UNIVERSITY OF MORATUWA

DEPARTMENT OF ELECTRONIC AND TELECOMMUNICATION

EN2570 – DIGITAL SIGNAL PROCESSING

FIR Filter Design Project

Bandpass Filter

Nilakshana D.M.L.

180422U

ABSTRACT

This is a report on Bandpass filter design using Kaiser Window

INTRODUCTION

There are two classical methods for the design of non-recursive filters.

1. Fourier series method/ Window method

   This method is built using Fourier series in conjunction with a class of functions known as window functions

2. Weighted Chebyshev method

   This is a multivariable optimization method.

This report contains the designing of a filter using Fourier series method using Kaiser window functions.

BASIC THEORY

A fundamental property of digital filters in general is that they have a periodic frequency response with period equal to the sampling frequency ω_s

$$H\left(e^{j(\omega+k\omega_s)T}\right) = H\left(e^{j\omega T}\right)$$

By taking the Fourier series of $H\left(e^{j\omega T}\right)$,

$$H\left(e^{j\omega T}\right) = \sum_{n=-\infty}^{\infty} h(nT)e^{-j\omega nT}$$

Where ,

$$h(nT) = \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} H\left(e^{j\omega T}\right) e^{j\omega nT} d\omega$$

Transfer function of the non-recursive filter is obtained by substituting z = $e^{j\omega T}$

$$H(z) = \sum_{n=-\infty}^{\infty} h(nT)z^{-n}$$

Since Fourier series coefficients are defined over an infinite range, following problems are associated with this method.

1. The non-recursive filter obtained is of infinite length.
2. The filter is noncausal because the impulse response is nonzero for negative time.

The impulse response of the causal filter can be obtained by,

$h(nT) = 0$ for $|n| > M$, where $M = (N-1)/2$

Then we can obtain the transfer function of the causal filter by,

$$H'(z) \quad = \quad z^{-M} \sum_{n=-M}^{M} h(nT) z^{-n}$$

The frequency response of the causal filter is given by,

$$H'(e^{j\omega T}) \quad = \quad e^{-jM\omega T} \sum_{n=-M}^{M} h(nT)\, e^{-jn\omega T}$$

Since $\left|e^{-jM\omega T}\right|$ = 1, the amplitude response remains unchanged due to the delaying of impulse response by M samples.

The oscillations in the amplitude response in passband and as well as stopband are known as Gibbs oscillations due to the truncation of the Fourier series.
It is not possible to reduce passband and stopband oscillations by increasing the filter length.

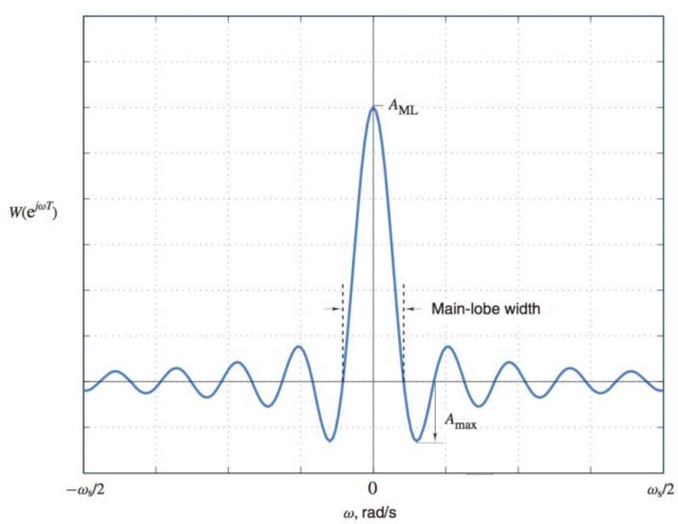
The standard technique for reducing these oscillations is using a discrete time window function, w(nT) ,
Let $\quad h_w(nT) = w(nT)$h(nT)

Modified transfer function can be obtained as

$$H_w(z) = Z[w(nT)h(nT)]$$

$$H_w(e^{j\omega T}) = \frac{T}{2\pi} \int_0^{\frac{2\pi}{T}} H(e^{j\omega' T}) W(e^{j(\omega-\omega')T}) d\omega'$$

Frequency spectrum of a usual window



Windows are characterized by their main-lobe width, *BML*, which is the bandwidth between the first negative and the first positive zero crossings, and by their ripple ratio r, which is defined as,

$$r = 100 \frac{A_{max}}{A_{ML}} \% \quad \text{or } R = 20\log \frac{A_{max}}{A_{ML}} \text{ dB}$$

A_{max} and A_{ML} are the maximum side lobe and main lobe amplitudes.

