

SIO 207A Final Project Report

Ruipu Ji

I. Data Set

This report provides an implementation of the complex basebanding and Fast Fourier Transform (FFT) architecture as shown in Figure 1. A discrete-time signal $x(n)$ with 4096 data points ($n = 0, 1, 2 \dots 4095$) is selected for analysis. The discrete-time signal $x(n)$ has the analog form $x(t)$ as defined in the following equation:

$$x(t) = \sum_{l=1}^3 A_l \cos(2\pi f_l t + \phi_l)$$

where the parameters are defined in Table 1. The sampling frequency of the signal is $f_s = 1000$ Hz.

Table 1. Parameters used in the analog form $x(t)$ for the discrete-time signal $x(n)$.

l	A_l	f_l	ϕ_l
1	100	160	0
2	10	237	0
3	1	240	0

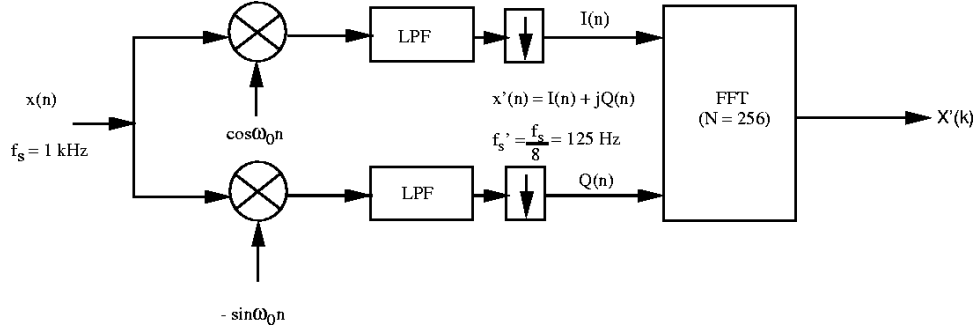


Figure 1. Complex basebanding and Fast Fourier Transform (FFT) architecture.

The first 256 data points of the discrete-time signal $x(n)$ are plotted in Figure 2. The logarithmic magnitude of its Fourier Transform using a Kaiser-Bessel window ($\alpha=2.5$ or $\beta=7.85$) with $N_{FFT} = 256$ is also presented in Figure 2. The analog frequency bin width of the FFT is 3.906 Hz, which is calculated based on the procedures as presented in Appendix 1. The bin indices for each analog frequency component are summarized in Table 2. These analog frequency components reside in 4 different frequency bins, which correspond to the 4 different peaks in the FFT magnitude plot. Because of the large analog frequency bin width (3.906 Hz), frequency components f_2 and f_3 both reside in bin #61. The two frequency components cannot be identified separately, so only one peak is observed for frequency components f_2 and f_3 (similarly $-f_2$ and $-f_3$ are also observed as one peak). Given the sampling frequency of 1000 Hz and $N_{FFT} = 256$, the frequency bin indices for f_2 and f_3 are always 61, which is not affected by the window applied to the signal. Therefore, rectangular windowed FFT will not indicate distinct spectral peaks for f_2 and f_3 .

Table 2. Frequency components of the original signal.

f_l [Hz]	Bin Index	$ X(f_l) $ [dB]	f_l [Hz]	Bin Index	$ X(f_l) $ [dB]
160	41	0	-160	-41	0
237	61	-20	-237	-61	-20
240	61	-20	-240	-61	-20

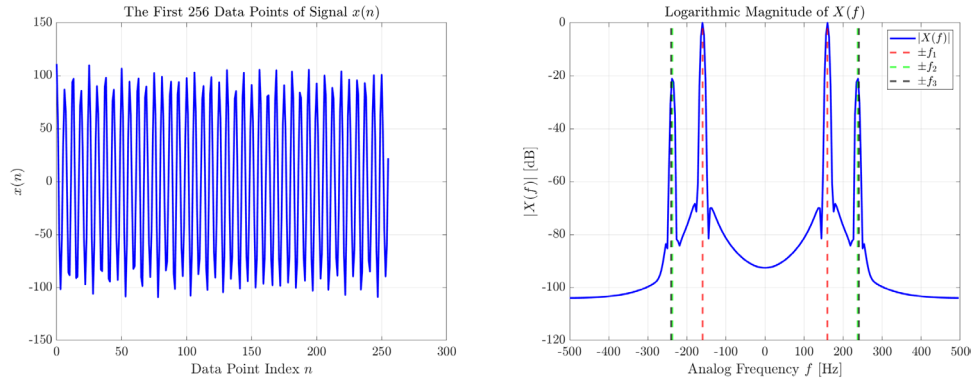


Figure 2. The first 256 data points of signal $x(n)$ and the logarithmic magnitude of its Fourier Transform $X(f)$.

II. Decimation Filter Design

A decimation filter is designed based on the equiripple FIR filter design algorithm. The filter is a 64-coefficient linear phase FIR low-pass filter with passband cutoff frequency of 40 Hz and stopband cutoff frequency of 85 Hz. The passband/stopband weight ratio of the filter is 50. Figure 3 shows the impulse response $h(n)$ and frequency response $H(f)$ of the decimation filter. The frequency response $H(f)$ is calculated by applying the Fourier Transform on the impulse response $h(n)$ using a rectangular window with $NFFT = 1024$. An expanded view of $H(f)$ with analog frequency $-40 \text{ Hz} \leq f \leq 40 \text{ Hz}$ (within passband frequency range) is also presented in Figure 3. Smooth magnitude is observed for the passband magnitude. The magnitude of the passband ripple is less than 4 dB.

