# LAB REPORT – 2
# Signal Processing

By – Dikshant (2022102038)

**AIM**: Build simple discrete time (DT) systems

## 2.1 Moving Average (MA)

MA system is used to detect trends from signal:

$$y[n] = \frac{1}{N} \sum_{k=n-N}^{n} x[k]$$

a) Function for movingAverage:

```
function y = movingAverage(N, xn, n)
    sum = zeros(1);
    y = zeros(1, length(n));
    for k = 1:length(n)
        if k<=N
            sum = sum+xn(k);
            y(k) = sum/N;
        end
        if k>N
            sum = sum + xn(k) - xn(k-N);
            y(k) = sum/N;
        end
    end
end
```

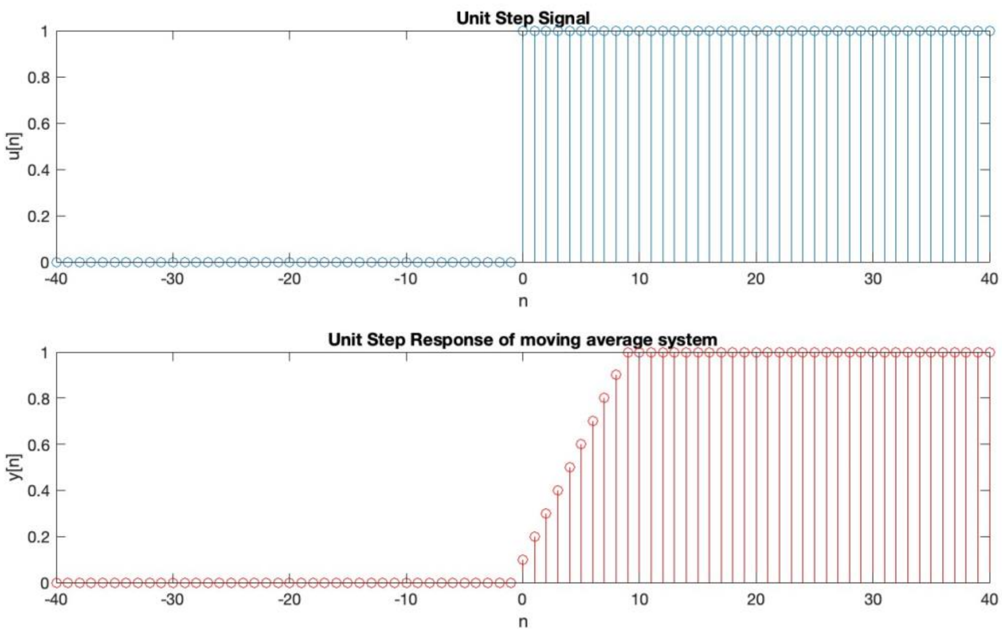
b) System with u[n] as input

Code:

```
clc; clearvars; clear all;
n = -40:1:40;
xn = (n>=0);
N = 10;
MA = movingAverage(N, xn, n);
subplot(2,1,1); stem(n, xn);
xlabel('n'); ylabel('u[n]'); title('Unit Step Signal');

subplot(2,1,2); stem(n,MA, 'r');
xlabel('n'); ylabel('y[n]'); title('Unit Step Response of moving average system');
```

Plot:

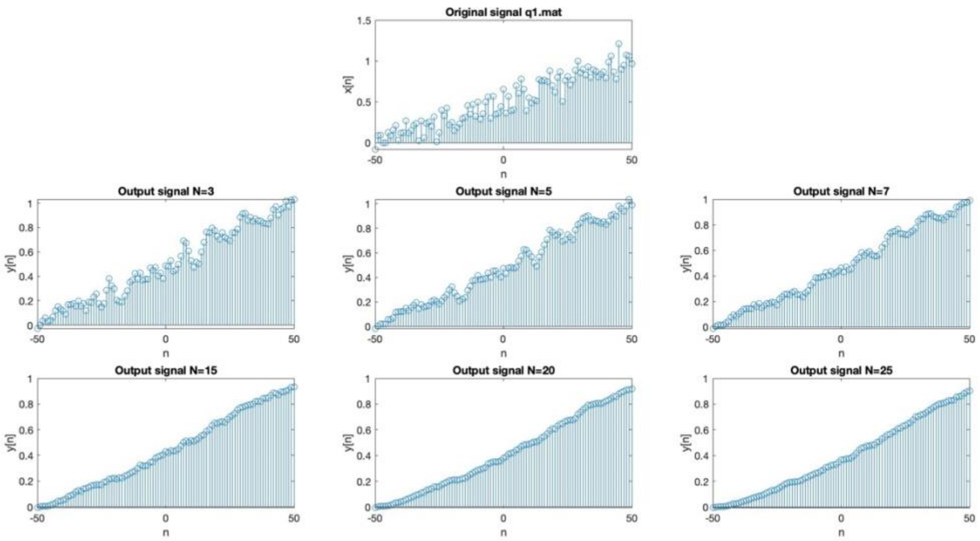



c) Finding Trend of s[n], given in q1.mat

Code:

```
clc; clearvars; clear all;
load('q1.mat');
n = -50:50;
ind = [3 5 7 15 20 25];
subplot(3,3,2); stem(n,x);
xlabel('n'); ylabel('x[n]'); title('Original signal q1.mat')
for k = 1:length(ind)
    N = ind(k);
    MA = movingAverage(N,x,n);
    subplot(3,3,k+3);
    stem(n,MA);
    xlabel('n'); ylabel('y[n]'); title(['Output signal N=', num2str(N)]);
end
```

Plot:




Around N=15 is appropriate to observe an almost monotonic increase. N=25 is also fine. But further increase in N will improve the monotonicity but also severely decrease the amplitude. So, N=15-20 is appropriate.

d) The trend in s[n] is that it increases in amplitude though not monotonically. We can adjust N such that the output can be made almost increasing monotonically.

