Department of Electronic and Telecommunication Engineering

University of Moratuwa, Sri Lanka

EN2570 - Digital Signal Processing

Design of a Finite Duration Impulse Response Bandpass Digital Filter

(For Prescribed Specifications Using the windowing method in conjunction with the Kaiser window)

Project Report

Submitted by

Thalagala B.P. 180631J

Submitted on

March 6, 2021

Abstract

Design procedure of a Finite Duration Impulse Response(FIR) bandpass Digital Filter which satisfies a set of prescribed specifications, is described in this report where windowing method in conjunction with the Kaiser window is used for the designing procedure. Operation of the filter was analyzed with a combination of sinusoidal signals. The design was implemented and tested using `MATLAB R2018a` of the MathWorks Inc.

Contents

List of Figures

List of Tables

****PDF is clickable***

Note:
All the materials and executable `MATLAB R2018a` *Live Script related to the project can also be found at* *https://github.com/bimalka98/Digital-Signal-Processing*

1 Introduction

This report describes the design procedure of an FIR bandpass digital filter which satisfies a predefined set of specifications as shown in the Table 1. ***Kaiser windowing*** method is used, due to its excellent capability to control filter's ripple ratio and main-lobe width to facilitate a given set of specification by simply varying the related parameters. And most importantly a method is available to calculate those parameters through empirical formulae. Therefore this Kaiser Window is heavily used to design filters with prescribed specifications.

`MATLAB R2018a` of the MathWorks Inc. is used to implement and analyze the digital filter. Filtering was done in the frequency domain rather than in the time domain since the time domain convolution is computationally expensive. Frequency domain analysis was done using ***Fast Fourier Transform*** algorithms(*using built-in* `fft()` *and* `ifft()` *functions*). Performance of the filter was analyzed using a combination of sinusoidal signal, whose frequency components lie in the middle of the three main regions(*lower stopband, passband and upper stopband*) of the bandpass filter.

2 Method

First the digital filter is implemented through the procedure described in the **section 2.1**. Then its performance is analyzed as described in the **section 2.2**.

2.1 Filter Implementation

Digital filter implementation consists of the steps that are mentioned below. Subsections of this section of the report describes each one of them thoroughly for designing the FIR bandpass filter using Kaiser Window method.

1. Identifying the prescribed filter specifications
2. Derivation of the filter Parameters
3. Derivation of the Kaiser Window Parameters
4. Derivation of The Ideal Impulse Response
5. Truncating the Ideal Impulse Response to obtain Finite Impulse Response i.e Windowing

2.1.1 Prescribed Filter specifications

Following table describes the desired specifications of the bandpass filter which need to be implemented. The notation used here is the same as the notation used in the reference material[1] and they will be used throughout the report rather than numerical values.

Parameter	Symbol	Value
Maximum passband ripple(*desired*)	$\tilde{A}_p$	0.09 dB
Minimum stopband attenuation(*desired*)	$\tilde{A}_a$	48 dB
Lower passband edge	ω_{p1}	400 rad/s
Upper passband edge	ω_{p2}	800 rad/s
Lower stopband edge	ω_{a1}	250 rad/s
Upper stopband edge	ω_{a2}	900 rad/s
Sampling frequency	ω_s	2600 rad/s

Table 1: Prescribed Filter specifications

Following Fig.1 illustrates the aforementioned specifications graphically for an idealized frequency response of a Bandpass filter. δ in the figure has the following relationship with peak to peak passband ripple(*practical*) A_p and the minimum stopband attenuation(*practical*) A_a.

$$\tilde{A_p} \geq A_p = 20\log\left(\frac{1+\delta}{1-\delta}\right) \tag{1}$$

$$\tilde{A_a} \leq A_a = -20\log(\delta) \tag{2}$$

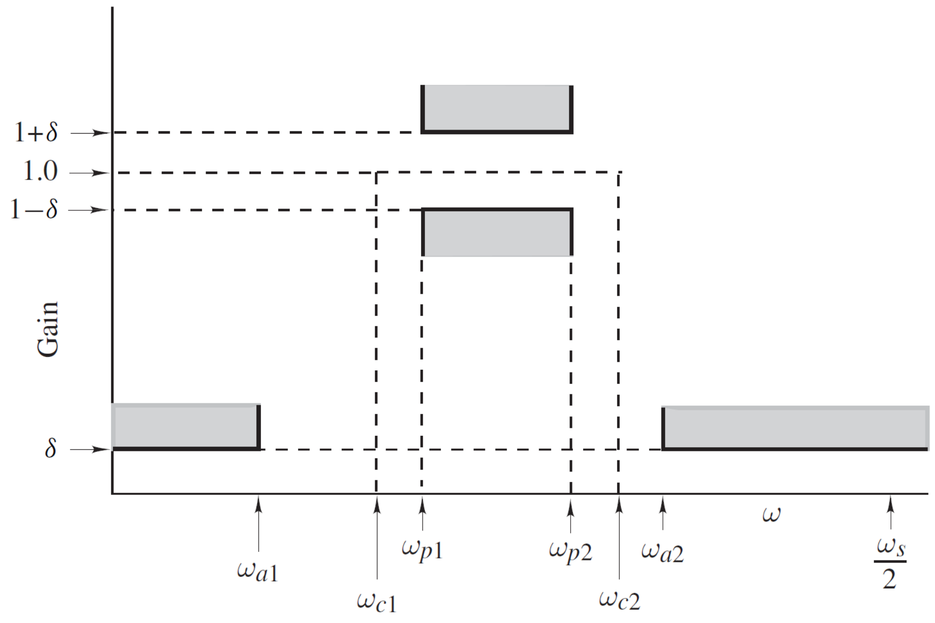

Figure 1: Idealized frequency response of a Bandpass filter[1]

2.1.2 Derivation of filter Parameters

According to the given specifications following parameters are calculated; which then will be used to derive the parameters of the Kaiser window.