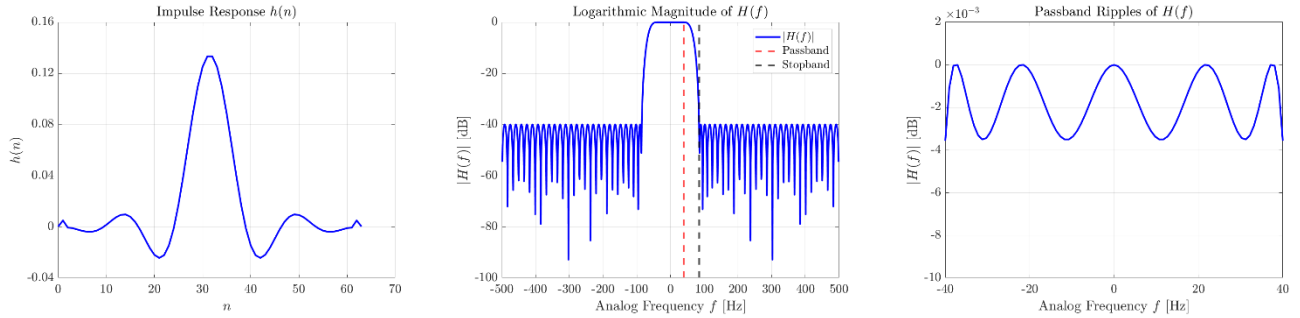


Figure 3. Impulse response and frequency response of the FIR low-pass filter.

III. Complex Basebanding and Desampling

The complex multiplication $e^{j\omega n}x(n)$ is implemented where $\omega = 2\pi(f_0/f_s)$ and the center frequency $f_0 = 250 \text{ Hz}$. The Fourier Transform of the first 256 data points in new complex sequence is calculated using a Kaiser Bessel window ($\alpha=2.5$ or $\beta=7.85$) with $NFFT = 256$. The logarithmic magnitude of the Fourier Transform is shown in Figure 4 (left). The frequency components for the signal after complex basebanding are presented in Table 3. It is observed that the frequency component after complex basebanding f_l' can be calculated from the frequency component of the original signal f_l and the center frequency f_0 based on the following equation (detailed calculations are presented in Appendix 1):

$$f_l' = f_l - f_0$$

The mathematical derivation of this frequency shift is presented as follows.

Signal after complex basebanding:

$$\tilde{x}(n) = x(n) \cdot e^{-j2\pi(f_0/f_s)n}$$

Fourier Transform of the original signal with frequency component f_i :

$$X(f_i) = \sum_{n=-\infty}^{\infty} x(n) \cdot e^{-j2\pi(f_i/f_s)n}$$

Fourier Transform of the signal after complex basebanding:

$$\begin{aligned} \tilde{X}(f) &= \sum_{n=-\infty}^{\infty} \tilde{x}(n) \cdot e^{-j2\pi(f/f_s)n} = \sum_{n=-\infty}^{\infty} x(n) \cdot e^{-j2\pi(f_0/f_s)n} \cdot e^{-j2\pi(f_i/f_s)n} = \sum_{n=-\infty}^{\infty} x(n) \cdot e^{-j2\pi(f_0+f_i/f_s)n} \\ \tilde{X}(f) &= X(f_i + f_0) \end{aligned}$$

Therefore, the spectrum of the signal translates to the left with the magnitude of f_0 after complex basebanding.

Table 3. Frequency components of the signal after complex basebanding.

f_i [Hz]	f'_i [Hz]	$ X(f'_i) $ [dB]	f_i [Hz]	f'_i [Hz]	$ X(f'_i) $ [dB]
160	-90	0	-160	-410	0
237	-13	-20	-237	-487	-20
240	-10	-20	-240	-490	-20

The sequence after complex basebanding is filtered by the FIR low-pass filter designed in Section II. The filtered signal is obtained as $y(n)$. The Fourier Transform of the windowed filtered signal $y(n)$ ($n = 256, 257 \dots 511$) is calculated using a Kaiser Bessel window ($\alpha=2.5$ or $\beta=7.85$) with NFFT = 256. The logarithmic magnitude of the Fourier Transform is shown in Figure 4 (right). The frequency components of the signal $y(n)$ are presented in Table 4. The frequency values are the same compared to the signal before filtering because filtering does not shift the frequency of the signal. However, the magnitude of the peak values outside the passband frequency range ($-40 \text{ Hz} \leq f \leq 40 \text{ Hz}$) are attenuated compared to the signal before filtering. Detailed calculations for the frequency components of the signal $y(n)$ are listed in Appendix 1.

Table 4. Frequency components of the signal after complex basebanding and low-pass filtering.

f_i [Hz]	f'_i [Hz]	$ Y(f'_i) $ [dB]	f_i [Hz]	f'_i [Hz]	$ Y(f'_i) $ [dB]
160	-90	-20	-160	-410	-20
237	-13	0	-237	-487	-40
240	-10	0	-240	-490	-40

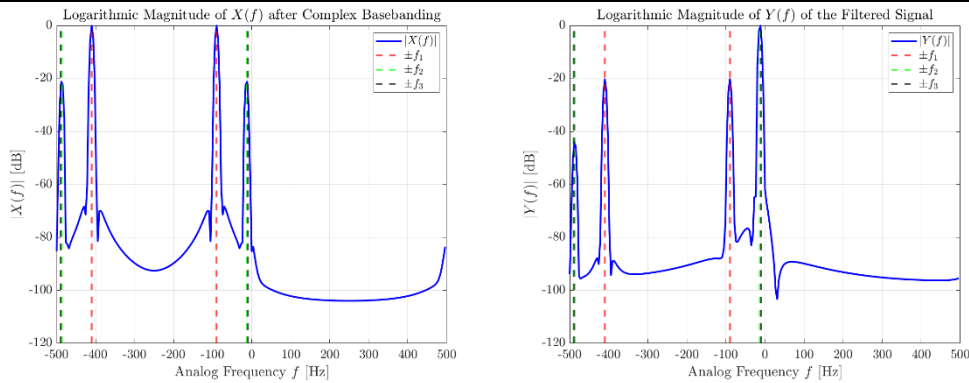


Figure 4. Logarithmic magnitude of the Fourier Transform for the complex baseband signal before (left) and after (right) passing the FIR low-pass filter.

