

Department of Electronic and Telecommunication Engineering

University of Moratuwa

EN2570 - Digital Signal Processing



FIR Filter Design Report

Submitted by

SIRITHUNGA M.R.A.

180609B

Submitted on

05/03/2021

Contents

Contents	1
List of Figures	3
1 Abstract	4
2 Introduction	5
3 Basic Theories	6
3.1 Design Using the Fourier Series	6
3.2 Use of Window Functions	7
3.2.1 Kaiser Window	8
3.3 Design of Prescribed Bandpass Filter	8
3.3.1 Determine the Impulse Response.	10
3.3.2 Choose the Delta	10
3.3.3 Chose the Alpha	10
3.3.4 Determine the parameter D	11
3.3.5 Apply the Kaiser Window	11
4 Results	12
5 Conclusion	17

Bibliography	18
A Appendix	19
A.1 Matlab Code	19

List of Figures

3.1	Bandpass filter parameters	9
4.1	Impulse response.	13
4.2	Bandpass kaiser window.	13
4.3	Impulse response of the kaiser window	14
4.4	Passband, lower stopband and upper stopband	14
4.5	Ideal outputs	15
4.6	sample signal's filtered output.	15
4.7	Filter output in the frequency domain	16

Abstract

This project deals with the designing of the finite impulse response (FIR) bandpass filter. The FIR filter is designed using windowing method in conjunction with the kaiser window. The corresponding theories are going to be explained according to the prescription that is provided. Matlab R2018a is used for demonstrations and, the figures shown below are the corresponding outputs given by the Matlab R2018a. The codes are included in the appendix.

Introduction

A filter is a component which eliminates the undesired parts of the input signal with respect to the users requirements. The digital filters can be seperated by considering the impulse response as[1],

- finite impulse response (FIR) filters and
- infinite impulse response (IIR) filters.

In this report it is going to deal with a FIR bandpass filter in conjunction with the kaiser window. Kiser window method is one of the most fascinating techniques.

Basic Theories

This chapter is cited: A.Antoniou's book [1].The design process of the finite impulse response filters can be categorise into two classical methods.

1. Fourier series method or Window method.
2. Weighted-Chebyshev method.

The fourier series is used in the first method and it will be discussed further with the basic theories later. The weighted-chebyshev method is multivariable optimization method. In order to that window functions are conjuncted with the fourier series method.

3.1 Design Using the Fourier Series

The digital filters have a periodic frequency response. The period is equal to the sampling frequency hence, the frequency response can be denoted as a fourier series as below.

$$H(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} h(nT)e^{-j\omega nT}$$
$$h(nT) = \frac{1}{\omega_s} \int_{-\omega_s/2}^{\omega_s/2} H(e^{j\omega T})e^{j\omega nT} d\omega$$

By substituting $z=e^{j\omega T}$ to the impulse response of the filter, transfer function can be obtained.

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

There are two major consequences with this fourier series method that can be shown because, the fourier series coefficients have been defined over the range -INF to +INF.

- The nonrecursive filter obtained is of infinite length.
- The filter is noncausal because the impulse response is nonzero for negative time.

To overcome these problems, the impulse response has been truncated. And a causal filter can be obtained by delaying the impulse response or by sampling periods appropriately.

$$h(nT) = 0 \quad \text{for } |n| > \frac{N-1}{2}$$

$$H(z) = h(0) + \sum_{n=1}^{(N-1)/2} [h(-nT)z^n + h(nT)z^{-n}]$$

3.2 Use of Window Functions

The oscillations can be found in the passband and the stopband. It is because of the slow convergence of the fourier series. These are called Gibb's oscillations and the effect of Gibb's oscillations[2] can be reduced by the complex-convolution with a discrete-time window function.

$$h_w(nT) = w(nT)h(nT)$$

$$H_w(z) = \mathcal{Z}[w(nT)h(nT)] = \frac{1}{2\pi j} \oint_{\Gamma} H(v)W\left(\frac{z}{v}\right)v^{-1}dv$$

The above equation can be interpreted in such a way that illustrates the effect of the window spectrum on the frequency response of the filter.

$$H_w(e^{j\omega T}) = \frac{T}{2\pi} \int_0^{2\pi/T} H(e^{j\varpi T})W(e^{j(\omega-\varpi)T})d\varpi$$

A variety of such windows has been observed and used over the years. Some significant and

common windows are listed below.