Parameter	Symbol	Calculation	Value
Lower transition width	B_{t1}	$\omega_{p1} - \omega_{a1}$	150 rad/s
Upper transition width	B_{t2}	$\omega_{a2} - \omega_{p2}$	100 rad/s
Critical transition width	B_t	$\min(B_{t1}, B_{t2})$	100 rad/s
Lower cutoff frequency	ω_{c1}	$\omega_{p1} - B_t/2$	350 rad/s
Upper cutoff frequency	ω_{c2}	$\omega_{p2} + B_t/2$	850 rad/s
Sampling period	T	$2\pi/\omega_s$	0.0024 s

Table 2: Derivation of filter Parameters

2.1.3 Derivation of the Kaiser Window Parameters

Following equation represents the Kaiser window which will be used to truncate the Infinite duration Impulse Response to obtain the Finite duration Impulse Response for our filter design. Further explanations of the same steps can be found in the *sections 9.4.5* and *9.4.6* in the reference material[1].

$$w_K(nT) = \begin{cases} \frac{I_0(\beta)}{I_0(\alpha)} & for|n| \leq \frac{N-1}{2} \\ 0 & Otherwise \end{cases} \tag{3}$$

where α is an independent parameter and $I_0(x)$ is the zeroth-order modified Bessel function of the first kind.

$$\beta = \alpha\sqrt{1-\left(\frac{2n}{N-1}\right)^2} \qquad\qquad I_0(x) = 1+\sum_{k=1}^{\infty}\left[\frac{1}{k!}\left(\frac{x}{2}\right)^k\right]^2$$

Now we have to calculate the required parameters as follows,

a.) Choose δ in Eqs. (1) and (2) such that $\delta = \min(\tilde{\delta_p}, \tilde{\delta_a})$ where,

$$\begin{aligned}\tilde{\delta_p} &= \frac{10^{0.05\tilde{A}_p}-1}{10^{0.05\tilde{A}_p}+1}\\ &= \frac{10^{0.05*0.09}-1}{10^{0.05*0.09}+1}\\ &= 5.181\times10^{-3}\end{aligned} \qquad \begin{aligned}\tilde{\delta_a} &= 10^{-0.05\tilde{A}_a}\\ &= 3.981\times10^{-3}\end{aligned}$$

$$\therefore\ \delta = 3.981\times10^{-3}$$

b.) With the required δ defined, the actual stopband loss(attenuation) A_a in dB can be calculated using Eq. 2.

$$\begin{aligned}\tilde{A_a} \le A_a &= -20\log(\delta)\\ &= -20\log\big(3.981\times10^{-3}\big)\\ A_a &= 48\ dB\end{aligned}$$

c.) Choose parameter α as,

$$\alpha = \begin{cases}0 & for\ A_a \le 21\ dB\\ 0.5842(A_a-21)^{0.4}+0.07886(A_a-21) & for\ 21 < A_a \le 50\ dB\\ 0.1102(A_a-8.7) & for\ A_a > 50\ dB\end{cases}$$

$$\begin{aligned}\therefore\ \alpha &= 0.5842(A_a-21)^{0.4}+0.07886(A_a-21)\\ &= 0.5842(48-21)^{0.4}+0.07886(48-21)\\ &= 4.3125\end{aligned}$$

d.) Choose parameter D as,

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

$$\begin{aligned}\therefore\ D &= \frac{A_a-7.95}{14.36}\\ &= \frac{48-7.95}{14.36}\\ &= 2.7890\end{aligned}$$

e.) Then select the lowest odd value of N that would satisfy the inequality,

$$\begin{aligned}N &\ge \frac{\omega_s D}{B_t}+1\\ &\ge \frac{2600*2.79}{100}+1\\ &\ge 73.51\end{aligned} \qquad\qquad \therefore\ N = 75$$

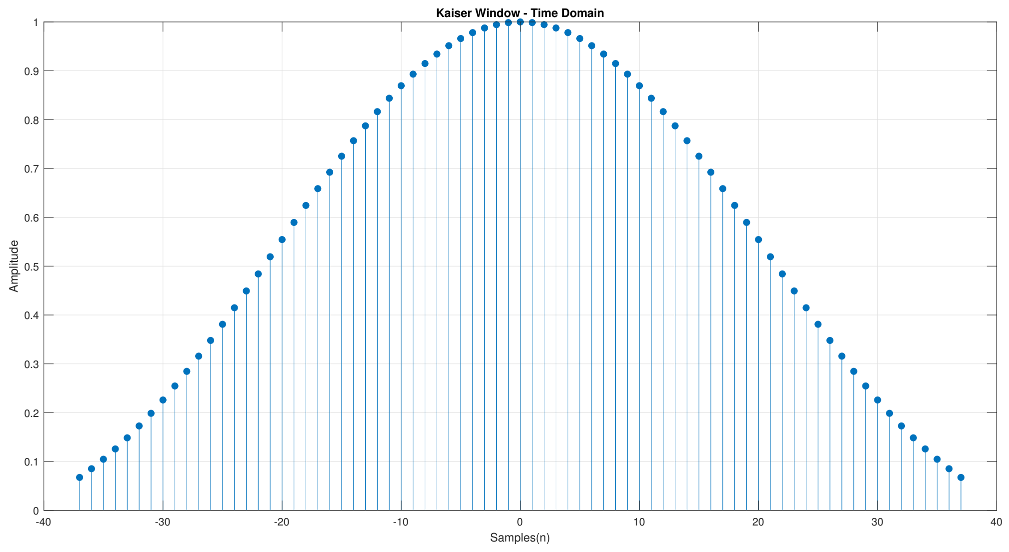

Figure 2: Derived Kaiser Window $w_K(nT)$ using calculated Parameters and Eq. (3)

2.1.4 Derivation of The Ideal Impulse Response

Note : Here subscript 'd' implies "desired", as it is the ideal response of the filter. Subscript d will be omitted to indicate a given expression is no longer ideal.

