Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Title:	Signals and Systems II
Course Number:	ELE632
Semester/Year (e.g.F2016)	W2022

Instructor:	Dimitri Androutsos

Assignment/Lab Number:	4
Assignment/Lab Title:	Discrete-Time Fourier Transform

Submission Date:	Sunday, March 27th, 2022
Due Date:	Sunday, March 27th, 2022

Student LAST Name	Student FIRST Name	Student Number	Section	Signature*
Fahmy	Ahmad	500913092	9	A.F.

**By signing above, you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work <u>will result in an investigation of Academic Misconduct and may result in a "0" on the work,</u> an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*

https://www.ryerson.ca/senate/policies/pol60.pdf

Appendix

A) Discrete Time Fourier Transform (DTFT)

B) Time Convolution

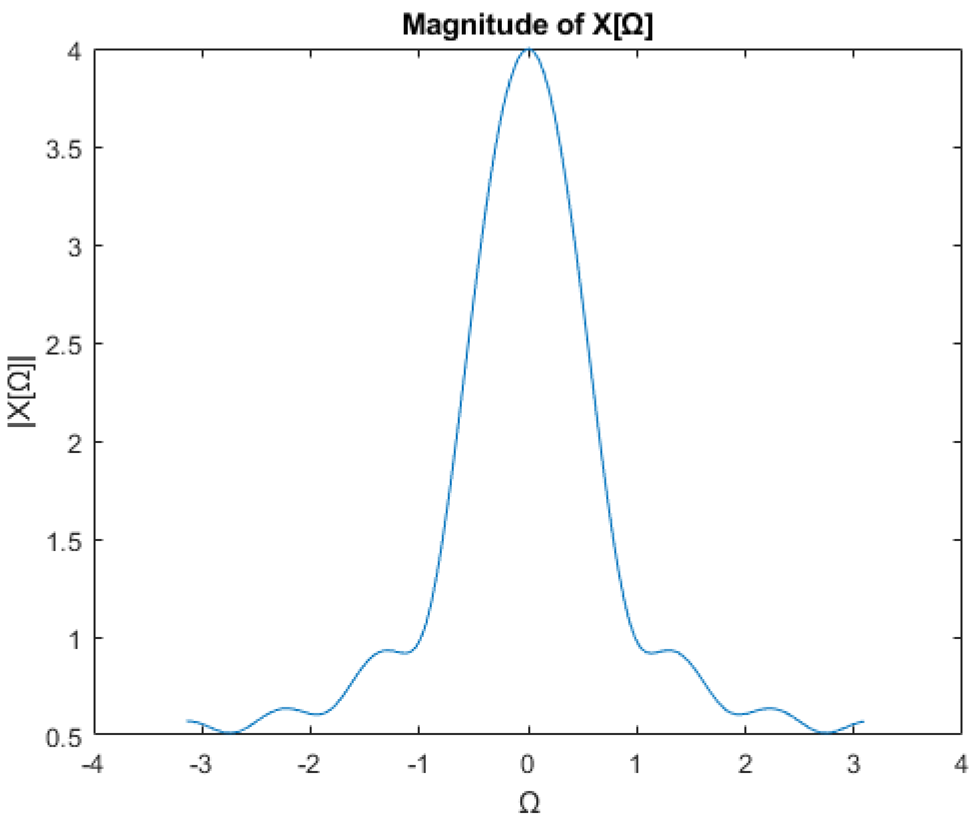
C) FIR Filter Design by Frequency Sampling

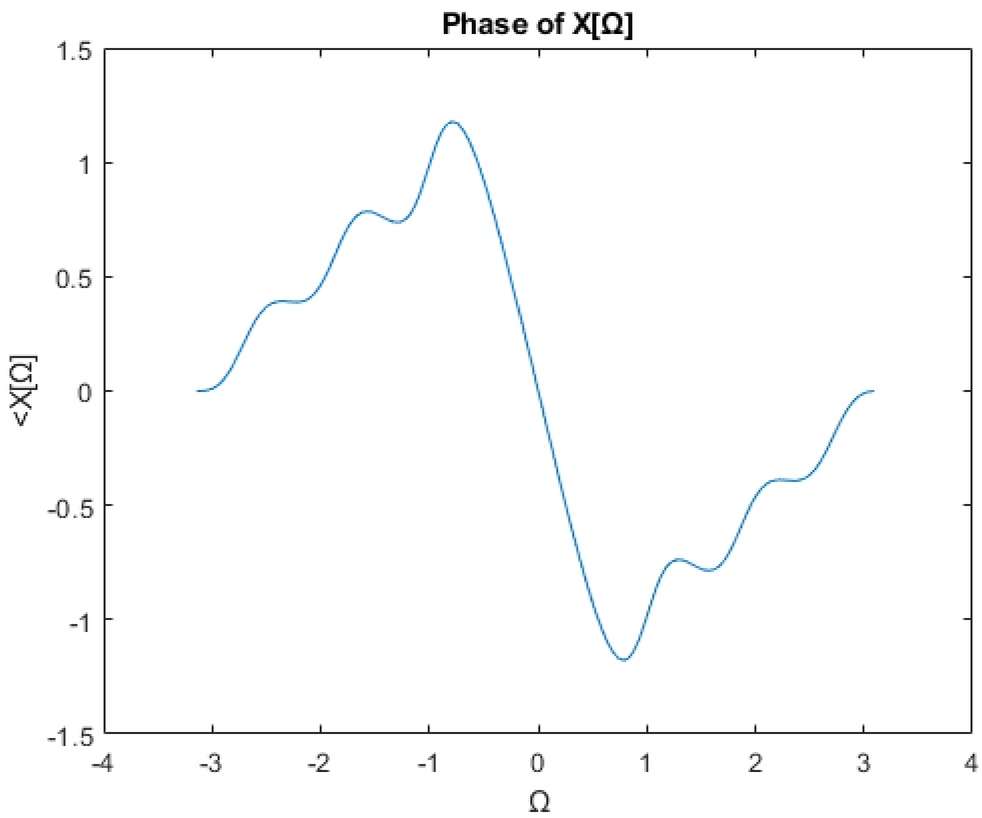
A) Discrete-Time Fourier Transform (DTFT)

Part 1 - DTFT

```
clc
clear
N0 = 128;
n = 0:N0-1;
ohm = (2*pi/128).*(-64:63);

x = [1 (6/7) (5/7) (4/7) (3/7) (2/7) (1/7) zeros(1, 121)];
X = fft(x);

figure;
plot(ohm, fftshift(abs(X)));
title('Magnitude of X[Ω]')
xlabel('Ω')
ylabel('|X[Ω]|')

figure;
plot(ohm, fftshift(angle(X)));
title('Phase of X[Ω]')
xlabel('Ω')
ylabel('<X[Ω]')

% figure;
% plot(ohm, abs(X));
% title('Magnitude of X[Ω]')
% xlabel('Ω')
% ylabel('|X[Ω]|')
%
% figure;
```

```
% plot(ohm, angle(X));
% title('Phase of X[Ω]')
% xlabel('Ω')
% ylabel('<X[Ω]')
```





Part 2 - FTDT by Hand

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\Omega n}$$

$$x(n) = \left(\frac{-1}{7}n + 1\right)\left(u(n) - u(n-6)\right)$$

$$X(\Omega) = \sum_{n=-\infty}^{\infty} \left(\frac{-1}{7}n+1\right)\left(u(n) - u(n-6)\right) e^{-j\Omega n}$$

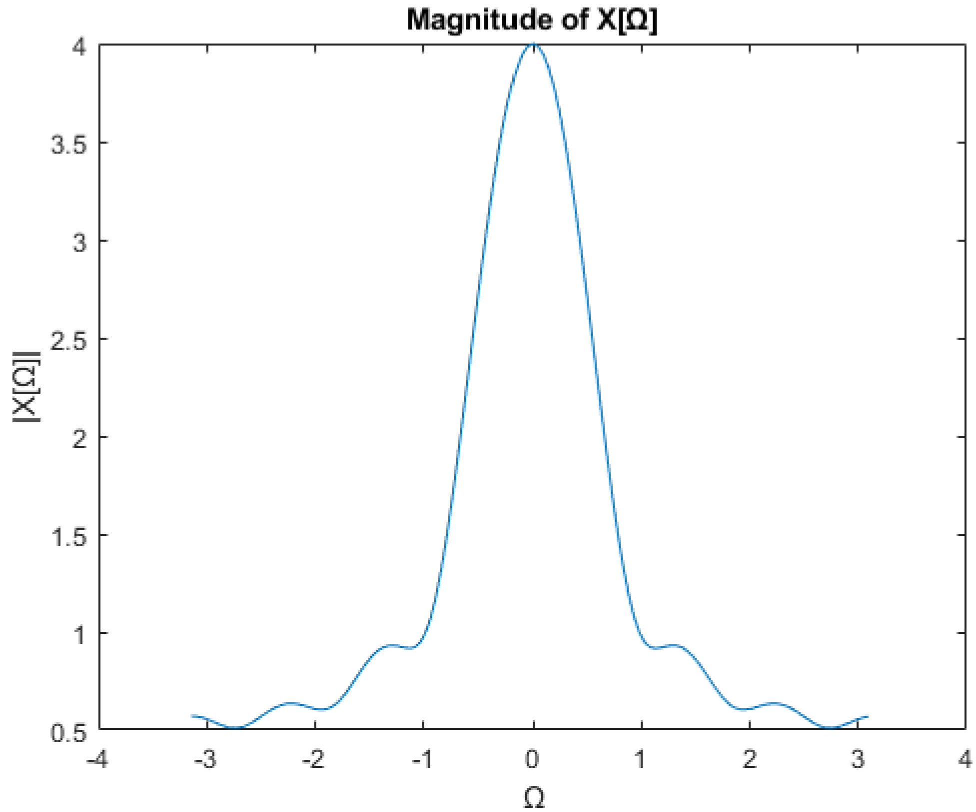
$$= \sum_{n=0}^{6} \left(\frac{-1}{7}n+1\right) e^{-j\Omega(n+6)}$$