IV. High Resolution Spectral Analysis

The complex filtered sequence is desampled by a factor of 8. The desampled sequence $x'(n)$ has a new sampling frequency $f'_s = 125$ Hz. The Fourier Transform of the windowed filtered signal $x'(n)$ ($n = 256, 257 \dots 511$) is calculated using a Kaiser Bessel window ($\alpha=2.5$ or $\beta=7.85$) with $NFFT = 256$. The logarithmic magnitude of the Fourier Transform is shown in Figure 5. The analog frequency bin width of the FFT is 0.488 Hz, which is 1/8 of the frequency bin width for the FFT of the original signal. The frequency values corresponding to the peaks of $|X'(f)|$ are presented in Table 5. Detailed calculations are presented in Appendix 1. Since the absolute values of some frequency components are greater than the Nyquist frequency ($f'_s/2 = 62.5$ Hz), aliasing is observed in the Fourier transform magnitude plot.

Table 5. Frequency components of the signal after complex basebanding, low-pass filtering and desampling.

f_i [Hz]	f'_i [Hz]	$ X'(f'_i) $ [dB]	f_i [Hz]	f'_i [Hz]	$ X'(f'_i) $ [dB]
160	35	-20	-160	-35	-20
237	-13	0	-237	13	-40
240	-10	-20	-240	10	-60

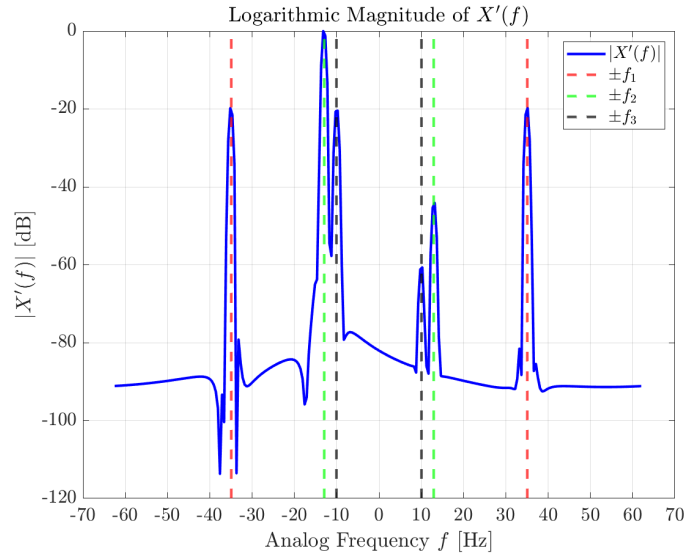


Figure 5. Logarithmic magnitude of the desampled signal $x'(n)$.

A second iteration in the design of the FIR low-pass filter is conducted. All requirements are to remain as originally given except for the passband/stopband weight ratio. Figure 6 and Figure 7 present the analysis result for the filter with passband/stopband weight ratio of 10 and 100 (the original weight ratio is 50). It is observed that a lower weight ratio has larger passband ripples and lower Fourier Transform magnitude in the stopband (larger stopband attenuation). In contrast, a higher weight ratio has smaller passband ripples and larger Fourier Transform magnitude in the stopband (smaller stopband attenuation). In practice, the selection of the passband/stopband weight ratio depends on the required behavior of the filter. A lower weight ratio is suggested if we want to remove the signal within the stopband (e.g. noise within the stopband has a significant effect on the results) regardless of the potential error introduced to the signal within the passband because of the large passband ripples. A higher weight ratio is suggested if we want to keep an accurate and smooth signal within the passband regardless of the potential effect from the signal within the stopband because of the smaller stopband attenuation.

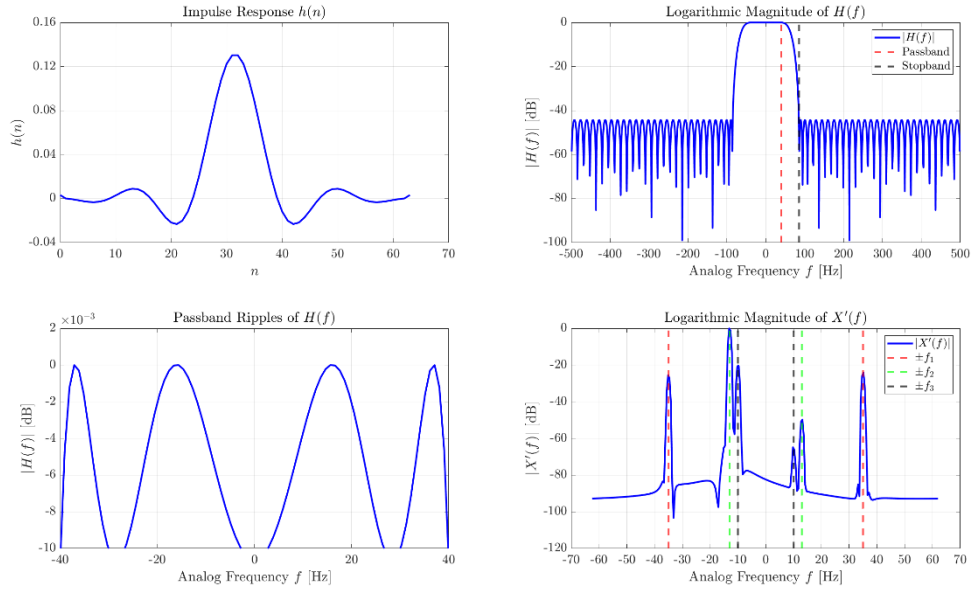


Figure 6. Analysis result for the case of passband/stopband weight ratio = 10.

