulse response samples
clf;
num = [1 -4 3]; den = [1 -1.7  1.0];
N = 200;
h = impz(num,den,N+1);
parsum = 0;
for k = 1:N+1;
    parsum = parsum + abs(h(k));
    if abs(h(k)) < 10^(-6), break, end
end
% Plot the impulse response
n = 0:N;
stem(n,h)
xlabel('Time index n'); ylabel('Amplitude');
% Print the value of abs(h(k))
disp('Value =');disp(abs(h(k)));
```

The impulse response generated by running the modified Program P2_8 is shown below:



The values of $|$h(K)$|$ here are - 2.0321 for k=200; the values are not decreasing.

From this value and the shape of the impulse response we can conclude that the system is – almost certainly unstable.

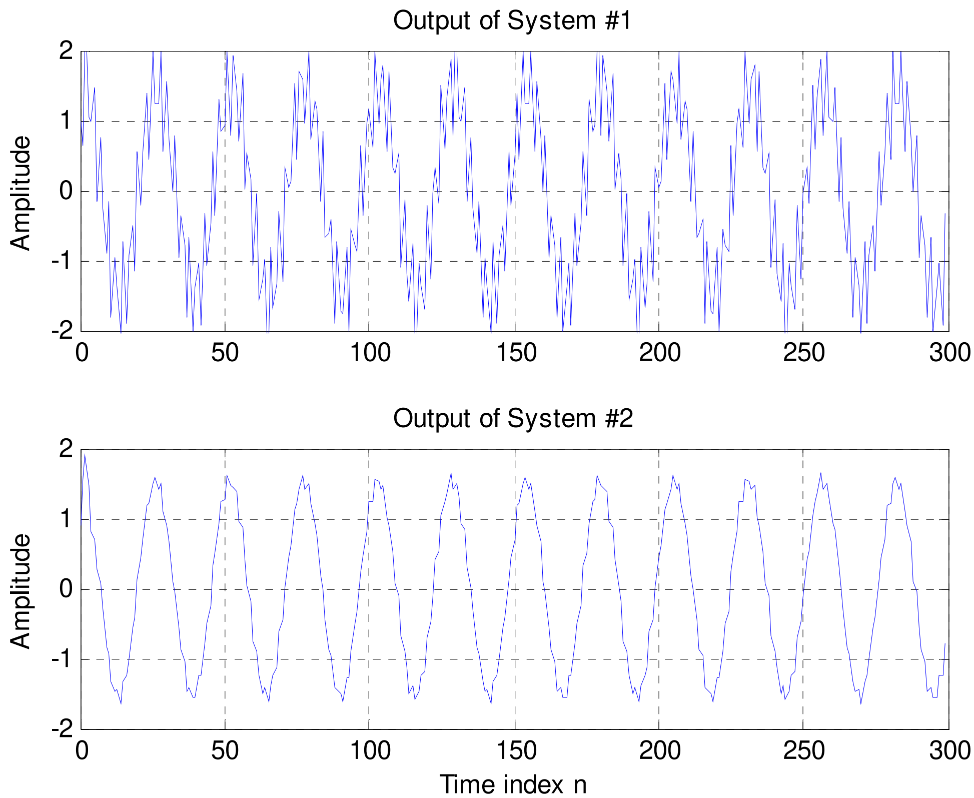
Project 2.9 Illustration of the Filtering Concept

A copy of Program P2_9 is given below:

```
% Program P2_9
% Generate the input sequence
clf;
n = 0:299;
x1 = cos(2*pi*10*n/256);
x2 = cos(2*pi*100*n/256);
x = x1+x2;
% Compute the output sequences
num1 = [0.5 0.27 0.77];
y1 = filter(num1,1,x); % Output of System #1
den2 = [1 -0.53 0.46];
num2 = [0.45 0.5 0.45];
y2 = filter(num2,den2,x); % Output of System #2
% Plot the output sequences
subplot(2,1,1);
plot(n,y1);axis([0 300 -2 2]);
ylabel('Amplitude');
title('Output of System #1'); grid;
subplot(2,1,2);
plot(n,y2);axis([0 300 -2 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Output of System #2'); grid;
```

Answers:

Q2.34 The output sequences generated by this program are shown below:

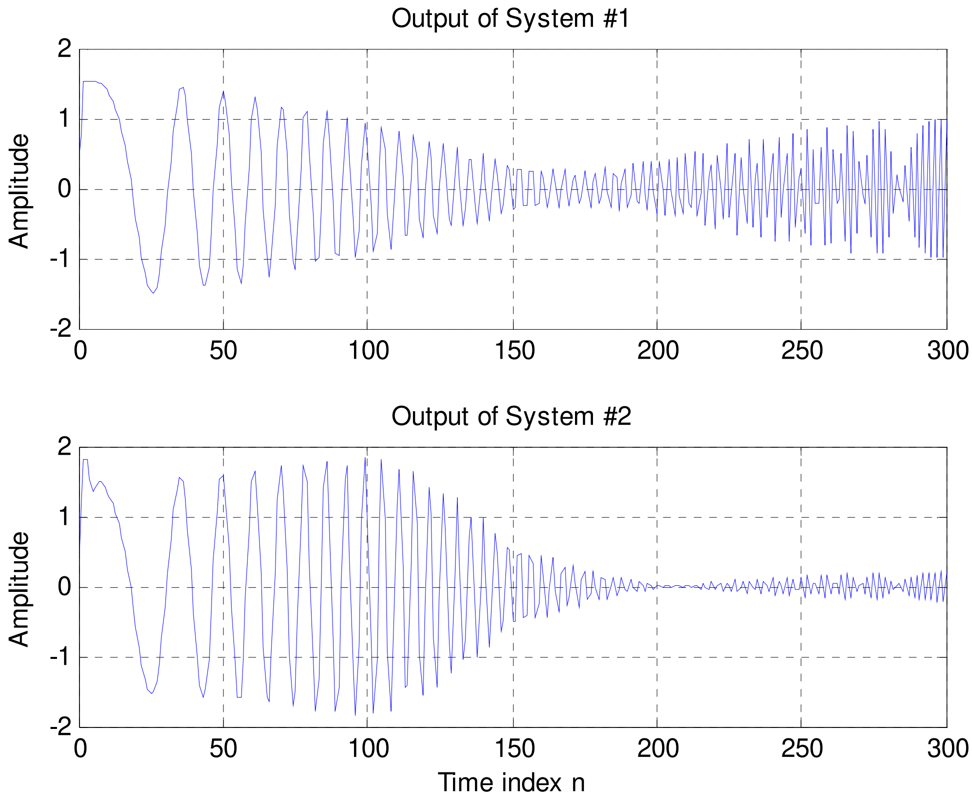


The filter with better characteristics for the suppression of the high frequency component of the input signal x[n] is – System #2.

Q2.35 The required modifications to Program P2_9 by changing the input sequence to a swept sinusoidal sequence (length 301, minimum frequency 0, and maximum frequency 0.5) are listed below along with the output sequences generated by the modified program:

```
% Program Q2_35
% Generate the input sequence
clf;
n = 0:300;
a = pi/600;
b = 0;
arg = a*n.*n + b*n;
x = cos(arg);
% Compute the output sequences
num1 = [0.5 0.27 0.77];
y1 = filter(num1,1,x); % Output of System #1
den2 = [1 -0.53 0.46];
num2 = [0.45 0.5 0.45];
y2 = filter(num2,den2,x); % Output of System #2
% Plot the output sequences
subplot(2,1,1);
plot(n,y1);axis([0 300 -2 2]);
ylabel('Amplitude');
title('Output of System #1'); grid;
subplot(2,1,2);
plot(n,y2);axis([0 300 -2 2]);
```

```
xlabel('Time index n'); ylabel('Amplitude');
title('Output of System #2'); grid;
```



The filter with better characteristics for the suppression of the high frequency component of the input signal x[n] is – System #2.

Date: 17 September, 2006 **Signature: HAVLICEK**