There are different types of window functions.

- Rectangular
- von Hann
- Hamming
- Blackman
- Dolph-Chebyshev
- Kaiser
- Ultraspherical

Kaiser window function is used to design the filter in this project.

KAISER WINDOW

The Kaiser window function is given by

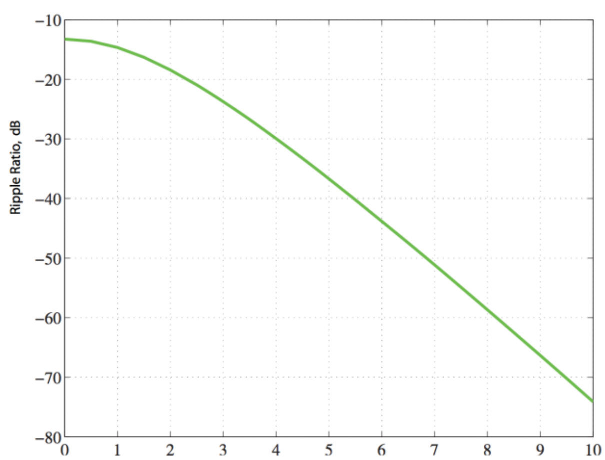
$$w_k(nT) = \begin{cases} \frac{I_0(\beta)}{I_0(\alpha)} & for\ |n| < \frac{N-1}{2} \\ 0 & otherwise \end{cases}$$

Where $\beta = \alpha\sqrt{1 - \left({2n}/{(N-1)}\right)^2}$

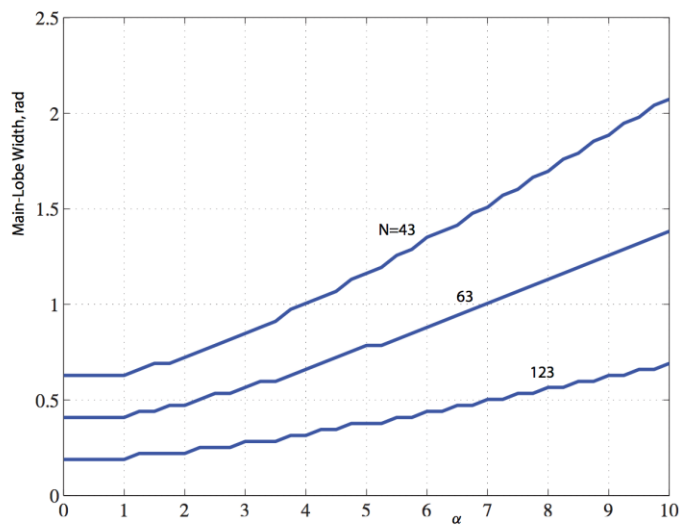
$$I_0 = 1 + \sum_{k=1}^{\infty}\left[\frac{1}{k!}\left(\frac{x}{2}\right)^k\right]^2$$

α is an independent parameter, and *Io*(*x*) is a zeroth-order modified Bessel function of the first kind.

Ripple ratio versus α



Main-lobe width versus α

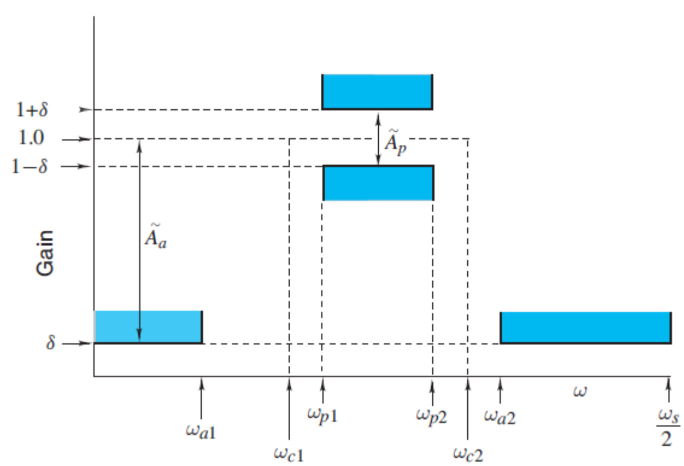


Kaiser's method can be used to design lowpass, highpass, bandpass and bandstop filters.