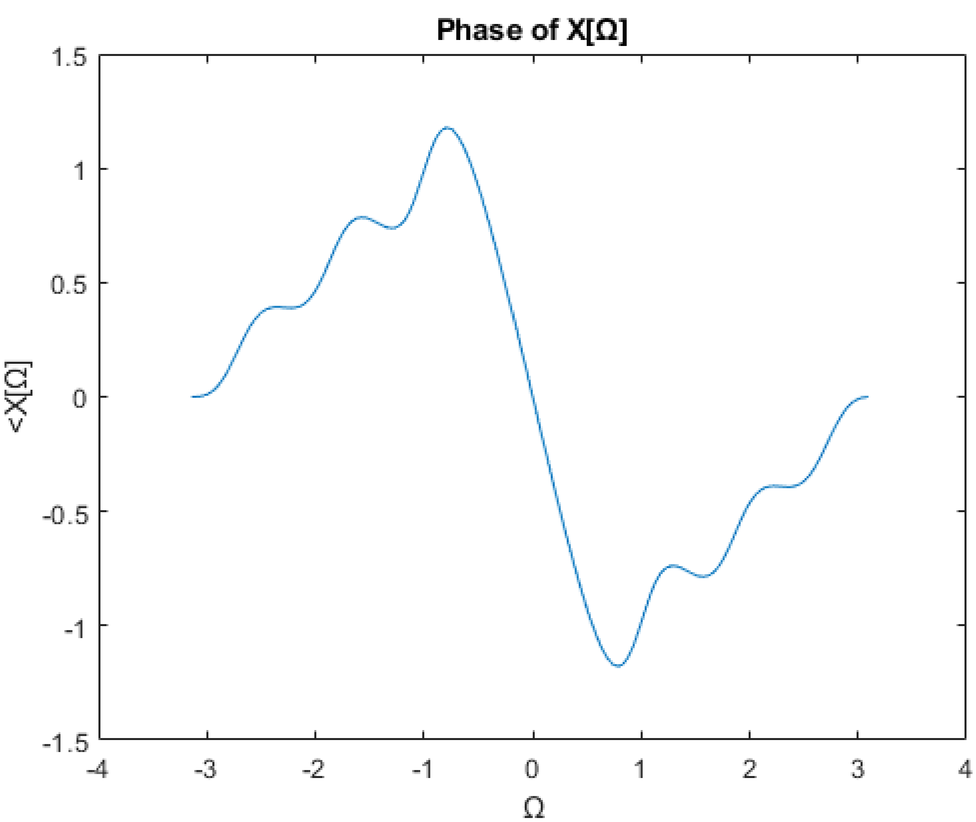
$$X(\Omega) = 1 + \frac{6}{7}e^{-j\Omega} + \frac{5}{7}e^{-2j\Omega} + \frac{4}{7}e^{-3j\Omega}$$

$$+ \frac{3}{7}e^{-4j\Omega} + \frac{2}{7}e^{-5j\Omega} + \frac{1}{7}e^{-6j\Omega}$$

```
clc
clear
```

```
ohm = (2*pi/128).*(-64:63);

X = 1 + (6/7).*exp(-1i.*ohm) ...
    + (5/7).*exp(-1i.*2.*ohm) ...
    + (4/7).*exp(-1i.*3.*ohm) ...
    + (3/7).*exp(-1i.*4.*ohm) ...
    + (2/7).*exp(-1i.*5.*ohm) ...
    + (1/7).*exp(-1i.*6.*ohm);

figure;
plot(ohm, abs(X));
title('Magnitude of X[Ω]')
xlabel('Ω')
ylabel('|X[Ω]|')



figure;
plot(ohm, angle(X));
title('Phase of X[Ω]')
xlabel('Ω')
ylabel('<X[Ω]')
```


Magnitude of X[Ω]
|X[Ω]|
Ω
4
3.5
3
2.5
2
1.5
1
0.5
-4
-3
-2
-1
0
1
2
3
4

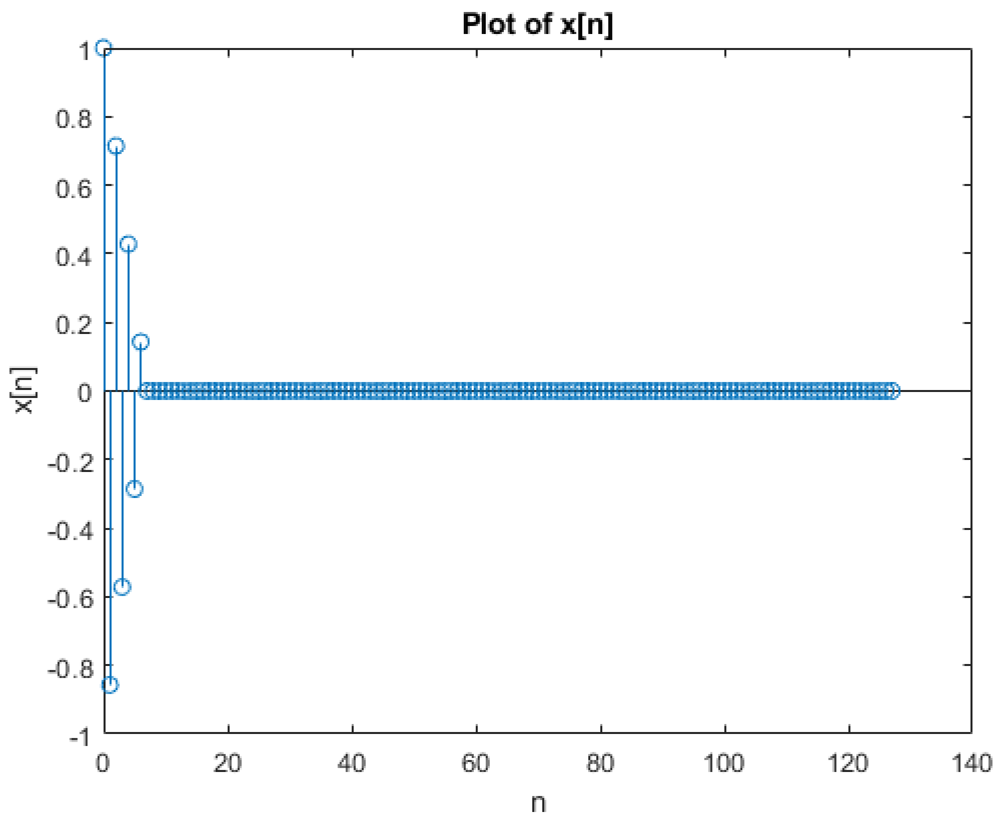

Phase of X[Ω]
<X[Ω]
Ω
1.5
1
0.5
0
-0.5
-1
-1.5
-4
-3
-2
-1
0
1
2
3
4

Response: Yes, the results of part 2 are consistent with part 1.

Part 3 - IFFT of Hand Calculated DTFT

```
clc
clear
N0 = 128;
n = 0:N0-1;
ohm = (2*pi/128).*(-64:63);

X = 1 + (6/7).*exp(-1i.*ohm) ...
    + (5/7).*exp(-1i.*2.*ohm) ...
    + (4/7).*exp(-1i.*3.*ohm) ...
    + (3/7).*exp(-1i.*4.*ohm) ...
    + (2/7).*exp(-1i.*5.*ohm) ...
    + (1/7).*exp(-1i.*6.*ohm);

% X = fftshift(X);

x = ifft(X);

figure;
stem(n, x);
title('Plot of x[n]')
xlabel('n')
ylabel('x[n]')
```

Warning: Using only the real component of complex data.



Response: No, the result isn't the same as x[n] since the definition of the Fourier Transform involves the multiplication by a complex exponential thus resulting in a shift when switching between domains.