## 2.1.1

Advantage of implementing it using convolution is that it is easier and more straightforward. The main disadvantage is that it might be less efficient for large signals or large values of N.

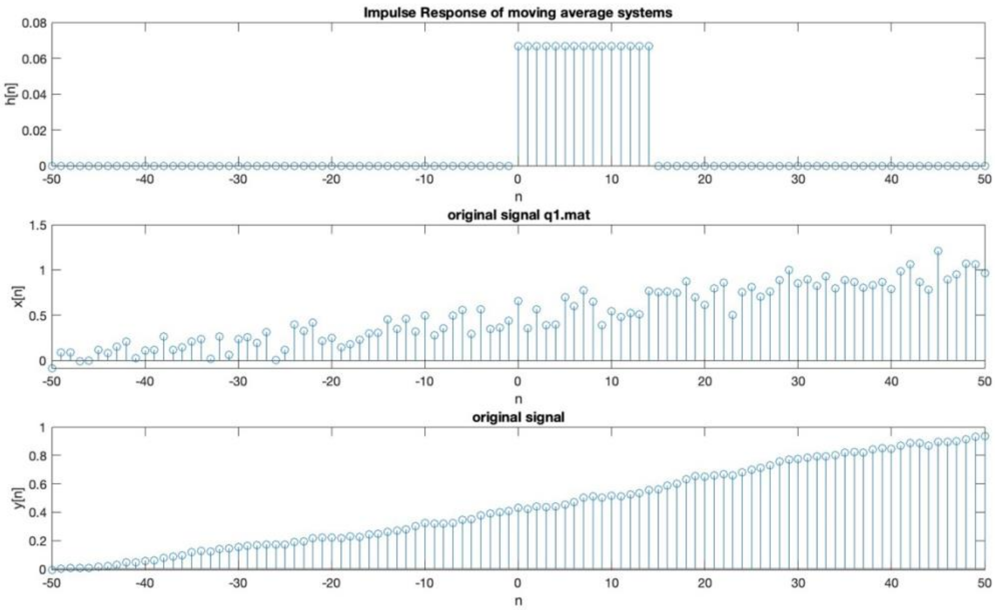
Code:

```
clc; clearvars; clear all;
n = -50:50;
xn  = (n==0);
N = 15;
index = 51:1:151;
imp_rep = movingAverage(15,xn,n);
y = conv(imp_rep, x);
subplot(3,1,1); stem(n, imp_rep);
xlabel('n'); ylabel('h[n]'); title('Impulse Response of moving average systems');

subplot(3,1,2); stem(n,x); xlabel('n'); ylabel('x[n]'); title('original signal q1.mat');

y = y(index);
subplot(3,1,3); stem(n,y);
xlabel('n'); ylabel('y[n]'); title('original signal');
```

Plot:




## 2.2 Upsampler System

Func to implement Zero order hold – STEP1:

```
function y = upSamplingS1(xn,n,M)
    m = n(1)* M:n(length(n))*M;
    y = zeros(1,length(m));
    count = 1;
    for k = 1:length(m)
        if rem(m(k),M)==0
            y(k) = xn(count);
            count = count + 1;
        else
            y(k) = 0;
        end
    end
end
```

Func to implement Linear interpolation – STEP2

```
function y = upSamplingS2(xn,n,M)
    m = n(1)*M:n(length(n))*M;
    y = zeros(1,length(m));
    temp = xn(1);
    count = 1;
    o = zeros(1); c = zeros(1);
    for k = 1:length(m)
        if rem(m(k),M)==0
            y(k) = temp;
            count = count+1;
            if(count~=(length(n)+1))
                temp =xn(count);
            end
        else
            if (m(k-1)~=(n(count)*M))
                o  = (y(k-1)-temp)/(m(k-1) - n(count)*M);
            else
                o = 0;
            end
            c = temp-(o*n(count)*M);
            y(k) = o*m(k)+c;
        end
    end
end
```

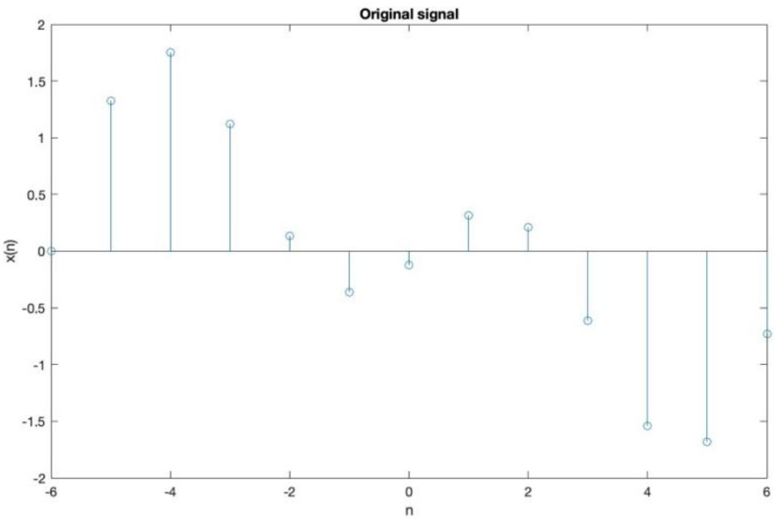
a. Implementing upsampler with M = 2 and 3