DESIGNING PROCEDURE OF THE FILTER

The specifications of the required bandpass filter are given below,

- Passband ripple $\leq \widetilde{A}p$
- Minimum stopband attenuation $\leq \widetilde{A}a$
- Lower stopband edge ω_{a1}
- Lower passband edge ω_{p1}
- Upper passband edge ω_{p2}
- Upper stopband edge ω_{a2}
- Sampling frequency ω_s



The more critical transition width is,

$$B_t = \min[(\omega_{a1} - \omega_{p1}), (\omega_{a2} - \omega_{p2})]$$

The ideal frequency response for the bandpass filter is ,

$$H(e^{j\omega T}) = \begin{cases} 1 & for - \omega_{c2} \le \omega \le -\omega_{c1} \\ 1 & for\ \omega_{c1} \le \omega \le \omega_{c2} \\ 0 & otherwise \end{cases}$$

where $\omega_{c1} = \omega_{p1} - \frac{B_t}{2}$ and $\omega_{c2} = \omega_{p2} + \frac{B_t}{2}$

By applying the Fourier series, we can get,

$$h(nT) = \begin{cases} \frac{2}{\omega_s}(\omega_{c2} - \omega_{c1}) & for\ n = 0 \\ \frac{1}{n\pi}(sin\omega_{c2}nT - sin\omega_{c1}nT) & otherwise \end{cases}$$

<u>Steps of designing the filter.</u>

1. Choosing value for δ

$$\delta = \min(\tilde{\delta}p, \tilde{\delta}a) \quad \text{where } \tilde{\delta}p = \frac{10^{0.05\tilde{A}p}-1}{10^{0.05\tilde{A}p}+1} \quad \text{and } \tilde{\delta}a = 10^{-0.05\tilde{A}a}$$

2. Calculation of actual stopband attenuation A_a

$$A_a = -20\log(\delta)$$

3. Choosing parameter α

$$\alpha = \begin{cases} 0 & ;for\ A_a \le 21 \\ 0.5842(A_a - 21)^{0.4} + 0.07886(A_a - 21) & ;for\ 21 < A_a \le 50 \\ 0.1102(A_a - 87) & ;for\ A_a > 50 \end{cases}$$

4. Choose parameter D

$$D = \begin{cases} 0.9222 & ;for\ A_a \le 21 \\ \frac{A_a - 7.95}{14.36} & ;for\ A_a > 21 \end{cases}$$

Then select the lowest odd value of N that satisfy the following equation

N $\geq \frac{\omega_s D}{B_t} + 1$ where B_t = min[$(\omega_{a1} - \omega_{p1})$,($\omega_{a2} - \omega_{p2}$)]

5. Form $w_k(nT)$ using Kaiser window functions

$$w_k(nT) = \begin{cases} \frac{I_0(\beta)}{I_0(\alpha)} & for\ |n| < \frac{N-1}{2} \\ 0 & otherwise \end{cases}$$

where β = α$\sqrt{1 - \left({2n}/{(N-1)}\right)^2}$ and $I_0 = 1 + \sum_{k=1}^{\infty} \left[\frac{1}{k!}\left(\frac{x}{2}\right)^k\right]^2$

