ELE632 Lab 4 Report

Table of Contents

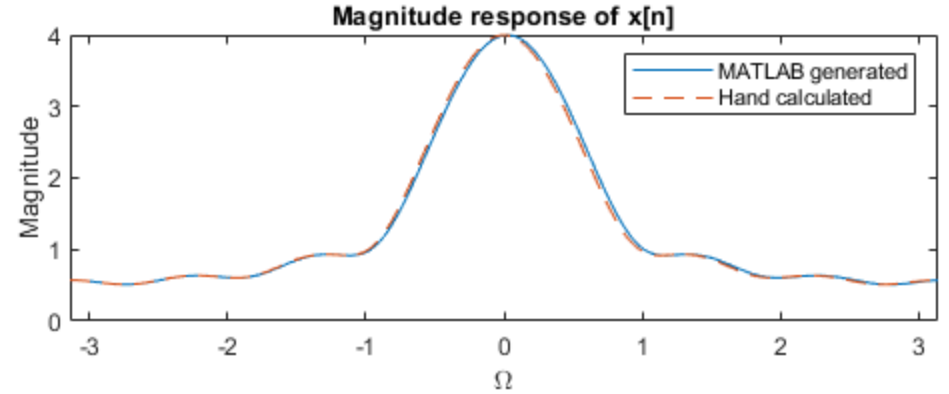
Authors: Mai Abdelhameed & Omar Ahmad

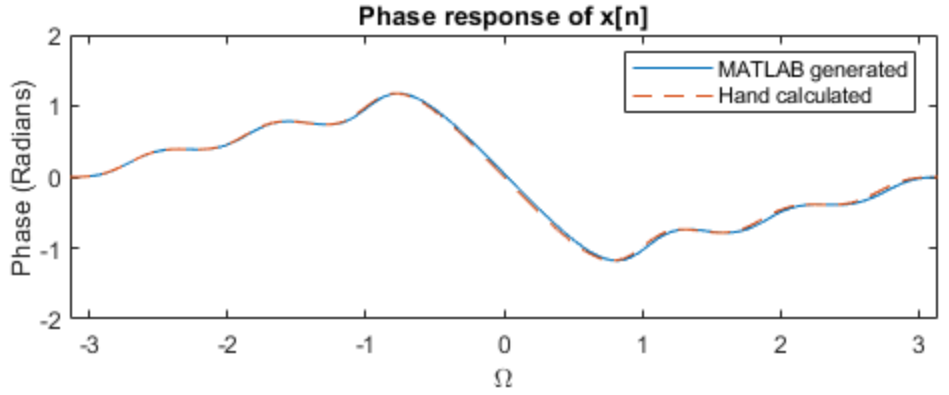
Part A - Discrete-Time Fourier Transform (DTFT)