B) Time Convolution

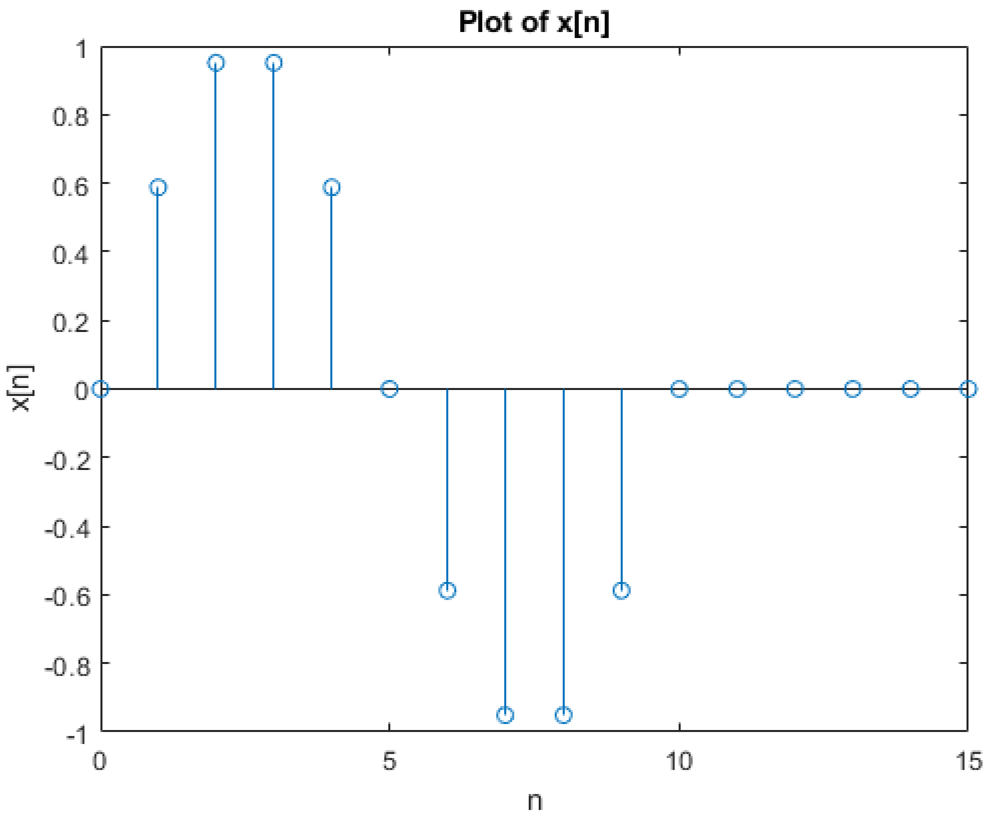
Part 1 - DTFT plot of x[n]

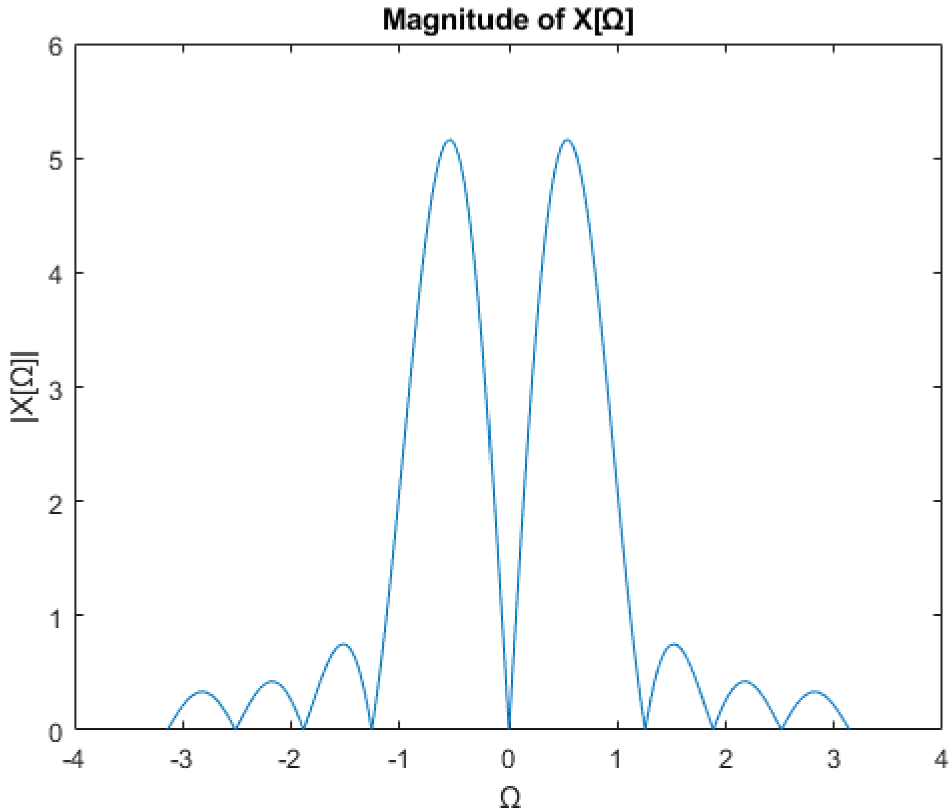
```
clc
clear
n = (0:15);
u_c = @(t) 1.0.*(t>=0);
u = @(n) u_c(n).*(mod(n,1) == 0);
x = sin(2*pi*n/10).*(u(n) - u(n-10));

omega= linspace(-pi,pi,1001);
W_omega = exp(-1i).^((0:length(x)-1)'*omega);
X = (x*W_omega);
```

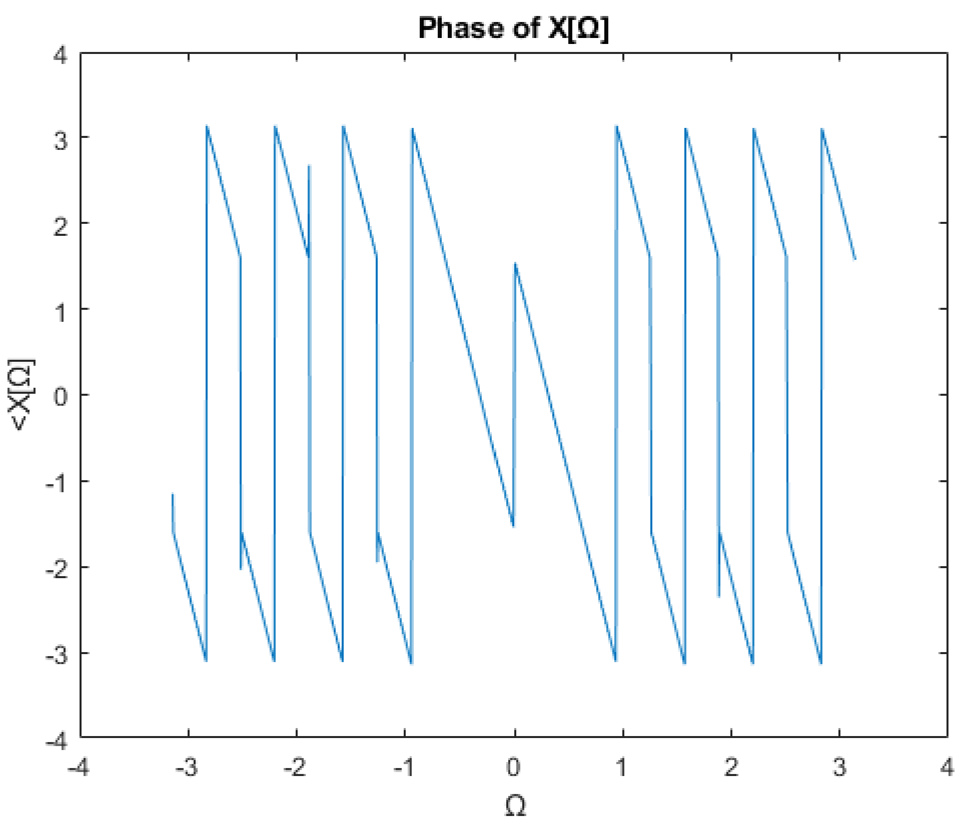
```
figure;
stem(n, x);
title('Plot of x[n]')
xlabel('n')
ylabel('x[n]')

figure;
plot(omega, abs(X));
title('Magnitude of X[Ω]')
xlabel('Ω')
ylabel('|X[Ω]|')

figure;
plot(omega, angle(X));
title('Phase of X[Ω]')
xlabel('Ω')
ylabel('<X[Ω]')



% Part 2 - DTFT plot of h[n]
n = (0:9);
x = u(n)-u(n-9);



omega= linspace(-pi,pi,1001);
W_omega = exp(-1i).^((0:length(x)-1)'*omega);
H = (x*W_omega);

figure;
```