The frequency response of an ideal bandpass filter with cutoff frequencies ω_{c1} and ω_{c2} is given by,

$$H_d(e^{j\omega T}) = \begin{cases} 1 & for \ -\omega_{c2} \leq \omega \leq -\omega_{c1} \\ 1 & for \quad \ \ \omega_{c1} \leq \omega \leq \omega_{c2} \\ 0 & Otherwise \end{cases}$$

Using the Inverse Fourier Transform, impulse response of the above $H(e^{j\omega T})$ is calculated.

$$\begin{aligned} h_d(nT) &= \frac{1}{\omega_s}\int_{-\omega_s/2}^{\omega_s/2} H(e^{j\omega T})e^{j\omega nT}\, d\omega \\ &= \frac{1}{\omega_s}\left[\int_{-\omega_{c2}}^{-\omega_{c1}} e^{j\omega nT}\, d\omega + \int_{\omega_{c1}}^{\omega_{c2}} e^{j\omega nT}\, d\omega\right] \\ &= \frac{1}{\omega_s}\left[\frac{e^{j\omega nT}}{jnT}\bigg|_{-\omega_{c2}}^{-\omega_{c1}} + \frac{e^{j\omega nT}}{jnT}\bigg|_{\omega_{c1}}^{\omega_{c2}}\right] \\ &= \frac{1}{j\omega_s nT}\left[e^{-j\omega_{c1}nT} - e^{-j\omega_{c2}nT} + e^{j\omega_{c2}nT} - e^{j\omega_{c1}nT}\right]; where\ \omega_s T = 2\pi \\ &= \frac{1}{\pi n}\left[\frac{(e^{j\omega_{c2}nT} - e^{-j\omega_{c2}nT})}{2j} - \frac{(e^{j\omega_{c1}nT} - e^{-j\omega_{c1}nT})}{2j}\right]; rearanging \\ &= \frac{1}{\pi n}\left[\sin(\omega_{c2}nT) - \sin(\omega_{c1}nT)\right]; from\ Euler's\ Eq. \end{aligned}$$

$$\therefore\ h_d(nT) = \begin{cases} \frac{1}{\pi n}\left[\sin(\omega_{c2}nT) - \sin(\omega_{c1}nT)\right] & \forall n \neq 0 \\ \\ \frac{2}{\omega_s}\left(\omega_{c2} - \omega_{c1}\right) & for\ n = 0 \end{cases} \tag{4}$$

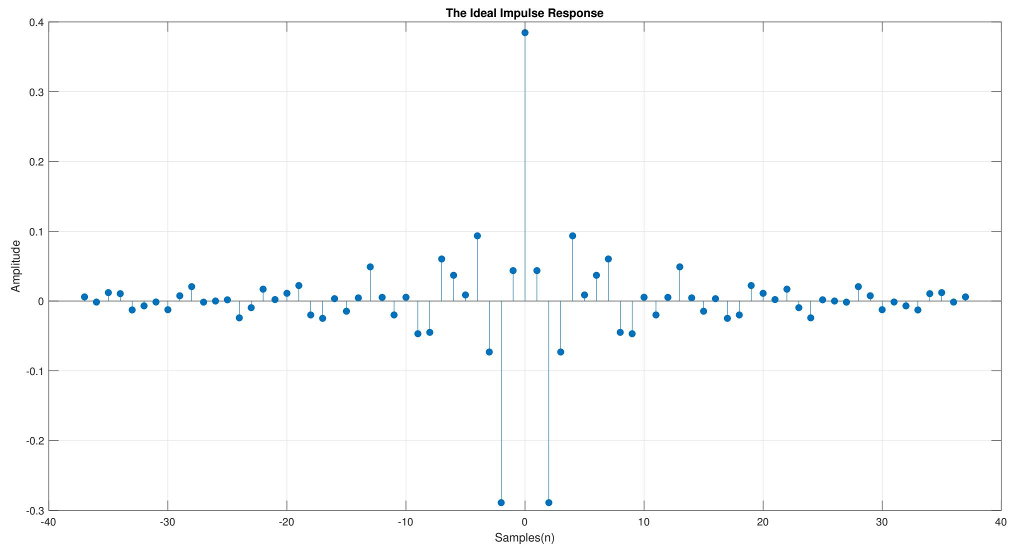

Figure 3: Anti-Causal Ideal Impulse Response $h_d(nT)$

2.1.5 Truncating the Ideal Impulse Response to obtain a Finite Impulse Response

By multiplying the ideal impulse response $h_d(nT)$ in Eq. (4) with the Kaiser window $w_K(nT)$ in Eq. (3), the ideal infinite impulse response can be truncated to obtain the finite impulse response $h(nT)$ for practical implementation.