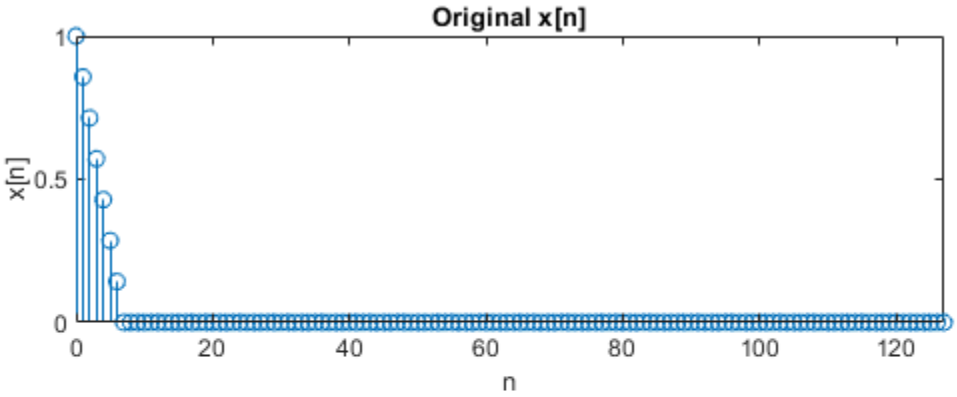
```
%A.1
x = zeros(1,128);
n = (0:127);
x(1:7) = 1-1/7.*n(1:7);

X=fft(x);
X=fftshift(X);
Omega = linspace(-pi,pi,128);

%A.2 Hand-calculated DTFT
w = @(a) (1 ...
    +   (6/7).*exp(-i.*a)...
    +   (5/7).*exp(-2.*i.*a)...
    +   (4/7).*exp(-3.*i.*a)...
    +   (3/7).*exp(-4.*i.*a)...
    +   (2/7).*exp(-5.*i.*a)...
    +   (1/7).*exp(-6.*i.*a));

figure;
subplot(2, 1, 1);
plot(Omega,abs(X));
hold on;
plot(Omega,abs(w(Omega)),'--'); %slightly off due to rounding errors
legend('MATLAB generated','Hand calculated');
hold off;
axis([-pi pi 0 4]);
title('Magnitude response of x[n]');
xlabel('\Omega');
ylabel('Magnitude');

subplot(2, 1, 2);
plot(Omega,angle(X));
hold on;
plot(Omega,angle(w(Omega)),'--');
legend('MATLAB generated','Hand calculated');
hold off;
```

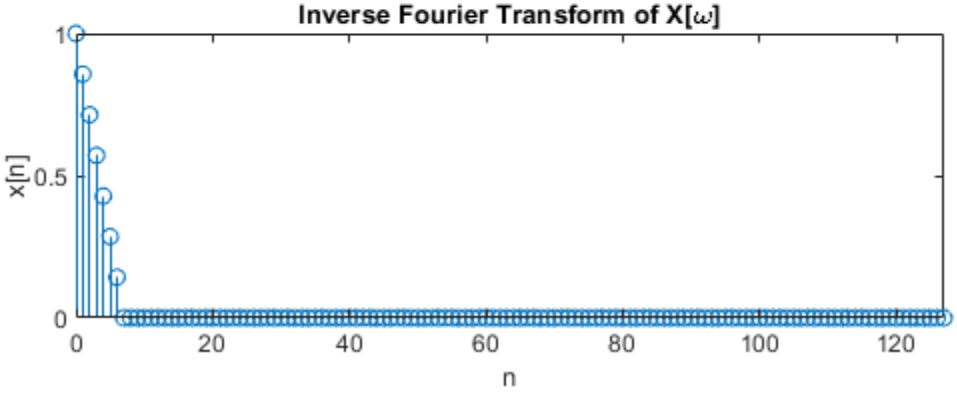
```
axis([-pi pi -2 2]);
title('Phase response of x[n]');
xlabel('\Omega');
ylabel('Phase (Radians)');

disp('The MATLAB generated and the hand calculated DTFT are both the
 same');
disp('with the slight differences attributed to small rounding
 errors.');

%A.3
X=ifftshift(X);
z=ifft(X);

figure;
subplot(2, 1, 1);
stem(n, x);
title('Original x[n]');
xlabel('n');
ylabel('x[n]');
axis([0 127 0 1]);

subplot(2, 1, 2);
stem(n,z);
title('Inverse Fourier Transform of X[\omega]');
xlabel('n');
ylabel('x[n]');
axis([0 127 0 1]);

disp('Yes, the inverse fourier transform resulted in the original
 x[n].');
```

```
The MATLAB generated and the hand calculated DTFT are both the same
with the slight differences attributed to small rounding errors.
Yes, the inverse fourier transform resulted in the original x[n].
```