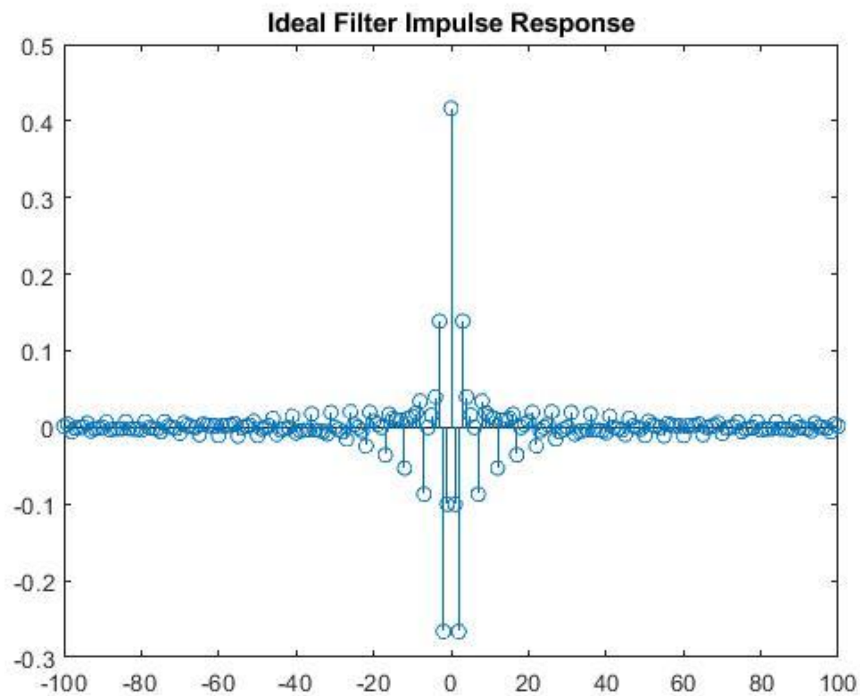
6. Form modified transfer function

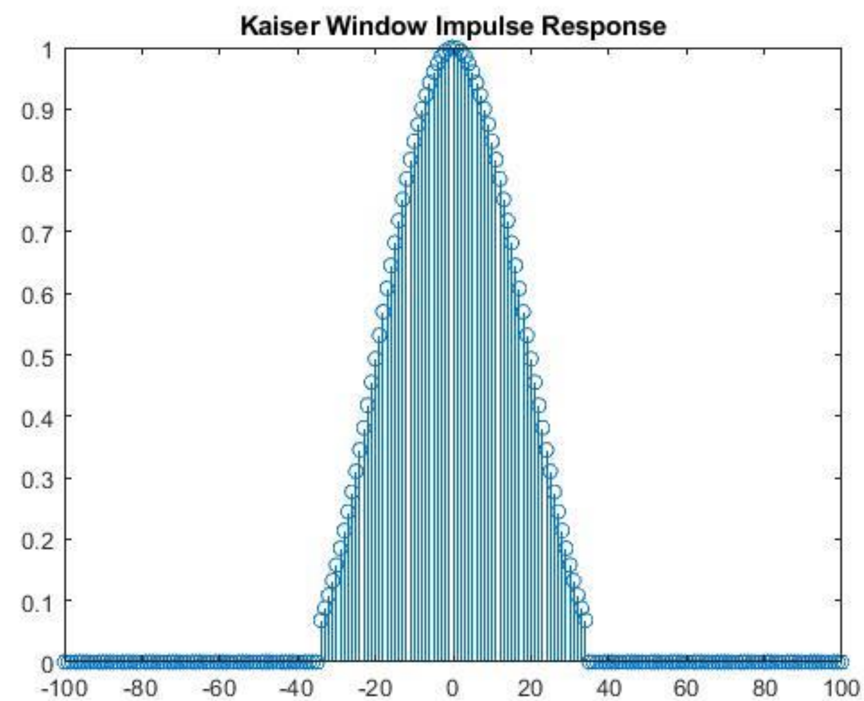
$$H'_w(z) \quad = \quad z^{-(N-1)/2} H_w(z)$$