$$h(nT) = w_K(nT).h_d(nT) \tag{5}$$

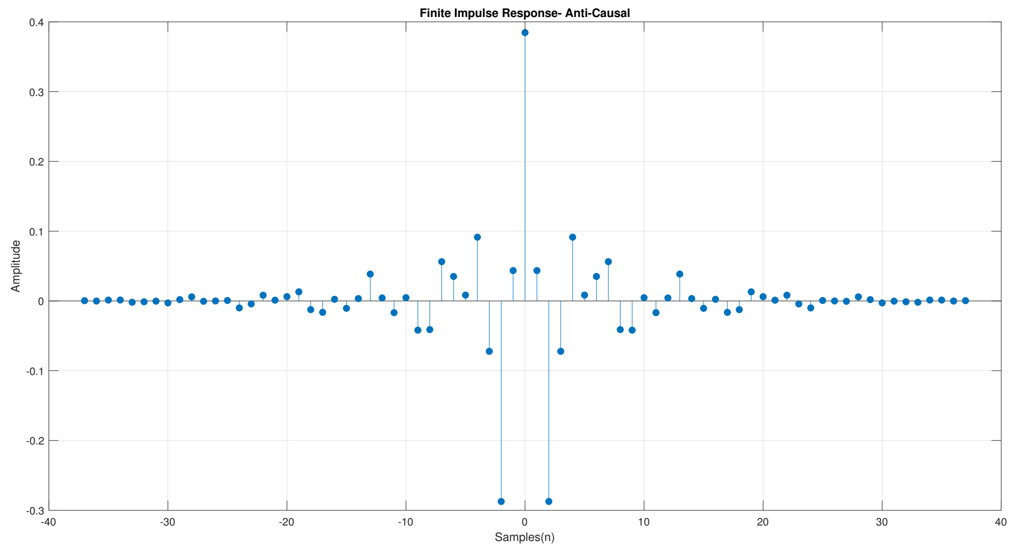

Figure 4: Anti-Causal Finite Impulse Response $h(nT)$

Obtaining the transfer function in the $\mathcal{Z}$ domain using $\mathcal{Z}$ Transformation,

$$\begin{aligned} H'(z) &= \mathcal{Z}\left\{h(nT)\right\} \\ &= \mathcal{Z}\left\{w_K(nT).h_d(nT)\right\} \end{aligned} \tag{6}$$

The response is anti-causal it need to be shifted in the time domain to make it causal and get the final filter response$H(Z)$, in $\mathcal{Z}$ domain it is represented as follows. Then the equation is evaluated at $e^{j\omega}$ to get the frequency response of the filter

$$H(z) = z^{-\left(\frac{N-1}{2}\right)}.H'(z) \tag{7}$$

2.2 Filter Performance Evaluation

Performance of the filter was evaluated by using the following excitation $x(nT)$ which is a combination of three sinusoidal signals. Frequencies of these three sinusoidal signals are specified as follows to cover all three bands(*lower stopband, passband and upper stopband*) in the bandpass filter.

$$x(nT) = \sum_{i=1}^{3} sin(w_i nT)$$

Parameter	**Symbol**	**Calculation**	**Value**
Middle frequency of the lower stopband	ω_1	$\frac{0+\omega_{a1}}{2}$	125 rad/s
Middle frequency of the passband	ω_2	$\frac{\omega_{p1}+\omega_{p2}}{2}$	600 rad/s
Middle frequency of the upper stopband	ω_3	$\frac{\omega_{a2}+\omega_s/2}{2}$	1100 rad/s

Table 3: Frequencies for Filter Performance Evaluation

The required filtered output can be obtained by time domain convolution of the input signal $x(nT)$ with the impulse response $h(nT)$ in Fig. 4. But as mentioned previously the convolution is computationally expensive operation. Therefore filtering is done in the frequency domain which then simplifies into a simple multiplication. For that Discrete Fourier Transform(DFT) of the input signal $x(nT)$ and DFT of $h(nT)$ are obtained through the `fft()` function in MATLAB. Then they are multiplied to get the frequency domain output and `ifft()` is applied on that to get the time domain output signal.

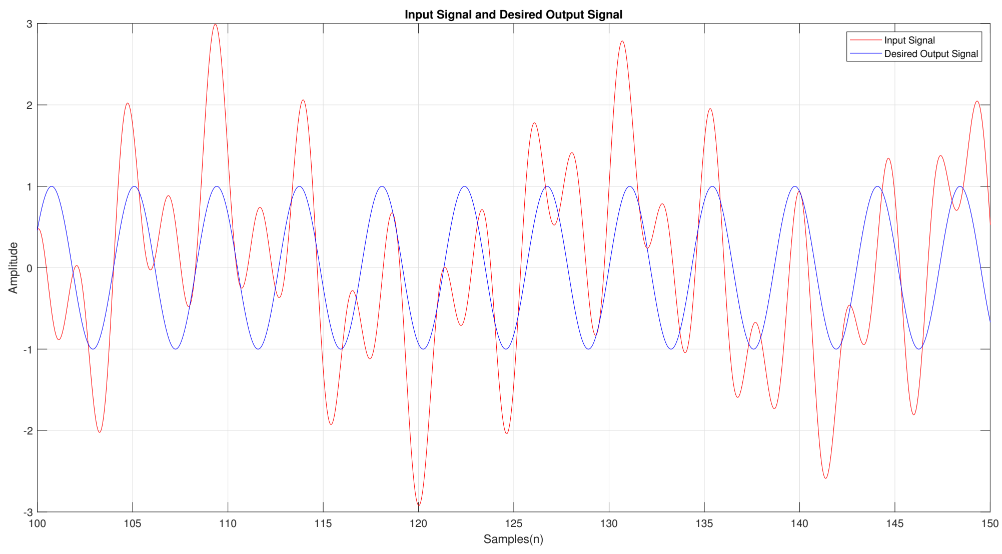

Figure 5: Input Signal and the Desired Output Signal

3 Results

3.1 FIR Filter Characteristics

Filter Characteristics showed in this section was obtained through the procedure explained in the *section 2.1 Filter Implementation.*

Anti-Causal Impulse Response of the Filter can be right shifted in time domain to obtain the Causal Impulse Response as shown below. Note that there, the samples(n) axis starts from 0 and continues up to 74. That is N-1. Whereas previously it was from -37 to 37.

