भारतीय प्रौद्योगिकी संस्थान तिरुपति

TIRUPATI

# LAB SHEET 03

**NAME : MADHU SAI NAIK P**

**ROLL NO : EE23B039**

## A. OBSERATION OF GIBBS PHENOMENON.

1. Fix a value of Wc = [0,pi] and M =1.
2. Compute Hm(e^jw) = sgm n = -M to M(sin(Wc.n)/pi.n . e^-j.Wc.n) for w = -pi:^w:pi.
3. Plot w vs Hm(e^jw) and compare with H(e^jw).
4. Repeat the experiment for M = 5,12,18 and 27.
5. Report your observations.

Question 1

```
Wc = 0+pi.*rand;
M = 1;
```

Question 2

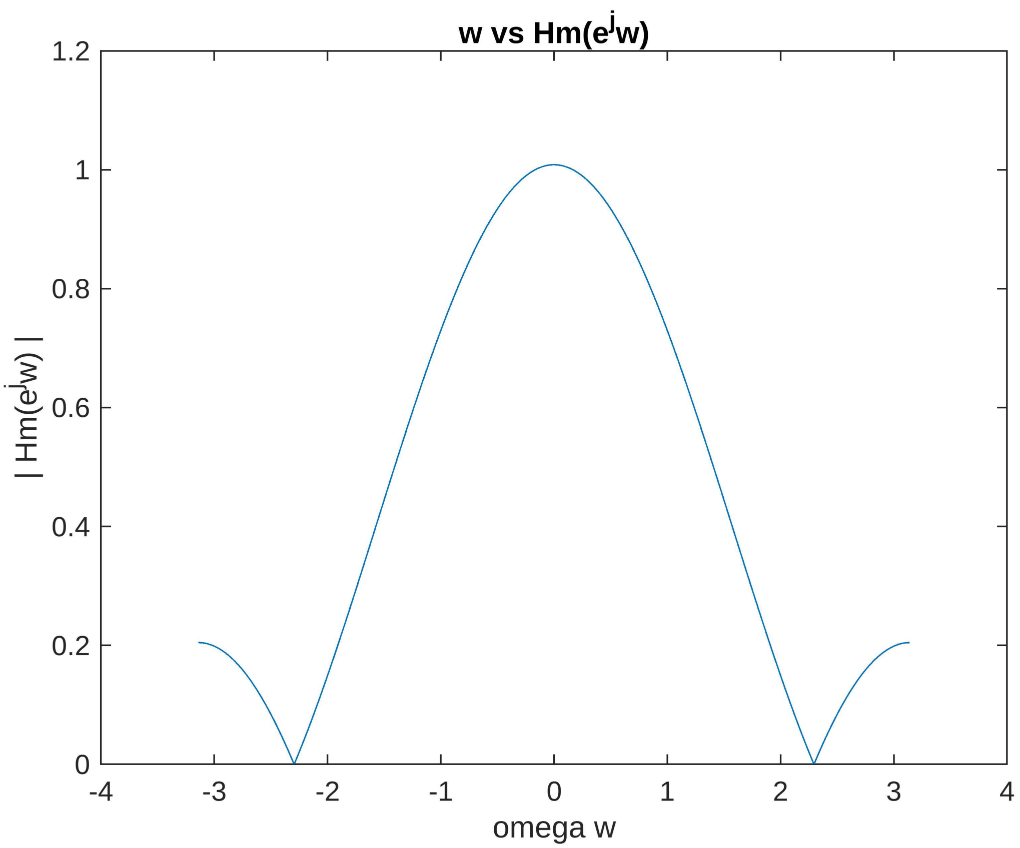
```
n = -M:M;
w = -pi:0.01:pi;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(Wc * n(i)))/ (pi * n(i));
 else
 H_n(i) = Wc/ pi;
 end
end
for k = 1:length(w)
 H_sum = 0;
 for i = 1:length(n)
 H_sum = H_sum + H_n(i) * exp(-1i * w(k) * n(i));
 end
 H_m(k) = H_sum;
end
```

```
H_m
```

```
H_m = 1×629
   -0.2048    -0.2047    -0.2046    -0.2045    -0.2043    -0.2040    -0.2037    -0.2033 ⋯
```

Question 3

```
plot(w,abs(H_m));
xlabel("omega w");
ylabel("| Hm(e^jw) |");
title("w vs Hm(e^jw)");
```



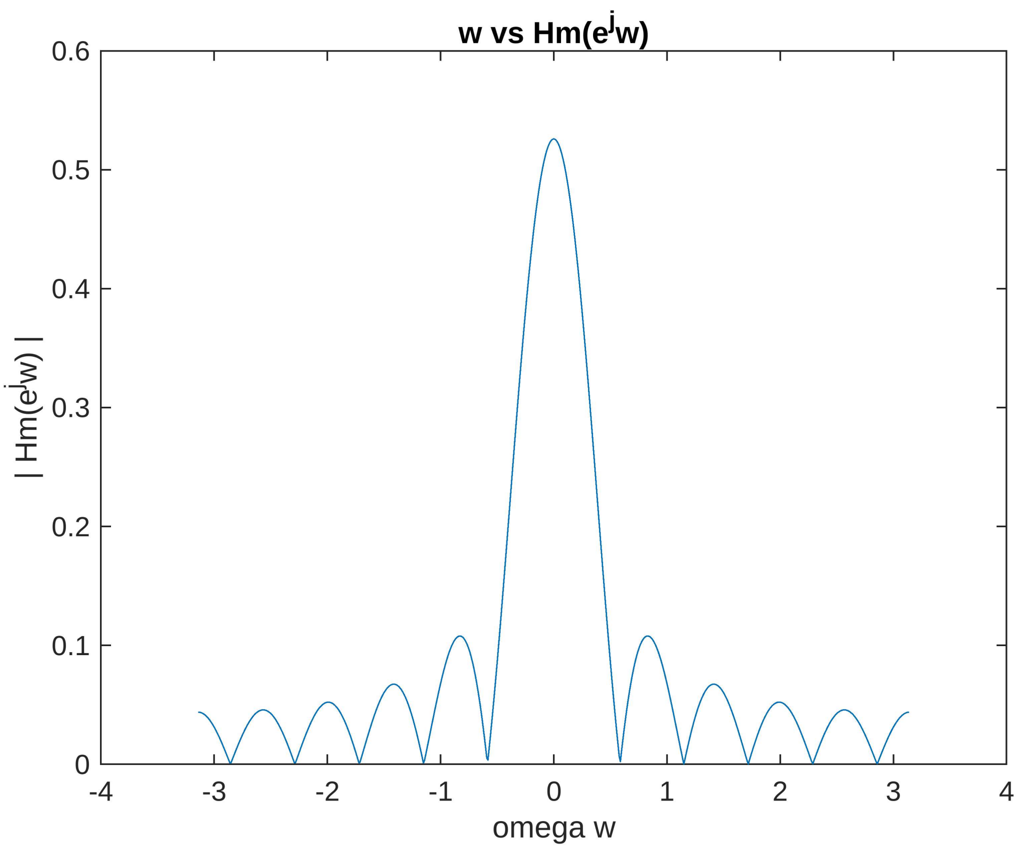

Question 4

```
M = 5;
Wc = 0+pi.*rand;
n = -M:M;
w = -pi:0.01:pi
```

```
w = 1×629
   -3.1416    -3.1316    -3.1216    -3.1116    -3.1016    -3.0916    -3.0816    -3.0716 ⋯
```