1. Rectangular.
2. von Hann2.
3. Hamming.
4. Blackman.
5. Dolph-Chebyshev.
6. Kaiser.

3.2.1 Kaiser Window

Kaiser window is an one of the most widely used window in fir filter designing. It can be easily achieved the prescribed specifications by using the kaiser window method. This is cited:[3]. The window function is given by,

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

where alpha is an independent parameter. The zero-th order, first kind of Bessel function is denoted by the I_0 . The value of beta and the bessel function are given in the following manner.

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

3.3 Design of Prescribed Bandpass Filter

The designing procedure of the prescribed filter by using kaiser window will be explained. The required parameters would give as below.

- Passband Ripple $\leq \tilde{A}_p$
- Minimum Stop Band Attenuation $\geq \tilde{A}_a$
- Lower passband edge ω_{p1}
- Lower stopband edge ω_{a1}
- Upper stopband edge ω_{a2}
- Upper passband edge ω_{p2}
- Sampling frequency ω_s

The above parameters are demonstrated by using a graph for further clarifications in the figure.

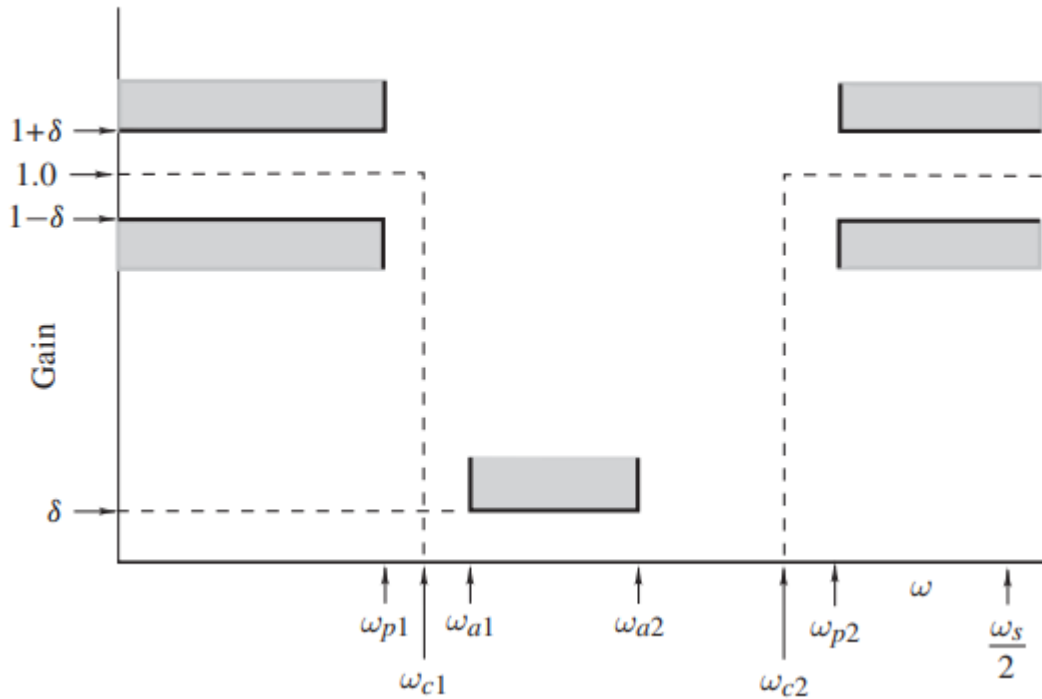


Figure 3.1: Bandpass filter parameters

Now, the design of prescribed bandpass filter design can be started.