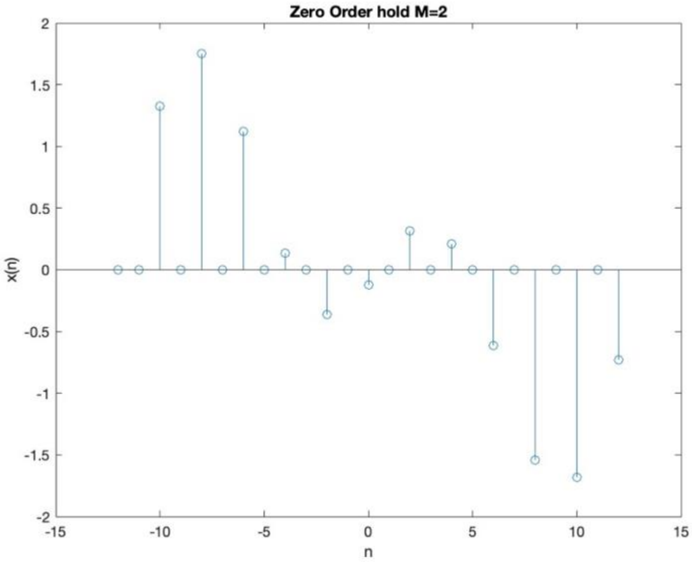
Code:

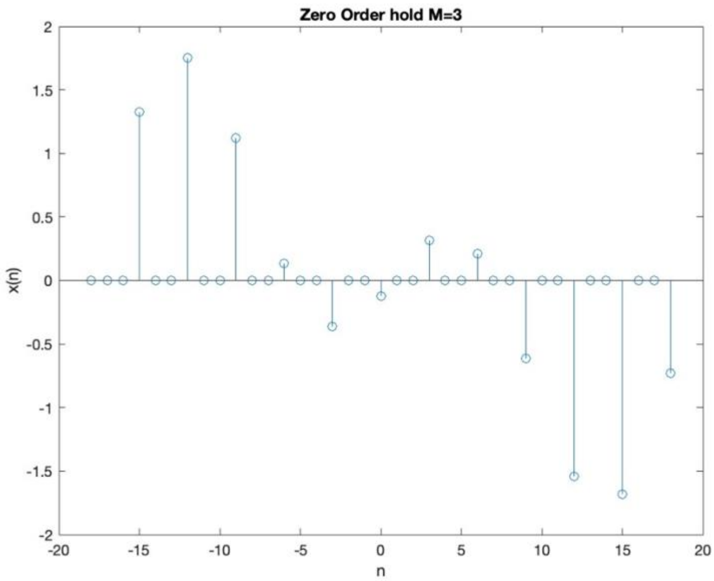
```
clc; clearvars;clear all;
load('q2_2.mat');
ind = [2 3];
if(rem(length(x),2)==0)
    a=length(x)/2;
    n = -1*a+1:a;
else
    a = (length(x)-1)/2;
    n = -1*a:a;
end
figure(1);
stem(n,x);
xlabel('n'); ylabel('x(n)'); title('Original signal');
for k =1:2
    M = ind(k);
    y = upSamplingS1(x,n,M);
    m = n(1)*M:n(length(n))*M;
    figure(k+1); stem(m,y);
    xlabel('n'); ylabel('x(n)'); title(['Zero Order hold M=', num2str(M)]);
end
for k=1:2
    M = ind(k);
    y = upSamplingS2(x,n,M);
    m = n(1)*M:n(length(n))*M;
    figure(k+3); stem(m,y);
    xlabel('n'); ylabel('x^(n)'); title(['Linear interpolation M=', num2str(M)]);
end
```

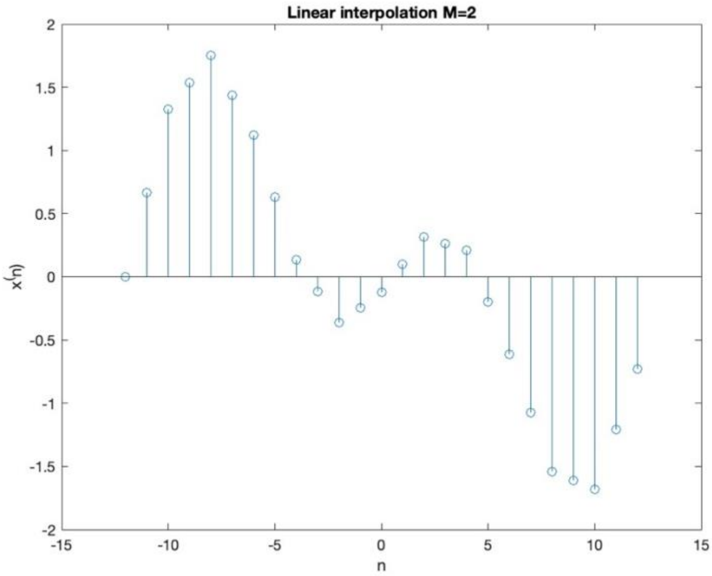
Plot:

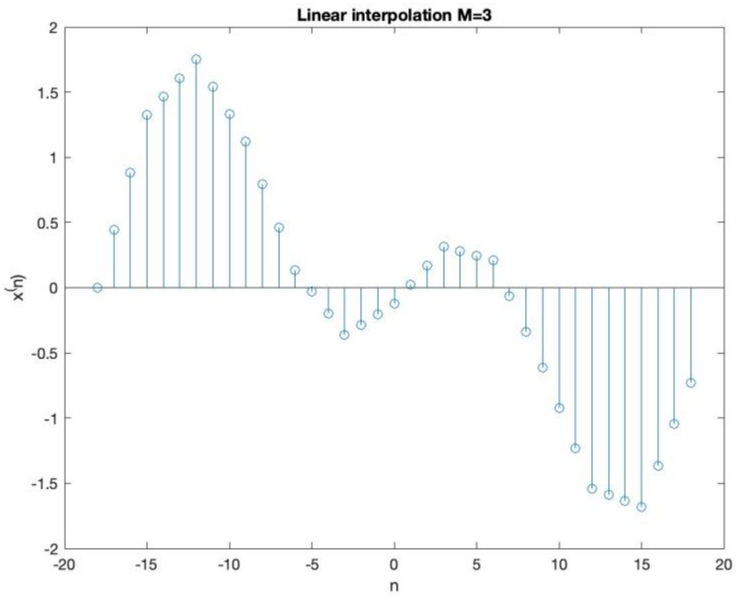
i) When input signal is used from q2_1.mat



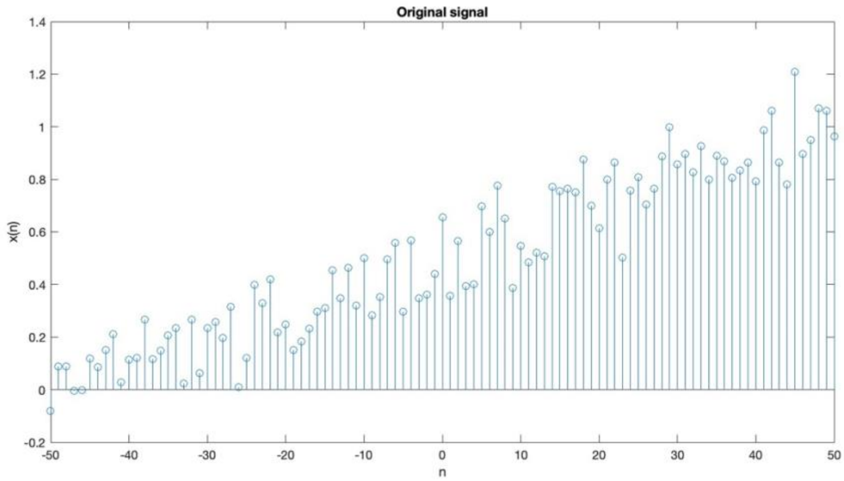

Zero Order hold M=2
x(n)
n
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-15
-10
-5
0
5
10
15


Zero Order hold M=3
x(n)
n
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-20
-15
-10
-5
0
5
10
15
20