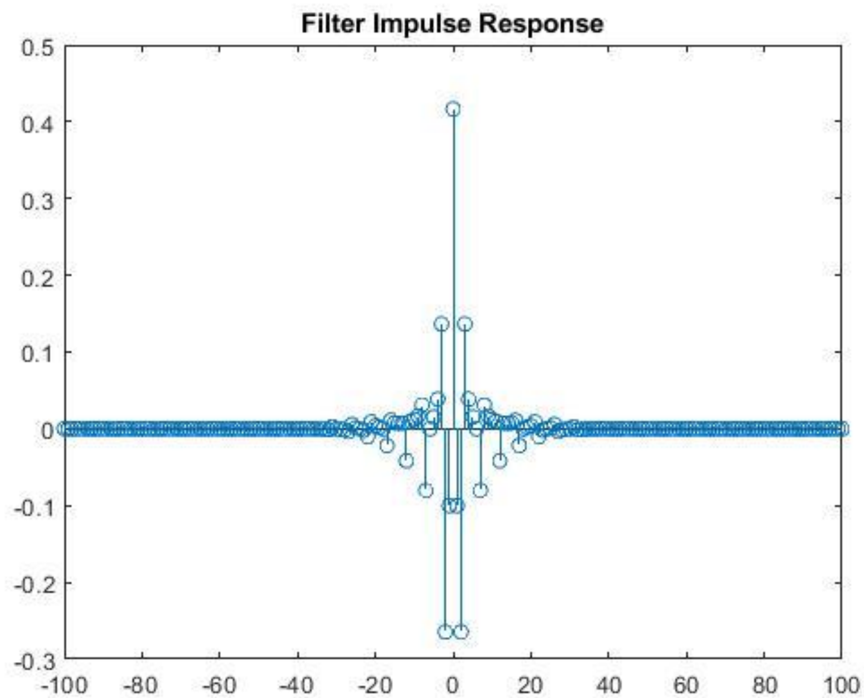
where $H_w(z)$ = Z[$w_k(nT)$h(nT)]