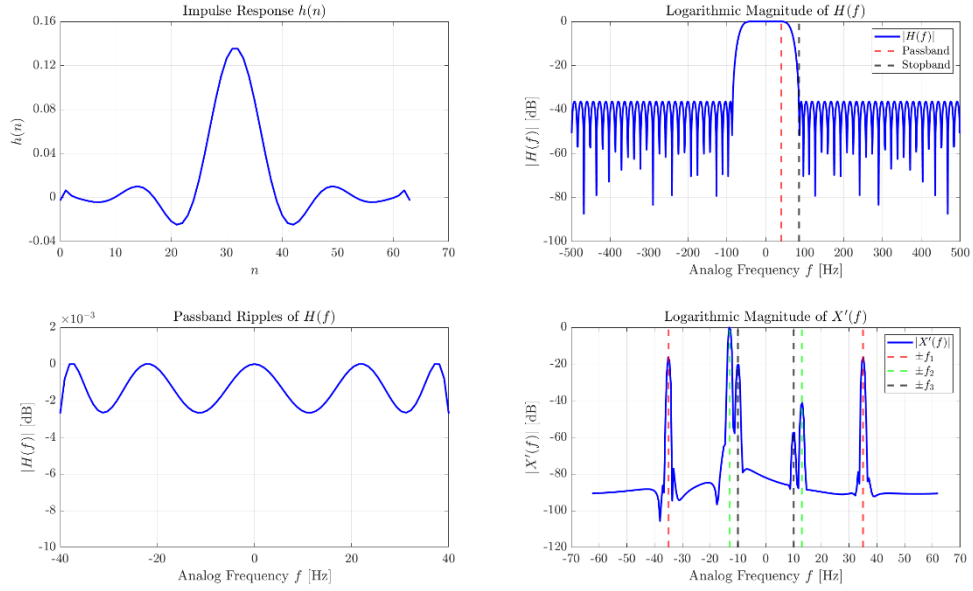


Figure 7. Analysis result for the case of passband/stopband weight ratio = 100.

Appendix 1. Hand Calculation

I. Data Set

$f_1 := 160 \text{ Hz}$ $f_2 := 237 \text{ Hz}$ $f_3 := 240 \text{ Hz}$ Analog frequency components of the signal

$A_1 := 100$ $A_2 := 10$ $A_3 := 1$ Linear amplitude for each analog frequency component

$f_s := 1000 \text{ Hz}$ Sampling frequency

$NFFT := 256$

$BinWidth := \frac{f_s}{NFFT} = 3.906 \text{ Hz}$ Analog frequency bin width of the FFT

$Bin_{f_1} := \text{round}\left(\frac{f_1}{BinWidth}\right) = 41$ Frequency bin index of analog frequency component f_1

$Bin_{f_2} := \text{round}\left(\frac{f_2}{BinWidth}\right) = 61$ Frequency bin index of analog frequency component f_2

$Bin_{f_3} := \text{round}\left(\frac{f_3}{BinWidth}\right) = 61$ Frequency bin index of analog frequency component f_3

Given the sampling frequency of 1000 Hz and $NFFT = 256$, the frequency bin indices for f_2 and f_3 are always 61, which is not affected by the window applied to the signal. Therefore, rectangular windowed FFT will not indicate distinct spectral peaks for f_2 and f_3 .

$20 \cdot \log(A_1) = 40$ Magnitude of the peak corresponding to $\pm f_1$

$20 \cdot \log(A_2) + 20 \cdot \log(A_3) = 20$ Magnitude of the peak corresponding to $\pm f_2$ and $\pm f_3$ (same frequency bin)

Therefore, after the normalization, the magnitude of the peak corresponding to $\pm f_1$ is 0 dB and the magnitude of the peak corresponding to $\pm f_2$ and $\pm f_3$ (same frequency bin) is -20 dB.

III. Complex Basebanding and Desampling

1. Complex basebanding:

$f_0 := 250 \text{ Hz}$ Center frequency

Frequency components after complex basebanding:

$f_1' := f_1 - f_0 = -90 \text{ Hz}$ $f_2' := f_2 - f_0 = -13 \text{ Hz}$ $f_3' := f_3 - f_0 = -10 \text{ Hz}$

$f_1'' := -f_1 - f_0 = -410 \text{ Hz}$ $f_2'' := -f_2 - f_0 = -487 \text{ Hz}$ $f_3'' := -f_3 - f_0 = -490 \text{ Hz}$

Complex basebanding does not change the magnitude of the signal. Therefore, the magnitude of peaks corresponding to each frequency component remain the same as previous.

2. Low-pass filtering:

Low-pass filtering does not shift the frequency of the signal. Therefore, frequency values of the signal remain the same as previous.

$$20 \cdot \log(A_1) - 40 = 0$$

Magnitude of the peak corresponding to f_1

$$20 \cdot \log(A_2) + 20 \cdot \log(A_3) = 20$$

Magnitude of the peak corresponding to f_2 and f_3 (same frequency bin)

$$20 \cdot \log(A_1) - 40 = 0$$

Magnitude of the peak corresponding to $-f_1$

$$20 \cdot \log(A_2) + 20 \cdot \log(A_3) - 40 = -20$$

Magnitude of the peak corresponding to $-f_2$ and $-f_3$ (same frequency bin)

Therefore, after the normalization, the magnitude of the peak corresponding to f_1 is -20 dB; the magnitude of the peak corresponding to f_2 and f_3 (same frequency bin) is 0 dB; the magnitude of the peak corresponding to $-f_1$ is -20 dB; the magnitude of the peak corresponding to $-f_2$ and $-f_3$ (same frequency bin) is -40 dB;