Magnitude response of x[n]
MATLAB generated
Hand calculated
Magnitude
Ω


Phase response of x[n]
MATLAB generated
Hand calculated
Phase (Radians)
Ω

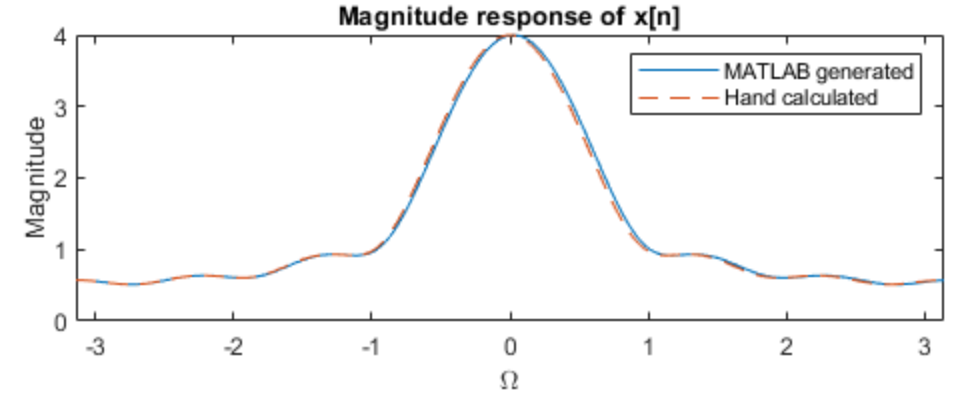

Original x[n]
x[n]
n

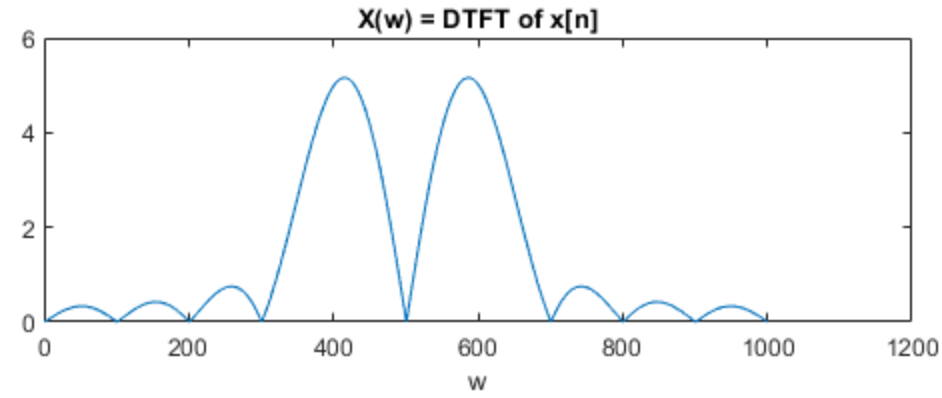

Inverse Fourier Transform of X[ω]
x[n]
n

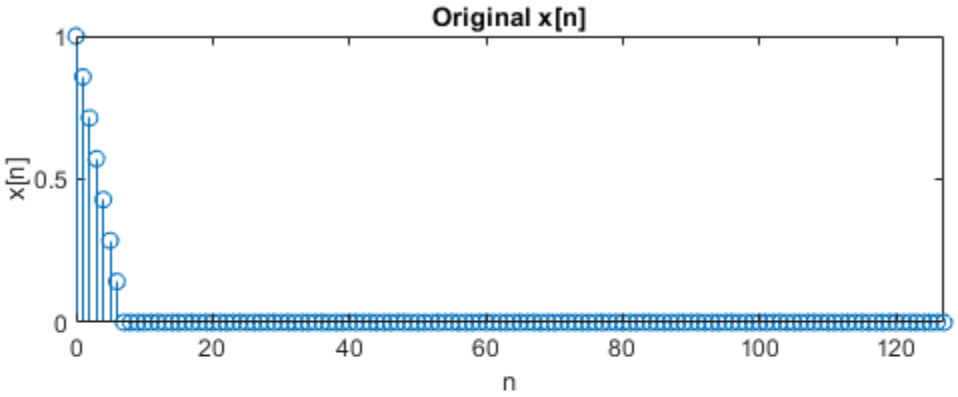
Part B - Time Convolution

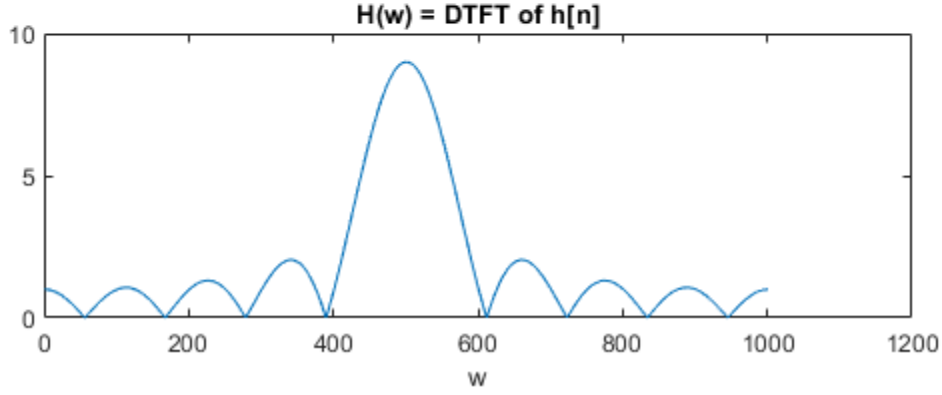
```
clear;
%B.1
n = 0:1000;

u = @(n) (n >= 0) * 1.0 .* (mod(n, 1) == 0);
x = @(n) sin(0.2 * pi * n).*(u(n) - u(n-10));
omegax = linspace(-pi, pi, 1001);
W_omegax = exp(-j).^((0:length(x(n))-1)'*omegax);
X = (x(n)*W_omegax);
figure(1);
plot(abs(X));
title('X(w) = DTFT of x[n]');
xlabel('w');

%B.2
h = @(n) u(n) - u(n-9);
omegah = linspace(-pi, pi, 1001);