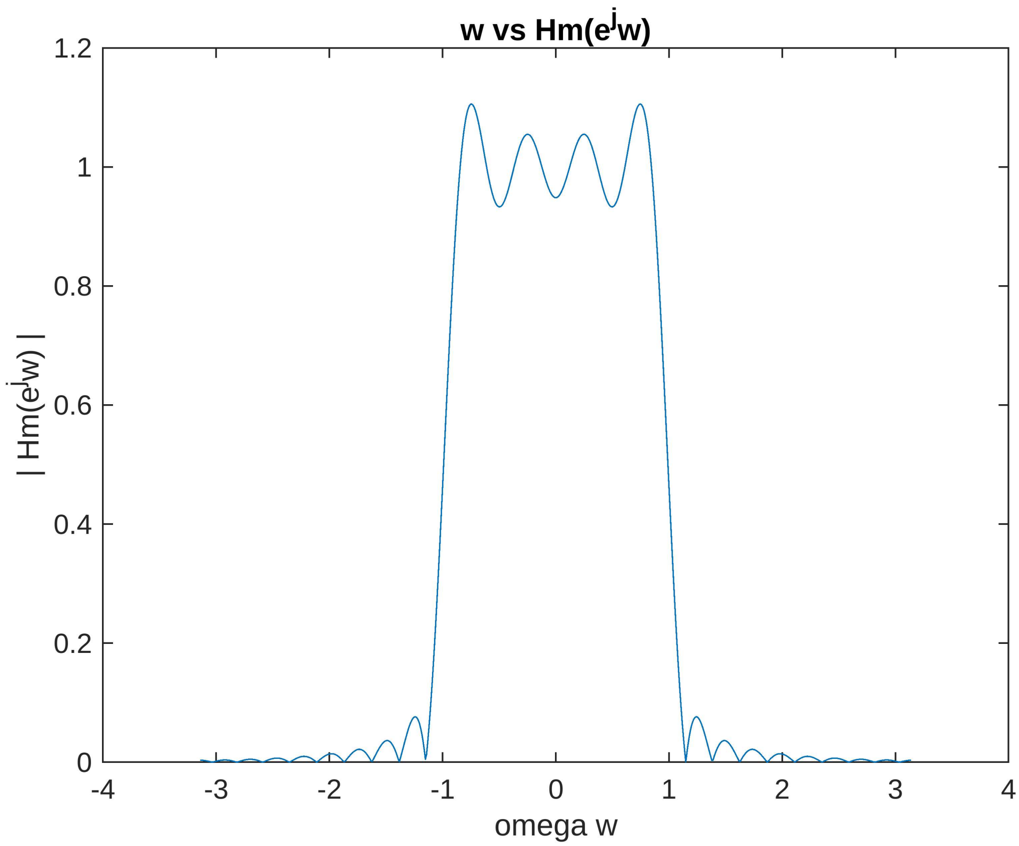
```
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(Wc * n(i)))/ (pi * n(i));
 else
```

```
 H_n(i) = Wc/ pi;
 end
end
for k = 1:length(w)
 H_sum = 0;
 for i = 1:length(n)
 H_sum = H_sum + H_n(i) * exp(-1i * w(k) * n(i));
 end
 H_m(k) = H_sum;
end
plot(w,abs(H_m));
xlabel("omega w");
ylabel("| Hm(e^jw) |");
title("w vs Hm(e^jw)");
```



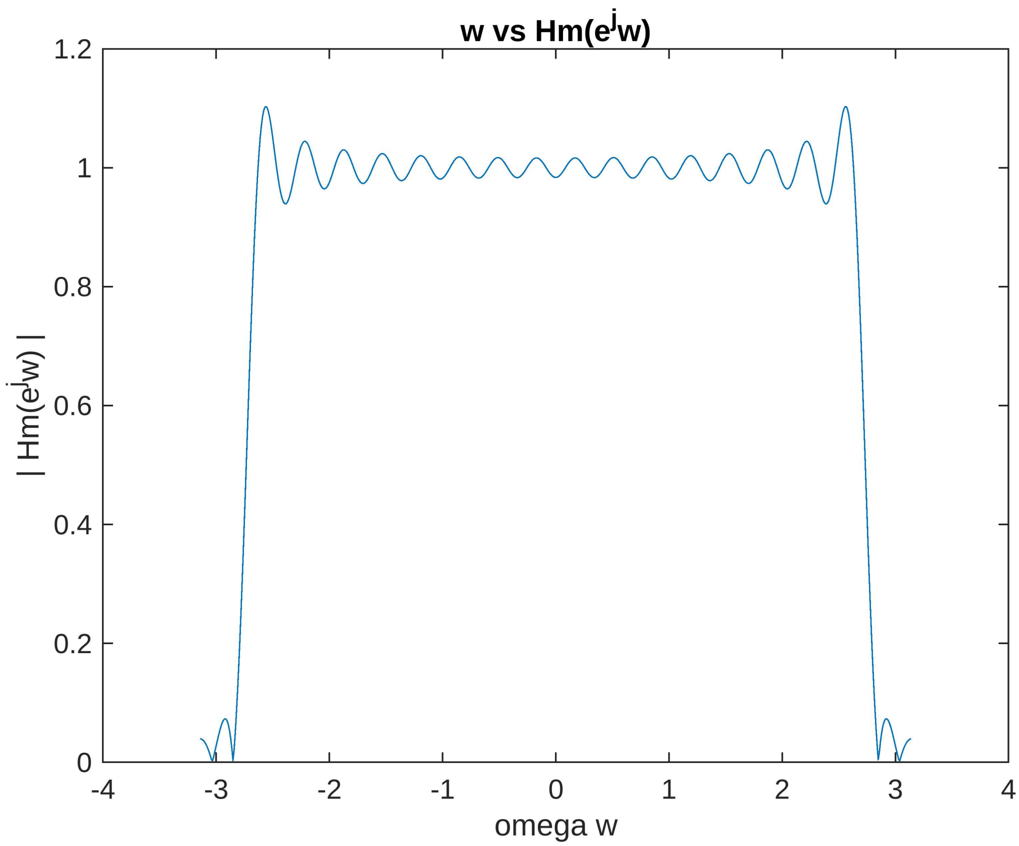

```
M = 12;
Wc = 0+pi.*rand;
n = -M:M;
w = -pi:0.01:pi;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(Wc * n(i)))/ (pi * n(i));
 else
 H_n(i) = Wc/ pi;
 end
```

```
end
for k = 1:length(w)
 H_sum = 0;
 for i = 1:length(n)
 H_sum = H_sum + H_n(i) * exp(-1i * w(k) * n(i));
 end
 H_m(k) = H_sum;
end
plot(w,abs(H_m));
xlabel("omega w");
ylabel("| Hm(e^jw) |");
title("w vs Hm(e^jw)");
```