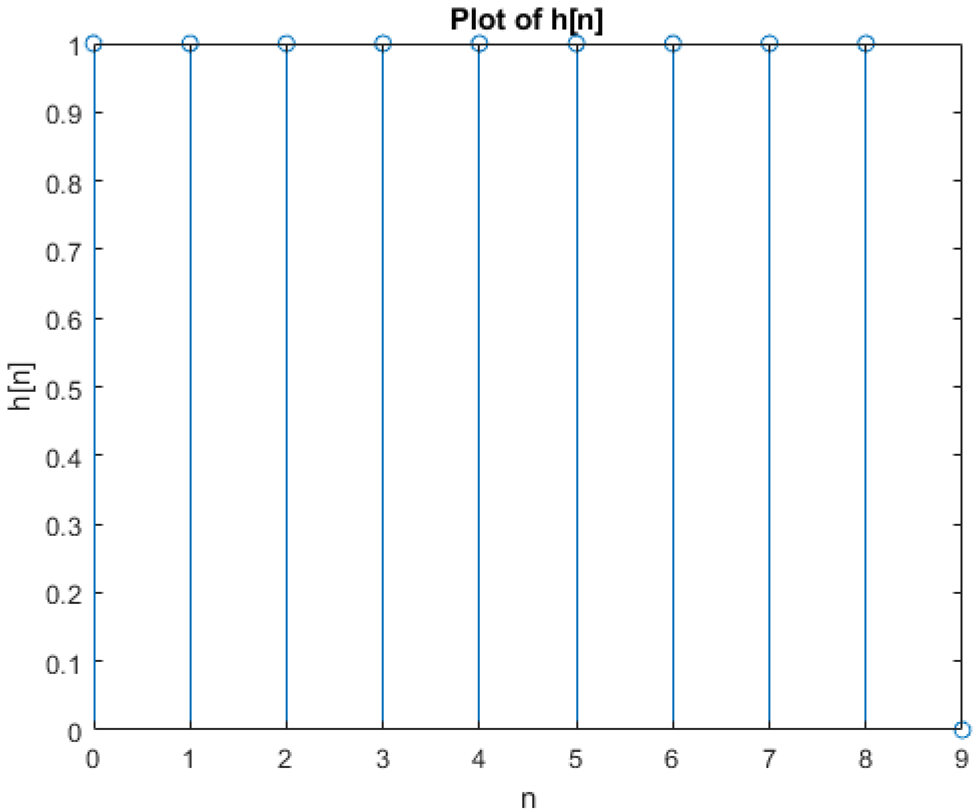
```
stem(n, x);
title('Plot of h[n]')
xlabel('n')
ylabel('h[n]')

figure;
plot(omega, abs(H));
title('Magnitude of H[Ω]')
xlabel('Ω')
ylabel('|H[Ω]|')

figure;
plot(omega, angle(H));
title('Phase of H[Ω]')
xlabel('Ω')
ylabel('<H[Ω]')

% Part 3 - Convolution Plot of X[Ω] and H[Ω]

Y = X.*H;

figure;
plot(omega, abs(Y));
title('Magnitude of Y[Ω]')
xlabel('Ω')
ylabel('|Y[Ω]|')

figure;
plot(omega, angle(Y));
title('Phase of Y[Ω]')
```