W_omegah = exp(-j).^((0:length(h(n))-1)'*omegah);
H = (h(n)*W_omegah);
figure(2);
plot(abs(H));
title('H(w) = DTFT of h[n]');
xlabel('w');

%B.3
Y_1 = X.*H;
figure(3);
plot(abs(Y_1));
hold on;

%B.4
y = conv(x(n),h(n));

%B.5
omegay = linspace(-pi, pi, 1001);
W_omegay = exp(-j).^((0:length(y)-1)'*omegay);
Y_2 = (y*W_omegay);
plot(abs(Y_2), '--');
hold off;
title('Comparision of Y(w) obtained through X(w)H(w) vs DTFT of
 y[n]');
legend('Y(w) = X(w)H(w)','Y(w) = DTFT of y[n]');
xlabel('w');

disp('B.6: We obtained the same graphs using both methods asked in
 part B.3 and part B.5 because they ');
```

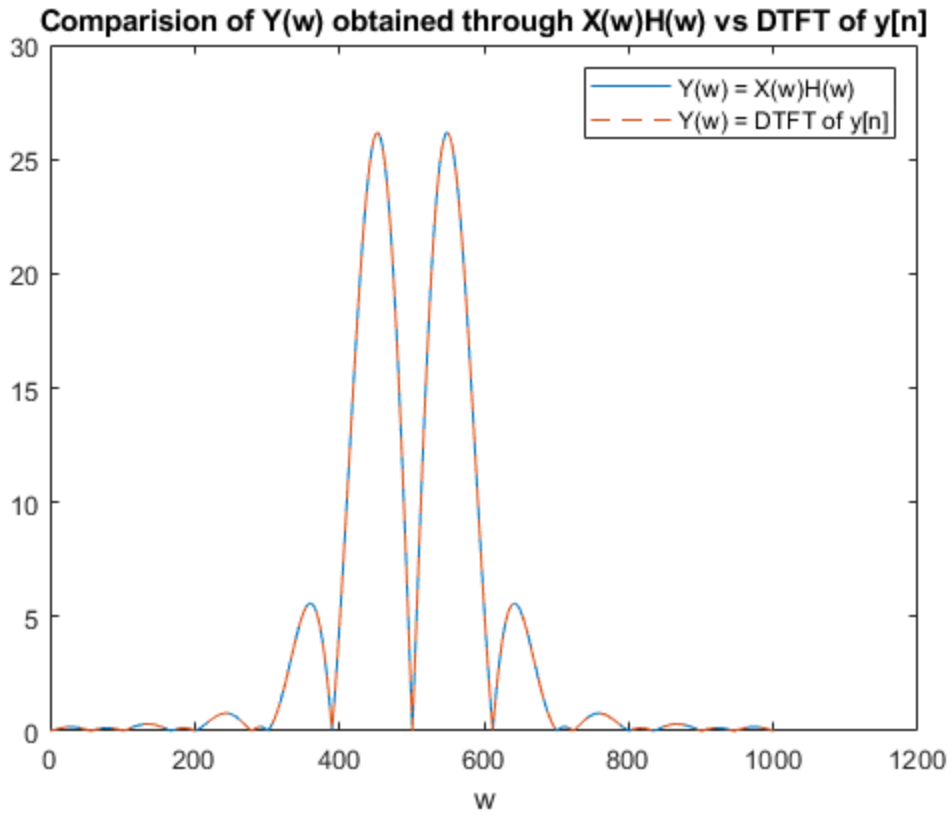
```
B.6: We obtained the same graphs using both methods asked in part B.3
 and part B.5 because they
```