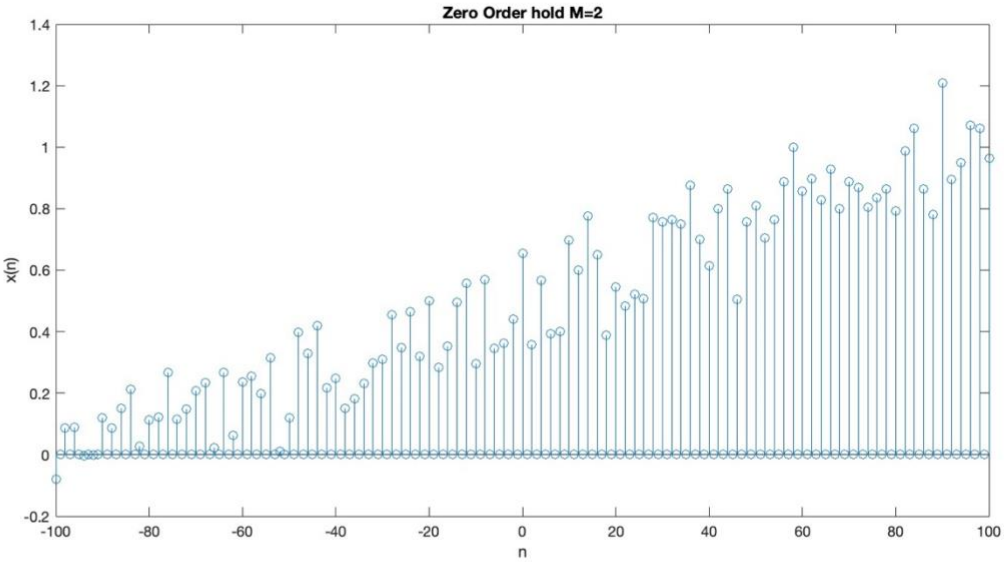


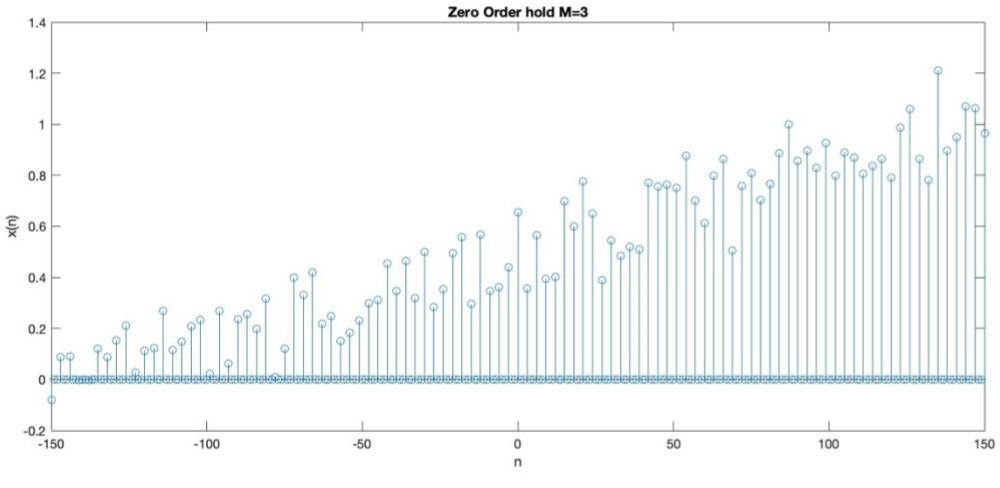

ii) When input signal is used from q2_2.mat



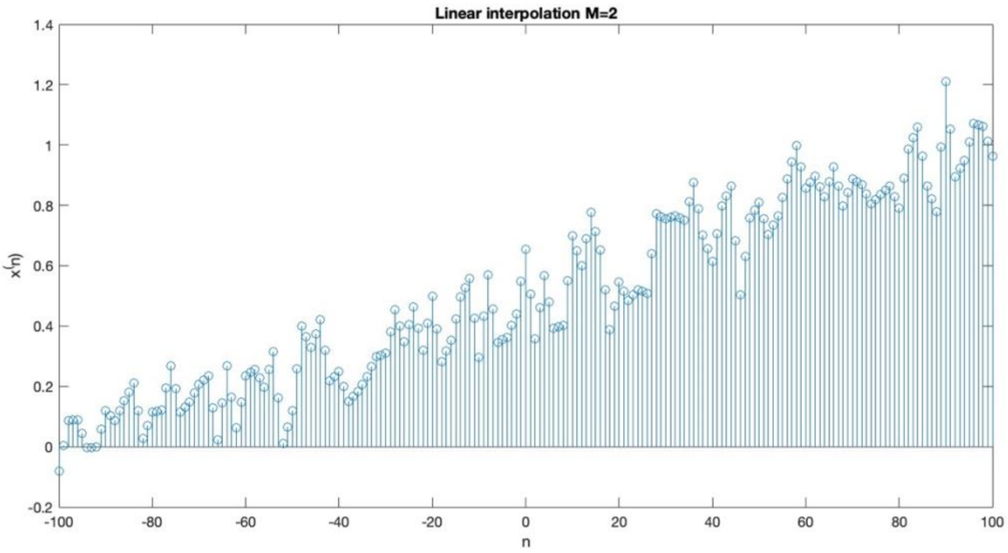
Original signal
x(n)
n
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-50
-40
-30
-20
-10
0
10
20
30
40
50

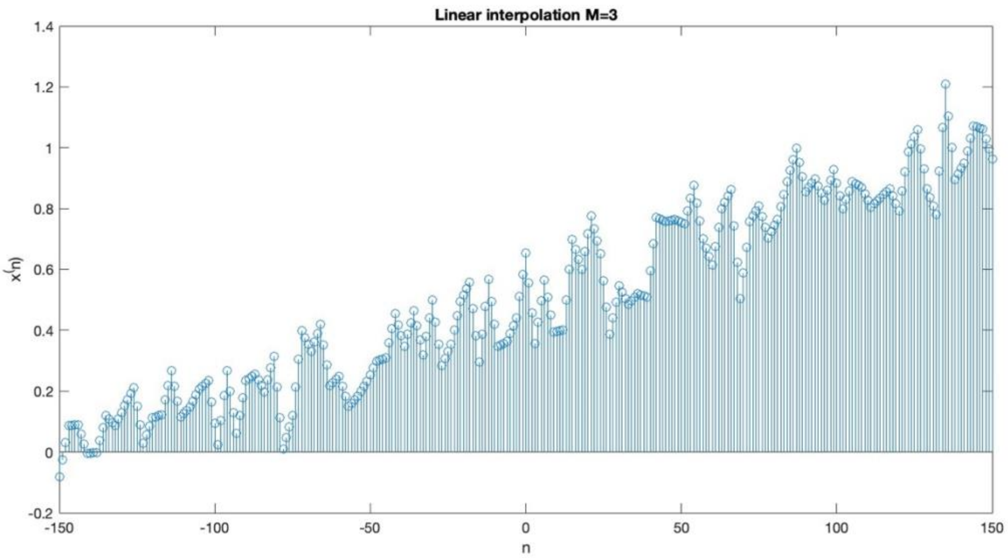



Zero Order hold M=2
x(n)
n
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-100
-80
-60
-40
-20
0
20
40
60
80
100

Zero Order hold M=3
x(n)
n
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-150
-100
-50
0
50
100
150




If we upsample it using zero-order hold with M = 2 and M = 3, we will observe that the upsampled signal has a staircase-like shape. If we upsample the same signal using linear interpolation with M = 2 and M = 3, we will observe that the upsampled signal is smoother compared to zero-order hold. The new samples inserted between each pair of adjacent samples in the original signal are obtained by linearly interpolating between them.

We observe that when we do Upsampling,we are able to spread our data on a much larger scale and it can also be able to increase the information in our hand by assuming first order or second order or so on upsampling depending on the application.