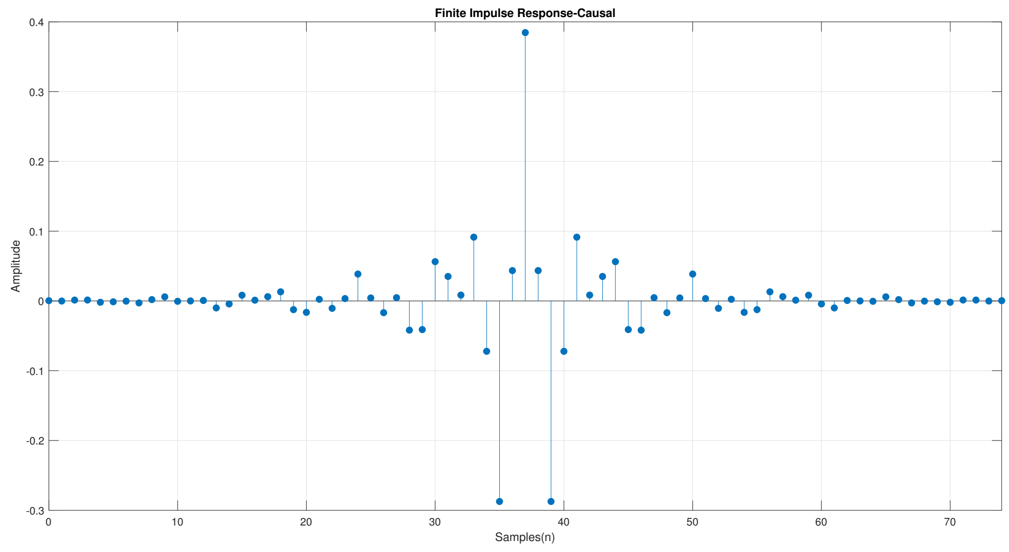

Figure 6: Causal Impulse Response of the Filter

Magnitude Response of the Filter was obtained through the MATLAB's `freqz()` function which computes the frequency, magnitude, and phase response of given digital filter(*impulse response*).

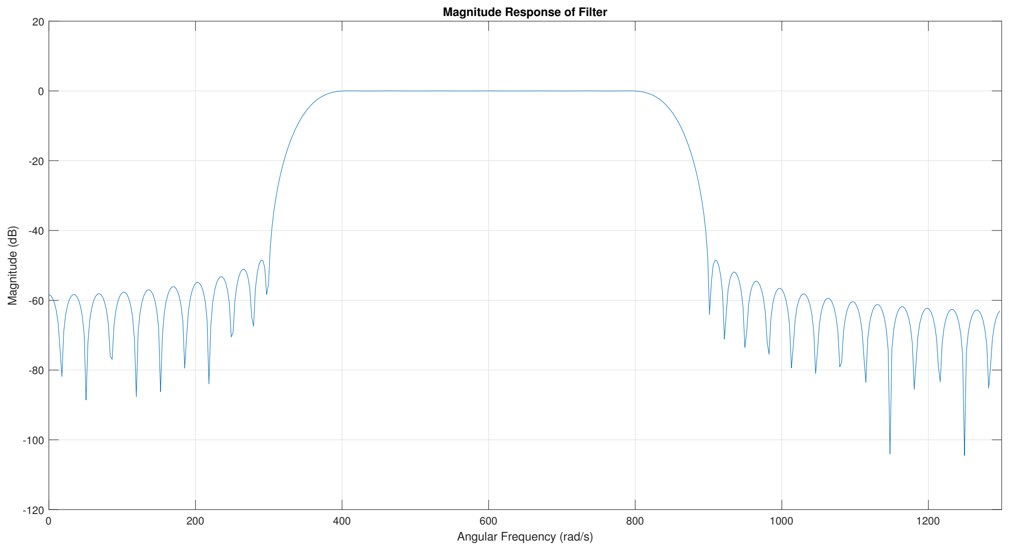

Figure 7: Magnitude Response of the Filter

Magnitude Response of the Filter for the frequencies in the Passband can be simply obtained through limiting the range of Angular Frequency axis using MATLAB's `xlim([`$\omega_{p1}\ \omega_{p2}$`])` function. It allows us to identify the nature of the passband ripples of a kaiser window of order 75. According to the prescribed specifications Maximum passband ripple(*desired*) $\tilde{A}_p$ was 0.09 dB. And we can observe that the designed filter has achieved that goal.