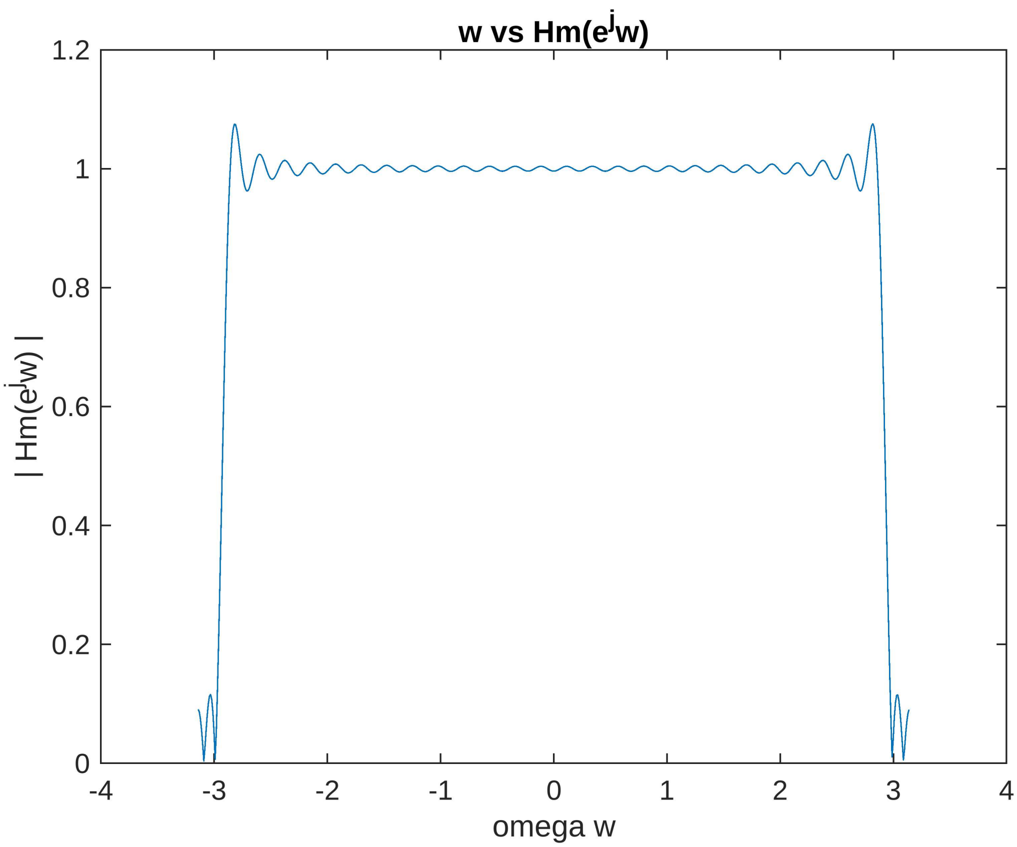

```
M = 18;
Wc = 0+pi.*rand;
n = -M:M;
w = -pi:0.01:pi;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(Wc * n(i)))/ (pi * n(i));
 else
 H_n(i) = Wc/ pi;
 end
end
```

```
for k = 1:length(w)
 H_sum = 0;
 for i = 1:length(n)
 H_sum = H_sum + H_n(i) * exp(-1i * w(k) * n(i));
 end
 H_m(k) = H_sum;
end
plot(w,abs(H_m));
xlabel("omega w");
ylabel("| Hm(e^jw) |");
title("w vs Hm(e^jw)");
```




```
M = 27;
Wc = 0+pi.*rand;
n = -M:M;
w = -pi:0.01:pi;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(Wc * n(i)))/ (pi * n(i));
 else
 H_n(i) = Wc/ pi;
 end
end
for k = 1:length(w)
```

```
  H_sum = 0;
  for i = 1:length(n)
  H_sum = H_sum + H_n(i) * exp(-1i * w(k) * n(i));
  end
  H_m(k) = H_sum;
end
plot(w,abs(H_m));
xlabel("omega w");
ylabel("| Hm(e^jw) |");
title("w vs Hm(e^jw)");
```




Question 5

```
% overshoot at Discontinuities: Fourier series approximations overshoot near
% discontinuities, about 9% of the jump.

%Oscillations Stay: The oscillations near discontinuities never disappear,
% no matter how many terms are added.

%Better in Smooth Areas: The approximation improves in smooth regions as
% more terms are included.

%More Terms, Sharper Overshoot: Adding more terms makes the overshoot sharper
% but doesn't reduce its size.
```

## B . Generation of random Sinusoid

1. Fix a value of Wc = [0,pi]
2. Fix a value of L1 and randomly genrate w1,w2,...wL where wl ~ u([0,wc]).
3. Fix a value of A and randomly generate A1,A2,A3,..Al where Al ~ u([1,A]).
4. Randomly generate phi1,phi2,phi3,...,phil, where phil~u([-pi,pi]).
5. Using {Wl},{Al} and {phil} construct the signal Xa[n] = sgm l = L1 to L2(Al*cos(We*n+phil)) 0 <= n <= N.
6. Similarly generate the signal Xb[n] = sgm k = L1 to L2 (Ak * cos(Wk*n+phik)) 0<=n<=N , where for some fixed L2 we have Wk ~ u([Wc,pi]) k = 1,2,..,L2 , Ak ~ u([1,A]) k = 1,2,....L2 phik ~u([-pi,pi]) k = 1,2,..,L2.
7. Obtain X[n] = Xa[n] + Xb[n].