3.3.1 Determine the Impulse Response.

The minimum transition width B_t is given by the first equation. The frequency response of a band pass filter is given below. The impulse response can be determined by applying the fourier series. The lower and upper cutoff frequencies approximate as follow.

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

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

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

The impulse response for a bandpass filter can be determined as below.

$$h(nT) = \frac{1}{n\pi}(\sin \omega_{c2}nT - \sin \omega_{c1}nT)$$

3.3.2 Choose the Delta

Delta can be calculated using the following equations. Then the actual stopband loss

$$\delta = \min(\tilde{\delta}_p, \tilde{\delta}_a)$$

$$\tilde{\delta}_p = \frac{10^{0.05\tilde{A}_p} - 1}{10^{0.05\tilde{A}_p} + 1} \quad \text{and} \quad \tilde{\delta}_a = 10^{-0.05\tilde{A}_a}$$

of the filter can be determined.

3.3.3 Chose the Alpha

The independent variable alpha should determine at this stage. The following conditions must consider when calculating the alpha.

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

3.3.4 Determine the parameter D

It is the last parameter to determine before move in to the kaiser window calculations. The parameter D can be obtained after checking these conditions.

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

The N must satisfy the following inequality as well.

$$N \geq \frac{\omega_s D}{B_t} + 1$$

3.3.5 Apply the Kaiser Window

Finally, compute the kaiser window by using the above determined parameters. It is the last step and the filter has been designed.

$$H'_w(z) = z^{-(N-1)/2} H_w(z) \quad \text{where } H_w(z) = \mathcal{Z}[w_K(nT)h(nT)]$$

Results

The given parameters and their calculated values are showing in the table. The index is taken as a user input and calculateed the rest by using Matlab R2018a installed version.

Parameter	Value
Passband Ripple	0.0900 dB
Minimum Stop Band Attenuation	45 dB
Lower passband edge	1200 rad/s
Lower stopband edge	1600 rad/s
Upper stopband edge	1050 rad/s
Upper passband edge	1700 rad/s
Sampling frequency	4200 rad/s

The required impulse response and its frequency domain representation is showing in the figure.

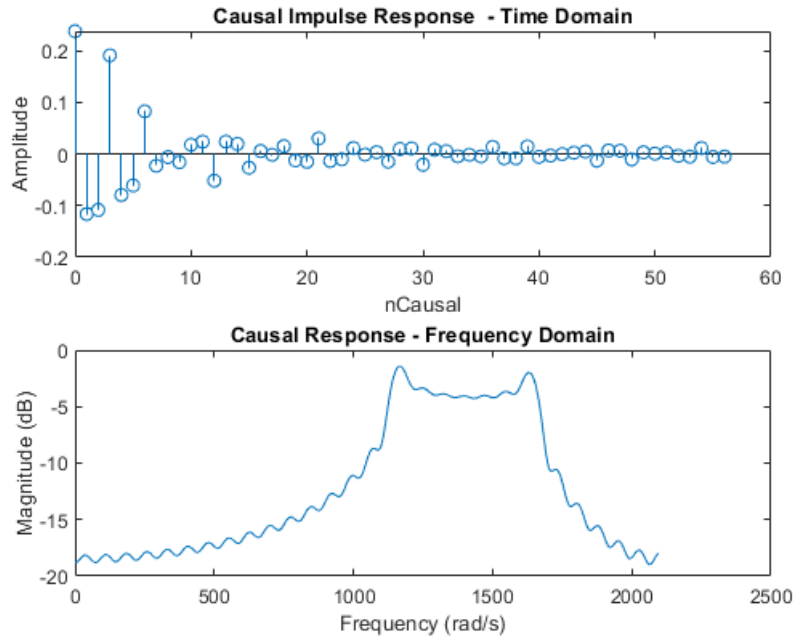


Figure 4.1: Impulse response.