## 2.3 Digital Differentiator

First order digital differentiator is given by:

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

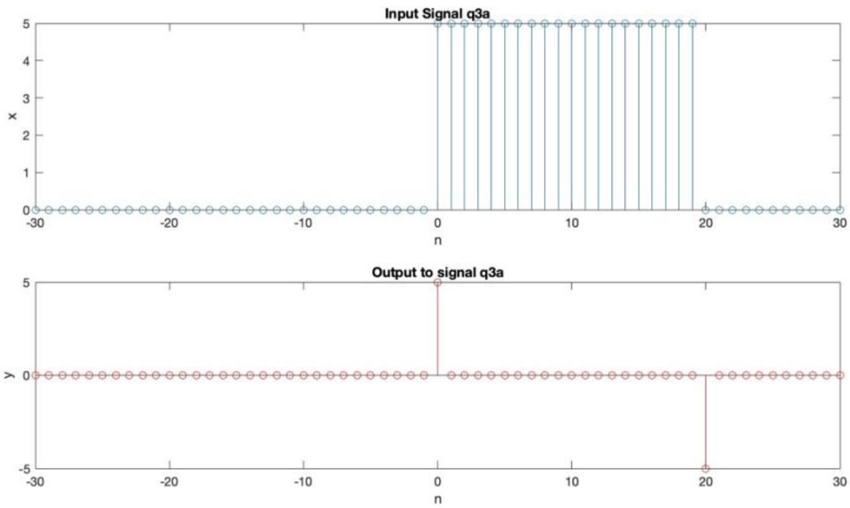
digital_differentiator function:

```
function y = digitalDifferentiator(n,xn)
    y = zeros(1, length(n));
    y(1) = xn(1);
    for k = 2:length(n)
        y(k) = xn(k) - xn(k-1);
    end
end
```

a) Code:

```
clc; clearvars; clear all;
n = -30:30;
x = (n>=0) & (n<20);
x = 5*x;
y = digitalDifferentiator(n,x);
subplot(2,1,1); stem(n,x);
xlabel('n'); ylabel('x'); title('Input Signal q3a');
subplot(2,1,2); stem(n,y, 'r');
xlabel('n'); ylabel('y');
title('Output to signal q3a');
```

Plot:

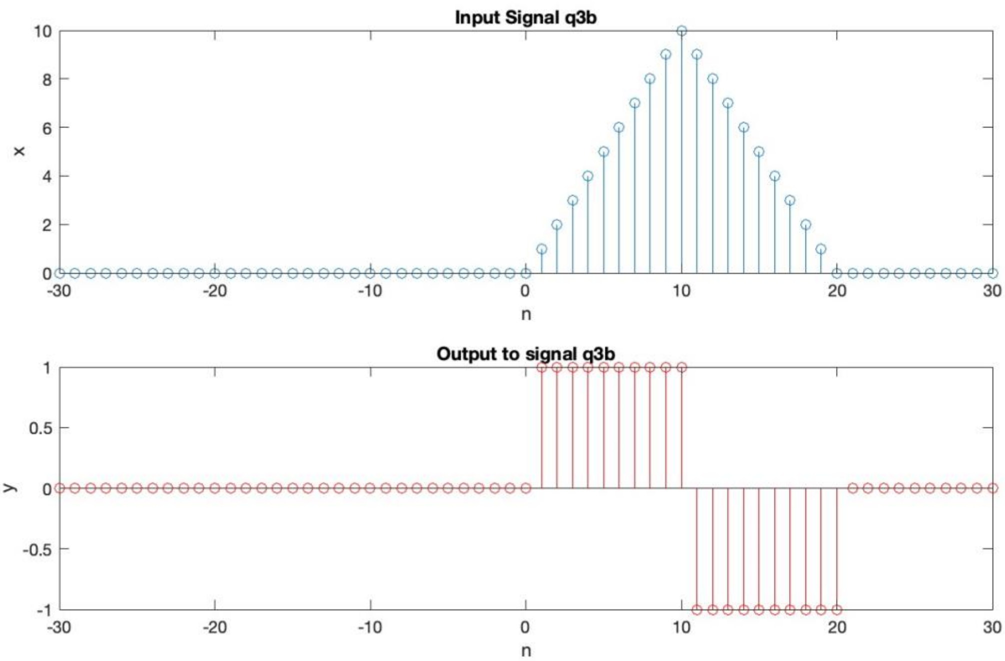



b) Code:

```
clc; clearvars; clear all;

n = -30:30;
u = (n>=0) & (n<10);
u1 = (n>=10) & (n<20);
x = n.*u+(20-n).*u1;
y = digitalDifferentiator(n,x);
subplot(2,1,1); stem(n,x);
xlabel('n'); ylabel('x'); title('Input Signal q3b');
subplot(2,1,2); stem(n,y,'r');
xlabel('n'); ylabel('y');
title('Output to signal q3b');
```

