**3.1 Nulling Filters for Rejection**

**A)**clear  
clc

wn1 = 0.44 \* pi; % First nulling frequency

wn2 = 0.7 \* pi; % Second nulling frequency

% Coefficients for the first nulling filter

b1\_0 = 1;

b1\_1 = -2 \* cos(wn1);

b1\_2 = 1;

% Coefficients for the second nulling filter

b2\_0 = 1;

b2\_1 = -2 \* cos(wn2);

b2\_2 = 1;

% Coefficients Displaying of Filters

disp('Filter 1 Coefficients:');

disp(['b1\_0 = ', num2str(b1\_0)]);

disp(['b1\_1 = ', num2str(b1\_1)]);

disp(['b1\_2 = ', num2str(b1\_2)]);

disp(' ');

disp('Filter 2 Coefficients:');

disp(['b2\_0 = ', num2str(b2\_0)]);

disp(['b2\_1 = ', num2str(b2\_1)]);

disp(['b2\_2 = ', num2str(b2\_2)]);

**A number and number equation

Description automatically generated with medium confidence**

**B)**clear

clc

N = 150; % Length of the Signal

n = 0:N-1; % Time Index

w1 = 0.3 \* pi; % Frequency of first sinusoid

w2 = 0.44 \* pi; % Frequency of second sinusoid

w3 = 0.7 \* pi; % Frequency of third sinusoid

phi1 = 0; % Phase of first sinusoid

phi2 = -pi/3; % Phase of second sinusoid

phi3 = -pi/4; % Phase of third sinusoid

x1 = 5 \* cos(w1 \* n); % First sinusodial component

x2 = 22 \* cos(w2 \* n + phi2); % Second sinusodial component

x3 = 22 \* cos(w3 \* n + phi3); % Third sinusodial component

x = x1 + x2 + x3; % Input Signal Calculation

% Plotting the Input Signal

figure;

stem(n, x);

xlabel('n');

ylabel('x[n]');

title('Input Signal x[n]');

A graph of a function

Description automatically generated with medium confidence

**C)**

clear

clc

wn1 = 0.44 \* pi; % First nulling frequency

wn2 = 0.7 \* pi; % Second nulling frequency

% Coefficients for the first nulling filter

b1\_0 = 1;

b1\_1 = -2 \* cos(wn1);

b1\_2 = 1;

% Coefficients for the second nulling filter

b2\_0 = 1;

b2\_1 = -2 \* cos(wn2);

b2\_2 = 1;

% Waveform Properties

N = 150; % Signal length

n = 0:N-1; % Time Index

w1 = 0.3 \* pi; % Frequency of first sinusoid

w2 = 0.44 \* pi; % Frequency of second sinusoid

w3 = 0.7 \* pi; % Frequency of third sinusoid

phi1 = 0; % Phase of first sinusoid

phi2 = -pi/3; % Phase of second sinusoid

phi3 = -pi/4; % Phase of third sinusoid

x1 = 5 \* cos(w1 \* n); % First sinusodial component

x2 = 22 \* cos(w2 \* n + phi2); % Second sinusodial component

x3 = 22 \* cos(w3 \* n + phi3); % Third sinusodial component

x = x1 + x2 + x3; % Input Signal Calculation

b1 = [b1\_0, b1\_1, b1\_2]; % First nulling filter coefficients

b2 = [b2\_0, b2\_1, b2\_2]; % Second nulling filter coefficients

% Nulling filter implementation

filtered\_x1 = conv(x, b1, 'same');

filtered\_x2 = conv(filtered\_x1, b2, 'same');

% Plotting the signals [Original & Filtered]

figure;

subplot(3,1,1);

plot(n, x);

xlabel('n');

ylabel('x[n]');

title('Original Signal');

subplot(3,1,2);

plot(n, filtered\_x1);

xlabel('n');

ylabel('Filtered Signal 1');

title('Filtered Signal after 1st Nulling Filter');

subplot(3,1,3);

plot(n, filtered\_x2);

xlabel('n');

ylabel('Filtered Signal 2');

title('Filtered Signal after 2nd Nulling Filter');

A screenshot of a computer screen

Description automatically generated

**D & E)**

clear

clc

wn1 = 0.44 \* pi; % First nulling frequency

wn2 = 0.7 \* pi; % Second nulling frequency

% Coefficients for the first nulling filter

b1\_0 = 1;

b1\_1 = -2 \* cos(wn1);

b1\_2 = 1;

% Coefficients for the second nulling filter

b2\_0 = 1;

b2\_1 = -2 \* cos(wn2);

b2\_2 = 1;

