Hanfei Geng

hgeng4

Oct 5th, 2016

Report Item 1:

code:

N = 20

n = -N:N

wc = pi/5

w0 = pi/2

lowpass = wc/pi \* sinc(wc/pi \* n)

highpass = [n==0] - lowpass

bandpass = cos(w0\*n) .\* lowpass

LOWPASS = fft(lowpass)

LOWPASS = fftshift(LOWPASS)

HIGHPASS = fft(highpass)

HIGHPASS = fftshift(HIGHPASS)

BANDPASS = fft(bandpass)

BANDPASS = fftshift(BANDPASS)

size = 2\*N+1

w = fftshift((0:size - 1)/size\*2\*pi);

w(1:size/2) = w(1:size/2) - 2\*pi;

figure(1)

subplot(121)

stem(n,lowpass)

title('impulse response:ideal lowpass filter')

xlabel('n')

ylabel('h(n)')

subplot(122)

plot(w,abs(LOWPASS))

title('magnitutde response:ideal lowpass filter')

xlabel('w')

ylabel('H(w)')

figure(2)

subplot(121)

stem(n,highpass)

title('impulse response:ideal highpass filter')

xlabel('n')

ylabel('h(n)')

subplot(122)

plot(w,abs(HIGHPASS))

title('magnitude response:ideal highpass filter')

xlabel('w')

ylabel('H(w)')

figure(3)

subplot(121)

stem(n,bandpass)

title('impulse response:ideal bandpass filter')

xlabel('n')

ylabel('h(n)')

subplot(122)

plot(w,abs(BANDPASS))

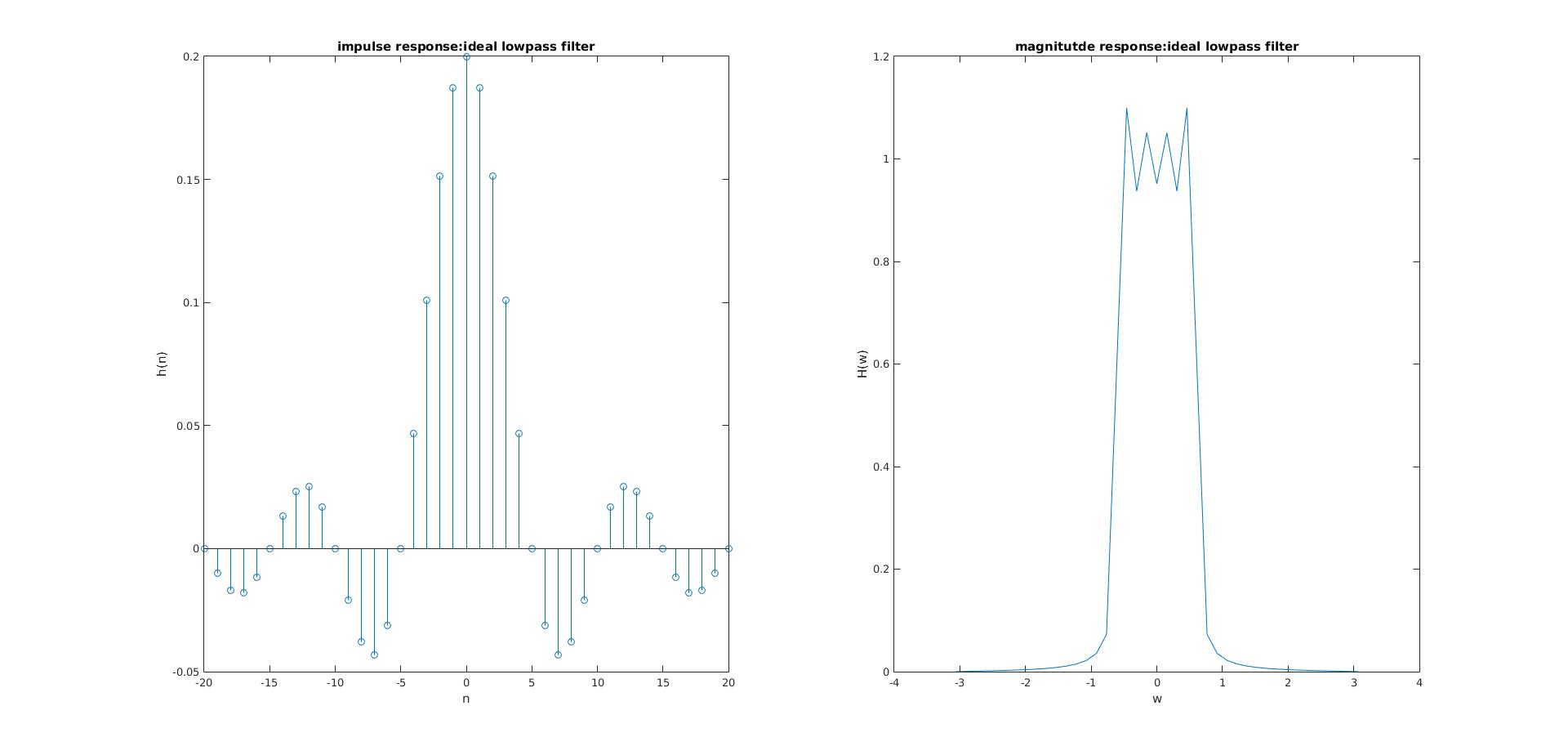
title('impulse response:ideal bandpass filter')

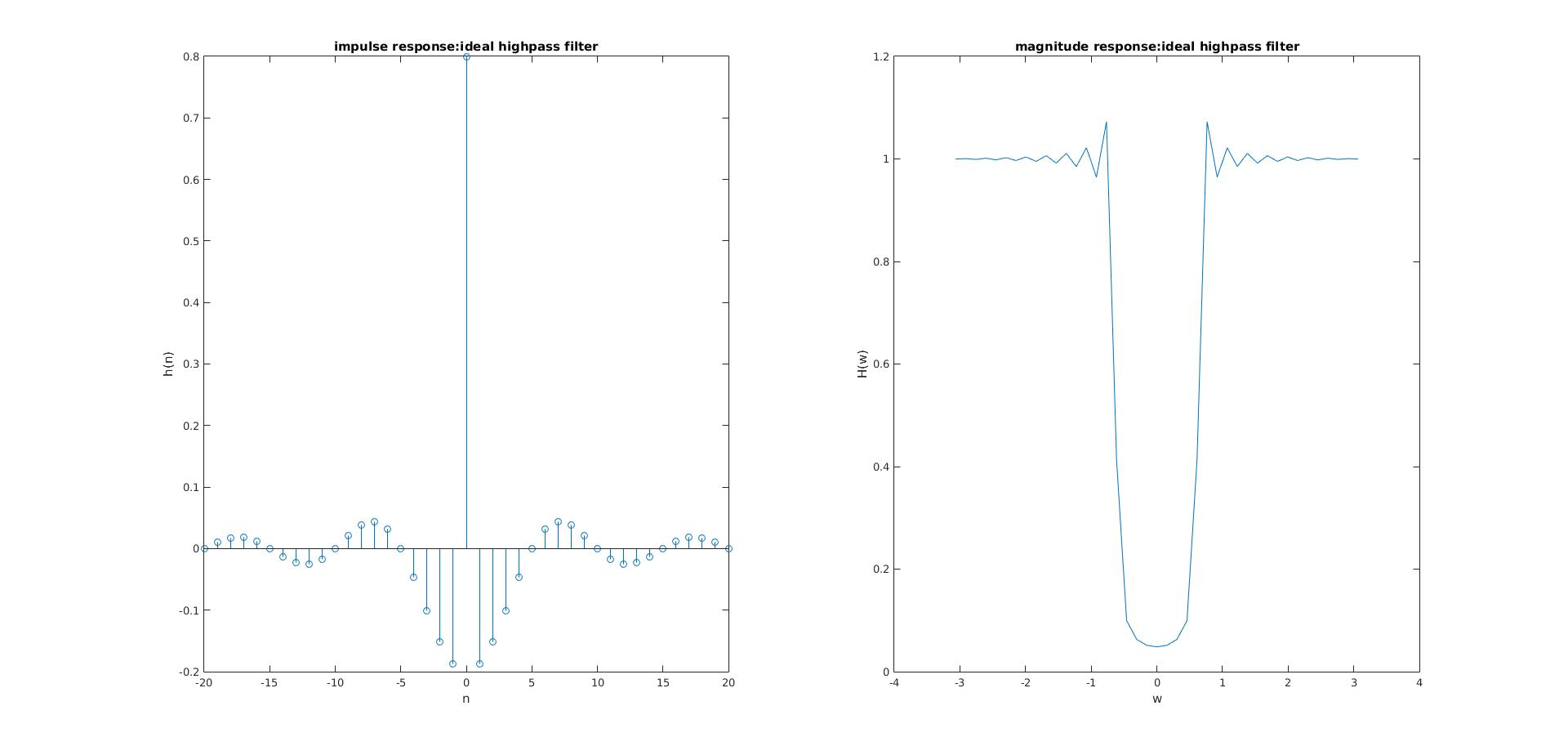
xlabel('w')