Question 1

```
W = 0+pi.*rand();
```

Question 2

```
L1 = 69;
Wl = W * rand(1,L1);
```

Question 3

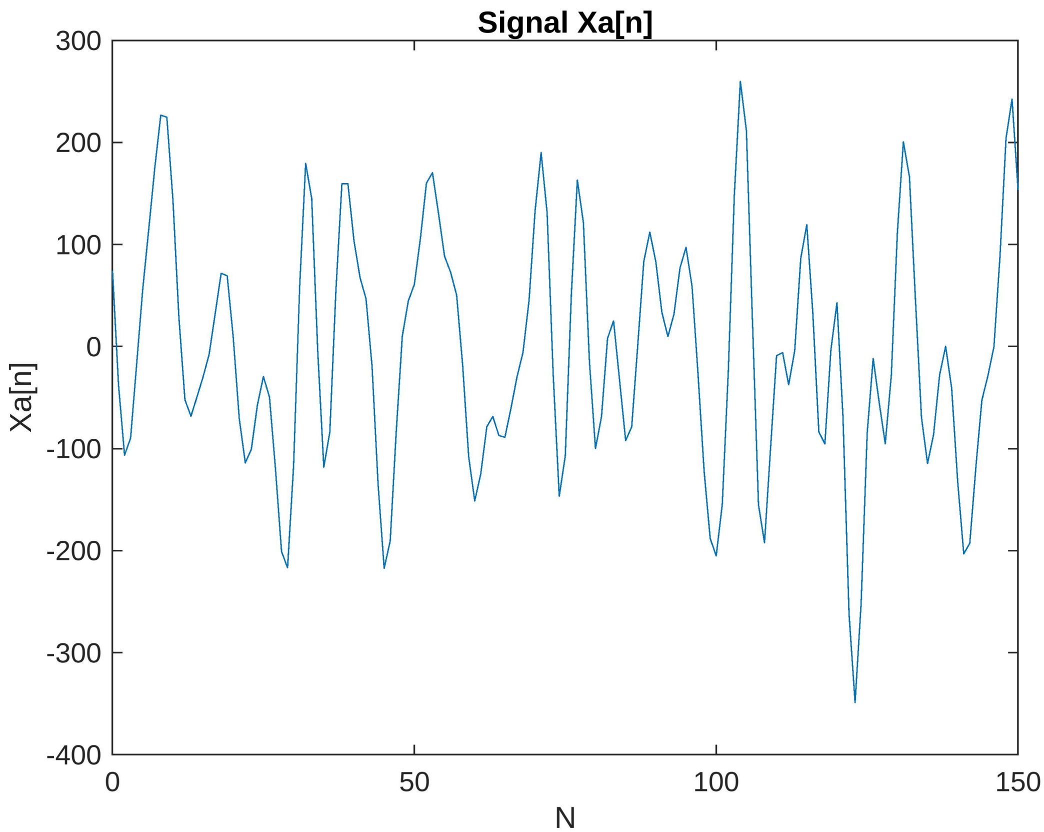
```
A = 30;
Al = 1+(A-1) * rand(1,L1);
```

Question 4

```
phi = -pi+(2*pi)*rand(1,L1);
```

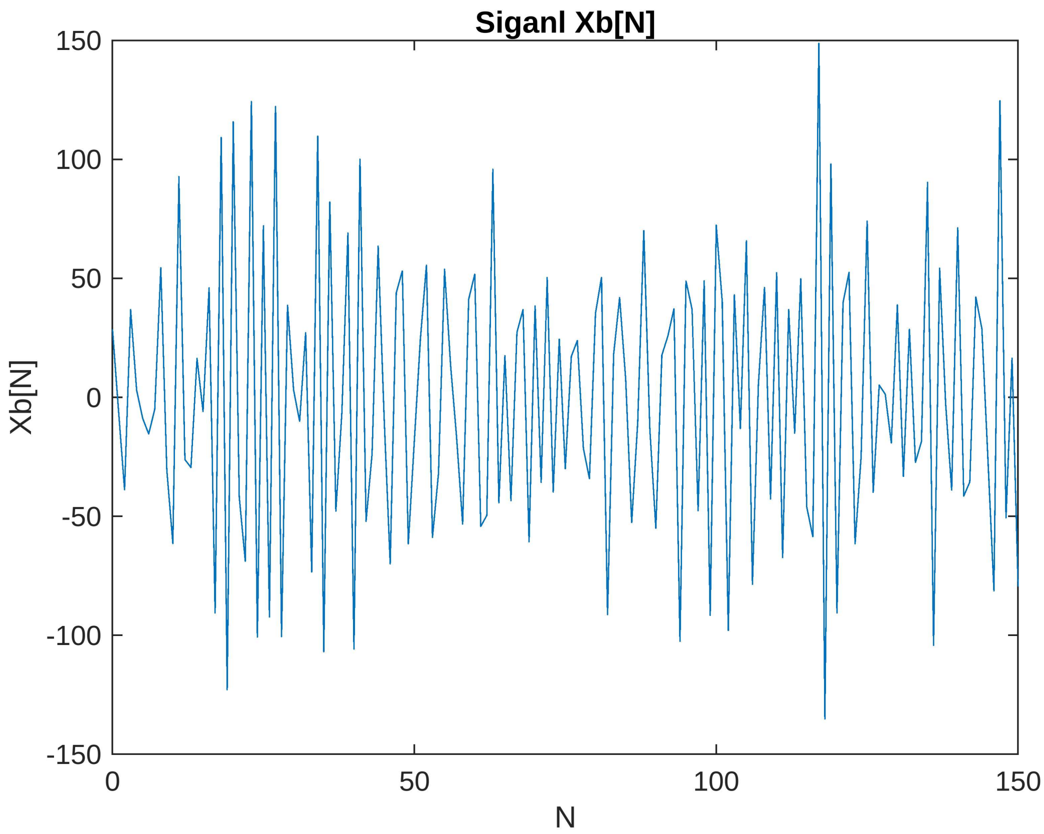
Question 5

```
N = 200;
n = 0:150;
Xa_n = 0;
for l = 1:L1
    Xa_n = Xa_n + Al(l).*cos(Wl(l)*n+phi(l));
end
plot(n,Xa_n);
xlabel("N");
ylabel("Xa[n]");
title("Signal Xa[n]");
```