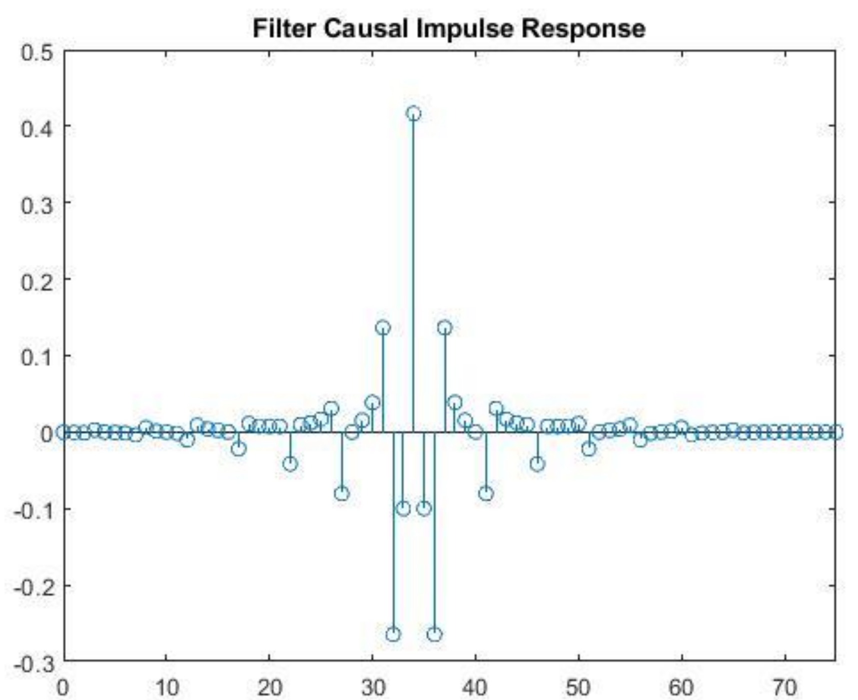
Results

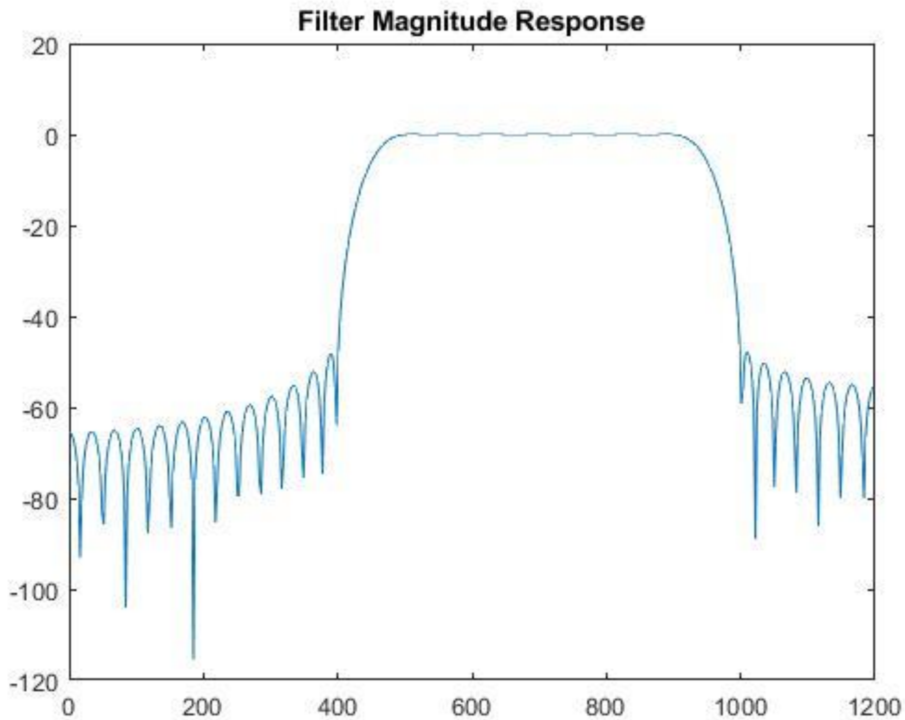
Following results were obtained when implementing the filter using Matlab R2018a.



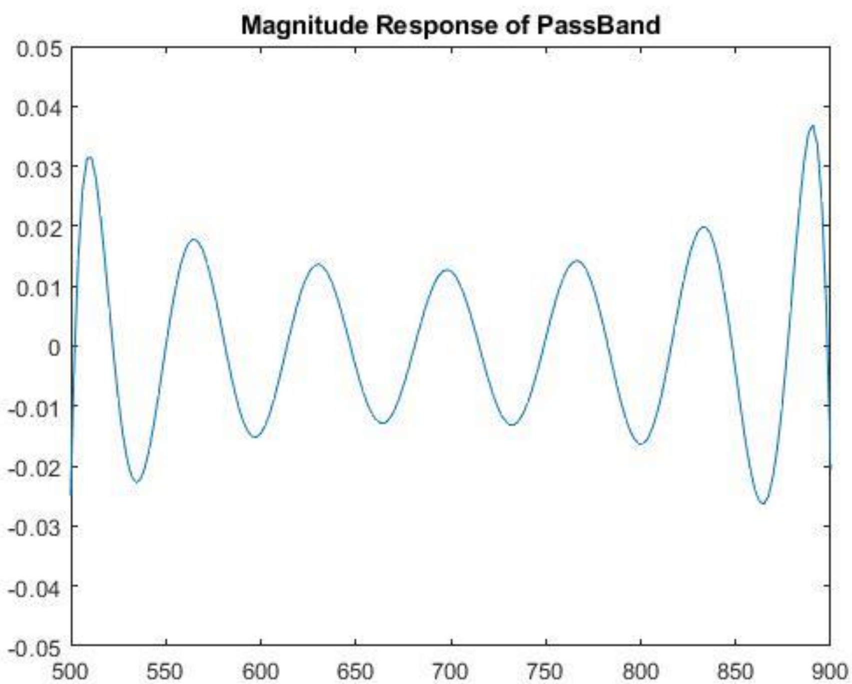



Filter Impulse Response
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-100
-80
-60
-40
-20
0
20
40
60
80
100