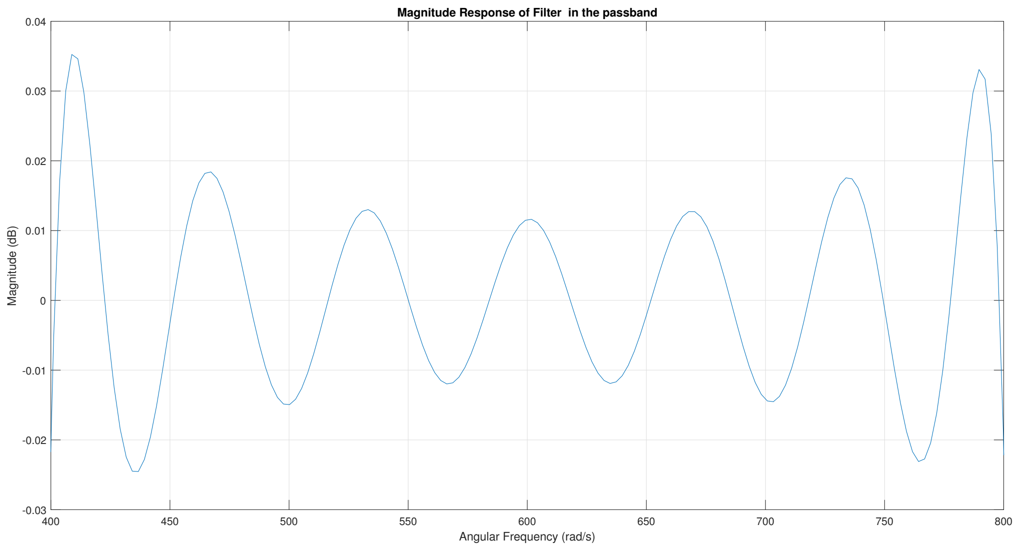

Figure 8: Magnitude Response of the Filter for the frequencies in the Passband

3.2 Input Output Signal Characteristics

As explained in the **Filter Performance Evaluation** section input signal is a combination of three sinusoidal signals which has its frequency components lie in the three bands of the bandpass filter. This fact is clearly visible in the frequency spectrum of the Input Signal.

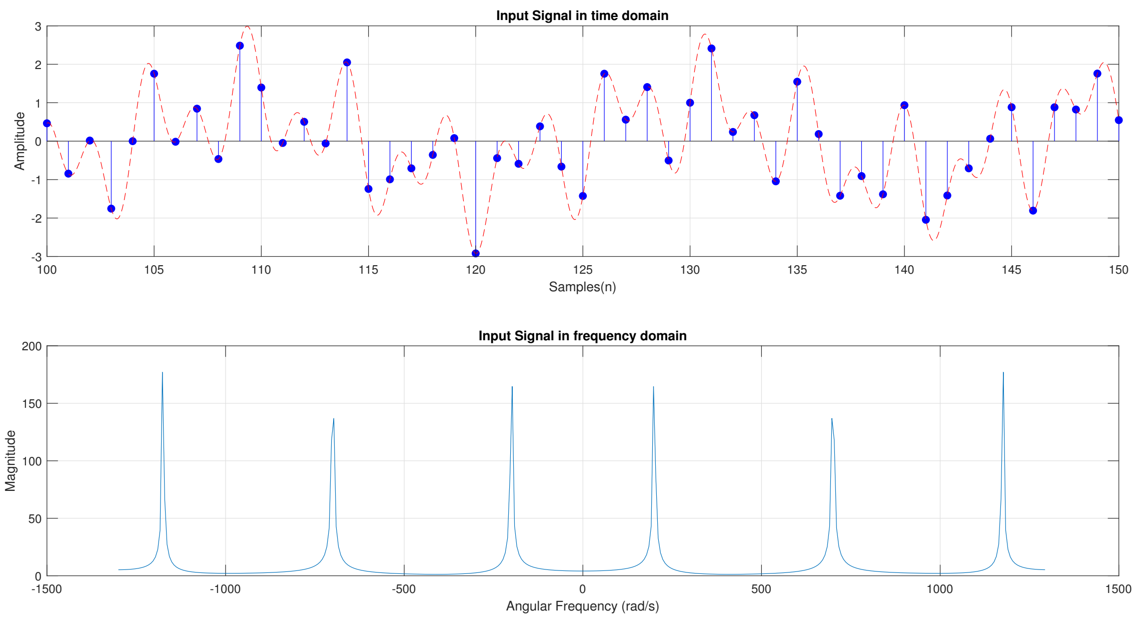

Figure 9: Input signal representation in Time and Frequency Domains

By inspecting the frequency spectrum of the output signal we can observe that output signal is a filtered version of the input signal and that the correct frequencies have been passed through the filter perfectly. Because both the frequencies that were in the lower and upper stop bands are filtered out and disappeared completely from the frequency spectrum of the output signal. This guarantees that the Kaiser window method has yield an almost ideal bandpass filter which satisfies the prescribed specifications.