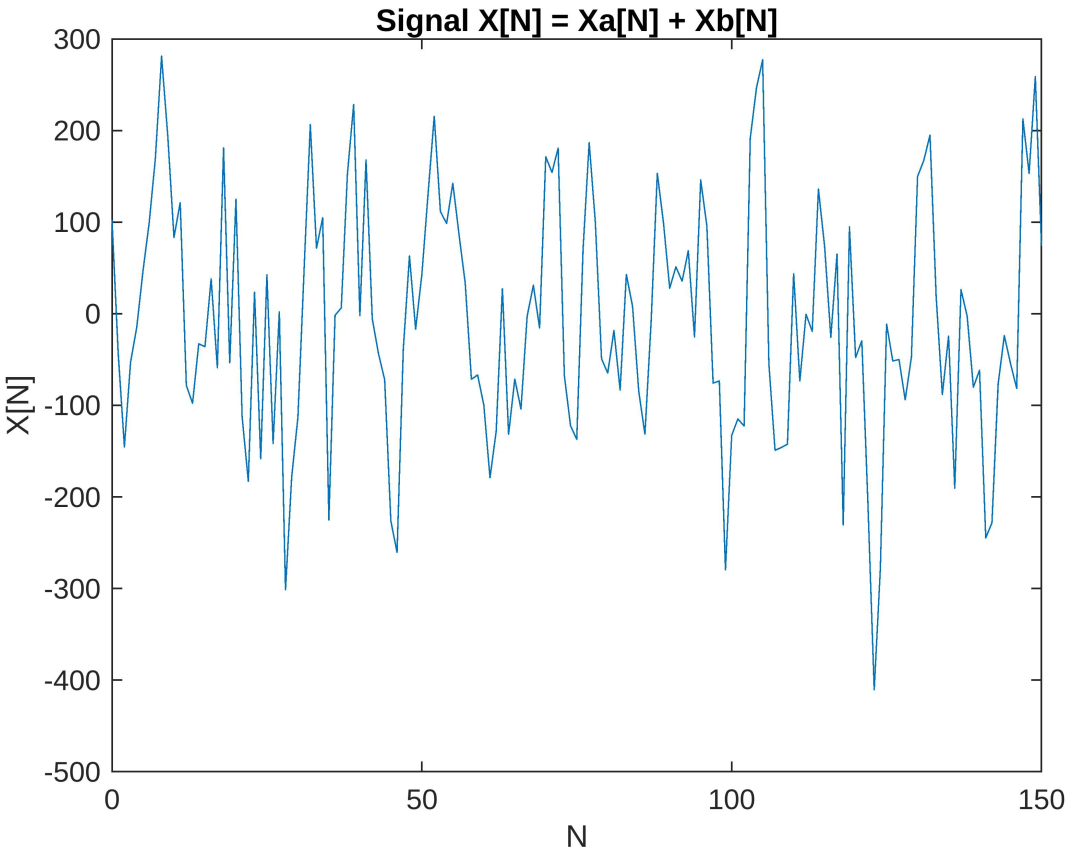

Question 6

```
L2 = 35;
Wk =   rand(1, L2) * (pi - W) + W;
Ak = 1+(A-1) * rand(1,L2);
phiK = rand(1, L2) * 2 * pi - pi ;
Xb_n = 0;
for l = 1:L2
    Xb_n = Xb_n + Ak(l).*cos(Wk(l).*n+phiK(l));
end
plot(n,Xb_n);
xlabel("N");
ylabel("Xb[N]");
title("Siganl Xb[N]");
```




Question 7

```
X_n = Xa_n + Xb_n;
plot(n,X_n);
xlabel("N");
ylabel("X[N]");
title("Signal X[N] = Xa[N] + Xb[N]");
```




## C. Filtering X[n] using truncated LPF

1. Fix a value of M.
2. Generate The impulse response of the truncated LPF : hlp[n] = sin(wc*n)/pi*n  -M<=n<=M.
3. Compute Ylp[n] = X[n] * hlp[n].
4. Compare Ylp[n] with X[n].
5. Repeat the experiement with few more values of M.
6. Report your observations.

Question 1

```
M = 37;
W = 0+pi*rand();
n = -M:M;
```

Question 2

```
H_n = 0;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(W * n(i)))/ (pi * n(i));
 else
 H_n(i) = W/ pi;
 end
```