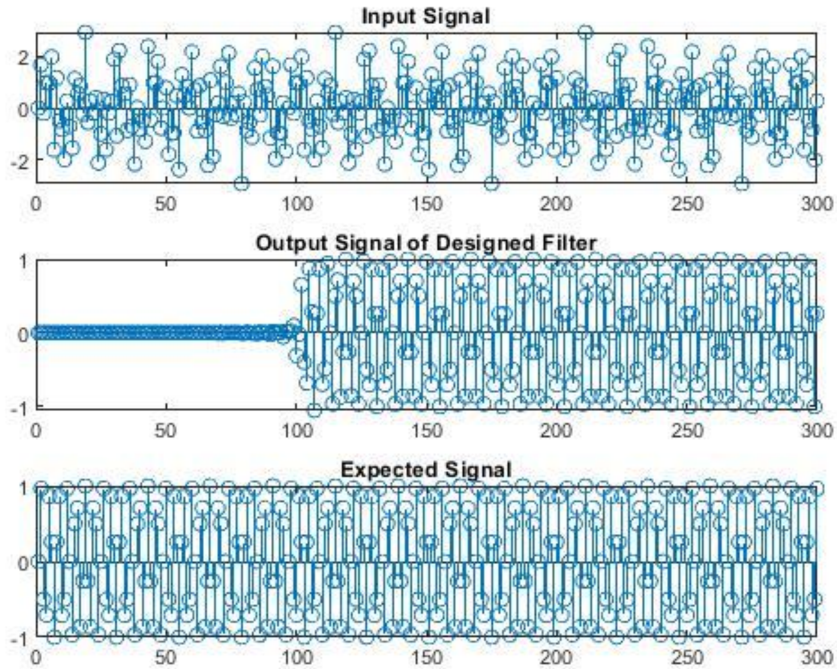

Filter Causal Impulse Response
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
10
20
30
40
50
60
70

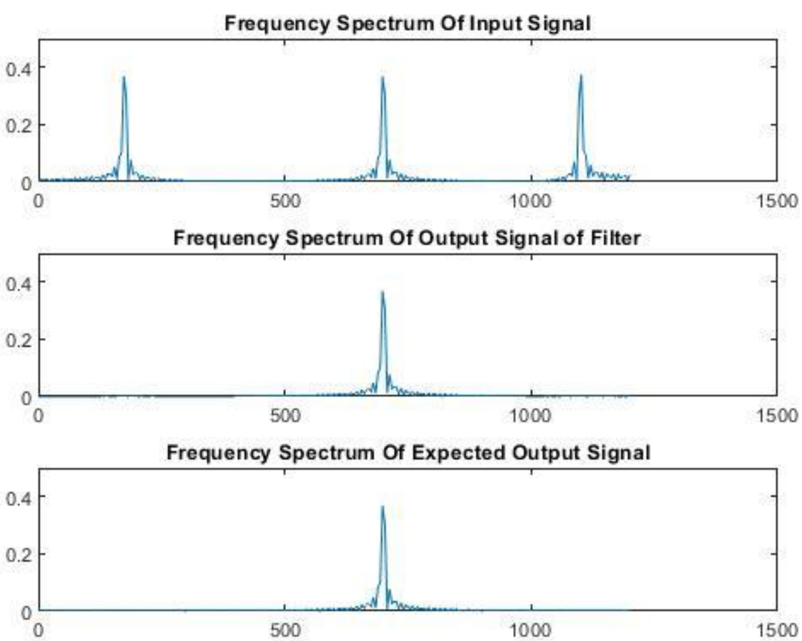


Filter Magnitude Response
20
0
-20
-40
-60
-80
-100
-120
0
200
400
600
800
1000
1200



Magnitude Response of PassBand
0.05
0.04
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04
-0.05
500
550
600
650
700
750
800
850
900


Input Signal
Output Signal of Designed Filter
Expected Signal


Frequency Spectrum Of Input Signal
Frequency Spectrum Of Output Signal of Filter
Frequency Spectrum Of Expected Output Signal

Conclusion

As we can see from above graphs, the filter designed using Kaiser window method gives almost the same output of the ideal filter. Hence this method is a very successful method of designing filters.

References

1. A. Antoniou, "Digital Signal Processing," 2005. [Online]. Available: www.ece.uvic.ca/~dsp. [Accessed 29 12 2016].
2. Discrete-Time Signal Processing by Alan V. Oppenheim, Ronald W. Schafer (z-lib.org)