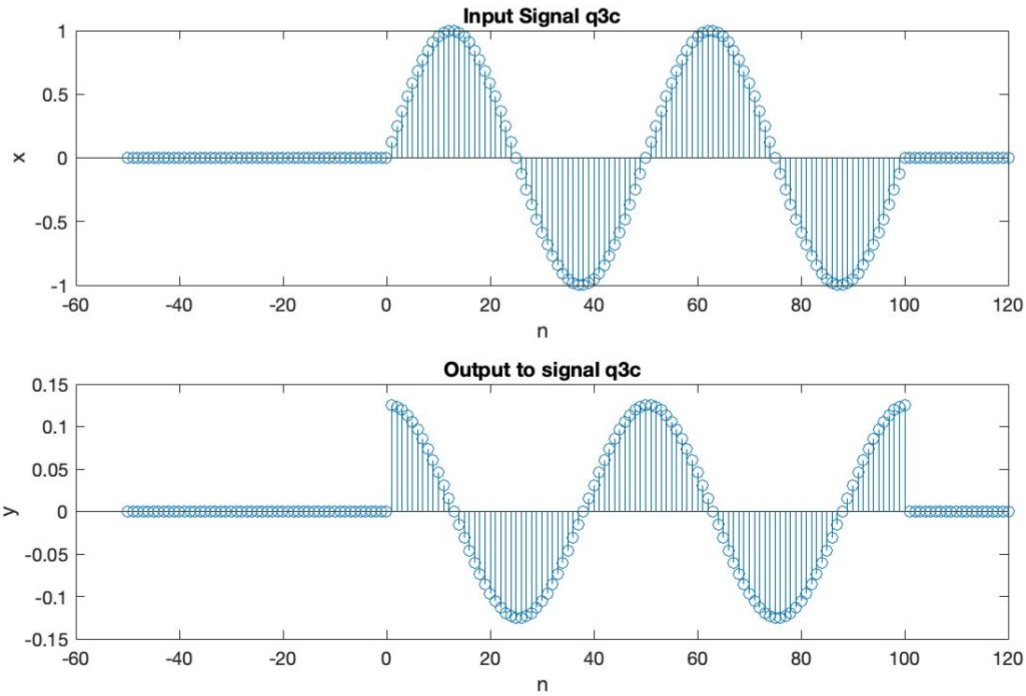
Plot:




c) Code:

```
clc; clearvars; clear all;
n = -50:120;
u = (n>=0) & (n<100);
x = sin(n*pi/25).*u;
y = digitalDifferentiator(n,x);
subplot(2,1,1); stem(n,x); xlabel('n'); ylabel('x');
title('Input Signal q3c');
subplot(2,1,2); stem(n,y);
xlabel('n'); ylabel('y'); title('Output to signal q3c');
```

Plot:




## 2.4 Plotting $\alpha$ $and$ $\beta$ :

```
n = 0:50;
a = -1.41421;
b = 1;
A = [1 a b];
B = [1];
y = impz(B, A, n);
stem(n, y); xlabel('n'); ylabel('h');
title(['q4b Impulse response for \alpha =', num2str(a),'\beta=', num2str(b)]);
```

4. $y[n] + \alpha y[n-1] + \beta y[n-2] = x[n]$.

For finding impulse response $h[n]$, we'll ~~put~~ put $x[n] = \delta[n]$.

for $n<0$, $y[n] = 0$ } Assuming Causal System.

for $n=0$, $y[0] = 1$

$n=1$, $y[1] = -\alpha$

$n=2$, $y[2] = \alpha^2 - \beta$

$n=3$, $y[3] = -\alpha(\alpha^2-\beta) - \beta(-\alpha) = -\alpha^3 + 2\alpha\beta$

$n=4$, $y[4] = -\alpha(\alpha^3 + 2\alpha\beta) - \beta(\alpha^2-\beta)$

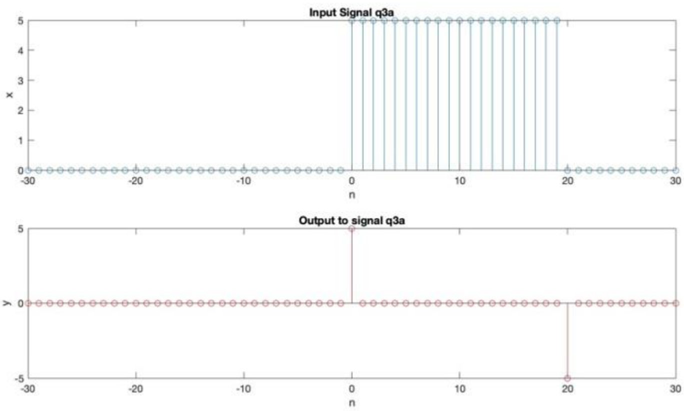
$= \alpha^4 - 3\alpha^2\beta + \beta^2$

$y[n] + \alpha y[n-1] + \beta y[n-2] = x[n]$.

In order to produce this system,

we will give $b = [1]$ $a = [1 \ \alpha \ \beta]$.

a)

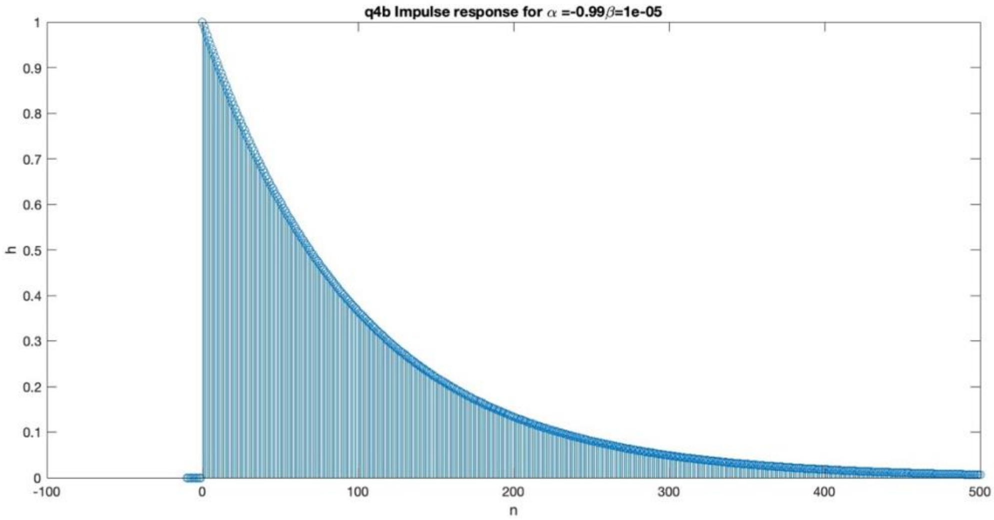


b) $\alpha = -.99,\ \beta = 0.00001$

Explanation

(b) for $f^n$ $h[n]$ to monotonically dec over time, $\alpha$ should be -ve . $\beta$ should be +ve sufficiently small such that $h[n]$ doesn't go b/w +ve & -ve values.