% Waveform Properties

N = 150; % Signal length

n = 0:N-1; % Time Index

w1 = 0.3 \* pi; % Frequency of first sinusoid

w2 = 0.44 \* pi; % Frequency of second sinusoid

w3 = 0.7 \* pi; % Frequency of third sinusoid

phi1 = 0; % Phase of first sinusoid

phi2 = -pi/3; % Phase of second sinusoid

phi3 = -pi/4; % Phase of third sinusoid

x1 = 5 \* cos(w1 \* n); % First sinusodial component

x2 = 22 \* cos(w2 \* n + phi2); % Second sinusodial component

x3 = 22 \* cos(w3 \* n + phi3); % Third sinusodial component

x = x1 + x2 + x3; % Input Signal Calculation

% Nulling filter implementations

filtered\_x1 = conv(x, [b1\_0, b1\_1, b1\_2], 'same');

filtered\_x2 = conv(filtered\_x1, [b2\_0, b2\_1, b2\_2], 'same');

% Define the mathematical equation for the output signal

% For n >= 5, y[n] = x[n] - 2 \* cos(wn1) \* x[n-1] + x[n-2] - 2 \* cos(wn2) \* y[n-1] + y[n-2]

y\_mathematical = zeros(1, N);

y\_mathematical(1:4) = filtered\_x2(1:4); % Initial values

for i = 5:N

y\_mathematical(i) = filtered\_x2(i) - 2 \* cos(wn1) \* filtered\_x2(i-1) + filtered\_x2(i-2) ...

- 2 \* cos(wn2) \* y\_mathematical(i-1) + y\_mathematical(i-2);

end

% Plot both the mathematical formula and FIR filter output over the range 5 <= n <= 40

range = 5:40;

figure;

plot(range, filtered\_x2(range), 'o-', 'DisplayName', 'FIR Filter Output');

hold on;

plot(range, y\_mathematical(range), 'x-', 'DisplayName', 'Mathematical Formula');

xlabel('n');

ylabel('Output Signal');

title('Comparison of FIR Filter Output and Mathematical Formula');

legend;

grid on;

A graph of a graph with numbers

Description automatically generated

**F)**

The output signal is not same for the first points due to the transient response or "start-up" attribute of the filters. This start-up behavior exists since the filters necessiate time to be balanced and attain to the steady-state after the input signal is applied.

In the FIR filters domain, the start-up points are the first points in the output signal where the filter response has not yet totally balanced. Points are affected by the initial situations and transient impacts of the filtering process.

For the cascade of two length-3 FIR filters designed in part (a), each single filter has a length of 3 coefficients. However, when cascaded, the length of the combined filter is the sum of the lengths of the filters minus one. Due to that, when convolving two filters, the length of output is the sum of the individual filter lengths minus one.

Hereby, for two length-3 FIR filters cascaded each other, the length of the combined filter is . Thus, there are 5 start-up points in the output signal, which is relevant with the lengths of the filters designed in part (a). These start-up points correspond to the first transient attributes of the filtering process, and they converge to the steady-state response.

**3.2 Simple Bandpass Filter Design**

**A)**

clear

clc

L = 10; % Filter length parameter

wc = 0.44\*pi; % Center frequency parameter

n = 0:L; % Index

h = 2/L \* cos(wc \* n); % Impulse response of bandpass filter

w = 0:pi/100:pi; % Angular frequency variable

H = freqz(h, 1, w); % Frequency response of filter

% Gain at interest interval

w\_interest = [0.3\*pi, wc, 0.7\*pi]; % Interest interval

gain\_interest = abs(interp1(w, H, w\_interest)); % Gain interest value

% Result Displaying & Plotting

disp('Gain at frequencies of interest:');

disp(['w = 0.3\*pi: ', num2str(gain\_interest(1))]);

disp(['w = 0.44\*pi: ', num2str(gain\_interest(2))]);

disp(['w = 0.7\*pi: ', num2str(gain\_interest(3))]);

figure;

plot(w, abs(H));

hold on;

stem(w\_interest, gain\_interest, 'r', 'LineWidth', 1.5);

xlabel('Frequency (rad/sample)');

ylabel('Magnitude');

title('Frequency Response of Bandpass Filter');

legend('Filter Frequency Response', 'Frequencies of Interest');

grid on;

**A screen shot of a computer

Description automatically generated**

**B)**

clear

clc

L\_values = [10, 20, 40]; % Filter length parameters

wc = 0.44\*pi; % Center frequency parameter

passband\_widths = zeros(1, length(L\_values)); % Arrays to store passband widths

% Frequency response for each of the filter lengths

figure;

for i = 1:length(L\_values)

L = L\_values(i);

n = 0:L; % Index

h = 2/L \* cos(wc \* n); % Impulse Response of the bandpass filter

w = 0:pi/1000:pi; % Angular frequency variable

H = freqz(h, 1, w); % Frequency response of filter

% Find frequencies on |H(e^jw)| >= 0.707 \* H\_max

passband\_indices = find(abs(H) >= 0.707 \* max(abs(H)));

passband\_widths(i) = w(passband\_indices(end)) - w(passband\_indices(1));

% Plotting of frequency response

subplot(length(L\_values), 1, i);

plot(w, abs(H), 'LineWidth', 1.5);

hold on;

stem(w(passband\_indices), abs(H(passband\_indices)), 'r', 'LineWidth', 1.5);

xlabel('Frequency (rad/sample)');

ylabel('Magnitude');

title(['Frequency Response for L = ', num2str(L)]);

legend('Filter Frequency Response', 'Passband Region');

grid on;

end

% Displaying results

disp('Passband widths at 0.707 level:');

disp(passband\_widths);

A screenshot of a computer

Description automatically generated

**C)**

The selectivity of a filter touch on its features to pass particular frequencies while filtering or rejecting others. For the bandpass filter located at , its selectivity can be explained by the attributes of its frequency response.