The time domain representation of the bandpass kaiser window is showing in the figure.

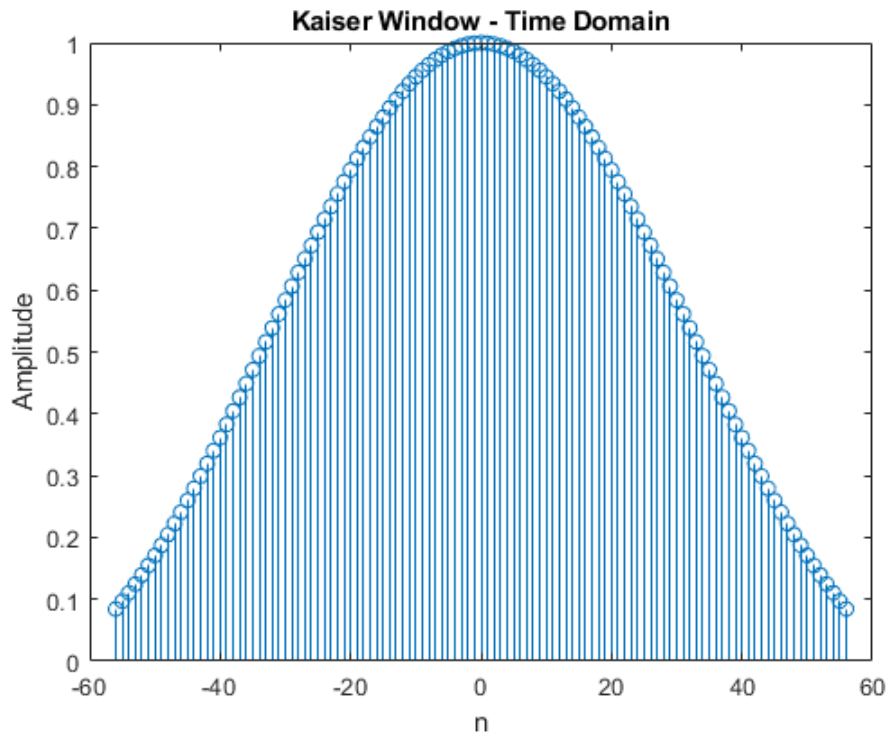


Figure 4.2: Bandpass kaiser window.

The impulse response of the kaiser window and its frequency domain representation are

showing in the figure.

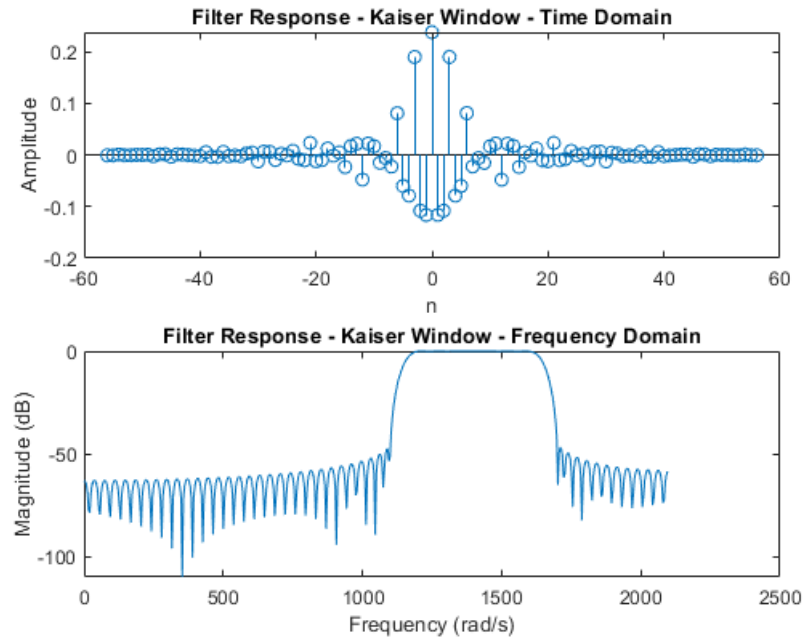


Figure 4.3: Impulse response of the kaiser window

The figure is showing the passband, lower stopband and upper stopband in the frequency domain.