Appendix

Filter Specifications

180422U

```
Ap = 0.07;
Aa = 47;
wp1 = 500;
wp2 = 900;
wa1 = 350;
wa2 = 1000;
ws = 2400;

Bt=min((wp1-wa1),(wa2-wp2));
wc1 = wp1-Bt/2;
wc2 = wp2+Bt/2;
T=2*pi/ws;

n = -100:100;
idealF = hnT(n,wc1,wc2,ws,T);
figure;
stem(n,idealF);
title('Ideal Filter Impulse Response')

deltaP=(10^(Ap/20)-1)/(10^(Ap/20)+1);
deltaA=10^(Aa/20);
delta=min(deltaP,deltaA);
ASBAtten = -20*log10(delta);
alpha=getAlpha(ASBAtten);
D= getD(ASBAtten);

N = ceil(ws*D/Bt+1);
if(mod(N,2)==0)
    N=N+1;
end
M=N-1;
tou=M/2;

kw=kaiserWindow(n,alpha,tou,N);
stem(n,kw);
title('Kaiser Window Impulse Response')

hn=idealF.*kw;
stem(n,hn);
title('Filter Impulse Response')

n1=n+tou;
stem(n1,hn);
xlim([0,75]);
title('Filter Causal Impulse Response')

[h,w]=freqz(hn);
w=w/T;
```

```
h=20*log10(abs(h));
plot(w,h);
title('Filter Magnitude Response')

l1=round(length(w)/(ws/2)*wc1);
l2=round(length(w)/(ws/2)*wc2);
plot(w(l1:l2),h(l1:l2));
axis([wp1 wp2 -0.05 0.05]);
title('Magnitude Response of PassBand')

w1=wa1/2;
w2=(wp1+wp2)/2;
w3=(wa2+ws/2)/2;
samples=300;
n=0:1:samples;
len = 2^nextpow2(samples);
frequency = ws*(0:1/len:1/2);
xt=sin(w1*n.*T)+sin(w2*n.*T)+sin(w3*n.*T);
Xw = fft(xt,len);
Xf = T*abs(Xw(1:len/2+1));
Hw = fft(hn,len);
yt=conv(hn,xt);
Yw=fft(yt,len);
Yf=T*abs(Yw(1:len/2+1));
yExp=sin(w2*n.*T);
YwExp=fft(yExp,len);
YfExp= T*abs(YwExp(1:len/2+1));

figure,
subplot(3,1,1);
t=(1:samples+1);
stem(t,xt(t));
title('Input Signal')
xlim([0,samples]);
subplot(3,1,2);
stem(t,yt(t));
title('Output Signal of Designed Filter')
xlim([0,samples]);
subplot(3,1,3);
stem(t,yExp(t));
title('Expected Signal')
xlim([0,samples]);

figure,
subplot(3, 1, 1) ;
plot(frequency,Xf);
title('Frequency Spectrum Of Input Signal');
axis([0, 1500, 0, 0.5]);
subplot(3, 1, 2);
plot(frequency, abs(Yf'));
title('Frequency Spectrum Of Output Signal of Filter');
```

```
axis([0, 1500, 0, 0.5]);
subplot(3, 1, 3);
plot(frequency, abs(YfExp'));
title('Frequency Spectrum Of Expected Output Signal');
axis([0, 1500, 0, 0.5]);
```

Functions

```
function [h] = hnT(n,wc1,wc2,ws,T)
h=zeros(size(n));
i1=0;
   for i = n
       i1=i1+1;
       if i==0
           h(i1)=2*(wc2-wc1)/ws;
       else
           h(i1)=(sin(wc2*i*T)-sin(wc1*i*T))/(i*pi);
       end
   end
end
function [alpha] = getAlpha(ASBAtten)
    if ASBAtten<=21
        alpha=0;
    else
        if ASBAtten<=50
            alpha=0.5842*(ASBAtten-21)^0.4+0.07886*(ASBAtten-21);
        else
            alpha = 0.1102*(ASBAtten - 8.7);
        end
    end
end
function [D]=getD(ASBAtten)
if (ASBAtten<=21)
    D = 0.9222;
else
    D = (ASBAtten-7.95)/14.36;
end
end
function [io]=Io(x)
k=1;
io=1;
while 1
    temp=(((x/2)^k)/factorial(k))^2;
    if temp<=10e-6
        break
    end
    io=io+temp;
    k=k+1;
end
end
```

```matlab
function [kw]=kaiserWindow(n,alpha,tou,N)
beta=alpha*sqrt(1-(2*n/(N-1)).^2);
kw=zeros(size(n));
for i=1:length(n)
    if (-tou>n(i)||n(i)>tou)
        kw(i)=0;
    else
        kw(i)=Io(beta(i))/Io(alpha);
    end
end
end
```