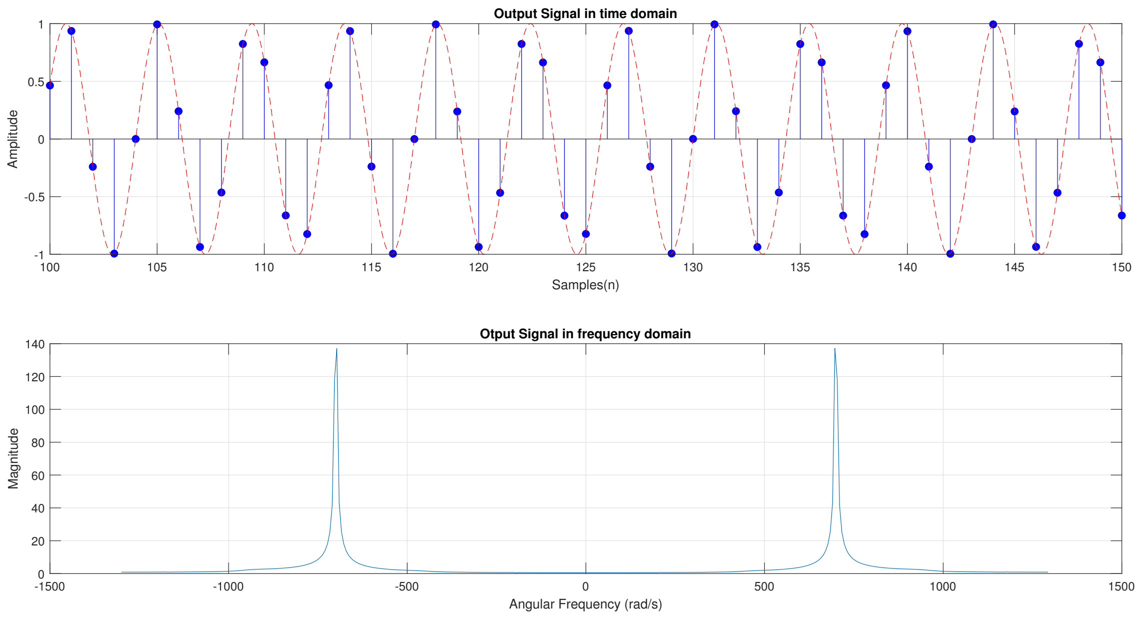

Figure 10: Output signal representation in Time and Frequency Domains

4 Conclusions

Depending on the results it can be concluded that the Kaiser Window method has yielded a near ideal passband filter characteristics which satisfies all the prescribed specifications. Even though there are some ripples in the passband and the stopband, the filter characteristics can be optimized without much effort through tuning the parameters (α and N) to obtain perfect and at the same time practical filter for a given application. In addition to that computational complexity of the parameters which are used to obtain the Kaiser Window is also much less and most importantly there is a clear method to obtain those parameters through empirical formulae.

But when a filter is practically implemented in hardware, usage of the computational resources plays a vital role and as the order of the filter grows resource requirement grows drastically. Therefore, as a major disadvantage of the Kaiser Window method, the high order of the filters can be pointed out. Filter of order 75 was required to achieve the given specs in our case while there are other filter designing methods which yield the same characteristics with much lower order and therefore much lower resource requirements.

Bibliography

[1] Andreas Antoniou. *Digital Signal Processing.* McGraw-Hill Professional, US, 2005.

Appendices

A Matlab Code for the Implementation

```
% Index number = 180631J
% Therefore the required parameters
A = 6;
B = 3;
C = 1;
```

Prescribed Filter specifications

```
tilde_A_p = 0.03+0.01*A;    % Maximum passband ripple
tilde_A_a = 45 + B;         % Minimum stopband attenuation
omega_p1 = C*100 + 300;     % Lower passband edge
omega_p2 = C*100 + 700;     % Upper passband edge
omega_a1 = C*100 + 150;     % Lower stopband edge
omega_a2 = C*100 + 800;     % Upper stopband edge
omega_s  = 2*(C*100 +1200); % Sampling frequency
```

Derivation of filter Parameters

```
B_t1 = omega_p1 - omega_a1; % Lower transition width
B_t2 = omega_a2 - omega_p2; % Upper transition width
B_t  = min(B_t1,B_t2);      % Critical transition width
omega_c1 = omega_p1-B_t/2;  % Lower cutoff frequency
omega_c2 = omega_p2+B_t/2;  % Upper cutoff frequency
T = 2*pi /omega_s;          % Sampling period
```

Derivation of the Kaiser Window Parameters

```
tilde_delta_p  =  (10^(0.05*tilde_A_p) -1)/(10^(0.05*tilde_A_p)  +1);
tilde_delta_a  = 10^(-0.05*tilde_A_a);
delta = min(tilde_delta_p, tilde_delta_a);

A_a = -20*log10(delta);% Actual stopband attenuation

% Choose parameter alpha as,
if A_a <=21
    alpha = 0;
elseif 21 < A_a && A_a <=50
    alpha = 0.5842*(A_a - 21)^0.4 + 0.07886*(A_a - 21);
else
    alpha = 0.1102*(A_a - 8.7);
end

% Choose parameter D as,
if A_a <= 21
    D = 0.9222;
else
    D = (A_a - 7.95)/14.36;
end
```

```
% Select the lowest odd value of N that satisfies the inequality
N = ceil(omega_s*D/B_t + 1);
if mod(N,2) ==0
    N = N+1; % If calculated N is evn, make it odd by adding 1
end
```

Creating the Kaise Window

```
n = -(N-1)/2:1:(N-1)/2;                        % Range of the Kaizer window
beta = alpha*sqrt(1-(2*n/(N-1)).^2);           % betas for I(beta)
terms = 100;                                   % Number of terms to be considered for bessel
w_k_nT = ZerothOrderModifiedBessel(beta,terms)...    % Custom function is defined at the end.
            /ZerothOrderModifiedBessel(alpha,terms); % Calculating window coefficients
stem(n,w_k_nT,'filled');
title('Kaiser Window - Time Domain');
xlabel('Samples(n)');
ylabel('Amplitude');
grid on
```

Derivation of The Ideal Impulse Response

```
h_d_nT = (sin(omega_c2*n*T) - sin(omega_c1*n*T))./(pi*n); % For each n != 0
h_d_nT((N+1)/2) = (omega_c2 - omega_c1)*(2/omega_s);      % For n = 0
stem(n,h_d_nT,'filled');
title('The Ideal Impulse Response');
xlabel('Samples(n)');
ylabel('Amplitude');
grid on
```