Inspecting frequency response plot, the filter display a peak in magnitude in the neighborhood of the center frequency of . This peak is the passband region where frequencies located neighborhood points of center frequency are admitted to pass. It means, the filter "passes" frequencies close to .

At frequencies lower () or higher () than the center frequency, the magnitude of the frequency response reduces reasonably. This filtering represents the filter's properties to reduce or reject frequencies other than the passband region.

The algorithm of this selectivity depends on the model of the filter's impulse response. By structuring the filter with a certain impulse response that indicates the desired frequency component at while attenuating others, the filter effectively behaves as a bandpass filter, admitting only the requested frequency to "pass through."

**D)**

clear

clc

wc = 0.44\*pi; % Center frequency parameter

desired\_attenuation = 10; % Desired attenuation factor

L = 5; % Starting value of filter length to initialize

step\_size = pi/100000; % Define step size for frequency response calculation

while true

% Generate impulse response of the bandpass filter

n = 0:L; % Index

h = 2/L \* cos(wc \* n); % Impulse Response of the bandpass filter

w = 0:step\_size:pi; % Angular frequency variable

H = freqz(h, 1, w); % Frequency response of filter

% Find frequencies on |H(e^jw)| <= 0.1 \* H\_max for w <= 0.3\*pi

passband\_indices\_low = find(abs(H) <= 0.1 \* max(abs(H)), 1, 'last');

% Find frequencies on |H(e^jw)| <= 0.1 \* H\_max for w >= 0.7\*pi

passband\_indices\_high = find(abs(H) <= 0.1 \* max(abs(H)), 1, 'first');

if passband\_indices\_low <= 0.3\*pi && passband\_indices\_high >= 0.7\*pi % Desired attenuation verifier

break;

end

L = L + 1; % Filter length increment

end

% Displaying Smallest Value

disp(['Smallest value of L: ', num2str(L)]);

**E)**

clear

clc

n = 0:99; % Time Index

x = 5\*cos(0.3\*pi\*n) + 22\*cos(0.44\*pi\*n - pi/3) + 22\*cos(0.7\*pi\*n - pi/4); % Signal

b = [1, -2\*cos(0.44\*pi), 1]; % Filter coefficient of y[n]

a = [1]; % Filter coefficient of x[n]

y = filter(b, a, x); % Input signal filtering

% 100 points plotting of input and output signals

subplot(2,1,1);

stem(n, x(1:100), 'b', 'LineWidth', 1.5);

xlabel('n');

ylabel('Amplitude');

title('Input Signal');

grid on;

subplot(2,1,2);

stem(n, y(1:100), 'r', 'LineWidth', 1.5);

xlabel('n');

ylabel('Amplitude');

title('Filtered Output Signal');

grid on;

A screenshot of a graph

Description automatically generated

Explanation:

• The input signal occurs with three sinusoidal elements: a cosine wave, a cosine wave with a phase shift of , and a cosine wave with a phase shift of

• The filter behaves as a second-order difference equation where the actual output is the difference between the input and twice the previous input. This filter has the impact of indicating differences in the input signal over time.

• By implementing this filter, the output signal is customized for attenuating or removing sinusoidal components with frequencies close to the resonant frequency of filter ().

• In conclusion, the output signal will display reduced amplitudes for sinusoidal components similar to , efficiently reducing or removing the cosine waves existing in the input signal. The cosine wave and the cosine wave with a phase shift of might exist in the output signal, relying on proximity of their frequencies to the resonant frequency of the filter.

**F)**

b = [1, -2\*cos(0.44\*pi), 1]; % Filter coefficient of y[n]

a = [1]; % Filter coefficient of x[n]

w = linspace(0, pi, 1000); % Frequency range from 0 to pi

H = freqz(b, a, w); % Frequency response

% Plot frequency response

plot(w, abs(H), 'LineWidth', 1.5);

xlabel('Frequency (rad/sample)');

ylabel('Magnitude');

title('Frequency Response of the Filter');

grid on;

**A graph with a line

Description automatically generated**

Explanation:

• The frequency response of a filter, demonstrated as explains how the filter impacts sinusoidal elements of different frequencies.

• The magnitude of the frequency response at a certain frequency shows how filter diminished or amplifies sinusoidal elements at that frequency.

• The magnitude response at desired frequency could be utilized to establish the size of sinusoidal elements in the output signal. Sinusoidal components at frequency peaks in the magnitude response would experience less filtering and will be more attenuated in the output signal for elements at frequencies on the dips or nulls in the magnitude response.

• Hence, by inspecting the frequency response plot, we can conclude which frequencies are amplified or filtered by the filter, and at the end, how the filter impacts the size of each sinusoidal elements in the output signal.