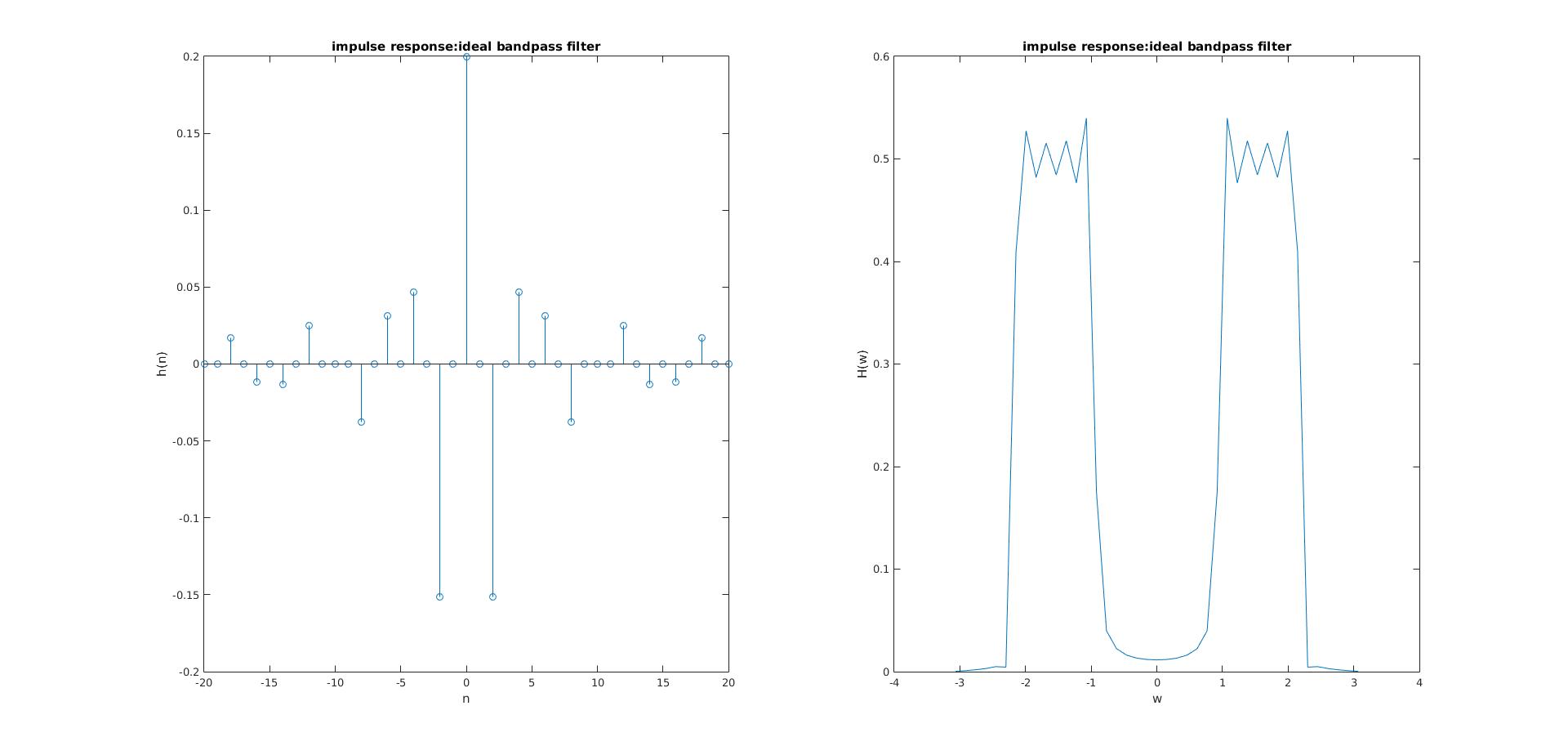
ylabel('H(w)')

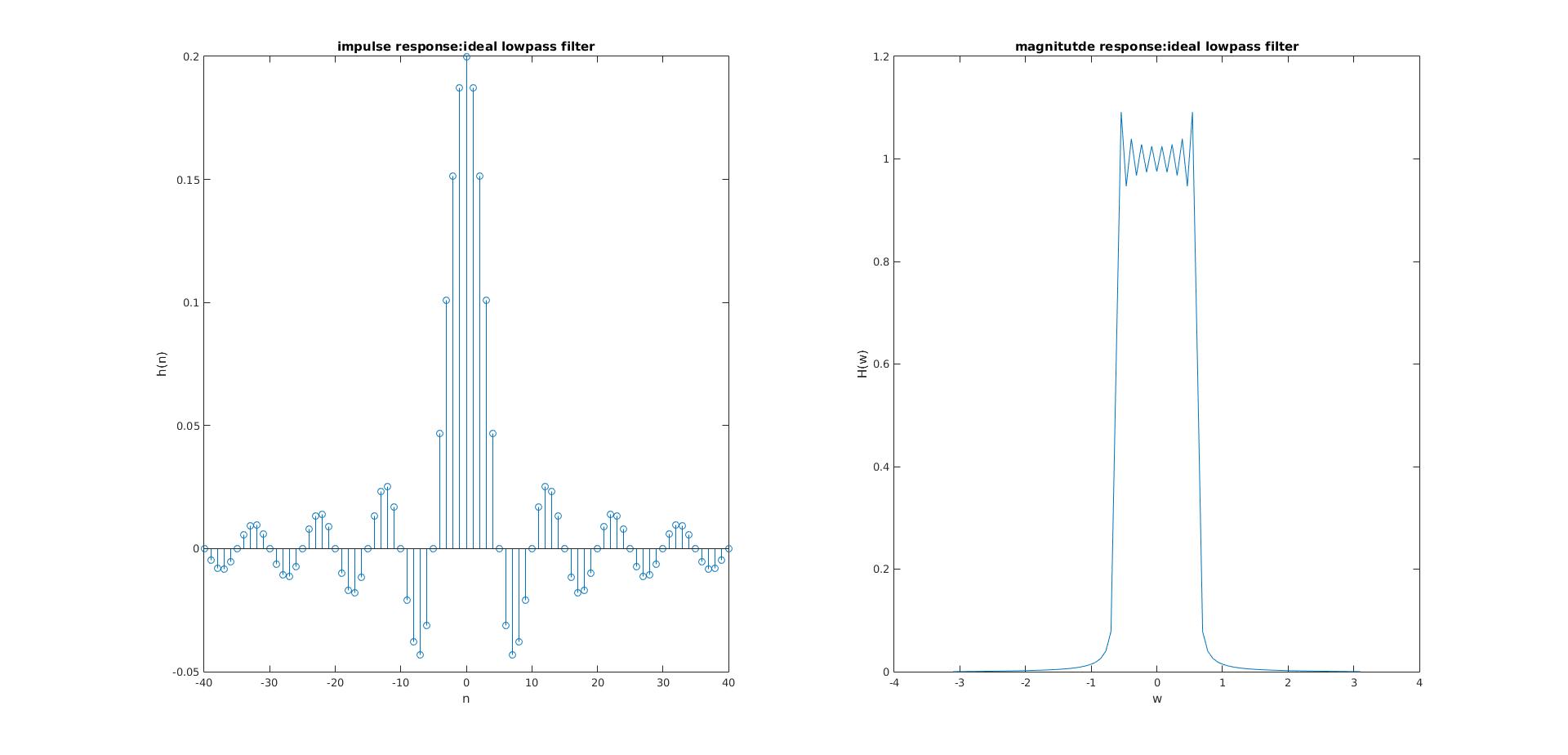
figure:

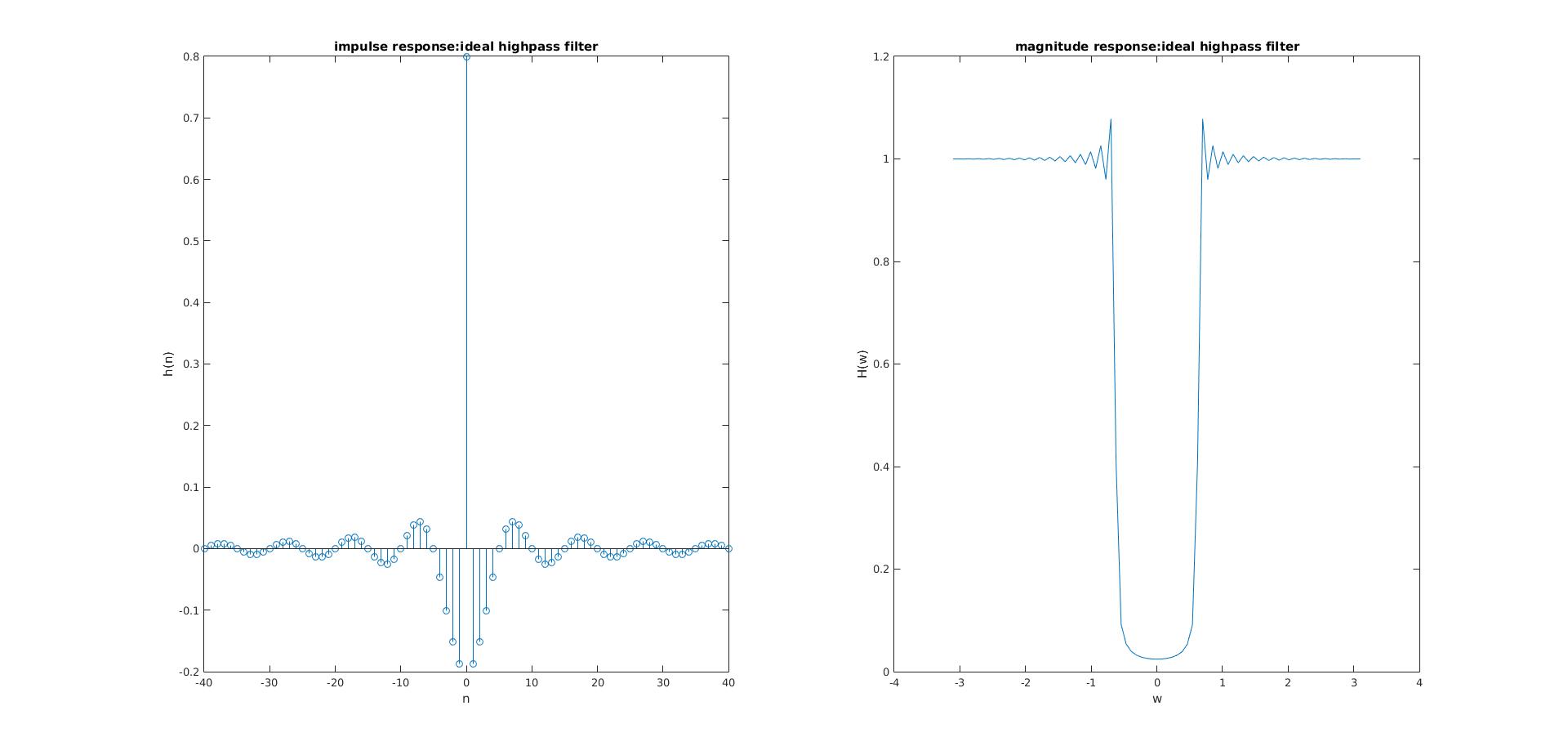
N = 20

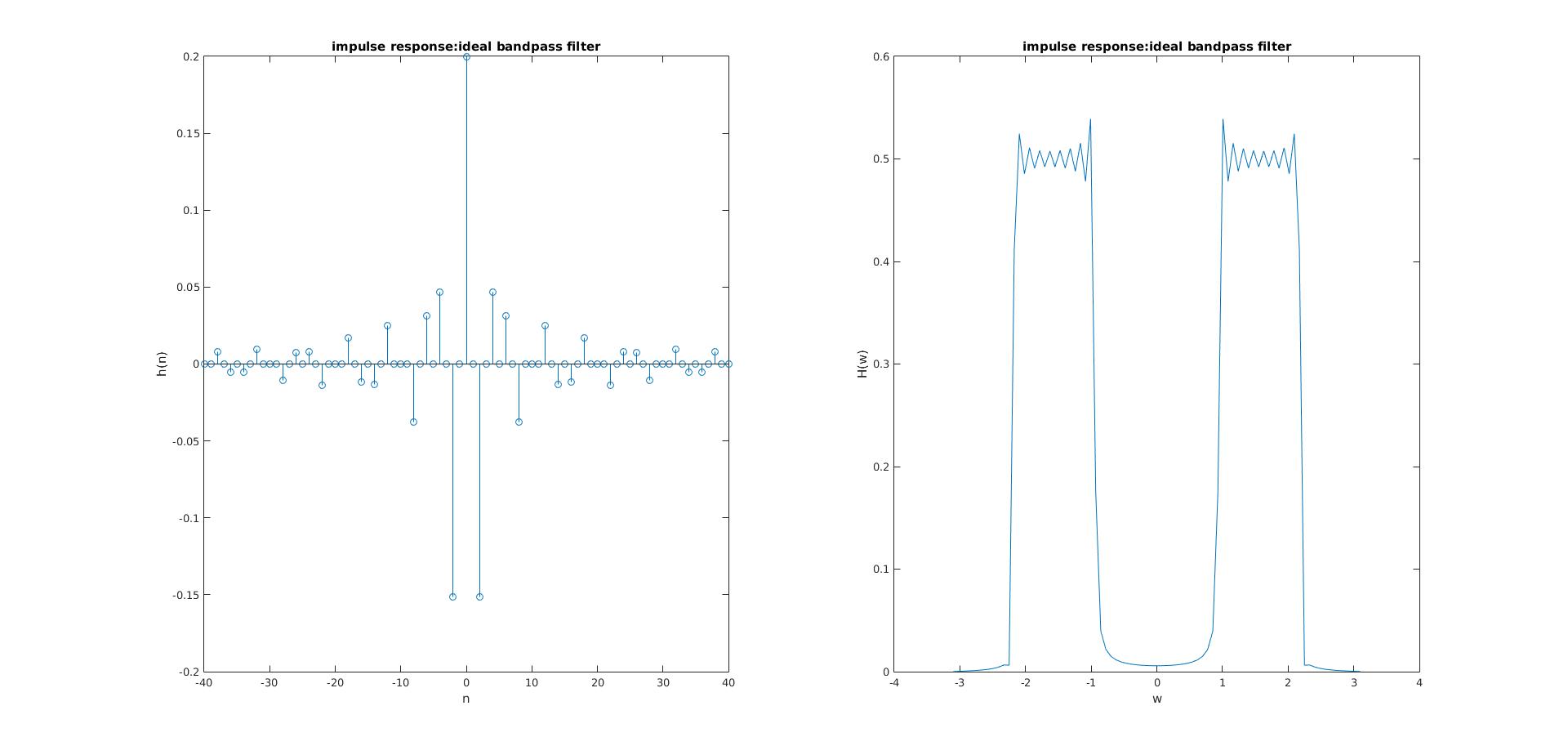




N = 40







The difference from the ideal magnitude response is that the magnitude responses in the figure have Gibbs phenomenon and there are also ripples in the pass-band. The main reason why this is happening is that the ideal filter is of infinite duration and not causal.

Report Item 2:

code:

load impulseresponse

N = length(h)

n = -(N-1)/2 : 1 : (N-1)/2;

n = n'

pad = 1024

H = fft(h,pad)

H = fftshift(H)

w = fftshift((0:pad - 1)/pad\*2\*pi);

w(1:pad/2) = w(1:pad/2) - 2\*pi;

db = mag2db(abs(H))

subplot(311)

stem(n,h)

title('impulse response')

xlabel('n')

ylabel('h(n)')

subplot(312)

plot(w,db)

title('magnitude response in dB')

xlabel('w')

ylabel('dB')

subplot(313)