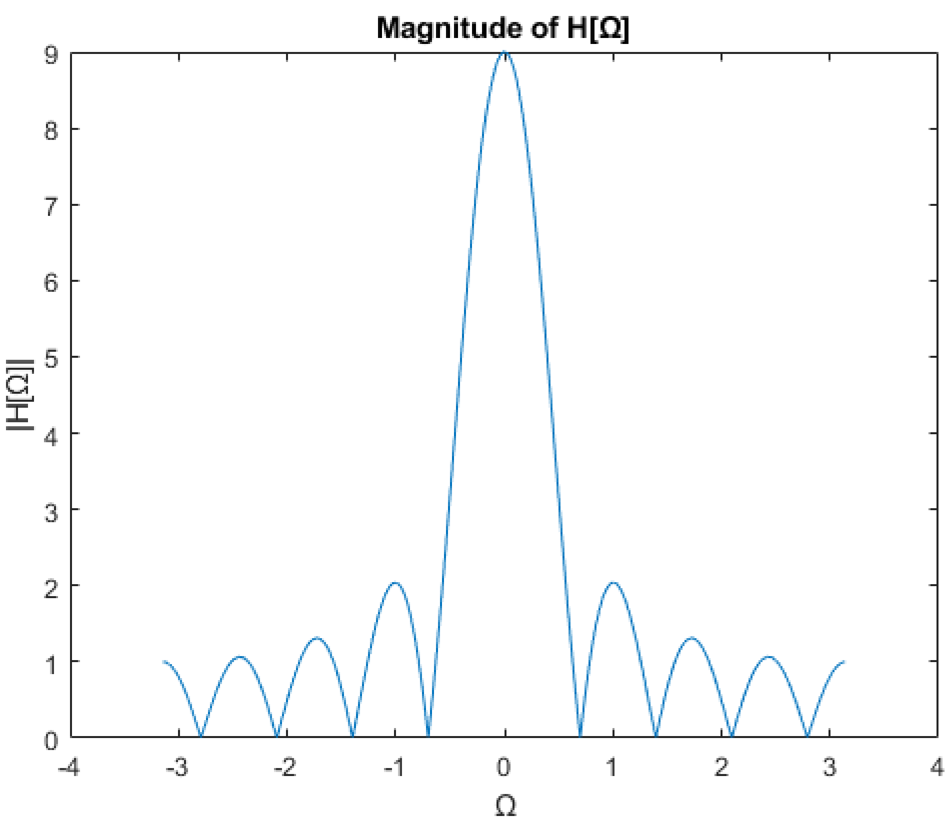
```
xlabel('Ω')
ylabel('<Y[Ω]')
```

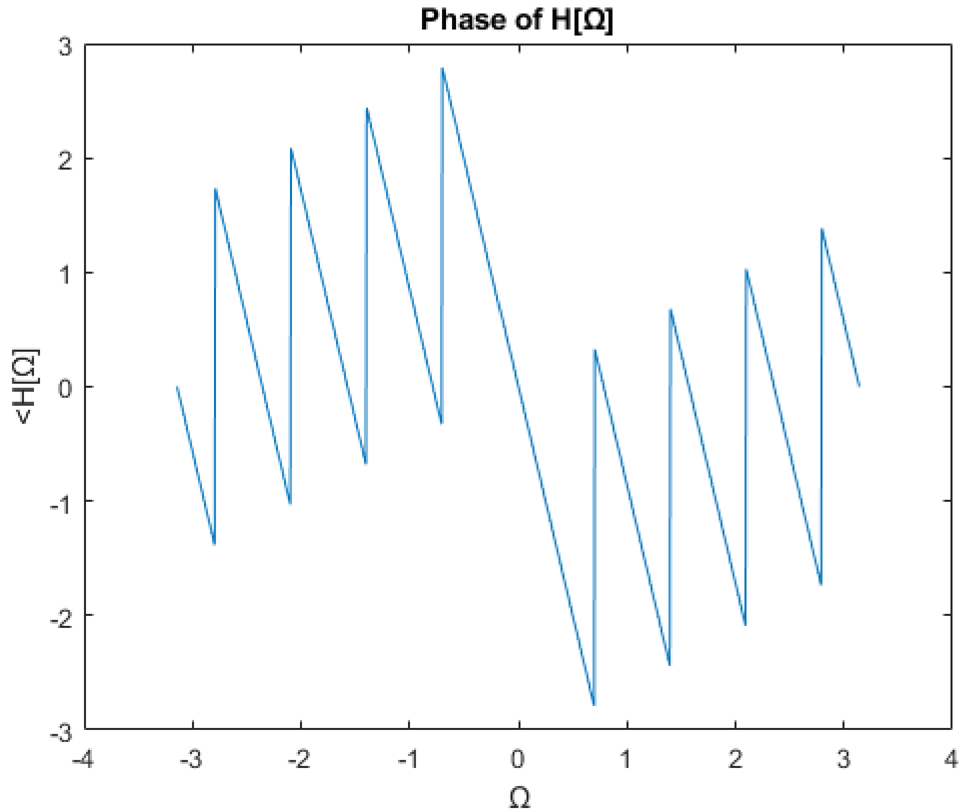


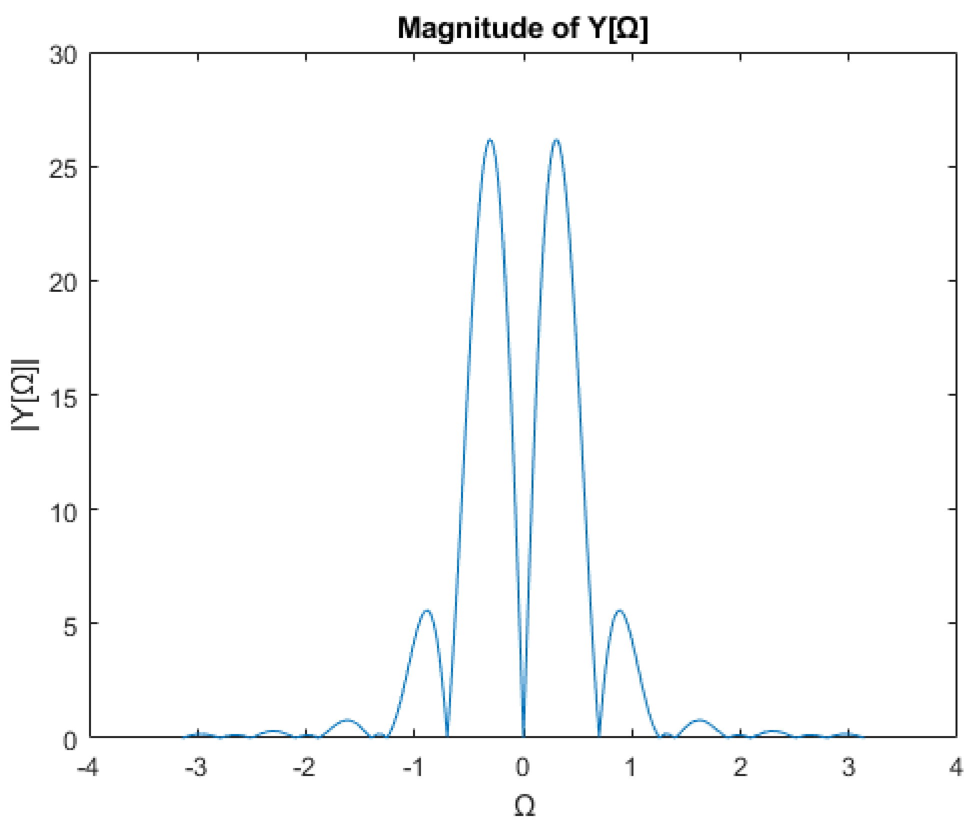

Magnitude of X[Ω]
|X[Ω]|
Ω
6
5
4
3
2
1
0
-4
-3
-2
-1
0
1
2
3
4

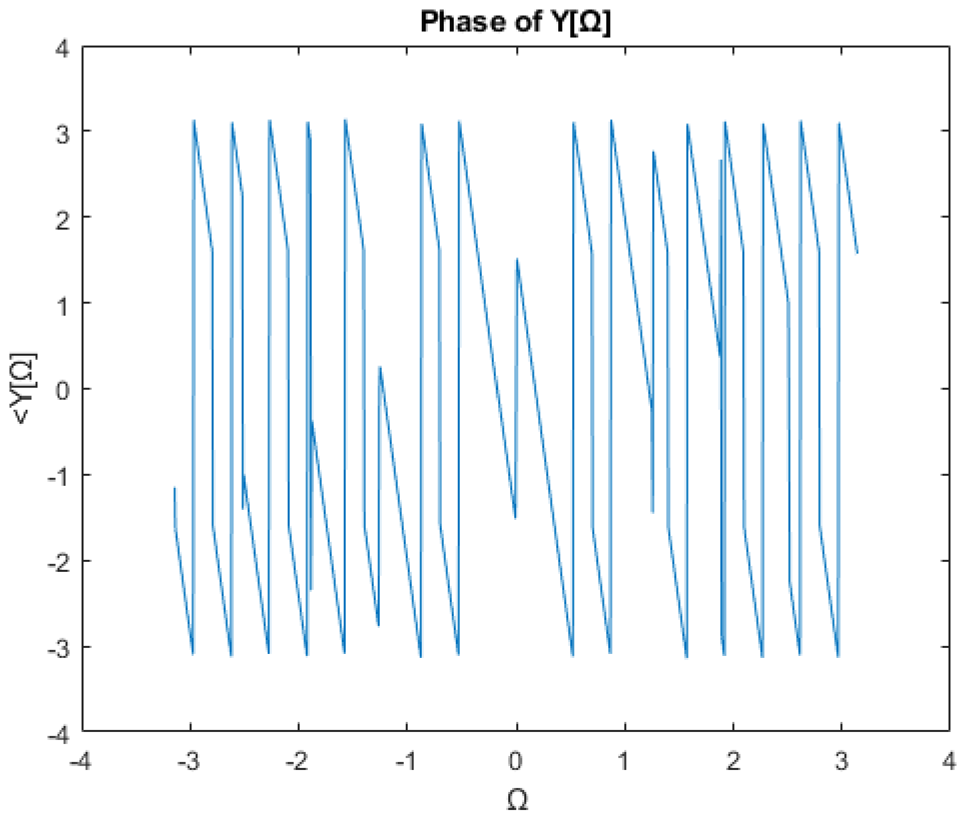

Phase of X[Ω]
<X[Ω]
Ω
4
3
2
1
0
-1
-2
-3
-4
-4
-3
-2
-1
0
1
2
3
4


Plot of h[n]
h[n]
n
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
4
5
6
7
8
9