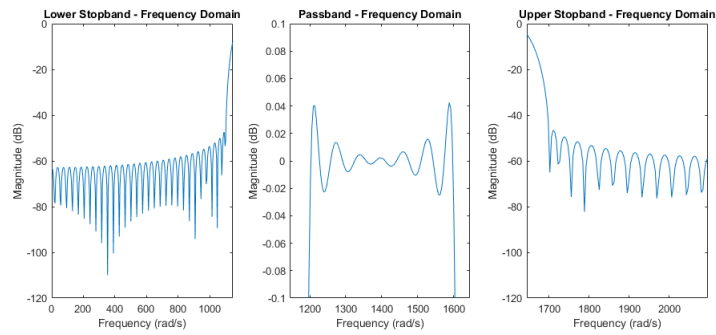


Figure 4.4: Passband, lower stopband and upper stopband

The figure is showing the ideal outputs of the filter.

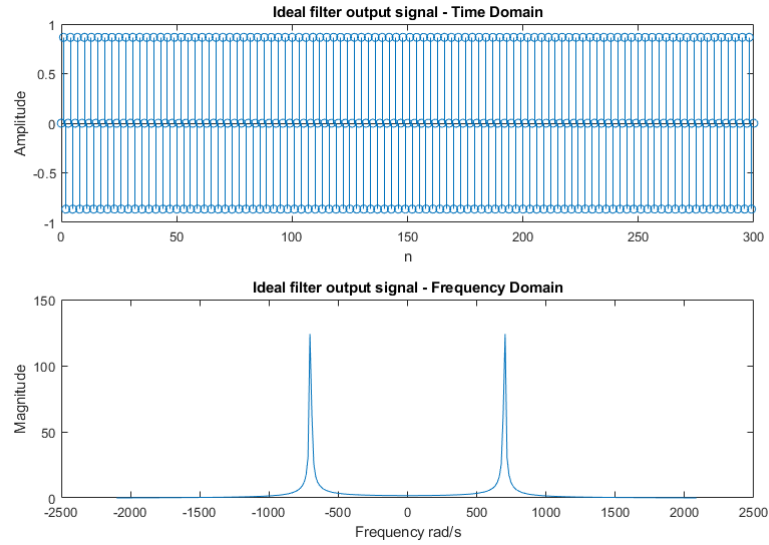


Figure 4.5: Ideal outputs

Finally, demonstrate the filter behaviour by using a sample signal and the output is showing in the figure.