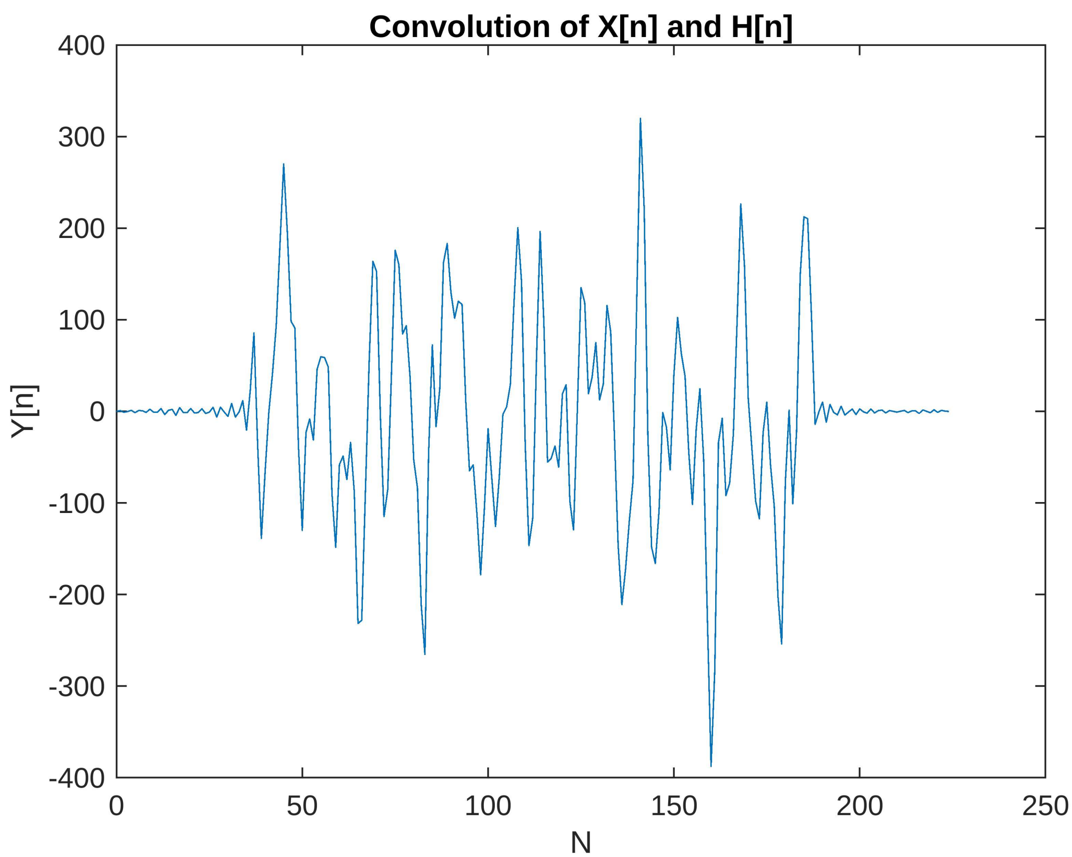
```
end
```

Question 3

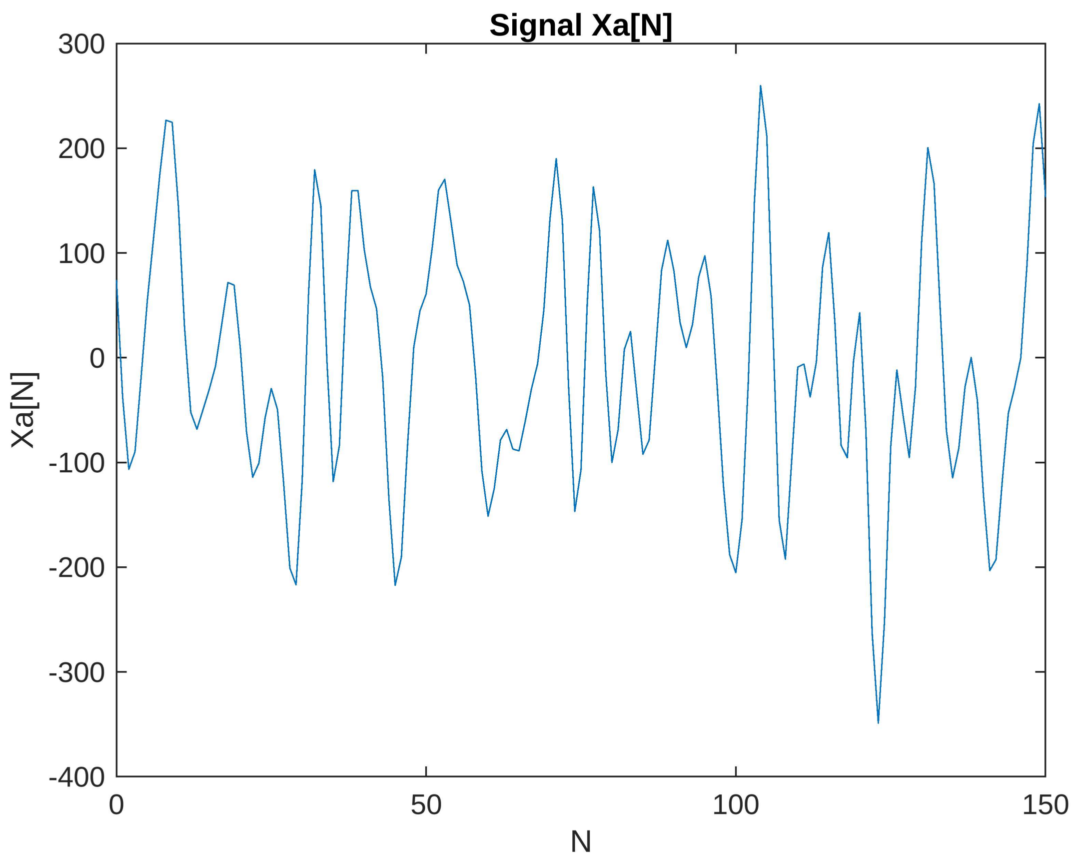
```
Y = conv(X_n,H_n);
```

Question 4

```
n = 0 : length(Y)-1;
plot(n,Y);
xlabel("N");
ylabel("Y[n]");
title("Convolution of X[n] and H[n]");
```

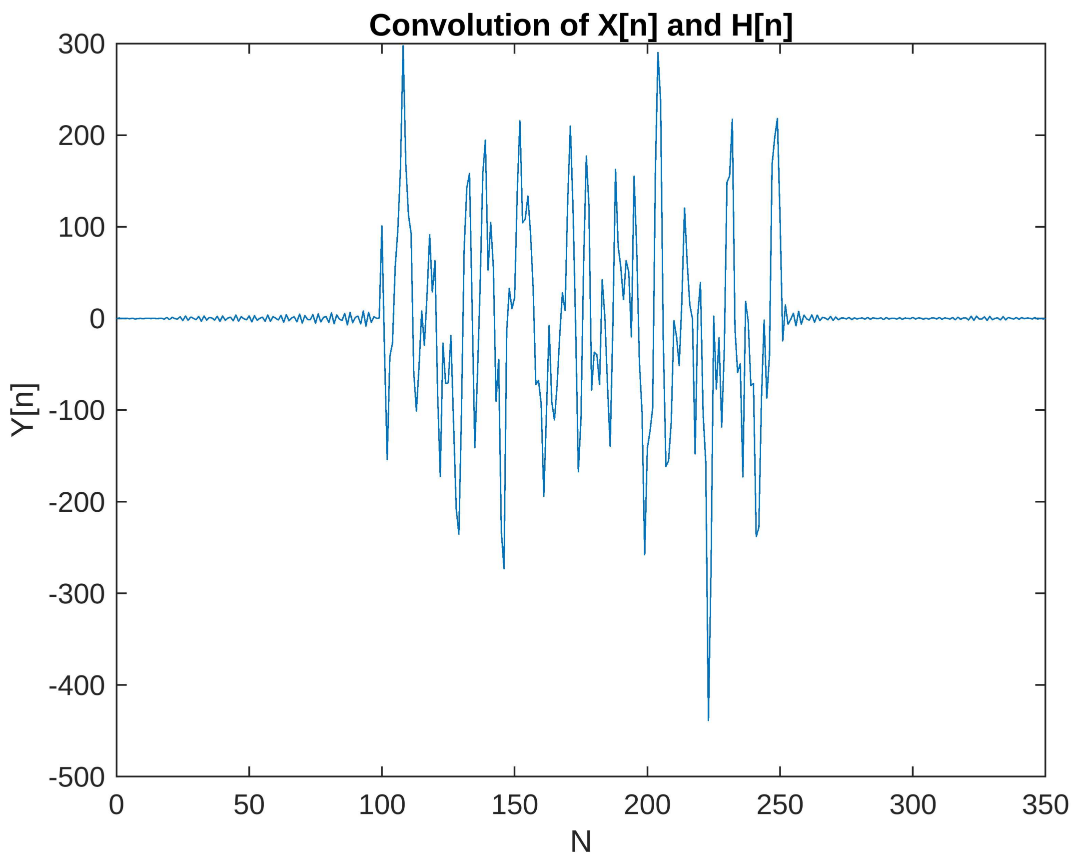



```
plot(0:length(X_n)-1 , Xa_n);
xlabel("N");
ylabel("Xa[N]");
title("Signal Xa[N]");
```




Question 5

```
M = 100;
W = 0+pi*rand();
n = -M:M;
H_n = 0;
for i = 1:length(n)
 if n(i) ~= 0
 H_n(i) = (sin(W * n(i)))/ (pi * n(i));
 else
 H_n(i) = W/ pi;
 end
end
Y = conv(X_n,H_n);
n = 0 : length(Y)-1;
plot(n,Y);
xlabel("N");
ylabel("Y[n]");
title("Convolution of X[n] and H[n]");
```




Question 6

```
%small M: Higher frequencies are not fully filtered out, resulting in less
% accurate filtering.

%Increasing M: The filter becomes more accurate, effectively removing
% high-frequency components.

%Trade-off: Larger M improves filtering but increases computational cost
% and filter length.

%Edge Effects: At signal edges, convolution can introduce artifacts due to
% truncation.
```

## D. Filtering X[n] using moving average system

1. Fix a value of M.

2. Generate the impulse response of the moving average system Hm[n] = 1/(M+1) 0<=n<=M.
3. Compute Y_ma[n] = X[n] *Hm[n].
4. Compare Y_ma[n] and Xa[n].
5. Repeat the experiment with a few more values of M.
6. Report your observations.