Truncating the Ideal Impulse Response to obtain Finite Impulse Response(Anti-Causal)

```
h_nT = h_d_nT.*w_k_nT; % Windowing using Kaiser window
stem(n,h_nT,'filled');
title('Finite Impulse Response- Anti-Causal')
xlabel('Samples(n)');
ylabel('Amplitude');
grid on
```

Finite Impulse Response(Causal)

```
n_causal = 0:1:N-1; % Making the range of n positive
stem(n_causal,h_nT,'filled');
xlim([0 N-1]);
title('Finite Impulse Response-Causal')
xlabel('Samples(n)');
ylabel('Amplitude');
```

```
grid on;
```

Plotting the magnitude response of filter in the range (0,omega_s/2)

```
%fvtool(h_nT,'magnitude')
[H_ejomegaT, omega] = freqz(h_nT); %compute the frequency, magnitude, and phase response
omega = (omega/pi)*(omega_s/2);          % rad/s = (normalized freq)*(sampling freq/2)
magnitude = 20*log10(abs(H_ejomegaT));  % converting the magnitude into dB
plot(omega, magnitude);
xlim([0 omega_s/2]);
title('Magnitude Response of Filter');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude (dB)');
grid on;
```

Magnitude response of the digital filter for the frequencies in the passband

```
plot(omega, magnitude);
xlim([omega_p1 omega_p2]);  % Passband = [Lower passband edge, Upper passband edge]
title('Magnitude Response of Filter  in the passband');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude (dB)');
grid on;
```

Input Signal Generation

```
omega_1 =  (omega_a1 + 0)/2;         % Middle frequency of the lower stopband
omega_2 =  (omega_p1 + omega_p2)/2;  % Middle frequency of the passband
omega_3 =  (omega_a2 + omega_s/2)/2; % Middle frequency of the upper stopband

samples = 400;      % To achieve a steady-state response we need much samples
n1 = 0:1:samples;   % Discrete Values for sampling
n2 = 0:0.01:samples;% Near Continuous values for envelope drawing

x_nT = sin(omega_1*n1*T) +  sin(omega_2*n1*T) +sin(omega_3*n1*T); % Sampled I/P signal
x_t  = sin(omega_1*n2*T) +  sin(omega_2*n2*T) +sin(omega_3*n2*T); % I/P signal envelope

%================ Visulaization of Input Signal in time domain ================
subplot(2,1,1)
stem(n1, x_nT,'filled', 'b');
title('Input Signal in time domain');
xlabel('Samples(n)');
ylabel('Amplitude');
hold on;
plot(n2, x_t,'--','Color','r');xlim([100 150]);

%============= Visulaization of Input Signal in frequency domain ==============
X_f = fft(x_nT);                         % Discrete Fourier Transform of the input signal
X_ff = ((n1/length(n1))*omega_s)-omega_s/2; % map the range into (-omega_s to omega_s)
```

```
subplot(2,1,2)
plot(X_ff, abs(X_f));
title('Input Signal in frequency domain');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude');

subplot(2,1,1)
grid on
subplot(2,1,2)
grid on
```

Filtering through frequency domain multiplication

```
% Convolution in time domain is computationally expensive.
% Therefore frequency domain multiplication is used in this implementation.

% x_nT and h_nT have different lenghts,
% Therefore we need to get their fft outputs to a common lenghts.
% In order to multiply them element wise matrix dimensions must agree.
n_points = length(x_nT) + length(h_nT) -1;

X_f = fft(x_nT, n_points); % Discrete Fourier Transform of the input signal
H_f = fft(h_nT, n_points); % Discrete Fourier Transform of the FIR bandpass filter
Y_f = X_f.*H_f;            % Filtering through freq domain multiplication
y_nT = ifft(Y_f,n_points); % Inverse Fourier Transform to get the output signal.

% Trunctaing to the required length to account added time shift
y_nT = y_nT(floor(N/2)+1:length(y_nT)-floor(N/2));
% Output of an Ideal Filter
y_t = sin(omega_2*n2*T);
```

Analyzing Output Signal

```
% ============Visulaization of Output Signal in time domain===============
figure;
subplot(2,1,1)
stem(n1, y_nT,'filled', 'b');
title('Output Signal in time domain');
xlabel('Samples(n)');
ylabel('Amplitude');
hold on;
plot(n2, y_t,'--','Color','r');xlim([100 150]);

% ===========Visulaization of Output Signal in frequency domain===========
Y_f = fft(y_nT);                          % Discrete Fourier Transform of the output signal
Y_ff = ((n1/length(n1))*omega_s)-omega_s/2; % map the range into (-omega_s to omega_s)

subplot(2,1,2)
plot(Y_ff, abs(Y_f));
```

```
title('Otput Signal in frequency domain');
xlabel('Angular Frequency (rad/s)');
ylabel('Magnitude');

subplot(2,1,1)
grid on
subplot(2,1,2)
grid on
```

Input Signal and Desired Output Signal

```
figure;
plot(n2, x_t,'Color','r')
hold on
plot(n2, y_t,'Color','b')
legend('Input Signal', 'Desired Output Signal')
xlim([100 150]);
title('Input Signal and Desired Output Signal')
xlabel('Samples(n)')
ylabel('Amplitude')
grid on
```

Local Function definitions

```
% function to calculate zeroth-order modified Bessel function of the first kind
% Upto a given terms.
```

```
function value = ZerothOrderModifiedBessel(x,terms)
value = 1;
for k = 1:terms
    value = value + ((1/factorial(k))*(x/2).^k).^2;
end
end
```