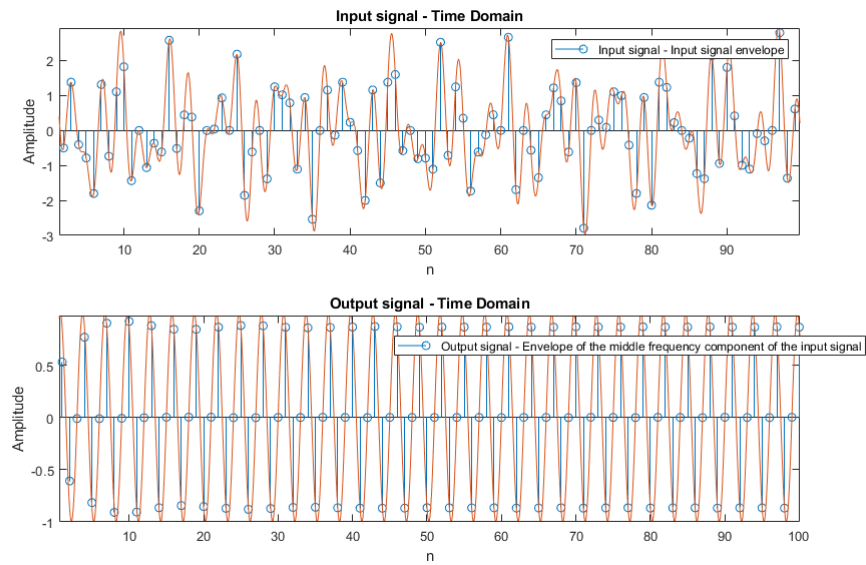


Figure 4.6: sample signal's filtered output.

The filter output in the frequency domain.

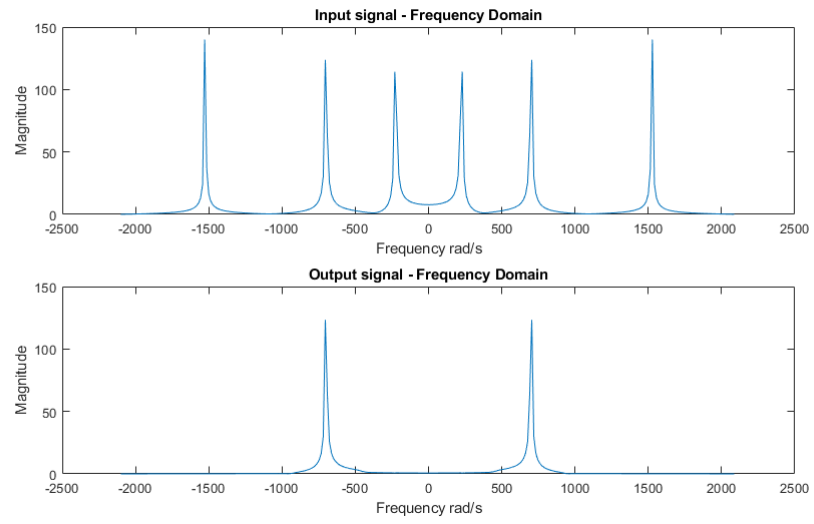


Figure 4.7: Filter output in the frequency domain

Conclusion

Kaiser Window technique is an effective method of planning a FIR filter under the closed form direct methodology, because of its adaptability, the design can oblige many plan prerequisites that are determined on the filter. It is additionally seen that the computational exertion of designing a filter in this strategy is little. Anyway the major downside of this filter configuration is the high order of filter required. There can be ideal filters of lower request that can accomplish similar determinations. At the point when actualized in equipment higher request filters can be wasteful because of the use of more unit delay adders and multipliers. In the product usage, higher order filters require more calculations per test. Further developed plan procedures must be utilized all together to get optimality.

Bibliography

- [1] A. Antoniou. "*Digital Signal Processing*," 2005 [Online].. Available: [www.ece.uvic.ca/ dsp](http://www.ece.uvic.ca/dsp).
- [2] K. Deergha Rao, M.N.S. Swamy *Digital Signal Processing*.. Available: <https://link.springer.com/>
- [3] UKEssays. (November 2018). Filter Design Using Rectangular Window And Kaiser Window Marketing Essay.
<https://www.ukessays.com/essays/marketing/filter-design-using-rectangular-window-and-kaiser-window-marketing-essay.php?vref=11>