Question 1

```
M = 69;
```

Question 2

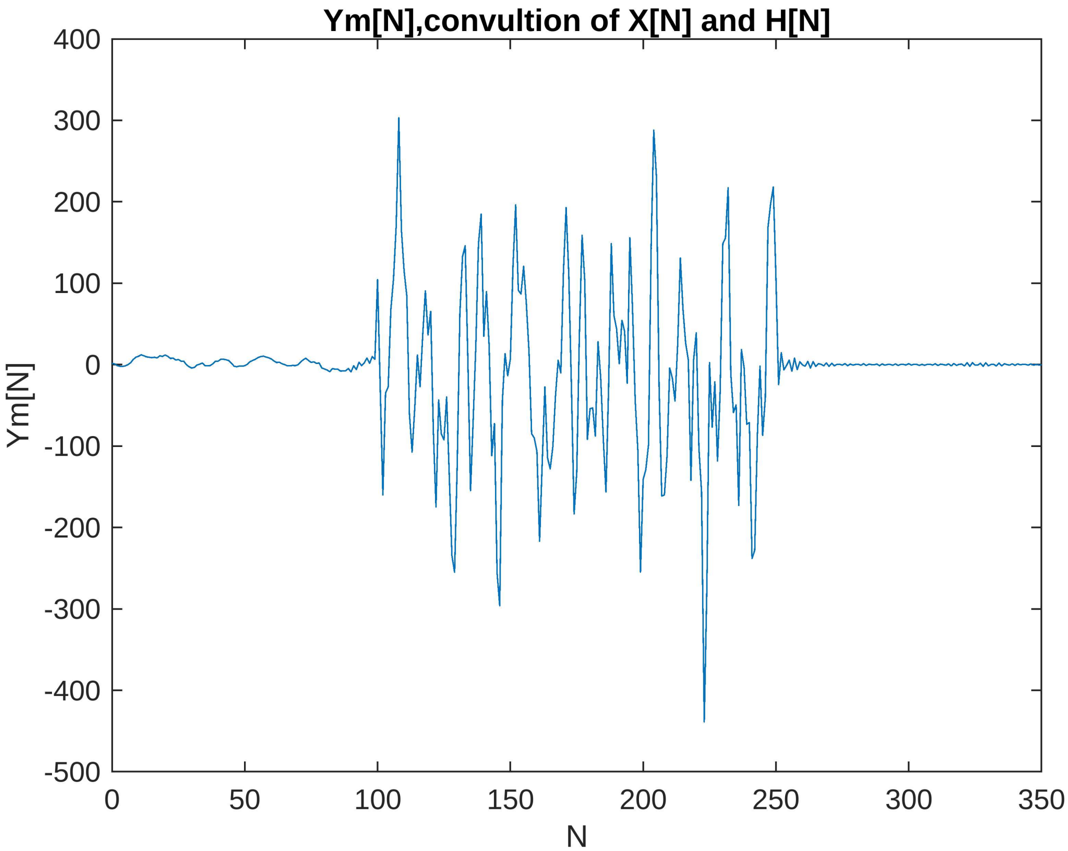
```
n= 0:M;
for i = 1:length(n)
 H_n(i) = 1/(M+1);

end
```

Question 3

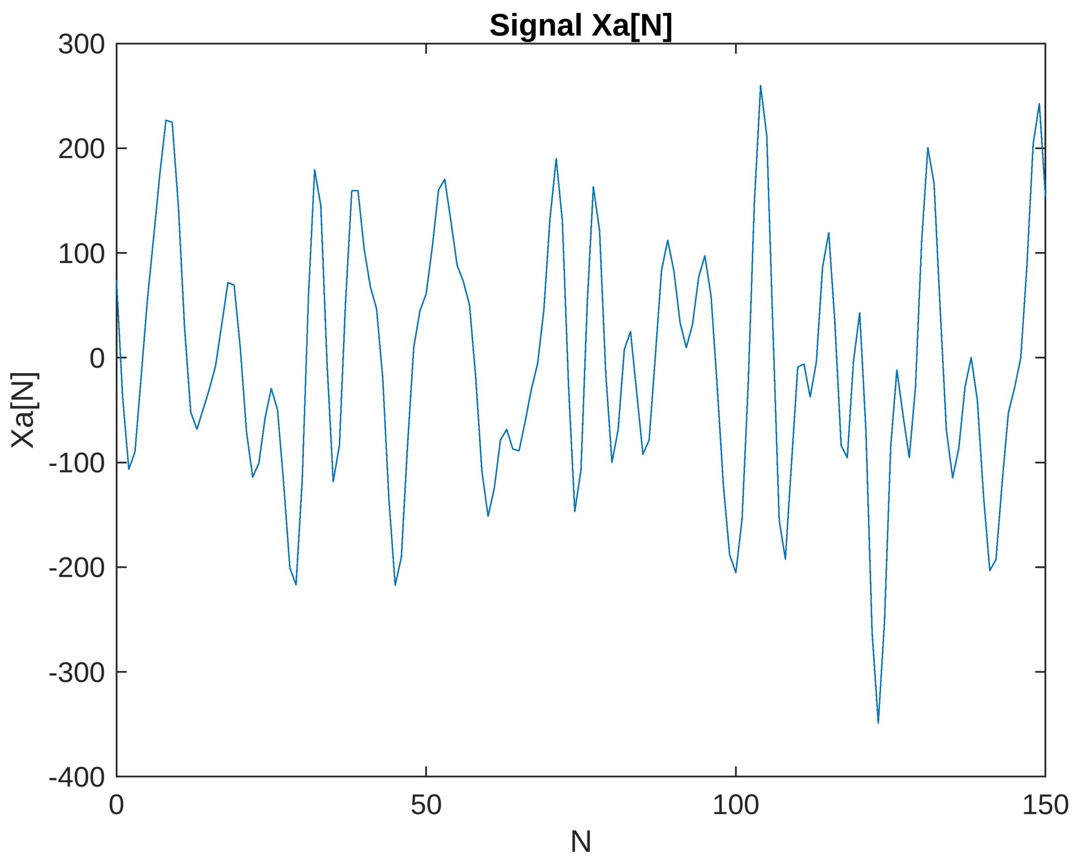
```
Y_m = conv(X_n,H_n);
```

Question 4

```
plot(0:length(Y_m)-1 , Y_m);
xlabel("N");
ylabel("Ym[N]");
title("Ym[N],convultion of X[N] and H[N]");
```