Magnitude of H[Ω]
|H[Ω]|
Ω
0
1
2
3
4
5
6
7
8
9
-4
-3
-2
-1
0
1
2
3
4


Phase of H[Ω]
<H[Ω]
Ω
3
2
1
0
-1
-2
-3
-4
-3
-2
-1
0
1
2
3
4

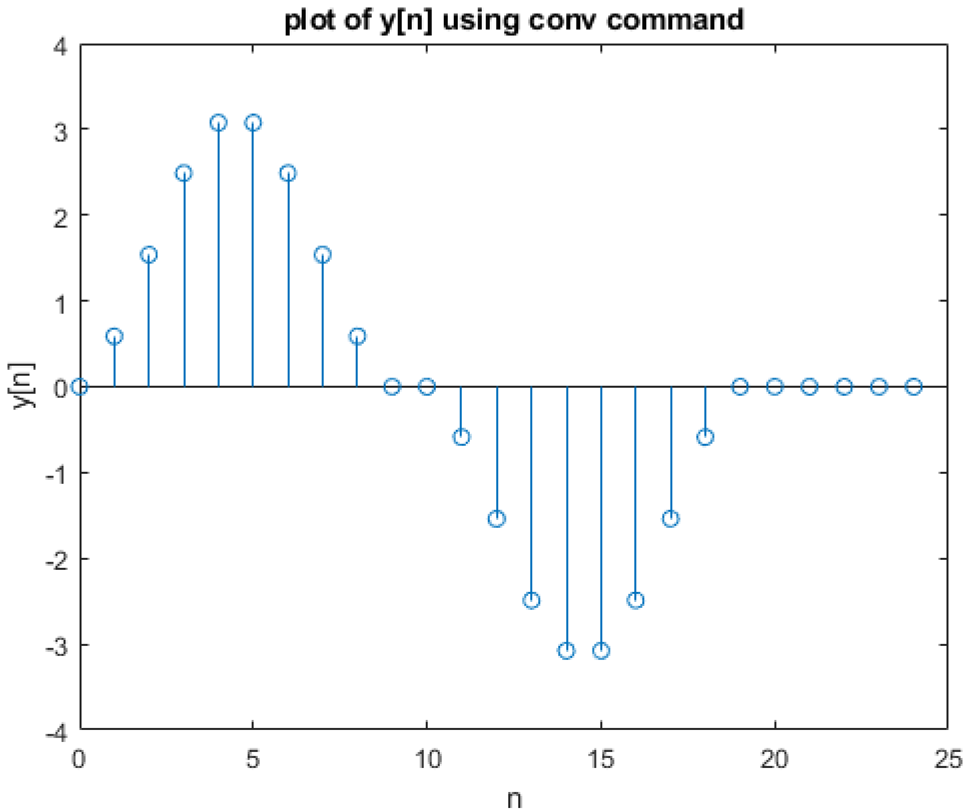

Magnitude of Y[Ω]
|Y[Ω]|
Ω
30
25
20
15
10
5
0
-4
-3
-2
-1
0
1
2
3
4



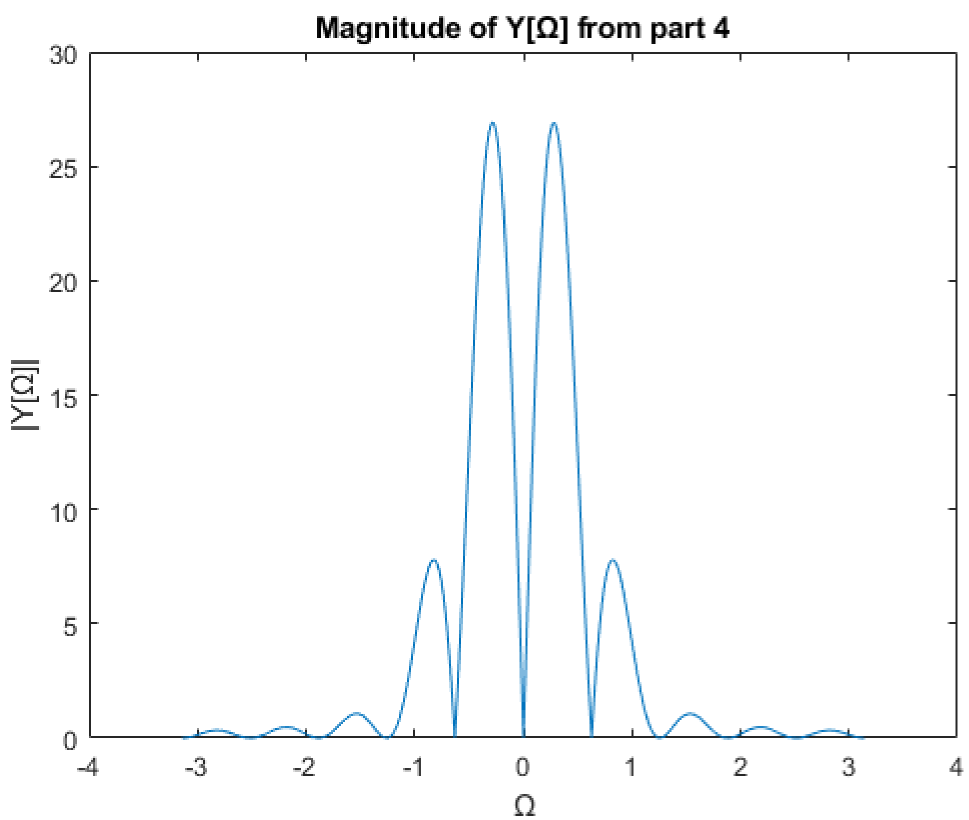
Part 4 - Convolution Plot of x[n] and h[n] by conv command

```
clc
clear
n = (0:15);
u_c = @(t) 1.0.*(t>=0);
u = @(n) u_c(n).*(mod(n,1) == 0);

h = u(0:9);
x = sin(2*pi*n/10).*(u(n) - u(n-10));
n=0:24;

y = conv(x, h);

figure;
stem(n, y);
```

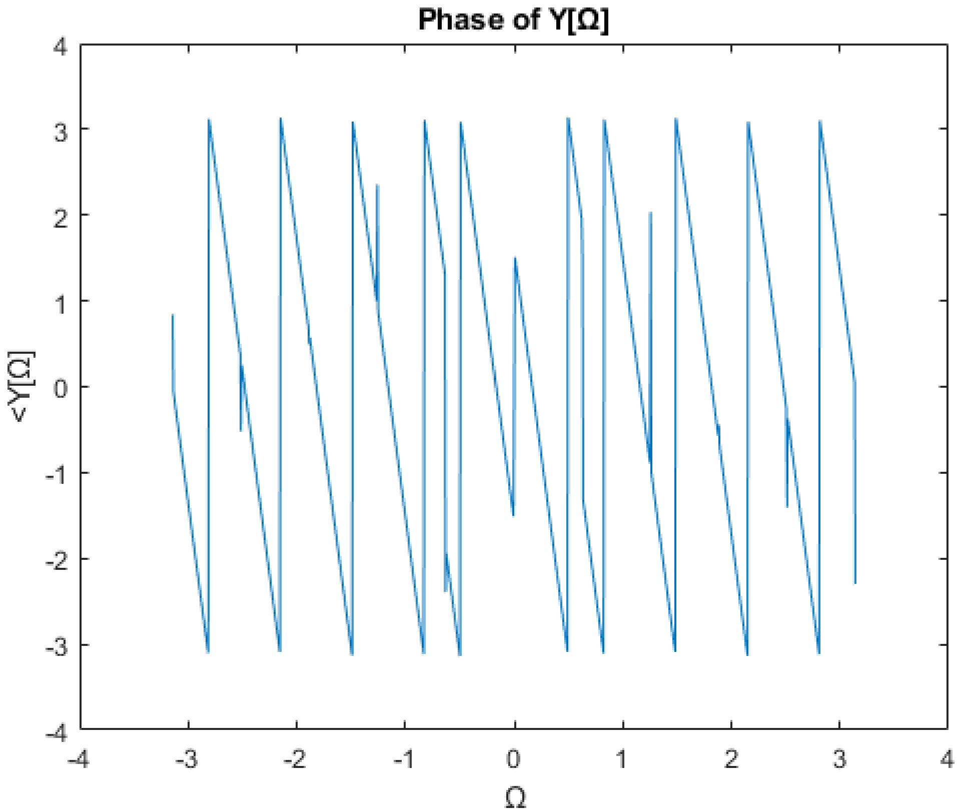
```
title('plot of y[n] using conv command')
xlabel('n')
ylabel('y[n]')

% Part 5 - DTFT Plot of y[n] from Part 4

omega= linspace(-pi,pi,1001);
W_omega = exp(-1i).^((0:length(y)-1)'*omega);
Y = (y*W_omega);

figure;
plot(omega, abs(Y));
title('Magnitude of Y[Ω] from part 4')
xlabel('Ω')
ylabel('|Y[Ω]|')

figure;
plot(omega, angle(Y));
title('Phase of Y[Ω]')
xlabel('Ω')
ylabel('<Y[Ω]')
```



plot of y[n] using conv command
4
3
2
1
0
-1
-2
-3
-4
y[n]
0
5
10
15
20
25
n



Magnitude of Y[Ω] from part 4
30
25
20
15
10
5
0
|Y[Ω]|
-4
-3
-2
-1
0
1
2
3
4
Ω



Part 6 – Question Response

Yes, the same results were achieved in part 3 and 5, this is due to a property of the Fourier transform; The convolution of two functions in the time domain is equivalent to the product in the frequency domain.

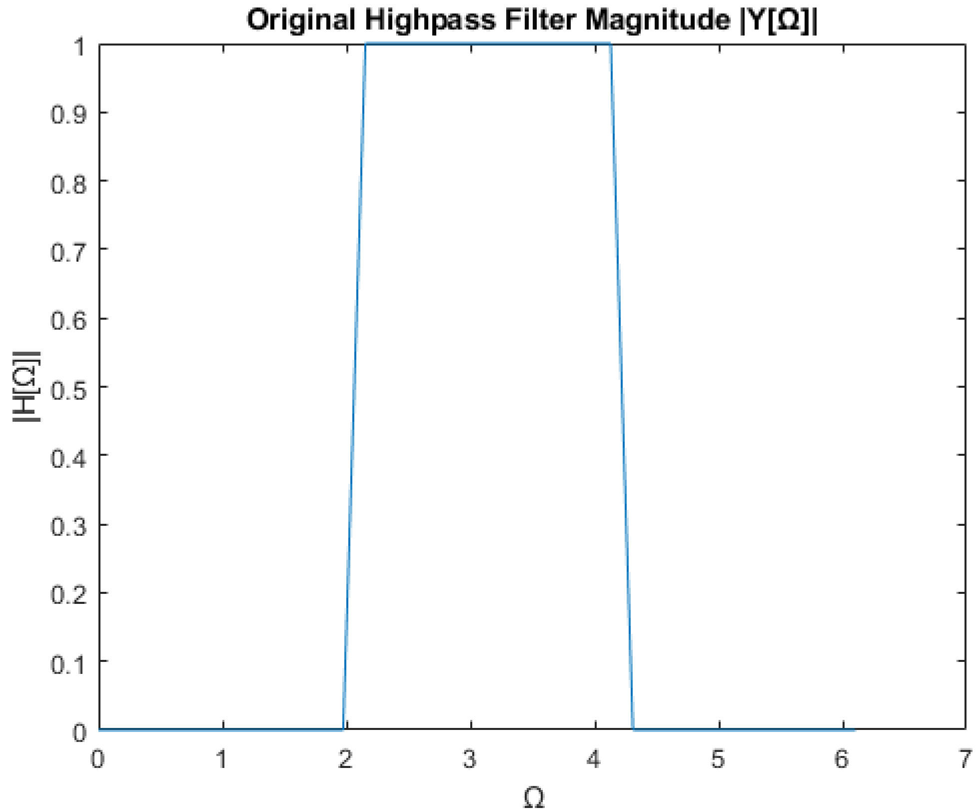
C) FIR Filter Design by Frequency Sampling