IV. High Resolution Spectral Analysis

$$f_s' := \frac{f_s}{8} = 125 \text{ Hz}$$

Sampling frequency after decimation

$$BinWidth := \frac{f_s'}{NFFT} = 0.488 \text{ Hz}$$

Analog frequency bin width of the FFT after desampling

Frequency components after desampling:

$$f_1' := f_1 - f_0 + f_s' = 35 \text{ Hz}$$

$$f_2' := f_2 - f_0 = -13 \text{ Hz}$$

$$f_3' := f_3 - f_0 = -10 \text{ Hz}$$

$$f_1'' := -f_1 - f_0 + 3 f_s' = -35 \text{ Hz}$$

$$f_2'' := -f_2 - f_0 + 4 f_s' = 13 \text{ Hz}$$

$$f_3'' := -f_3 - f_0 + 4 f_s' = 10 \text{ Hz}$$

$$20 \cdot \log(A_1) - 40 = 0$$

Magnitude of the peak corresponding to f_1 (-20 dB after normalization)

$$20 \cdot \log(A_2) = 20$$

Magnitude of the peak corresponding to f_2 (0 dB after normalization)

$$20 \cdot \log(A_3) = 0$$

Magnitude of the peak corresponding to f_3 (-20 dB after normalization)

$$20 \cdot \log(A_1) - 40 = 0$$

Magnitude of the peak corresponding to $-f_1$ (-20 dB after normalization)

$$20 \cdot \log(A_2) - 40 = -20$$

Magnitude of the peak corresponding to $-f_2$ (-40 dB after normalization)

$$20 \cdot \log(A_3) - 40 = -40$$

Magnitude of the peak corresponding to $-f_3$ (-60 dB after normalization)

Appendix 2. MATLAB Code

```
1 % SIO 207A Final Project
2 % Ruipu Ji
3
4 % Initialization and default plot settings.
5 clear; clc; close all;
6
7 set(0, 'DefaultAxesFontSize', 15);
8 set(0, 'DefaultTextFontSize', 15);
9
10 set(0, 'DefaultTextInterpreter', 'latex');
11 set(0, 'DefaultLegendInterpreter', 'latex');
12 set(0, 'DefaultAxesTickLabelInterpreter', 'latex');
13
14 %% Data Set.
15 % Generate a 4096-point discrete-time signal x(n). -----
16 A = [100 10 1]; % Magnitude.
17 f = [160 237 240]; % Analog frequency [Hz].
18 Phi = [0 0 0]; % Phase angle [rad].
19
20 fs = 1000; % Sampling frequency [Hz].
21 n = 4096; % Number of data points.
22 t = (0:1:n-1)'*1/fs; % Time vector.
23
24 x = zeros(n,1); % Initialize the signal sequence as an empty array.
25
26 % Generate the discrete-time signal sequence x(n).
27 for idx = 1:size(A,2)
28     x = x + A(idx)*cos(2*pi*f(idx)*t+Phi(idx));
29 end
30
31 % Plot the first 256 data points of the signal x(n). -----
32 NFFT = 256;
33
34 figure('Position', [0, 0, 1800, 600]);
35
36 subplot(1,2,1);
37 hold on;
38 plot(0:1:NFFT-1, x(1:NFFT), 'b', 'LineWidth', 2);
39 grid on;
40 box on;
41 xlim([0 300]);
42 xticks(0:50:300);
43 ylim([-150 150]);
44 yticks(-150:50:150);
45 xlabel('Data Point Index $n$');
46 ylabel('$x(n)$');
47 title('The First 256 Data Points of Signal $x(n)$');
48
49 % 256-point FFT of the signal using the Kaiser-Bessel window. -----
50 % Kaiser-Bessel window (alpha = 2.5 or beta = alpha*pi = 7.85).
51 KaiserBesselWindow = kaiser(NFFT, 7.85);
52
53 % Window the first 256 data points of the signal x(n).
54 x_256_KB = x(1:NFFT) .* KaiserBesselWindow;
55
56 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
```



```

magnitude.
57 X_256_KB_magnitude = 20*log10(abs(fftshift(fft(x_256_KB, NFFT))));
58 X_256_KB_magnitude = X_256_KB_magnitude - max(X_256_KB_magnitude); % Normalize the results.
59
60 % Calculate bin width of the FFT.
61 BinWidth = fs/NFFT;
62
63 % Calculate the frequency bin index for analog frequencies.
64 f_idx = zeros(1, size(f,2));
65
66 for idx = 1:size(f_idx,2)
67     f_idx(1,idx) = round(f(idx)/BinWidth);
68 end
69
70 % Plot the result.
71 f_fft = (-fs/2:BinWidth:fs/2-BinWidth)';
72
73 subplot(1,2,2);
74 hold on;
75 plot(f_fft, X_256_KB_magnitude, 'b', 'LineWidth', 2);
76 xline(f(1), 'r--', 'LineWidth', 2);
77 xline(f(2), 'g--', 'LineWidth', 2);
78 xline(f(3), 'k--', 'LineWidth', 2);
79 xline(-f(1), 'r--', 'LineWidth', 2);
80 xline(-f(2), 'g--', 'LineWidth', 2);
81 xline(-f(3), 'k--', 'LineWidth', 2);
82 grid on;
83 box on;
84 xlim([-500 500]);
85 xticks(-500:100:500);
86 ylim([-120 0]);
87 yticks(-120:20:0);
88 xlabel('Analog Frequency $f$ [Hz]');
89 ylabel('$|X(f)|$ [dB]');
90 legend('$|X(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location', 'northeast');
91 title('Logarithmic Magnitude of $X(f)$');
92
93 exportgraphics(gcf, 'Figure1-PartI.png', 'ContentType', 'image');
94
95 %% Decimation Filter Design.
96 N_coefficients = 64; % Number of coefficients.
97 fc_passband = 40; % Passband cutoff frequency [Hz].
98 fc_stopband = 85; % Stopband cutoff frequency [Hz].
99 Weight = [50 1]; % Define weight ratio vector [passband stopband].
100
101 % Filter design using an equiripple FIR filter design algorithm.
102 f_parameter = [0 fc_passband/(fs/2) fc_stopband/(fs/2) 1];
103 a_parameter = [1 1 0 0];
104 h = firpm(N_coefficients-1, f_parameter, a_parameter, Weight)'; % Filter design.
105
106 % Calculate the frequency response of the filter.
107 NFFT_filter = 1024; % NFFT for the filter calculation.
108 f_filter_fft = (-0.5:1/NFFT_filter:0.5-1/NFFT_filter)' * fs; % Frequency vector for the plot.
109 RectangularWindow = rectwin(NFFT_filter); % Rectangular window.