Magnitude response of x[n]
MATLAB generated
Hand calculated
Magnitude
Ω


X(w) = DTFT of x[n]
w


Original x[n]
x[n]
n


H(w) = DTFT of h[n]
w

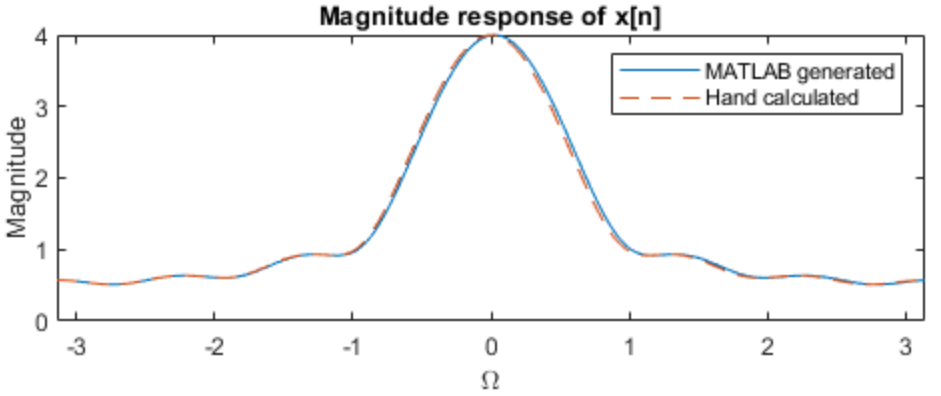


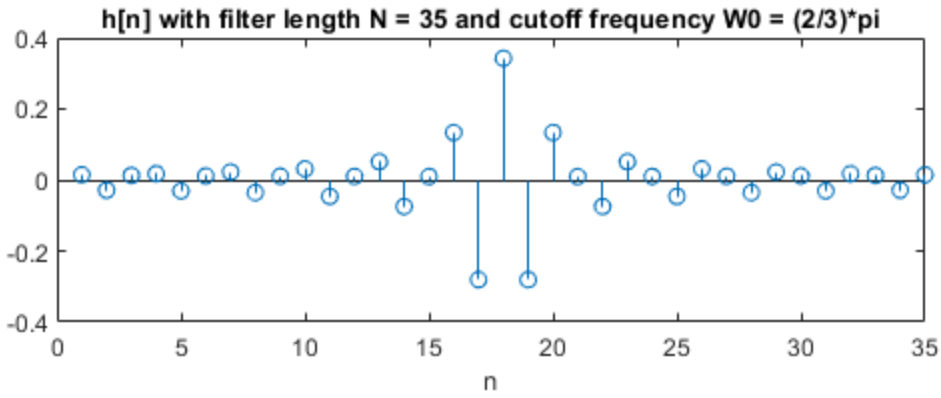
Part C - FIR Filter Design by Frequency Sampling