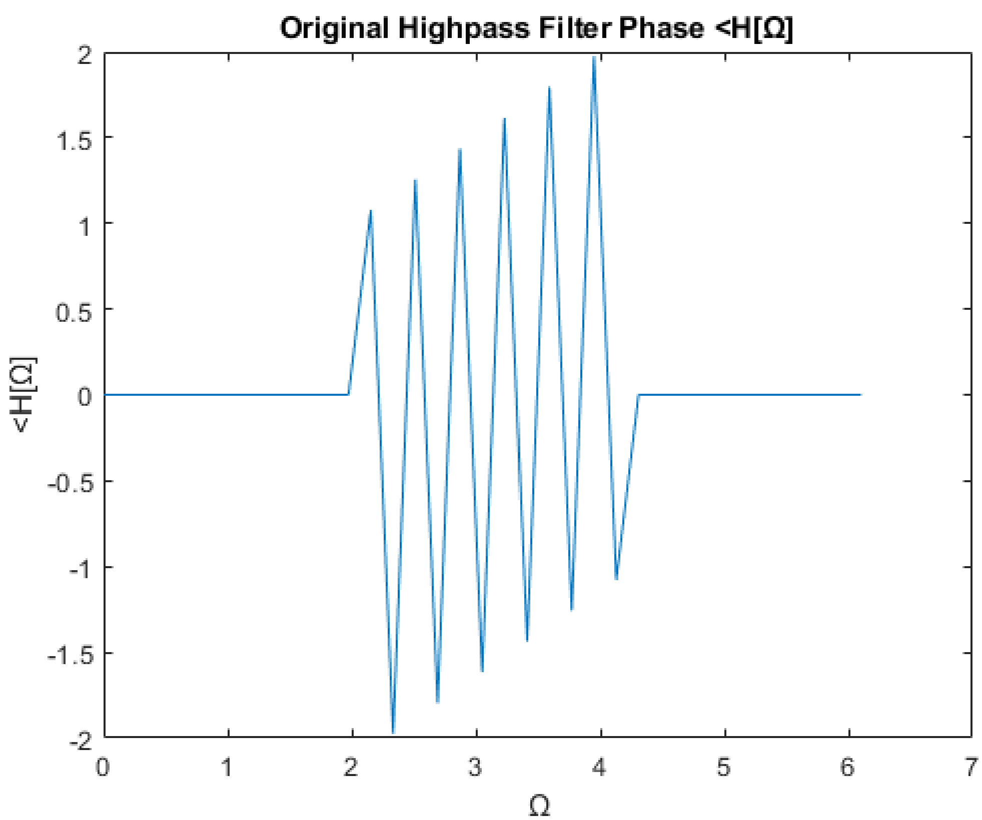
Part 1 - High Pass FIR Filter

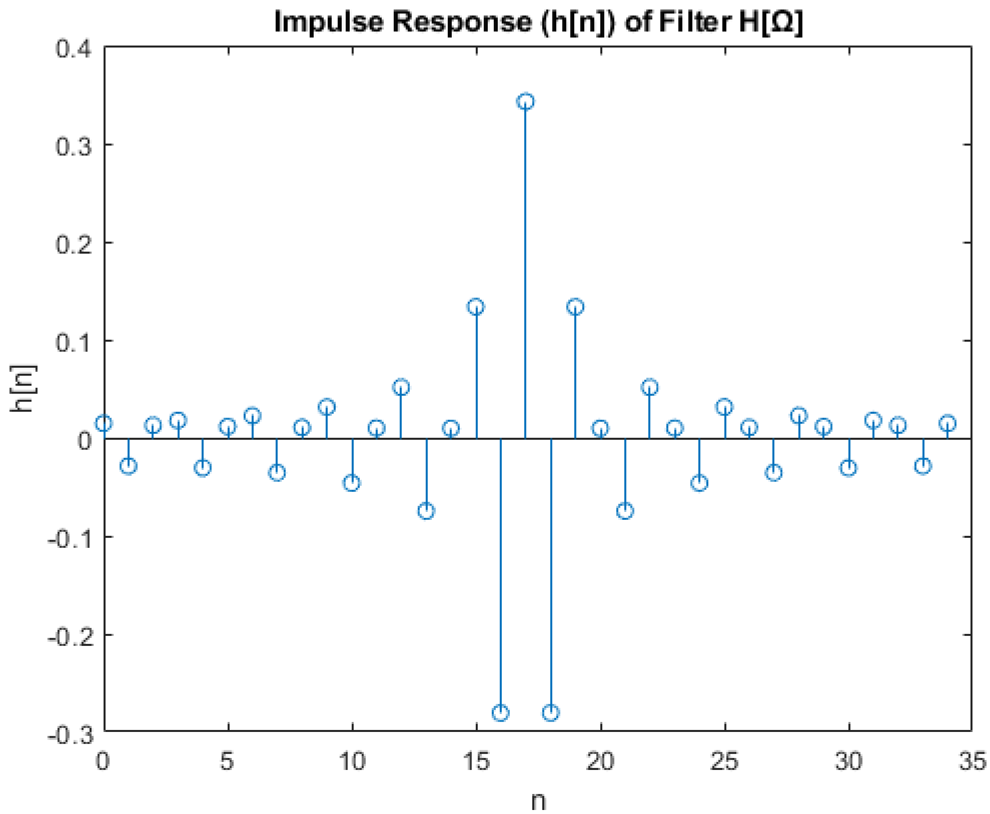
```
ohm0 = 2*pi/3;
N = 35;
n = 0:N-1;
Omega = linspace(0,2*pi*(1-1/N),N);
H_d = @(Omega) (mod(Omega,2*pi)>ohm0).*(mod(Omega,2*pi)<2*pi-ohm0);

H = H_d(Omega).*exp(-1i.*Omega.*((N-1)/2));
h = ifft(H);
```

```
figure;
plot(Omega, abs(H));
title('Original Highpass Filter Magnitude |Y[Ω]|')
xlabel('Ω')
ylabel('|H[Ω]|')

figure;
plot(Omega, angle(H));
title('Original Highpass Filter Phase <H[Ω]')
xlabel('Ω')
ylabel('<H[Ω]')

figure;
stem(n, h);
title('Impulse Response (h[n]) of Filter H[Ω]')
xlabel('n')
ylabel('h[n]')
```

Warning: Using only the real component of complex data.

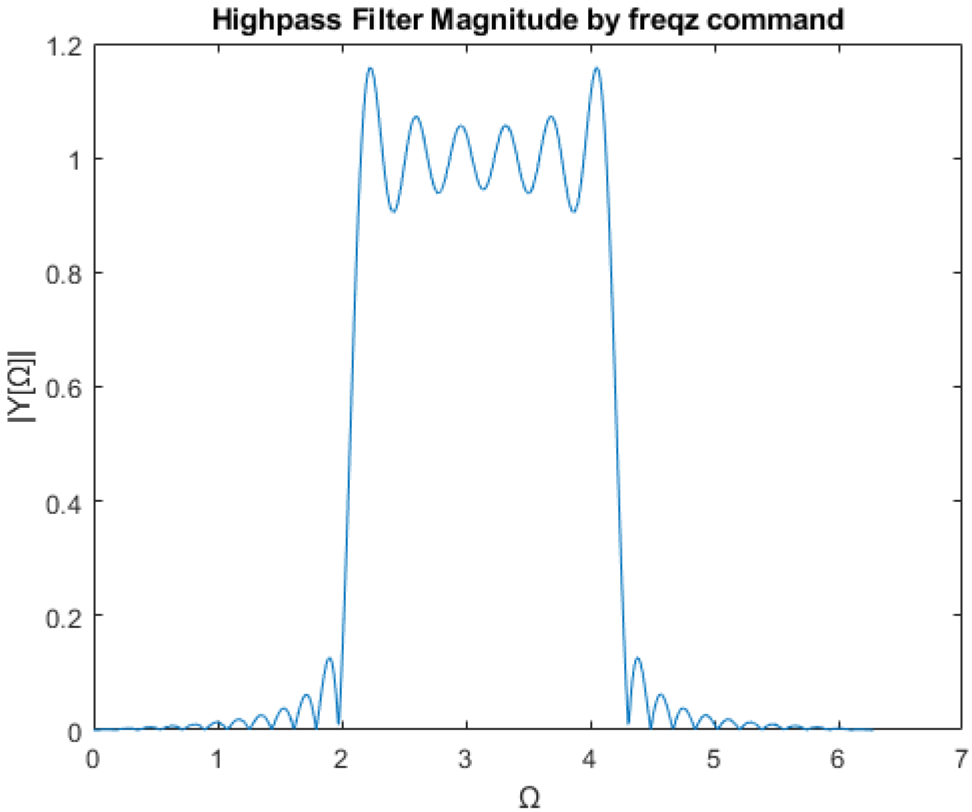

Original Highpass Filter Magnitude |Y[Ω]|
|H[Ω]|
Ω
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0 1 2 3 4 5 6 7


Original Highpass Filter Phase <H[Ω]
<H[Ω]
Ω
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
0 1 2 3 4 5 6 7



Part 2 - Frequency Response from h[n] by freqz Command

```
H = freqz(h,1, 0:2*pi/1001:2*pi);
figure;
plot(0:2*pi/1001:2*pi, abs(H));
title('Highpass Filter Magnitude by freqz command')
xlabel('Ω')
ylabel('|Y[Ω]|')
```