```

```

110 h_padded = padarray(h, [NFFT_filter-size(h,1) 0], 'post'); % Pad the filter to the
same length as NFFT.
111
112 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
magnitude.
113 H_magnitude = 20*log10(abs(fftshift(fft(h_padded.*RectangularWindow, NFFT_filter))));
114 H_magnitude = H_magnitude - max(H_magnitude); % Normalize the result.
115
116 % Plot the result.
117 figure('Position', [0, 0, 2500, 500]);
118
119 subplot(1,3,1);
120 hold on;
121 plot(0:1:N_coefficients-1, h, 'b', 'LineWidth', 2);
122 grid on;
123 box on;
124 xlim([0 70]);
125 xticks(0:10:70);
126 ylim([-0.04 0.16]);
127 yticks(-0.04:0.04:0.16);
128 xlabel('$n$');
129 ylabel('$h(n)$');
130 title('Impulse Response $h(n)$');
131
132 subplot(1,3,2);
133 hold on;
134 plot(f_filter_fft, H_magnitude, 'b', 'LineWidth', 2);
135 xline(fc_passband, 'r--', 'LineWidth', 2);
136 xline(fc_stopband, 'k--', 'LineWidth', 2);
137 grid on;
138 box on;
139 xlim([-500 500]);
140 xticks(-500:100:500);
141 ylim([-100 0]);
142 yticks(-100:20:0);
143 xlabel('Analog Frequency $f$ [Hz]');
144 ylabel('$|H(f)|$ [dB]');
145 legend('$|H(f)|$', 'Passband', 'Stopband', 'Location', 'northeast');
146 title('Logrithmic Magnitude of $H(f)$');
147
148 subplot(1,3,3);
149 hold on;
150 plot(f_filter_fft, H_magnitude, 'b', 'LineWidth', 2);
151 grid on;
152 box on;
153 xlim([0 40]);
154 xticks(0:10:40);
155 ylim([-0.01 0.002]);
156 yticks(-0.01:0.002:0.002);
157 xlabel('Analog Frequency $f$ [Hz]');
158 ylabel('$|H(f)|$ [dB]');
159 title('Passband Ripples of $H(f)$');
160
161 exportgraphics(gcf, 'Figure2-PartII.png', 'ContentType', 'image');
162
163 %% Complex Basebanding and Desampling.

```

```

164 % Complex multiplication on the discrete-time signal x(n). -----
165 f0 = 250; % Center frequency [Hz].
166 w0 = 2*pi*f0/fs; % Center frequency in rad.
167 x_complex = zeros(size(x,1), 1); % Initialization for the complex sequence.
168
169 for idx = 1:size(x,1)
170     x_complex(idx) = exp(-1i*w0*idx)*x(idx);
171 end
172
173 % 256-point FFT of the complex sequence. -----
174 % Window the first 256 data points of the complex sequence.
175 x_complex_256_KB = x_complex(1:NFFT) .* KaiserBesselWindow;
176
177 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
178 X_complex_256_KB_magnitude = 20*log10(abs(fftshift(fft(x_complex_256_KB, NFFT))));
179 X_complex_256_KB_magnitude = X_complex_256_KB_magnitude - max
180 (X_complex_256_KB_magnitude); % Normalize the results.
181
182 % Plot the results.
183 figure('Position', [0, 0, 1800, 600]);
184 subplot(1,2,1);
185 hold on;
186 plot(f_fft, X_complex_256_KB_magnitude, 'b', 'LineWidth', 2);
187 xline(f(1)-f0, 'r--', 'LineWidth', 2);
188 xline(f(2)-f0, 'g--', 'LineWidth', 2);
189 xline(f(3)-f0, 'k--', 'LineWidth', 2);
190 xline(-f(1)-f0, 'r--', 'LineWidth', 2);
191 xline(-f(2)-f0, 'g--', 'LineWidth', 2);
192 xline(-f(3)-f0, 'k--', 'LineWidth', 2);
193 grid on;
194 box on;
195 xlim([-500 500]);
196 xticks(-500:100:500);
197 ylim([-120 0]);
198 yticks(-120:20:0);
199 xlabel('Analog Frequency $f$ [Hz]');
200 ylabel('$|X(f)|$ [dB]');
201 legend('$|X(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location', 'northeast');
202 title('Logrithmic Magnitude of $X(f)$ after Complex Basebanding');
203
204 % Low pass filter the complex sequence in the time domain. -----
205 y_complex = filter(h, 1, x_complex);
206
207 % Window the data points (n = 256-511) of the signal y(n).
208 y_complex_256_KB = y_complex(NFFT+1:2*NFFT) .* KaiserBesselWindow;
209
210 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
211 Y_complex_256_KB_magnitude = 20*log10(abs(fftshift(fft(y_complex_256_KB, NFFT))));
212 Y_complex_256_KB_magnitude = Y_complex_256_KB_magnitude - max
213 (Y_complex_256_KB_magnitude); % Normalize the results.
214
215 % Plot the results.
216 subplot(1,2,2);