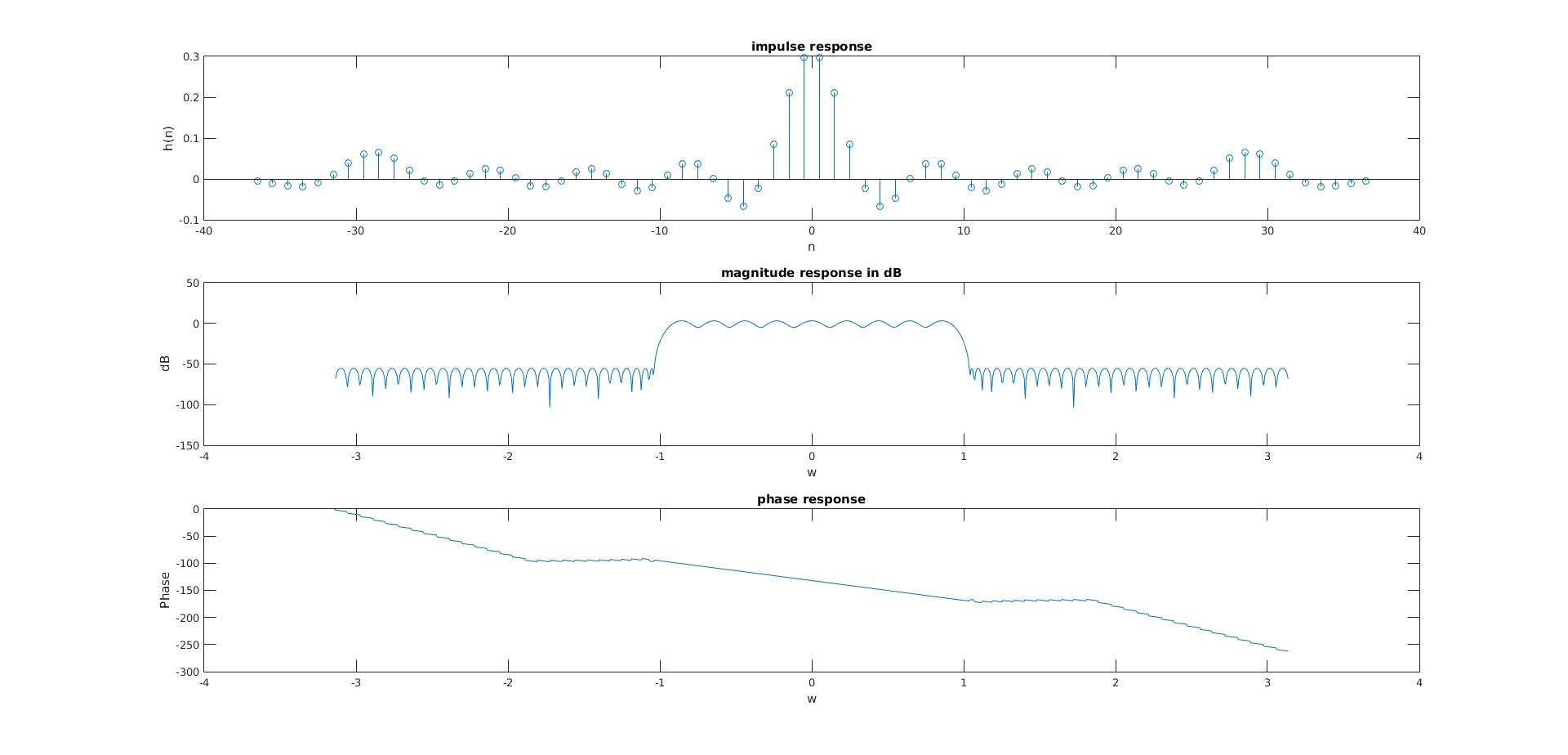
plot(w,phase(H))

title('phase response')

xlabel('w')

ylabel('Phase')

figure:

stop-band ripple:30dB

pass-band ripple:7dB

transition bandwidth:0.2

Report Item 3:

code:

N = 25;

cutoff = pi/3

M = (N-1)/2

w = fftshift((0:N-1)/N\*2\*pi);

w(1:N/2) = w(1:N/2)-2\*pi;

Dw = zeros(1,length(w))

Dw(9:17) = 1

Gw = Dw.\*exp(-i\*M\*w)

gn = real(ifft(ifftshift(Gw)))

wn = hamming(N)

hn = wn .\* real(gn')

figure(1)

stem([0:N-1],hn)

grid on

title('impulse response')

xlabel('n')

ylabel('h(n)')

size = 1024

H = fft(hn,size)

H = fftshift(H)

o = fftshift((0:size-1)/size\*2\*pi)

o(1:size/2) = o(1:size/2) - 2\*pi;

figure(2)

subplot(211)

plot(o,mag2db(abs(H)))

title('magnitude response')

xlabel('w')

ylabel('dB')

grid on

subplot(212)

plot(o,phase(H))

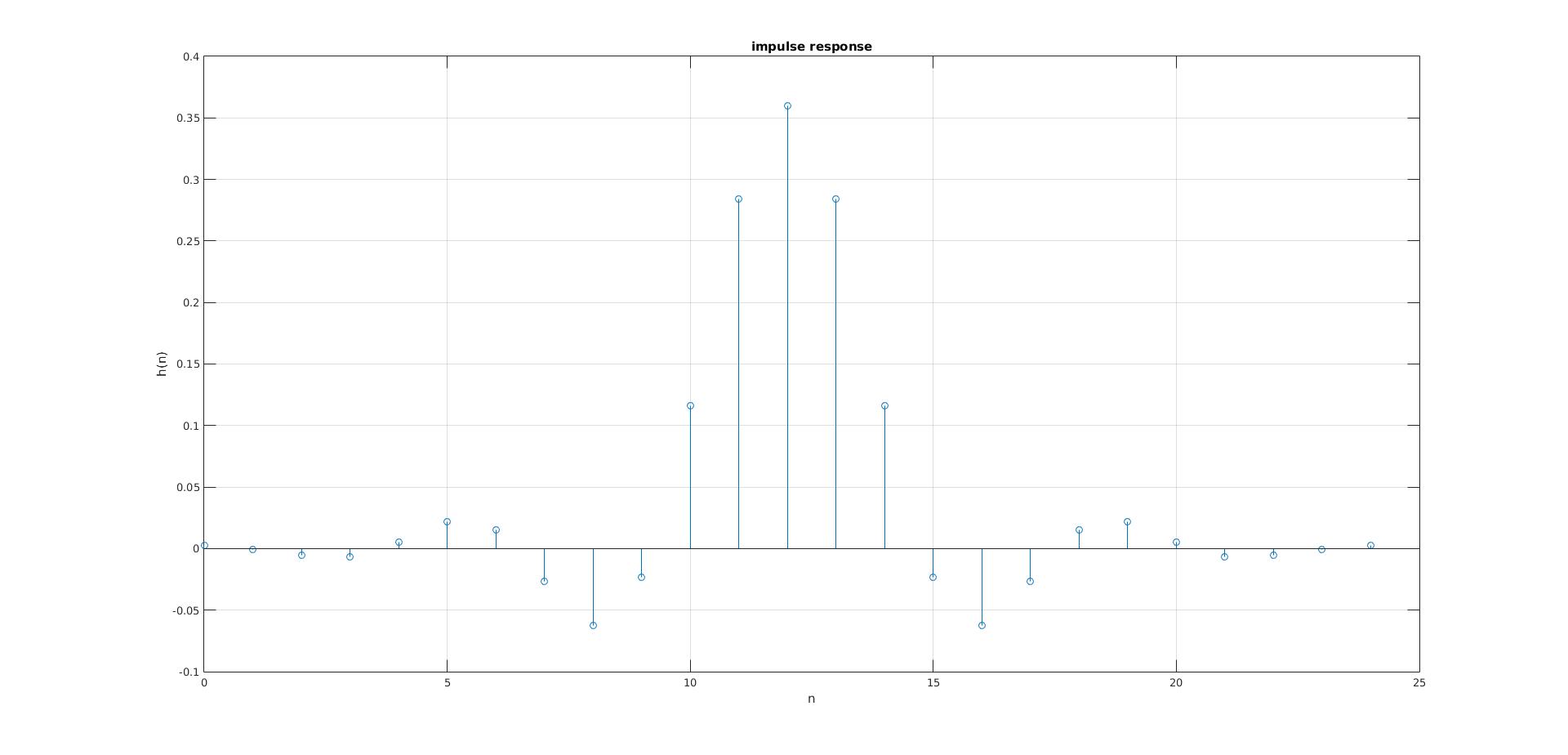
title('phase response')