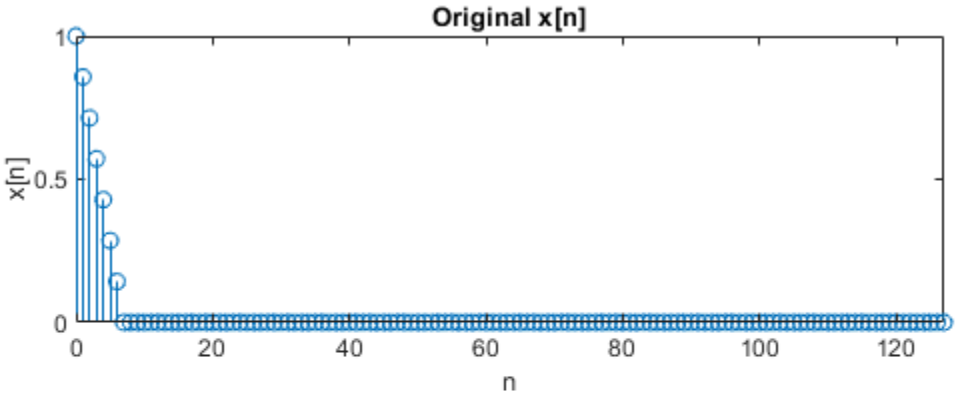
```
clear;
%C.1, h[n], N = 35
N = 35;
Omega = linspace(0,2*pi*(1-1/N),N);
H_d_35 = @(Omega) (mod(Omega,2*pi)>(2/3)*pi).*(mod(Omega,2*pi)<2*pi-
(2/3)*pi);
H_35 = H_d_35(Omega);
H_35 = H_35.*exp(-j*Omega*((N-1)/2));
h_35 = real(ifft(H_35));
figure(1);
stem(h_35);
title('h[n] with filter length N = 35 and cutoff frequency W0 =
 (2/3)*pi');
xlabel('n');

%part of C.2, H(w), N = 35
H_35 = freqz(h_35,1,0:2*pi/1001:2*pi);

%C.3
```

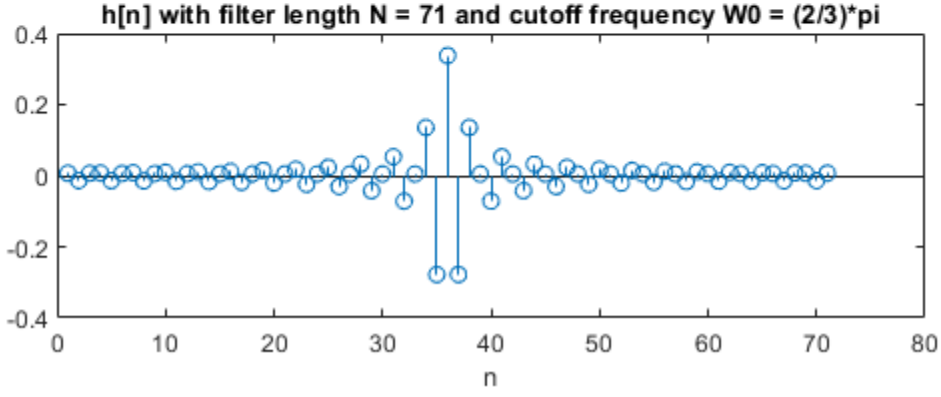
```
disp('C.3: The ideal filter is basically perfect; there is no noise
 around the edges (bandwidth length), while the regular filter would
 has noise around the edges and would affect the input signal.');

%C.4, h[n], N=71
N = 71;
Omega = linspace(0,2*pi*(1-1/N),N);
H_d_71 = @(Omega) (mod(Omega,2*pi)>(2/3)*pi).*(mod(Omega,2*pi)<2*pi-
(2/3)*pi);
H_71 = H_d_71(Omega);
H_71 = H_71.*exp(-j*Omega*((N-1)/2));
h_71 = real(ifft(H_71));
figure(2);
stem(h_71);
title('h[n] with filter length N = 71 and cutoff frequency W0 =
 (2/3)*pi');
xlabel('n');

%C.2 and C.4, plots of H(w) with N = 35, 71
H_71 = freqz(h_71,1,0:2*pi/1001:2*pi);
figure(3);
plot(abs(H_35));
hold on;
plot(abs(H_71));
hold off;
title('Comparision of H(w) with different filter lengths N');
xlabel('w');
legend('N = 35', 'N = 71');

disp('C.5: The filter of N = 71 is more accurate as it has more
 samples than that of N = 35.');
```