```

```

216 hold on;
217 plot(f_fft, Y_complex_256_KB_magnitude, 'b', 'LineWidth', 2);
218 xline(f(1)-f0, 'r--', 'LineWidth', 2);
219 xline(f(2)-f0, 'g--', 'LineWidth', 2);
220 xline(f(3)-f0, 'k--', 'LineWidth', 2);
221 xline(-f(1)-f0, 'r--', 'LineWidth', 2);
222 xline(-f(2)-f0, 'g--', 'LineWidth', 2);
223 xline(-f(3)-f0, 'k--', 'LineWidth', 2);
224 grid on;
225 box on;
226 xlim([-500 500]);
227 xticks(-500:100:500);
228 ylim([-120 0]);
229 yticks(-120:20:0);
230 xlabel('Analog Frequency $f$ [Hz]');
231 ylabel('$|Y(f)|$ [dB]');
232 legend('$|Y(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location', 'northeast');
233 title('Logarithmic Magnitude of $Y(f)$ of the Filtered Signal');
234
235 exportgraphics(gcf, 'Figure3-PartIII.png', 'ContentType', 'image');
236
237 % Desample the complex filtered sequence y(n) by a factor of 8. -----
238 factor_downsample = 8;
239 x_prime = y_complex(1:factor_downsample:end);
240
241 %% High Resolution Spectral Analysis.
242 % Window the data points (n = 256-511) of the signal x'(n).
243 x_prime_256_KB = x_prime(NFFT+1:2*NFFT) .* KaiserBesselWindow;
244
245 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
magnitude.
246 X_prime_256_KB_magnitude = 20*log10(abs(fftshift(fft(x_prime_256_KB, NFFT))));
247 X_prime_256_KB_magnitude = X_prime_256_KB_magnitude - max(X_prime_256_KB_magnitude);
% Normalize the results.
248
249 % Calculate the updated bin width of the FFT.
250 fs_prime = fs/factor_downsample;
251 BinWidth_prime = fs_prime/NFFT;
252 f_prime_fft = (-fs_prime/2:BinWidth_prime:fs_prime/2-BinWidth_prime)';
253
254 % Plot the results.
255 figure('Position', [0, 0, 800, 600]);
256 hold on;
257 plot(f_prime_fft, X_prime_256_KB_magnitude, 'b', 'LineWidth', 2);
258 xline(f(1)-f0+fs_prime, 'r--', 'LineWidth', 2);
259 xline(f(2)-f0, 'g--', 'LineWidth', 2);
260 xline(f(3)-f0, 'k--', 'LineWidth', 2);
261 xline(-f(1)-f0+3*fs_prime, 'r--', 'LineWidth', 2);
262 xline(-f(2)-f0+4*fs_prime, 'g--', 'LineWidth', 2);
263 xline(-f(3)-f0+4*fs_prime, 'k--', 'LineWidth', 2);
264 grid on;
265 box on;
266 xlim([-70 70]);
267 xticks(-70:10:70);
268 ylim([-120 0]);
269 yticks(-120:20:0);

```

```

270 xlabel('Analog Frequency $f$ [Hz]');
271 ylabel('$|X^{\prime}(f)|$ [dB]');
272 legend('$|X^{\prime}(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location',
'northeast');
273 title('Logrithmic Magnitude of $X^{\prime}(f)$');
274
275 exportgraphics(gcf, 'Figure4-PartIV.png', 'ContentType', 'image');
276
277 %% Second Iteration on LPF Design.
278 % Trial #1: Use passband/stopband weight ratio = 10. -----
279 Weight2 = [10 1]; % Define weight ratio vector [passband stopband].
280 h2 = firpm(N_coefficients-1, f_parameter, a_parameter, Weight2); % Filter design.
281
282 % Calculate the frequency response of the filter.
283 h2_padded = padarray(h2, [NFFT_filter-size(h2,1) 0], 'post'); % Pad the filter to the
same length as NFFT.
284
285 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
magnitude.
286 H2_magnitude = 20*log10(abs(fftshift(fft(h2_padded.*RectangularWindow,
NFFT_filter))));
287 H2_magnitude = H2_magnitude - max(H2_magnitude); % Normalize the result.
288
289 % Plot the result.
290 figure('Position', [0, 0, 1800, 1000]);
291
292 subplot(2,2,1);
293 hold on;
294 plot(0:1:N_coefficients-1, h2, 'b', 'LineWidth', 2);
295 grid on;
296 box on;
297 xlim([0 70]);
298 xticks(0:10:70);
299 ylim([-0.04 0.16]);
300 yticks(-0.04:0.04:0.16);
301 xlabel('$n$');
302 ylabel('$h(n)$');
303 title('Impulse Response $h(n)$');
304
305 subplot(2,2,2);
306 hold on;
307 plot(f_filter_fft, H2_magnitude, 'b', 'LineWidth', 2);
308 xline(fc_passband, 'r--', 'LineWidth', 2);
309 xline(fc_stopband, 'k--', 'LineWidth', 2);
310 grid on;
311 box on;
312 xlim([-500 500]);
313 xticks(-500:100:500);
314 ylim([-100 0]);
315 yticks(-100:20:0);
316 xlabel('Analog Frequency $f$ [Hz]');
317 ylabel('$|H(f)|$ [dB]');
318 legend('$|H(f)|$', 'Passband', 'Stopband', 'Location', 'northeast');
319 title('Logrithmic Magnitude of $H(f)$');
320
321 subplot(2,2,3);