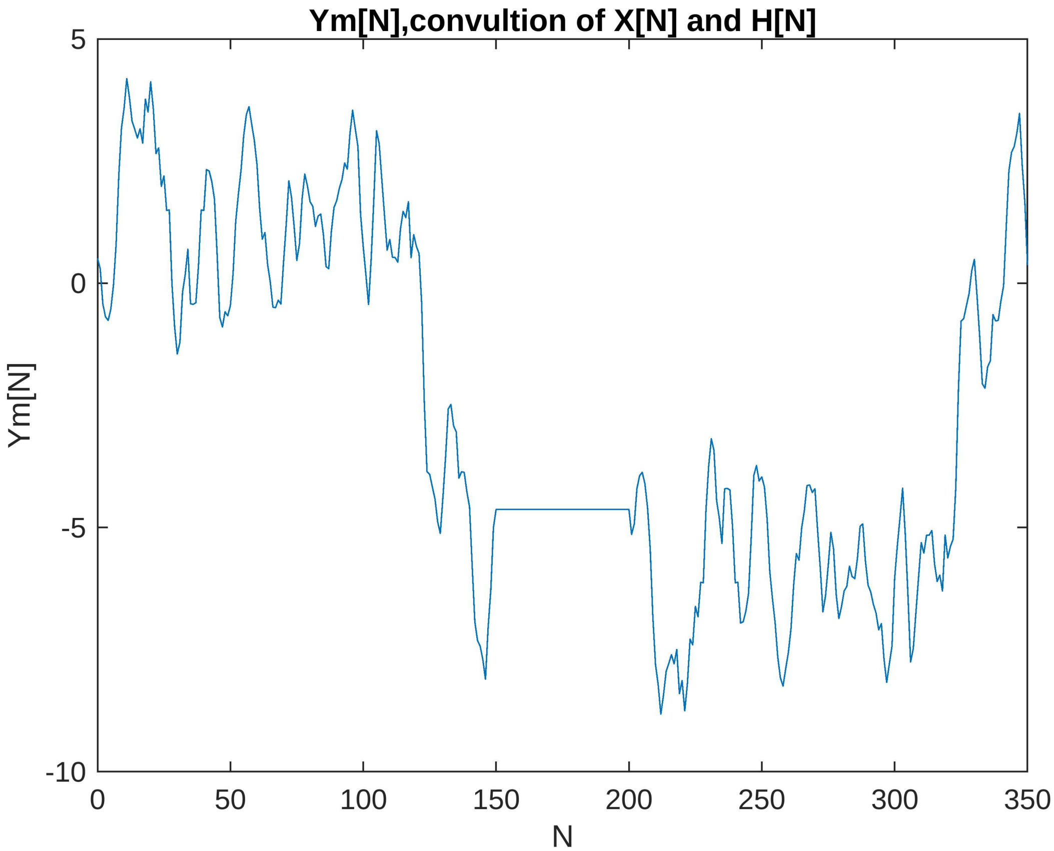



```
plot(0:length(Xa_n)-1 , Xa_n);
xlabel("N");
ylabel("Xa[N]");
title("Signal Xa[N]");
```

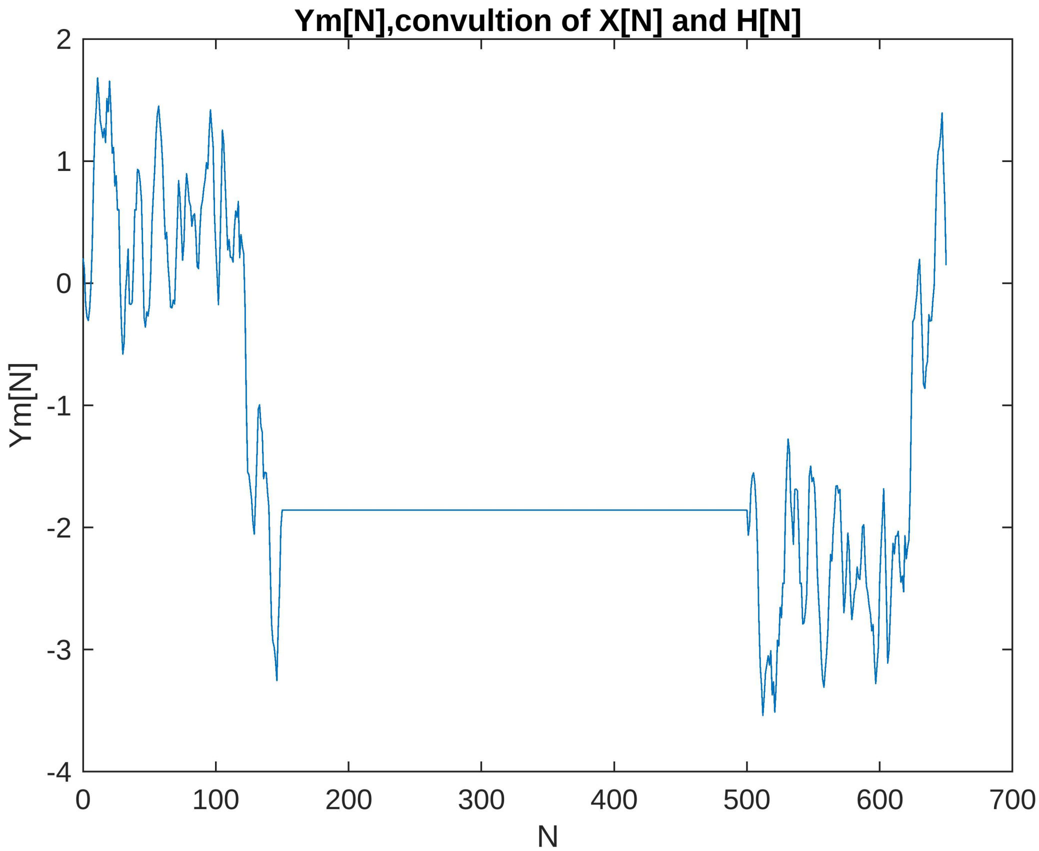



Question 5

```
M = 200;
n= 0:M;
for i = 1:length(n)
 H_n(i) = 1/(M+1);

end
Y_m = conv(X_n,H_n);
plot(0:length(Y_m)-1 , Y_m);
xlabel("N");
ylabel("Ym[N]");
title("Ym[N],convultion of X[N] and H[N]");
```




```
M = 500;
n= 0:M;
for i = 1:length(n)
 H_n(i) = 1/(M+1);

end
Y_m = conv(X_n,H_n);
plot(0:length(Y_m)-1 , Y_m);
xlabel("N");
ylabel("Ym[N]");
title("Ym[N],convultion of X[N] and H[N]");
```




Question 6

```
% Small M: The filter provides a basic smoothing effect, but high-frequency
% components may still be present.

%Increasing M: As M increases, the smoothing improves, reducing both
% high-frequency noise and signal variations.

%Trade-off: A larger M results in better smoothing but causes more signal
% delay and loss of fine details.

%Edge Effects: The moving average can distort the signal near the
% boundaries due to fewer available data points for averaging.
```