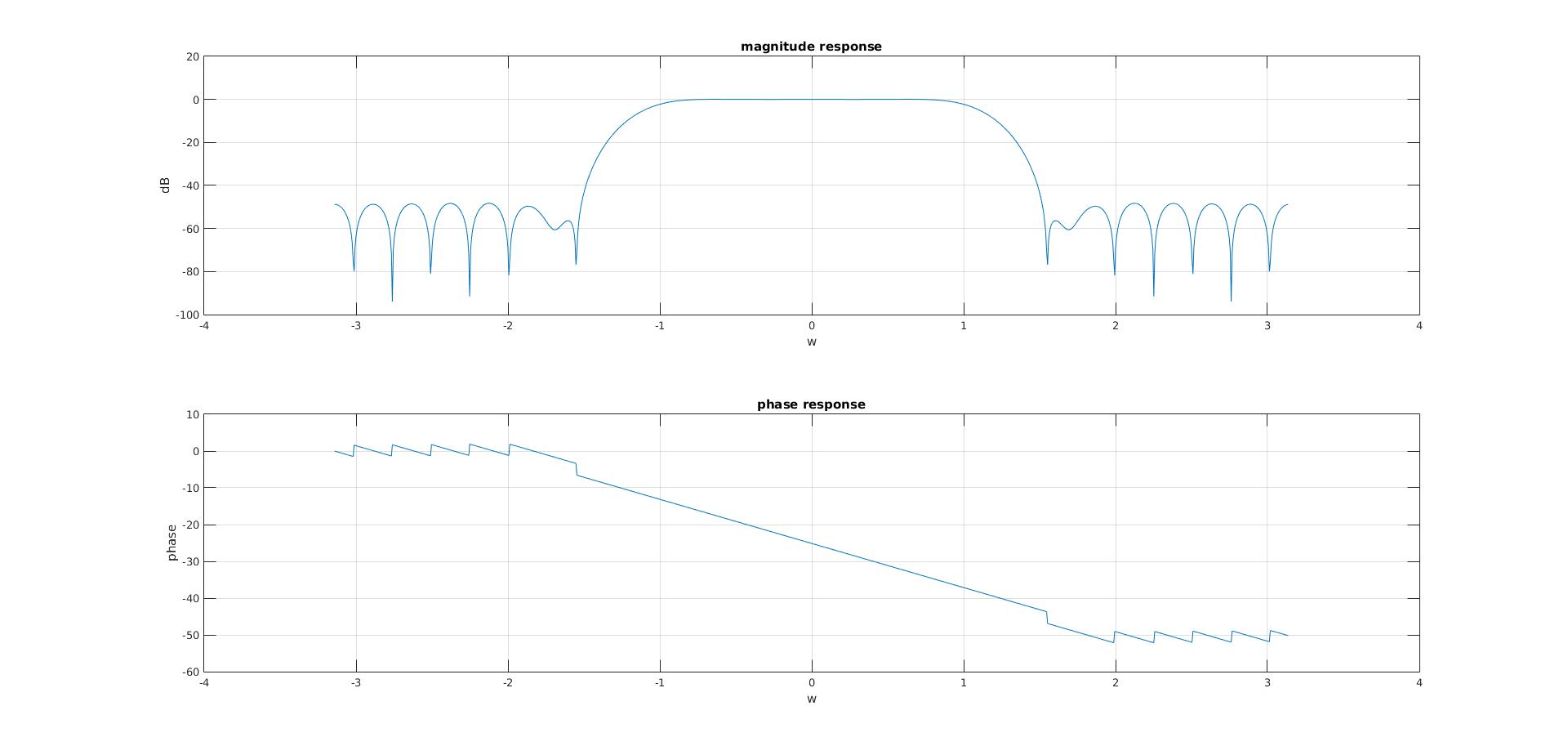
xlabel('w')

ylabel('phase')

grid on

figure:



pass-band ripple:0.26dB

stop-band attenuation:50dB

pass-band edge frequency:0.9

stop-band edge frequency:1.5

Report Item 4:

code:

rp = 2

a = [1,0]

fs = 1

f = [0.3/2,0.36/2]

rs = 50

dev = [(10^(rp/20)-1)/(10^(rp/20)+1) 10^(-rs/20)]

[n,fo,mo,w] = firpmord(f,a,dev,fs)

b = firpm(n,fo,mo,w)

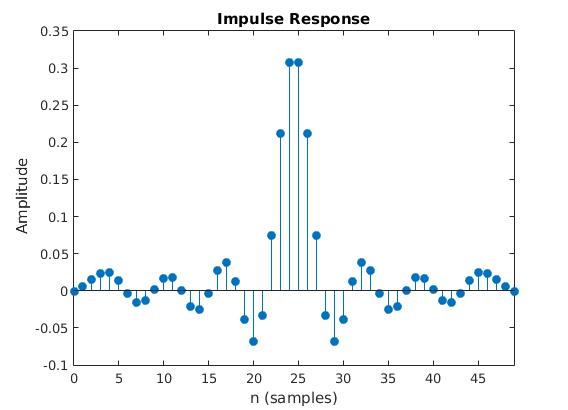
freqz(b,1,1024)

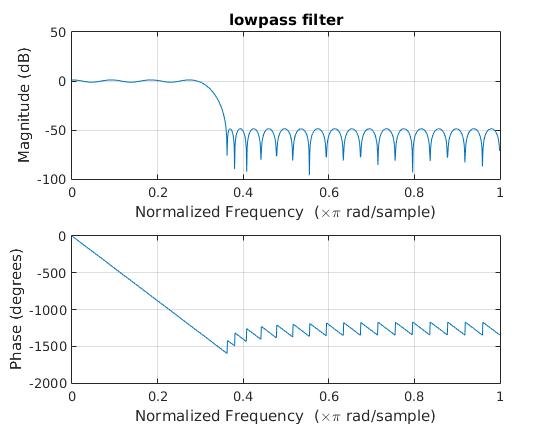
title('lowpass filter')

figure;

impz(b,1)

figure:





Report Item 5

code:

function [stdft,Omega,shift] = mySTDFT(x,M,D,P,fs)

N = length(x);

n = 0:length(x)-1;

row = ceil(P/2);

col = floor((N-M)/D)+1;

stdft = zeros(row,col);

for i = 1:col

y = x(((i-1)\*D +1):(i-1)\*D+M);

Y = fft(y,P);

stdft(:,i) = Y(1:row);

end

%shift

m = [0:col-1];

shift = D\*m/fs;

%omega

w = [0:col-1]/col\*2\*pi;

Omega = fs\*w;

end

load spectrogram

M = 5

P = 128

D = 5

[stdft,analog,shift] = mySTDFT(x,M,D,P,fs)

imagesc(shift,analog/2/pi,abs(stdft))

axis('xy')

title('spectrogram')

xlabel('time(s)')

ylabel('frequency(Hz)')

load spectrogram

M = 30

P = 128

D = 5

[stdft,analog,shift] = mySTDFT(x,M,D,P,fs)

imagesc(shift,analog/2/pi,abs(stdft))

axis('xy')

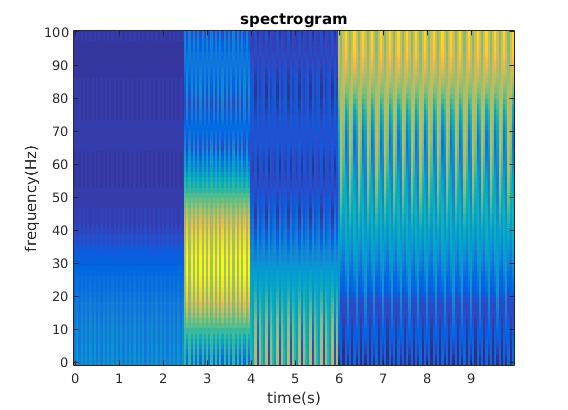
title('spectrogram')

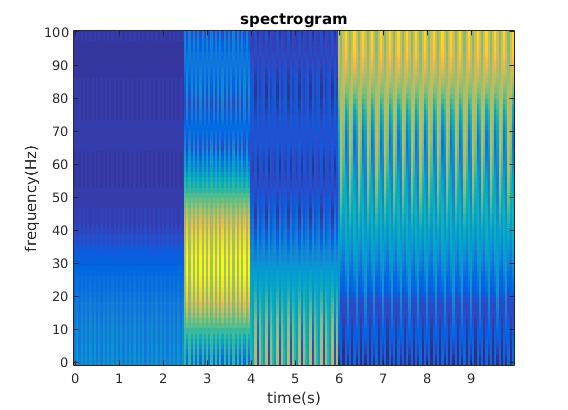
xlabel('time(s)')

ylabel('frequency(Hz)')

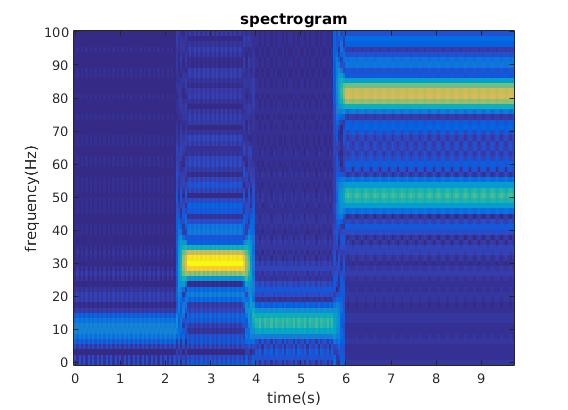
figure:

M = 5:





M = 30



o to 100 Hz is present in the spectrogram

10 to 50 Hz occurs from 2.5 to 4s

60 to 100 Hz occurs after 6s

When M = 30

The frequency resolution increases. The minimum frequency captured is lower.