Appendix

A.1 Matlab Code

```
1 close all;
2 clear;
3 clc
4
5 %prescribed parameters A,B, and C are calculated by the user input
6
7 indexIn = inputdlg ( ' Enter Index Number' , 'Index Number' , 1 ) ;
8 indexNo = str2double (indexIn{1}(1:end-1));
9 C = mod(indexNo , 10 ) ;
10 B = mod( floor(indexNo/10) ,10 ) ;
11 A = mod( floor(indexNo/100) ,10 ) ;
12
13 %parameters for the Bandpass filter
14
15 A_tilda_p=0.03+0.01*A;%Maximum passband ripple-:0.0900
16 A_tilda_a=45+B;%Minimum stopband attenuation-:45
17 wp1=C*100+300;%Lower passband edge-:1200
18 wp2=C*100+700;%Upper passband edge-:1600
19 wa1=C*100+150;%Lower stopband edge-:1050
20 wa2=C*100+800;%Upper stopband edge-:1700
21 ws=2*(C*100+1200);%Sampling frequency-:4200
22
23 % Derived Specifications
24
25 bt1 = wp1-wa1; %lower transition width
26 bt2 = wa2-wp2; % upper transisiton width
27 bt = min(bt1,bt2); %critical transition width
28 wc1 = wp1-bt/2; % lower cutoff frequency
29 wc2 = wp2+bt/2; % upper cutoff frequency
30 T = 2*pi/ws; % sampling period
31
32 % Kaiser Window Parameters
33
34 deltaP = (10^(0.05*A_tilda_p) - 1)/ (10^(0.05*A_tilda_p) + 1); % calculating
```

```

    delta
35 deltaA = 10^(-0.05*A_tilda_a);
36 delta = min(deltaP,deltaA);
37
38 Aa = -20*log10(delta); % Actual stopband attenuation
39
40 if Aa<=21 % Calculating alpha
41     alpha = 0;
42 elseif Aa>21 && Aa<= 50
43     alpha = 0.5842*(Aa-21)^0.4 + 0.07886*(Aa-21);
44 else
45     alpha = 0.1102*(Aa-8.7);
46 end
47
48 if Aa <= 21 % Calculating D
49     D = 0.9222;
50 else
51     D = (Aa-7.95)/14.36;
52 end
53
54 N = ceil(ws*D/bt +1); % order of the filter
55 if mod(N,2) == 0
56     N = N+1;
57 end
58
59 n = -(N-1)/2:1:(N-1)/2; % length of the filter
60
61 beta = alpha*sqrt(1-(2*n/(N-1)).^2);
62
63 % Generating I(alpha)
64
65 bessellimit = 50;
66
67 Ialpha = 1;
68 for k = 1:bessellimit
69     termk = (1/factorial(k)*(alpha/2).^k).^2;
70     Ialpha = Ialpha + termk;
71 end
72
73 % Generating I(beta) %%
74
75 Ibeta = 1;
76 for k = 1:bessellimit
77     termk = (1/factorial(k)*(beta/2).^k).^2;
78     Ibeta = Ibeta + termk;
79 end
80
81 % Obtaining Kaiser Window
82
83 wknt = Ibeta/Ialpha;
84

```

```

85 figure
86 stem(n,wknt)
87 xlabel('n')
88 ylabel('Amplitude')
89 title('Kaiser Window - Time Domain');
90
91 % Generating Impulse Response
92
93 nleft = -(N-1)/2:-1;
94 hntleft = 1./(nleft*pi).*(sin(wc2*nleft*T)-sin(wc1*nleft*T));
95
96 nright = 1:(N-1)/2;
97 hntright = 1./(nright*pi).*(sin(wc2*nright*T)-sin(wc1*nright*T));
98
99 hnt0 = 2/ws*(wc2-wc1);
100
101 hnt = [hntleft,hnt0,hntright];
102 figure
103 stem(n,hnt)
104 xlabel('n')
105 ylabel('Amplitude')
106 title(strcat('Filter Response - Rectangular window - Time Domain'));
107
108 figure
109 [hi_f,wi_f] = freqz(hnt);
110 wi_f = wi_f/T;
111 hi_f = 20*log10(abs(hi_f));
112 plot(wi_f,hi_f)
113 xlabel('Frequency (rad/s)')
114 ylabel('Magnitude (dB)')
115 title(strcat('Filter Response - Rectangular Window - Frequency Domain'));
116
117 % Applying the window to the filter
118
119
120 filter = hnt.*wknt;
121 figure
122 stem(n,filter)
123 xlabel('n')
124 ylabel('Amplitude')
125 title(strcat('Filter Response - Kaiser Window - Time Domain'));
126
127 figure
128 [h,w] = freqz(filter);
129 w = w/T;
130 h = 20*log10(abs(h));
131 plot(w,h)
132 xlabel('Frequency (rad/s)')
133 ylabel('Magnitude (dB)')
134 title(strcat('Filter Response - Kaiser Window - Frequency Domain'));
135