Part 3 – Question Response

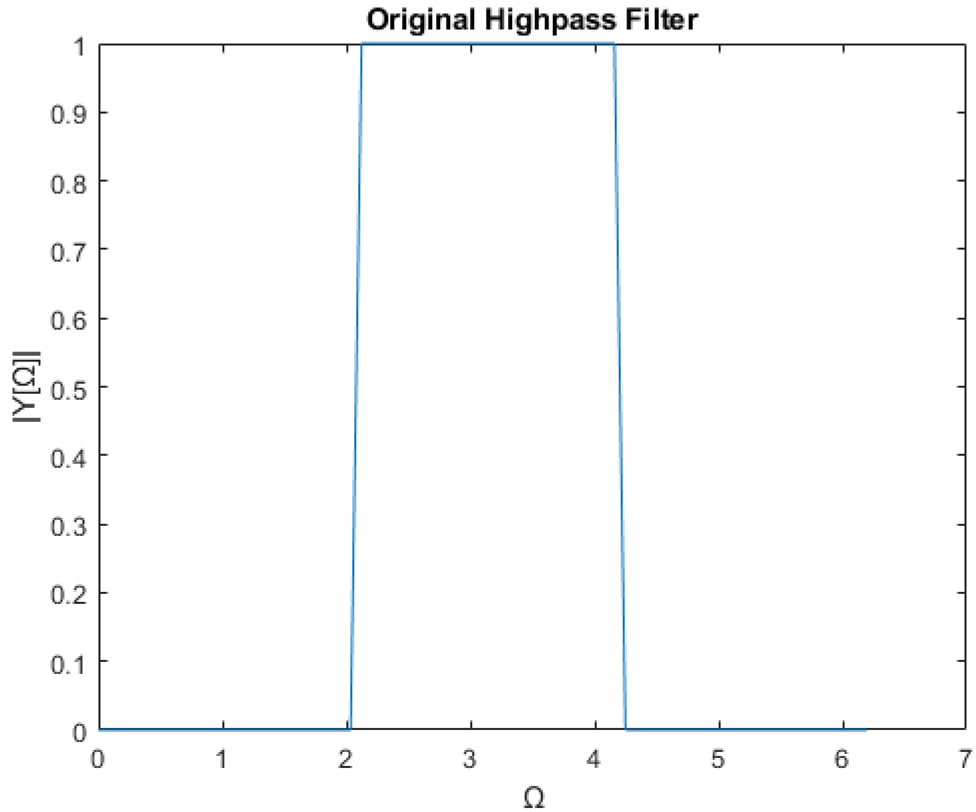
The result in part 2 is different from the ideal filter we started with in terms of the frequencies it permits. The ideal filter has a strict cut-off frequency, whereas the result in part 2 has ripples beyond the desired cut-off frequency. This is due to the impulse response of an ideal high pass filter being noncausal and therefore physically unrealizable.

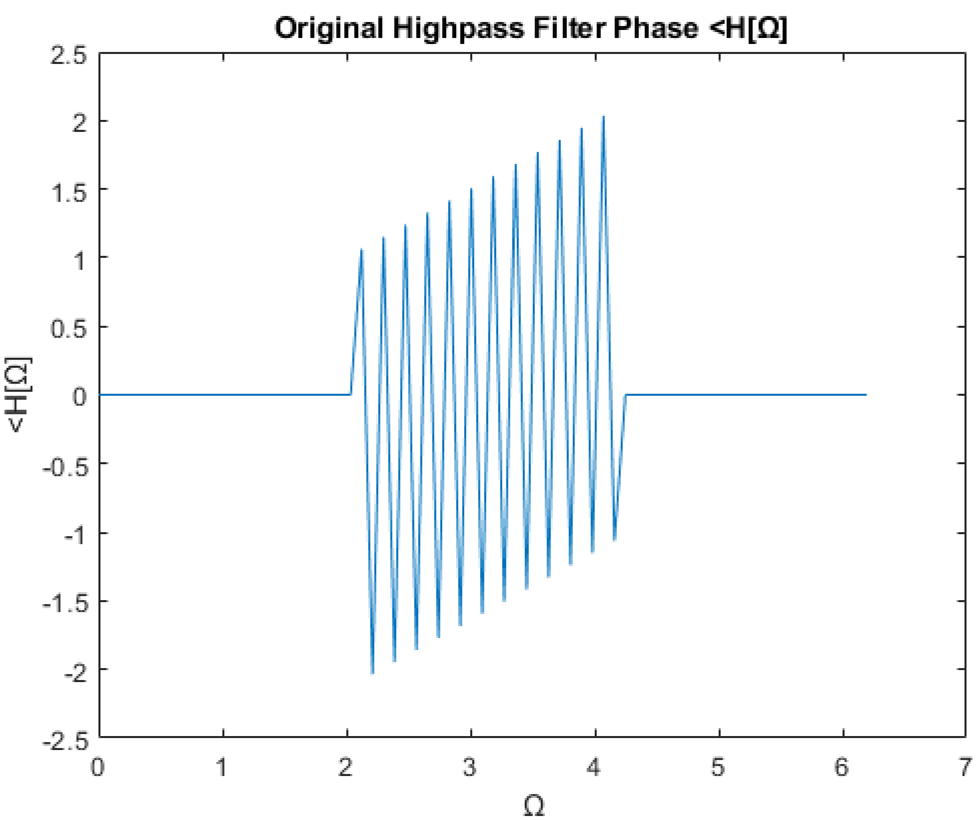
Part 4 – Increase to 71 points

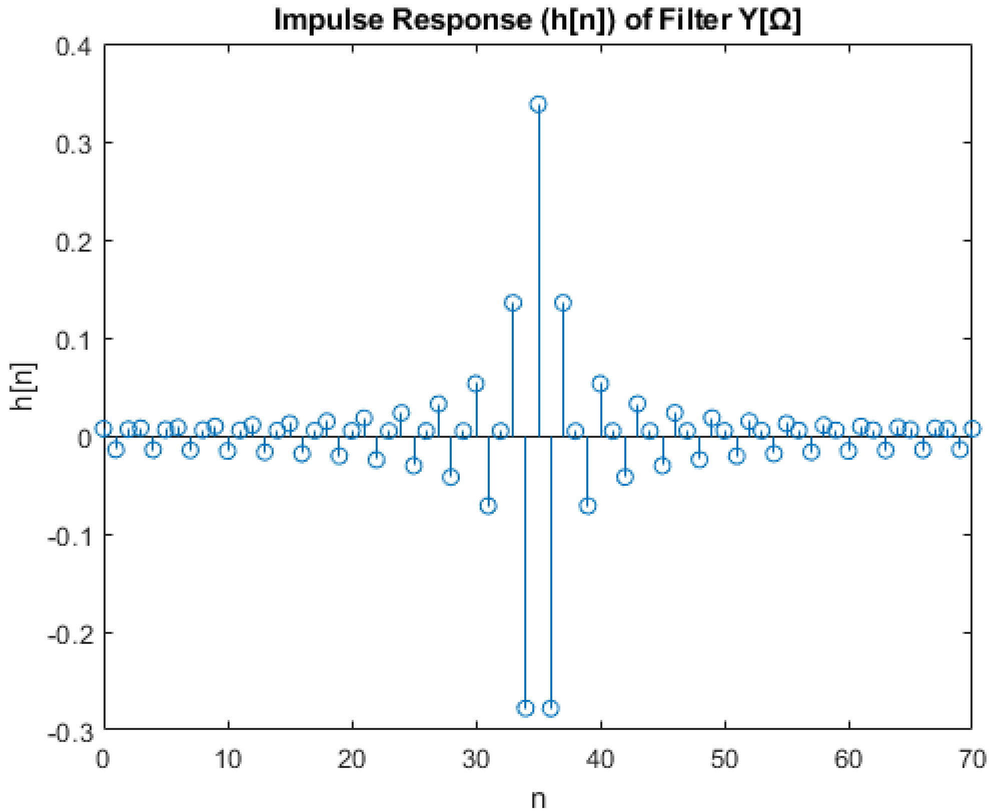
```
clc
clear
ohm0 = 2*pi/3;
N = 71;
n = 0:N-1;
Omega = linspace(0,2*pi*(1-1/N),N);
H_d = @(Omega) (mod(Omega,2*pi)>ohm0).*(mod(Omega,2*pi)<2*pi-ohm0);

H = H_d(Omega).*exp(-1i.*Omega.*((N-1)/2));
```

```
h = ifft(H);

figure;
plot(Omega, abs(H));
title('Original Highpass Filter')
xlabel('Ω')
ylabel('|Y[Ω]|')

figure;
plot(Omega, angle(H));
title('Original Highpass Filter Phase <H[Ω]')
xlabel('Ω')
ylabel('<H[Ω]')

figure;
stem(n, h);
title('Impulse Response (h[n]) of Filter Y[Ω]')
xlabel('n')
ylabel('h[n]')
```

Warning: Using only the real component of complex data.


Original Highpass Filter
|Y[Ω]|
Ω
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
4
5
6
7


Original Highpass Filter Phase <H[Ω]
<H[Ω]
Ω
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
0
1
2
3
4
5
6
7



Part 5 – Question Response

Increasing N increases the 'resolution' since we get a more accurate representation of the impulse response due to the existence of more points and a more unique signal representation.