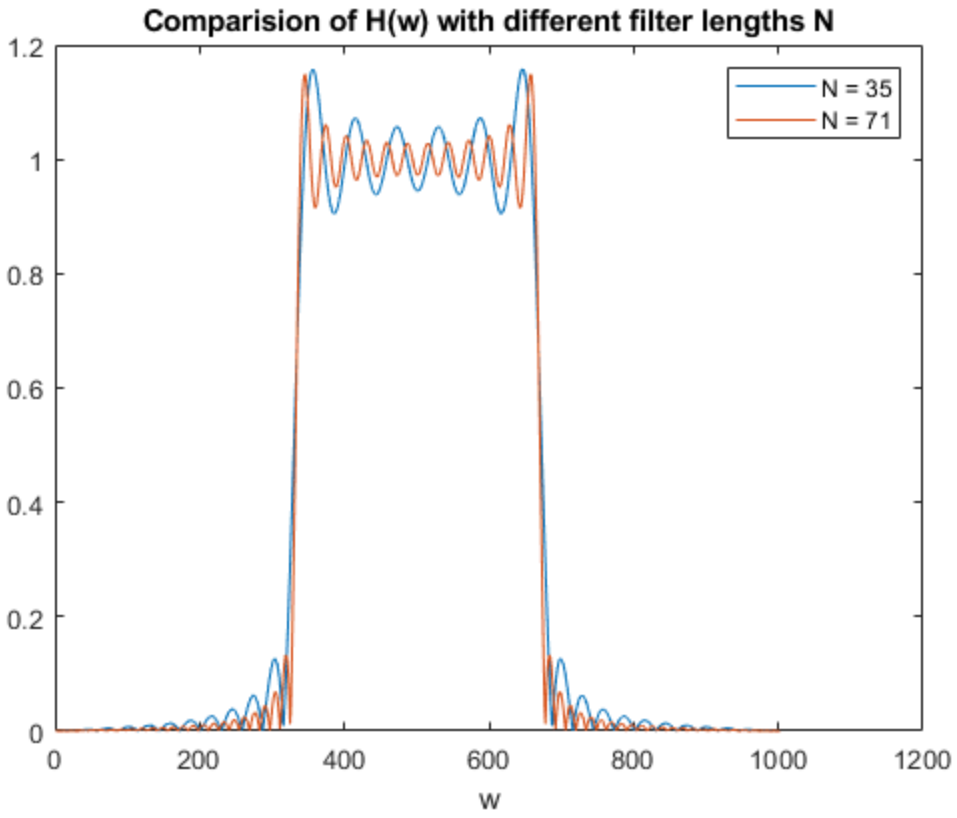
```
C.3: The ideal filter is basically perfect; there is no noise around
 the edges (bandwidth length), while the regular filter would has
 noise around the edges and would affect the input signal.
C.5: The filter of N = 71 is more accurate as it has more samples than
 that of N = 35.
```


Magnitude response of x[n]
MATLAB generated
Hand calculated
Magnitude
Ω


h[n] with filter length N = 35 and cutoff frequency W0 = (2/3)*pi
n


Original x[n]
x[n]
n


h[n] with filter length N = 71 and cutoff frequency W0 = (2/3)*pi
n



Published with MATLAB® R2018b