```

```

136 % Plotting the Passband %%
137
138 figure
139 start = round(length(w)/(ws/2)*wc1);
140 finish = round((length(w)/(ws/2)*wc2));
141 wpass = w(start:finish);
142 hpass = (h(start:finish));
143 plot(wpass,hpass)
144 axis([-inf, inf, -0.1, 0.1]);
145 xlabel('Frequency (rad/s)')
146 ylabel('Magnitude (dB)')
147 title('Passband - Frequency Domain');
148
149
150
151 % Input signal generation %%
152
153 w1 = wc1/2;                                %% component frequencies of the input
154 w2 = wc1 + (wc2-wc1)/2;
155 w3 = wc2 + (ws/2-wc2)/2;
156 wi = [w1,w2,w3];
157
158 n1 = 0:1:500;                                %% x axis for discrete signal
159 n2 = 0:0.1:500;                                %% x axis for envelope
160
161 xnt = 0;
162 xdash =0;
163
164 for each = wi
165     xnt = xnt+ sin(each.*n1.*T);    %% generate discrete signal
166     xdash = xdash+sin(each.*n2.*T); %% generate envelope
167 end
168
169
170 % Using DFT to check the filtering %%
171
172 % Filtering using frequency domain multiplication - see function getfiltered.m
    %%
173
174 Npoint = length(xnt) + length(filter) - 1; % length for fft in x dimension
175 xfft = fft(xnt,Npoint);
176 filterfft = fft(filter,Npoint);
177 outfft = filterfft .* xfft;
178 out = ifft(outfft,Npoint);
179
180 out1 = out(floor(N/2)+1:length(out)-floor(N/2)); % account for shifting delay
181
182
183 % Frequency domain representation of input signal before filtering
184
185 figure

```

```

186 subplot(2,1,1)
187
188 Npoint = length(xnt);
189 xfft = fft(xnt,Npoint);
190 x1 = n1/length(n1)*ws-ws/2;
191 plot(x1,abs(xfft))
192 xlabel('Frequency rad/s')
193 ylabel('Magnitude')
194 title(strcat(['Input signal',' ','- Frequency Domain']));
195
196 % Frequency domain representation of input signal after filtering
197
198 subplot(2,1,2)
199 Npoint = length(out1); % length for fft in x dimension
200 xffout = fft(out1,Npoint);
201 x1 = n1/length(n1)*ws-ws/2;
202 plot(x1,abs(xffout))
203 xlabel('Frequency rad/s')
204 ylabel('Magnitude')
205 title(strcat('Output signal',' ','- Frequency Domain'));
206
207
208
209 % time domain representation of input signal before filtering
210
211 figure
212 subplot(2,1,1)
213 stem(n1,xnt)
214 xlabel('n')
215 ylabel('Amplitude')
216 title(strcat(['Input signal',' ','- Time Domain']));
217 hold on
218 plot(n2,xdash)
219 legend('Input signal','Input signal envelope');
220
221
222
223 % Time domain representation of input signal after filtering
224
225 subplot(2,1,2)
226 stem(n1,out1)
227 xlabel('n')
228 ylabel('Amplitude')
229 title(strcat('Output signal',' ','- Time Domain'));
230
231 hold on
232 plot(n2,sin(w2.*n2.*T))
233 legend('Output signal','Envelope of the middle frequency component of the input
    signal');

```