Rayleigh limit states:

M/fs < 1/fmin

fmin > fs/M

10Hz component is not captured when M = 5 but it is present when M = 30

Report Item 6:

code:

[Y,fs] = wavread('sound1')

N = length(Y);

time = length(Y)/fs;

Y\_transform = fft(Y);

Y\_transform = fftshift(Y\_transform);

w = fftshift((0:N - 1)/N\*2\*pi);

w(1:N/2) = w(1:N/2) - 2\*pi;

omega = fs\*w/1000/2/pi;%analog kHz

plot(omega,abs(Y\_transform))

title('magnitude spectrum of sound1')

xlabel('f(kHz)')

ylabel('X(f)')

grid on

M = 200

P = 2048

D = 5

[stdft,analog,shift] = mySTDFT(Y,M,D,P,fs);

figure

imagesc(shift,analog/2/pi/1000,abs(stdft));

axis('xy')

title('spectrogram of sound 1')

xlabel('time')

ylabel('frequency(kHz)')

%filter

rp = 1

a = [1 0];

fs = 44100

f = [10000 10050];

rs = 60

dev = [(10^(rp/20)-1)/(10^(rp/20)+1) 10^(-rs/20)];

[n,fo,mo,w] = firpmord(f,a,dev,fs);

b = firpm(n,fo,mo,w);

B = fft(b,N);

newY = B'.\*fft(Y);

newy = ifft(newY);

figure

plot(omega,abs(fftshift(newY)))

title('magnitude spectrum of filtered sound1')

xlabel('f(kHz)')

ylabel('X(f)')

grid on

soundsc(newy,fs)

[stdft,analog,shift] = mySTDFT(newy,M,D,P,fs);

figure

imagesc(shift,analog/2/pi/1000,abs(stdft));

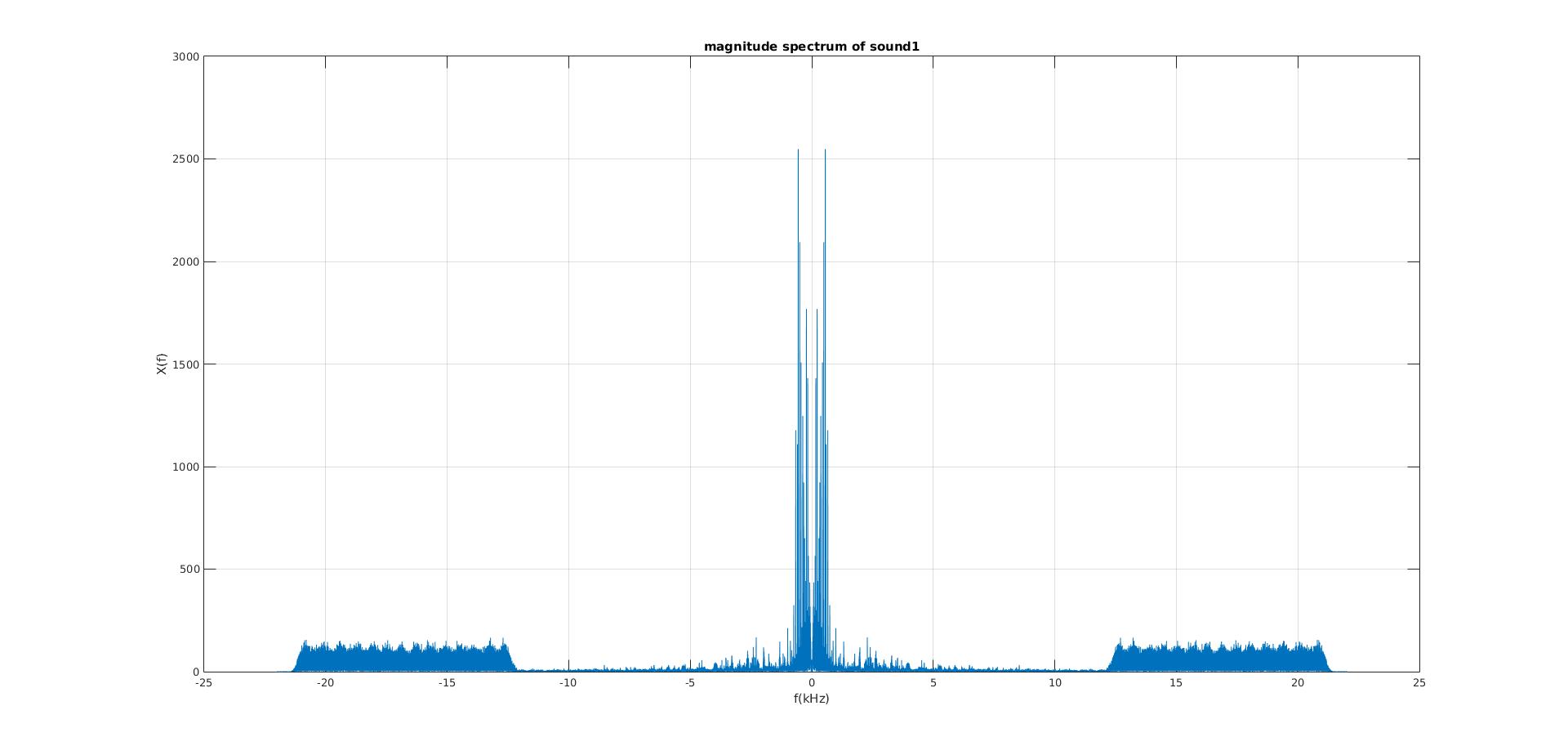
axis('xy')

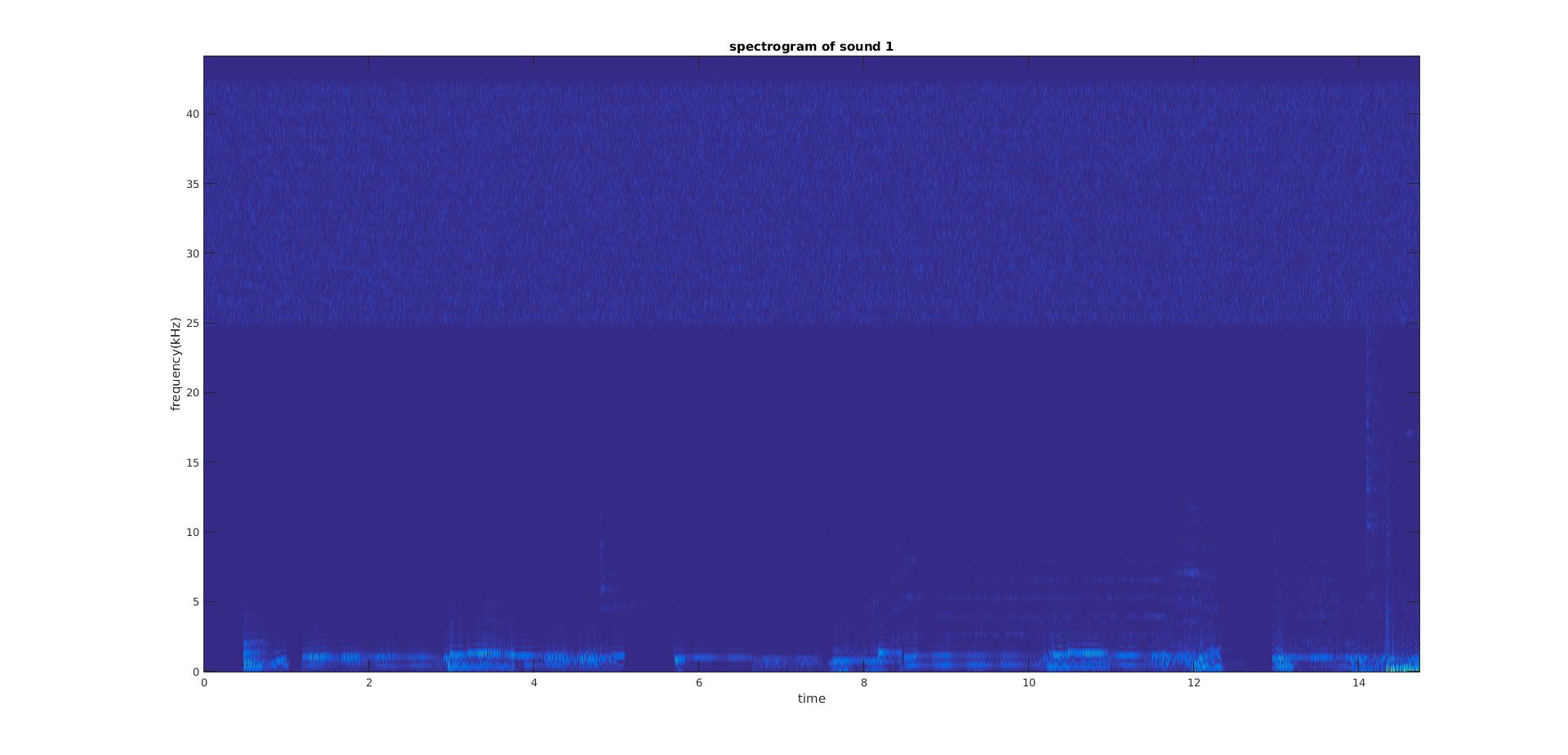
title('filtered spectrogram of sound 1')