Ex $\alpha = -.99$ , $\beta = 0.00001$




Explanation

(b) for $f^n$ $h[n]$ to monotonically dec over time, $\alpha$ should be -ve . $\beta$ should be +ve sufficiently small such that $h[n]$ doesn't go b/w +ve & -ve values.

Ex $\alpha = -.99$ , $\beta = 0.00001$

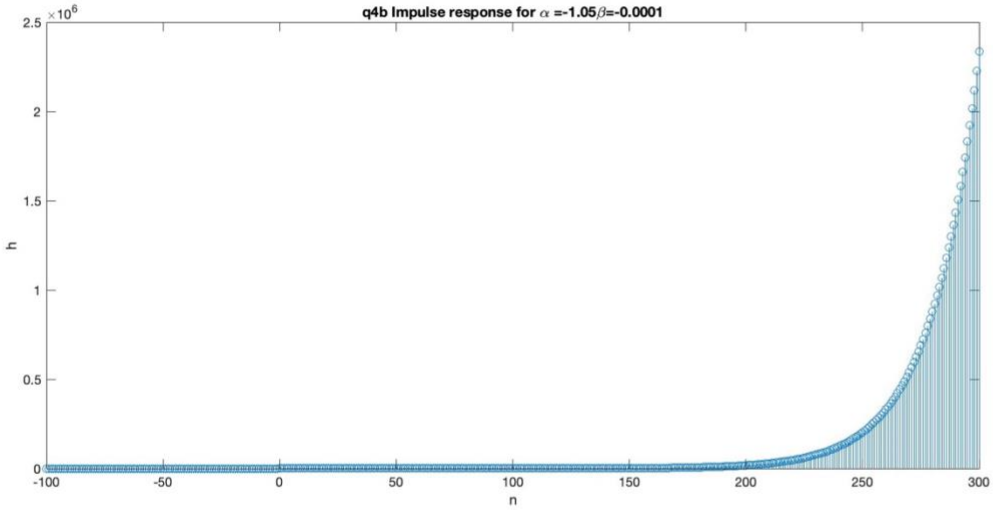
C) $\alpha = -1.05$ *and* $\beta = -0.0001$

(c) for $h[n]$ to diverge monotonically . $\alpha, \beta$ should be -ve $|\alpha| > 1$ (just greater) $|\beta|$ decides the growth

In Order to show steady yet stable growth

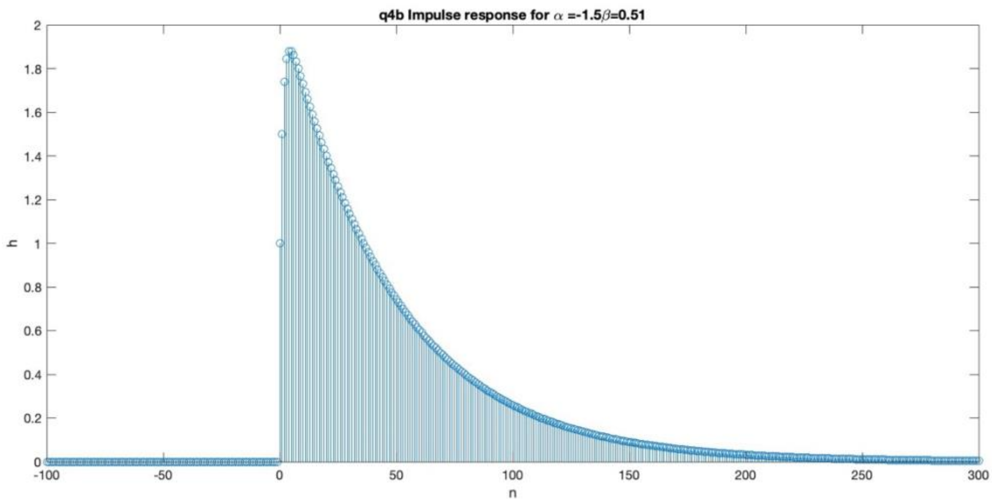
$|\beta|$ should be small.

Ex: $\alpha = -1.05$ , $\beta = -0.0001$




d) $\alpha\ =\ -1.5\ and\ \beta\ =\ 0.51$

d) for $h[n]$ to grow initially & then decay as $n \to \infty$ $\alpha$ should be $-ve$ & $|\alpha| > \beta$ should be $+ve$ & high enough so that, after an initial growth, it eliminates & $h[n]$ decays as $n \to \infty$.
Ex $\alpha = -1.5$ , $\beta = 0.505$.




e) $\alpha = -\sqrt{2}\ and\ \beta\ =\ 1$

e) The characteristic Eq$^n$ for this system is $r^2 + ar + b = 0$. for $h[n]$ to oscillate $\forall n$, both roots must be complex conjugate with magnitude 1.

Suppose $\theta = \pi/4$ Then $r_1 = e^{i\theta} = \frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}} = \frac{1}{\sqrt{2}}(1+i)$

$r_2 = e^{-i\theta} = \frac{1}{\sqrt{2}}(1-i)$

Solving linear Eq$^{ns}$ of $a$ & $b$, we get $a = -\sqrt{2}$, $b = 1$.