```

```

322 hold on;
323 plot(f_filter_fft, H2_magnitude, 'b', 'LineWidth', 2);
324 grid on;
325 box on;
326 xlim([0 40]);
327 xticks(0:10:40);
328 ylim([-0.01 0.002]);
329 yticks(-0.01:0.002:0.002);
330 xlabel('Analog Frequency $f$ [Hz]');
331 ylabel('$|H(f)|$ [dB]');
332 title('Passband Ripples of $H(f)$');
333
334 % Low pass filter the complex sequence in the time domain.
335 y2_complex = filter(h2, 1, x_complex);
336
337 % Desample the complex filtered sequence y(n) by a factor of 8.
338 x2_prime = y2_complex(1:factor_downsample:end);
339
340 % Window the data points (n = 256-511) of the signal x'(n).
341 x2_prime_256_KB = x2_prime(NFFT+1:2*NFFT) .* KaiserBesselWindow;
342
343 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
344 magnitude.
345 X2_prime_256_KB_magnitude = 20*log10(abs(fftshift(fft(x2_prime_256_KB, NFFT))));
346 X2_prime_256_KB_magnitude = X2_prime_256_KB_magnitude - max
347 (X2_prime_256_KB_magnitude); % Normalize the results.
348
349 % Plot the results.
350 subplot(2,2,4);
351 hold on;
352 plot(f_prime_fft, X2_prime_256_KB_magnitude, 'b', 'LineWidth', 2);
353 xline(f(1)-f0+fs_prime, 'r--', 'LineWidth', 2);
354 xline(f(2)-f0, 'g--', 'LineWidth', 2);
355 xline(f(3)-f0, 'k--', 'LineWidth', 2);
356 xline(-f(1)-f0+3*fs_prime, 'r--', 'LineWidth', 2);
357 xline(-f(2)-f0+4*fs_prime, 'g--', 'LineWidth', 2);
358 xline(-f(3)-f0+4*fs_prime, 'k--', 'LineWidth', 2);
359 grid on;
360 box on;
361 xlim([-70 70]);
362 xticks(-70:10:70);
363 ylim([-120 0]);
364 yticks(-120:20:0);
365 xlabel('Analog Frequency $f$ [Hz]');
366 ylabel('$|X^{\prime}(f)|$ [dB]');
367 legend('$|X^{\prime}(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location',
368 'northeast');
369 title('Logarithmic Magnitude of $X^{\prime}(f)$');
370
371 exportgraphics(gcf, 'Figure5-PartIV-Trial#1.png', 'ContentType', 'image');
372
373 % Trial #2: Use passband/stopband weight ratio = 100. -----
374 Weight3 = [100 1]; % Define weight ratio vector [passband stopband].
375 h3 = firpm(N_coefficients-1, f_parameter, a_parameter, Weight3); % Filter design.
376
377 % Calculate the frequency response of the filter.

```

```

375 h3_padded = padarray(h3, [NFFT_filter-size(h3,1) 0], 'post'); % Pad the filter to the
same length as NFFT.
376
377 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
magnitude.
378 H3_magnitude = 20*log10(abs(fftshift(fft(h3_padded.*RectangularWindow,
NFFT_filter))));
379 H3_magnitude = H3_magnitude - max(H3_magnitude); % Normalize the result.
380
381 % Plot the result.
382 figure('Position', [0, 0, 1800, 1000]);
383
384 subplot(2,2,1);
385 hold on;
386 plot(0:1:N_coefficients-1, h3, 'b', 'LineWidth', 2);
387 grid on;
388 box on;
389 xlim([0 70]);
390 xticks(0:10:70);
391 ylim([-0.04 0.16]);
392 yticks(-0.04:0.04:0.16);
393 xlabel('$n$');
394 ylabel('$h(n)$');
395 title('Impulse Response $h(n)$');
396
397 subplot(2,2,2);
398 hold on;
399 plot(f_filter_fft, H3_magnitude, 'b', 'LineWidth', 2);
400 xline(fc_passband, 'r--', 'LineWidth', 2);
401 xline(fc_stopband, 'k--', 'LineWidth', 2);
402 grid on;
403 box on;
404 xlim([-500 500]);
405 xticks(-500:100:500);
406 ylim([-100 0]);
407 yticks(-100:20:0);
408 xlabel('Analog Frequency $f$ [Hz]');
409 ylabel('$|H(f)|$ [dB]');
410 legend('$|H(f)|$', 'Passband', 'Stopband', 'Location', 'northeast');
411 title('Logrithmic Magnitude of $H(f)$');
412
413 subplot(2,2,3);
414 hold on;
415 plot(f_filter_fft, H3_magnitude, 'b', 'LineWidth', 2);
416 grid on;
417 box on;
418 xlim([0 40]);
419 xticks(0:10:40);
420 ylim([-0.01 0.002]);
421 yticks(-0.01:0.002:0.002);
422 xlabel('Analog Frequency $f$ [Hz]');
423 ylabel('$|H(f)|$ [dB]');
424 title('Passband Ripples of $H(f)$');
425
426 % Low pass filter the complex sequence in the time domain.
427 y3_complex = filter(h3, 1, x_complex);

```

```

428
429 % Desample the complex filtered sequence y(n) by a factor of 8.
430 x3_prime = y3_complex(1:factor_downsample:end);
431
432 % Window the data points (n = 256-511) of the signal x'(n).
433 x3_prime_256_KB = x3_prime(NFFT+1:2*NFFT) .* KaiserBesselWindow;
434
435 % Perform 256-point FFT on the windowed signal and calculate the logarithmic
magnitude.
436 X3_prime_256_KB_magnitude = 20*log10(abs(fftshift(fft(x3_prime_256_KB, NFFT))));
437 X3_prime_256_KB_magnitude = X3_prime_256_KB_magnitude - max
(X3_prime_256_KB_magnitude); % Normalize the results.
438
439 % Plot the results.
440 subplot(2,2,4);
441 hold on;
442 plot(f_prime_fft, X3_prime_256_KB_magnitude, 'b', 'LineWidth', 2);
443 xline(f(1)-f0+fs_prime, 'r--', 'LineWidth', 2);
444 xline(f(2)-f0, 'g--', 'LineWidth', 2);
445 xline(f(3)-f0, 'k--', 'LineWidth', 2);
446 xline(-f(1)-f0+3*fs_prime, 'r--', 'LineWidth', 2);
447 xline(-f(2)-f0+4*fs_prime, 'g--', 'LineWidth', 2);
448 xline(-f(3)-f0+4*fs_prime, 'k--', 'LineWidth', 2);
449 grid on;
450 box on;
451 xlim([-70 70]);
452 xticks(-70:10:70);
453 ylim([-120 0]);
454 yticks(-120:20:0);
455 xlabel('Analog Frequency $f$ [Hz]');
456 ylabel('$|X^{\prime}(f)|$ [dB]');
457 legend('$|X^{\prime}(f)|$', '$\pm f_1$', '$\pm f_2$', '$\pm f_3$', 'Location',
'northeast');
458 title('Logarithmic Magnitude of $X^{\prime}(f)$');
459
460 exportgraphics(gcf, 'Figure6-PartIV-Trial#2.png', 'ContentType', 'image');
461

```