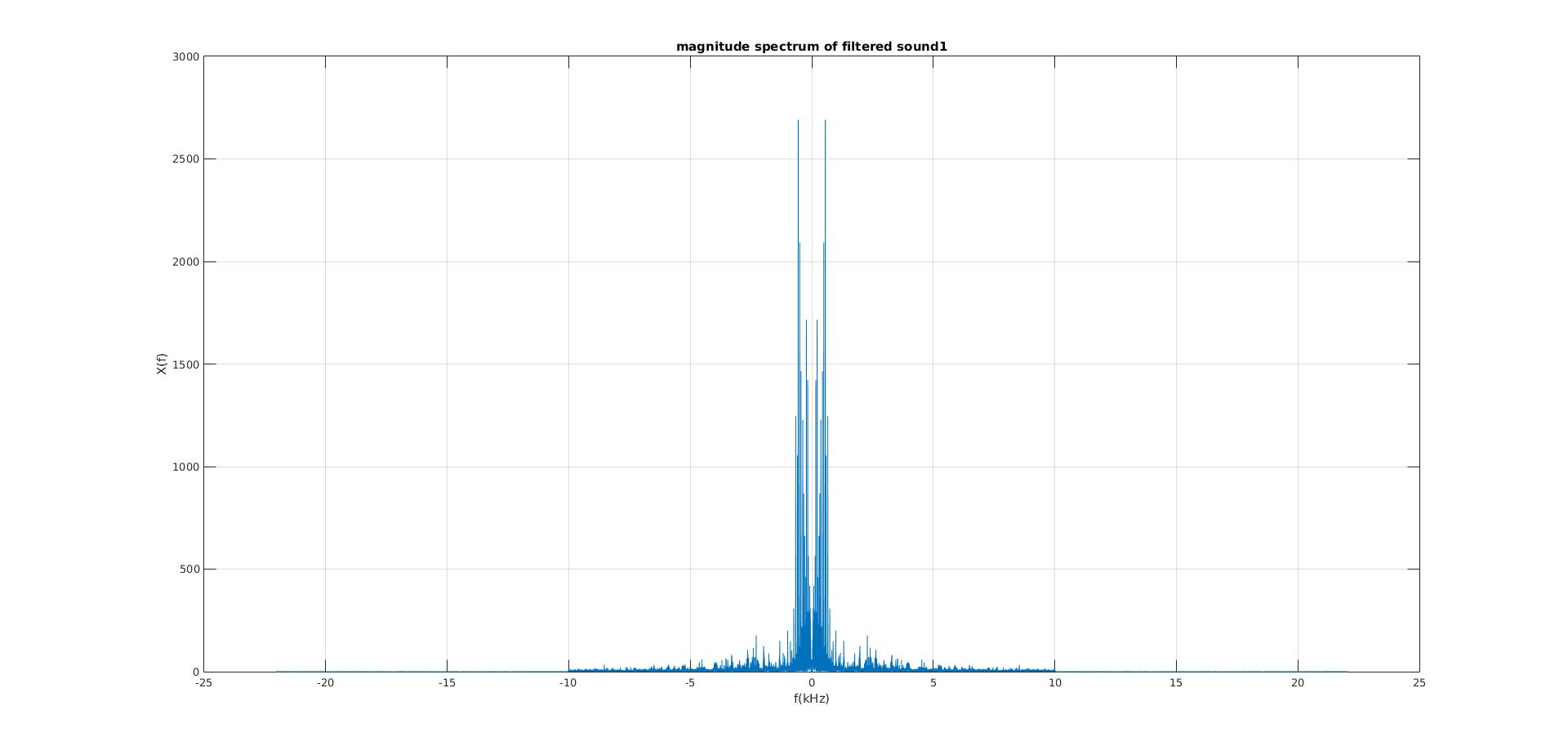
xlabel('time')

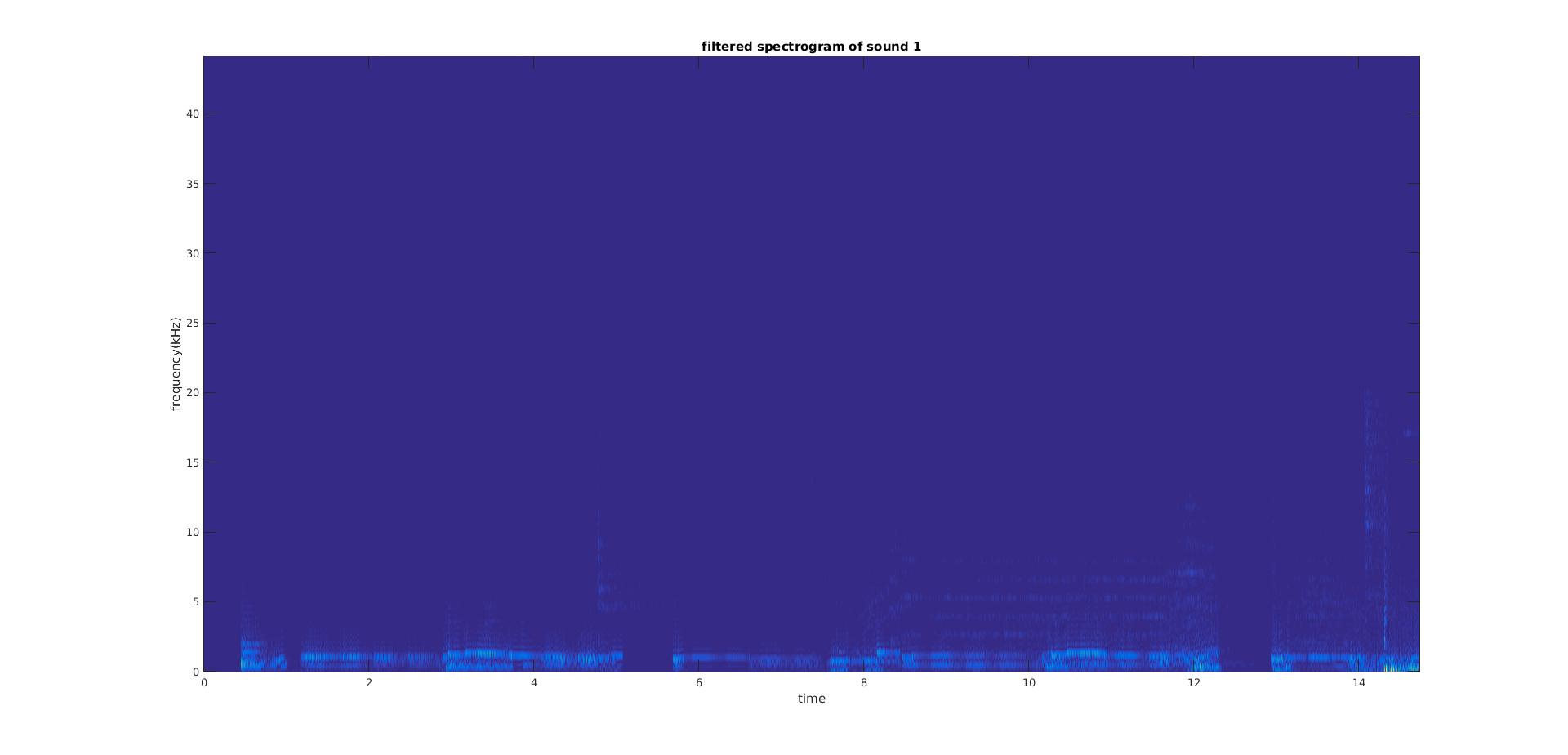
ylabel('frequency(kHz)')

figure:









M = 200

P = 2048

D = 5

fs = 44100

Effectiveness:

White noise is mostly gone. The sound is more pleasant after the filter

Report Item 7:

code:

[Y,fs] = wavread('sound2')

N = length(Y);

time = length(Y)/fs;

Y\_transform = fft(Y);

Y\_transform = fftshift(Y\_transform);

w = fftshift((0:N - 1)/N\*2\*pi);

w(1:N/2) = w(1:N/2) - 2\*pi;

omega = fs\*w/1000/2/pi;%kHz

plot(omega,abs(Y\_transform))

title('magnitude spectrum of sound2')

xlabel('f(kHz)')

ylabel('X(f)')

grid on

M = 200

P = 2048

D = 1

[stdft,analog,shift] = mySTDFT(Y,M,D,P,fs);

figure

imagesc(shift,analog/2/pi/1000,abs(stdft));

axis('xy')

title('spectrogram of sound 2')

xlabel('time')

ylabel('frequency(kHz)')

rp = 1

a = [1 0];

fs = 44100

f = [2000,10000];

rs = 60

dev = [(10^(rp/20)-1)/(10^(rp/20)+1) 10^(-rs/20)];

[n,fo,mo,w] = firpmord(f,a,dev,fs);

b = firpm(n,fo,mo,w)

B = fft(b,N);

newY = B'.\*fft(Y);

newy = ifft(newY);

figure

plot(omega,abs(fftshift(newY)))

title('magnitude spectrum of filtered sound 2')

xlabel('f(kHz)')

ylabel('X(f)')

grid on

soundsc(newy,fs)

[stdft,analog,shift] = mySTDFT(newy,M,D,P,fs);

figure

imagesc(shift,analog/2/pi/1000,abs(stdft));

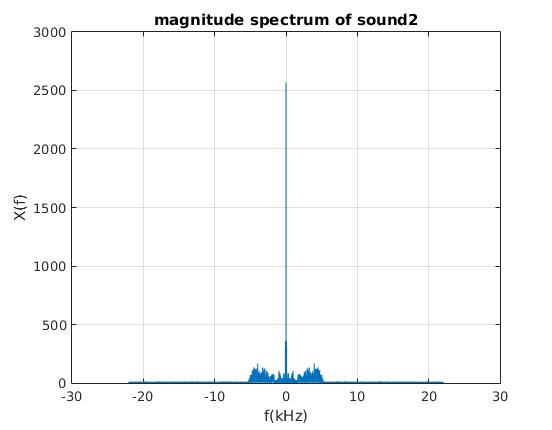
axis('xy')

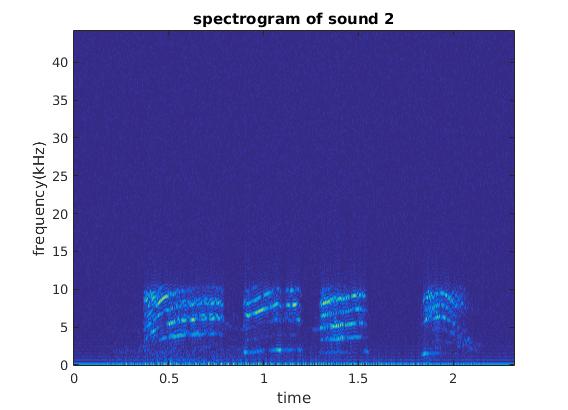
title('filtered spectrogram of sound 2')

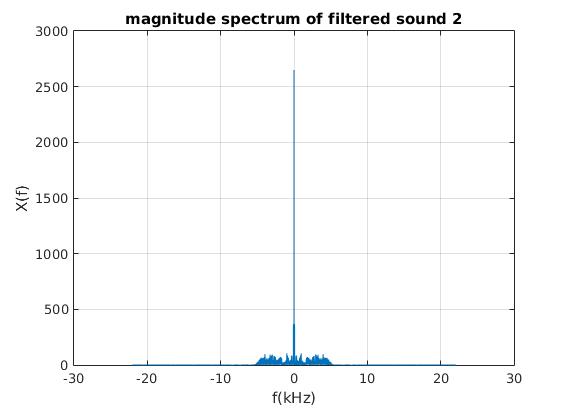
xlabel('time')

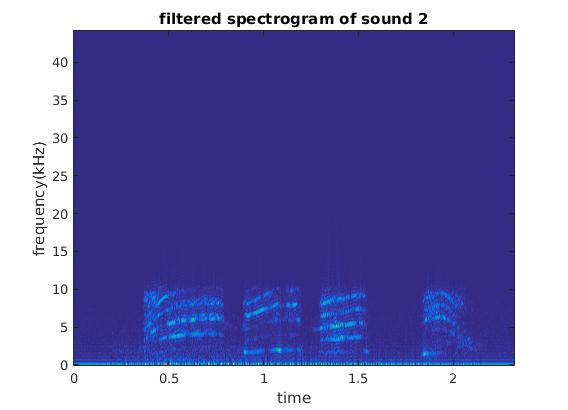
ylabel('frequency(kHz)')

figure:









White noise is present from 0Hz t0 40KHz

The spectrum fails to capture the noise under 10kHz since it's hidden inside the desired signal and spectrum cannot show how frequency varies over time while spectrogram can do so.

Effectiveness

Filter removes most of the signal while there is still some noise present under 10kHz

Report Item 8:

code:

function [stdft,Omega,shift] = mySTDFTHamming(x,M,D,P,fs)

N = length(x);

n = 0:length(x)-1;

row = ceil(P/2);

col = floor((N-M)/D)+1;

stdft = zeros(row,col);

for i = 1:col

y = x(((i-1)\*D +1):(i-1)\*D+M);

w = hamming(M)

Y = fft(w.\*y,P);

stdft(:,i) = Y(1:row);

end

%shift

m = [0:col-1];

shift = D\*m/fs;

%omega

w = [0:col-1]/col\*2\*pi;

Omega = fs\*w;

end

load speechsig

P = 128

step = 2

M = 128

fftlength = 128

stepsize = 2

fs = 1

[stdftH,analogH,shiftH] = mySTDFTHamming(x,M,stepsize,fftlength,fs)

imagesc(shiftH,analogH/2/pi/1000,abs(stdftH))

axis('xy')

title('spectrogram with Hamming window')

xlabel('time')

ylabel('frequency(kHz)')

[stdft,analog,shift] = mySTDFT(x,M,stepsize,fftlength,fs)

figure

imagesc(shift,analog/2/pi/1000,abs(stdft))

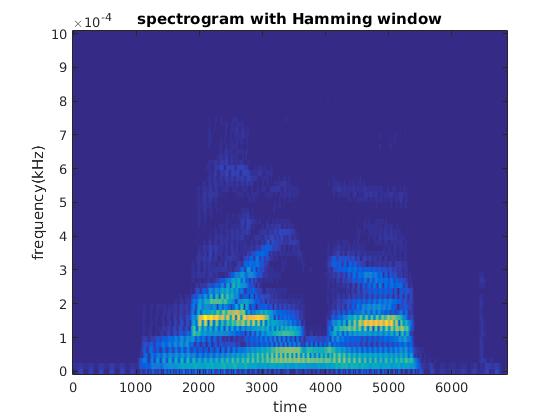
axis('xy')

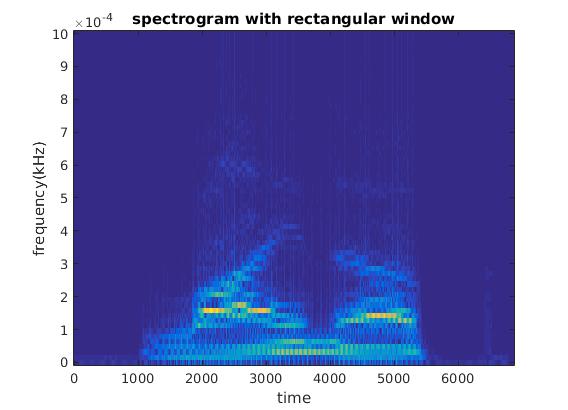
title('spectrogram with rectangular window')

xlabel('time')

ylabel('frequency(kHz)')

figure:





Spectrogram created by Hamming window does not show the leakage as the one created by rectangular window as Hamming window attenuates the data in the future. Also the main component around 2kHz is clearer when Hamming window is applied