DSP lab 08
Name: Basil khowaja bk08432

Task 1)

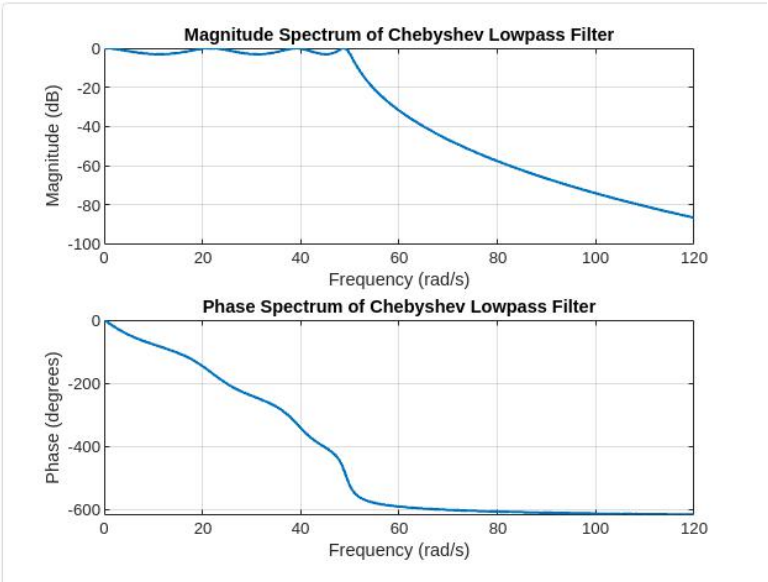
code:

```
omega1 = 50;
omega2 = 60;
Rp = 3;
As = 30;
omega0 = omega2 / omega1;
epsilon = sqrt(10^(Rp/10) - 1);
F = sqrt((10^(As/10) - 1) / (10^(Rp/10) - 1));


N_temp = acosh(F) / acosh(omega0);
N = ceil(N_temp);


fprintf('Computed filter order N = %d\n', N);
fprintf('Computed ripple factor epsilon = %f\n', epsilon);




beta = asinh(1/epsilon) / N;
p_all = [];
for k = 0:(2*N - 1)
alpha = (2*k + 1) * pi / (2 * N);
pk = 1i * omega1 * (cos(alpha) * cosh(beta) - 1i * sin(alpha) * sinh(beta));
p_all = [p_all; pk];
end


p = p_all(real(p_all) < 0);


if length(p) ~= N
error('Unexpected number of stable poles. Check the calculations.');
end


K = prod(-p);
```

```
den = poly(p);
num = K;


w = linspace(0, 2*omega2, 1000);
H = freqs(num, den, w);


figure;


subplot(2,1,1);
plot(w, 20*log10(abs(H)), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');
title('Magnitude Spectrum of Chebyshev Lowpass Filter');
grid on;


subplot(2,1,2);
plot(w, unwrap(angle(H)) * (180/pi), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase (degrees)');
title('Phase Spectrum of Chebyshev Lowpass Filter');
grid on;
```

Output:

```
Computed filter order N = 7
Computed ripple factor epsilon = 0.997628
```



Observation:

From the magnitude response plot, we can see that the filter has a ripple in the passband (0–50 rad/s), which is a characteristic of Chebyshev filters. This ripple is due to the way the filter achieves a sharper transition between the passband and stopband compared to a Butterworth filter. At around 50 rad/s, the gain begins to drop rapidly, and by 60 rad/s, the signal is attenuated by at least 30 dB, meaning higher frequencies are strongly suppressed.

The phase response plot shows how the filter affects the phase of the input signal. The phase decreases with frequency, meaning that higher frequency components of the input signal will experience a phase shift. This is expected in most practical filters and can cause distortion in signals that require phase linearity.

Physically, this filter is designed to allow low-frequency signals to pass while blocking high-frequency noise or unwanted components. The ripples in the passband occur because the Chebyshev filter prioritizes a faster roll-off at the cutoff frequency over a perfectly flat response in the passband. The phase shift indicates that different frequency components will experience different time delays when passing through the filter, which can be important in applications like audio processing, where phase distortion might affect signal quality.

In summary, the Chebyshev filter achieves a sharper transition between passband and stopband by introducing controlled ripple in the passband, and it successfully filters out frequencies beyond 60 rad/s while maintaining gain in the lower frequencies.

Task 2)

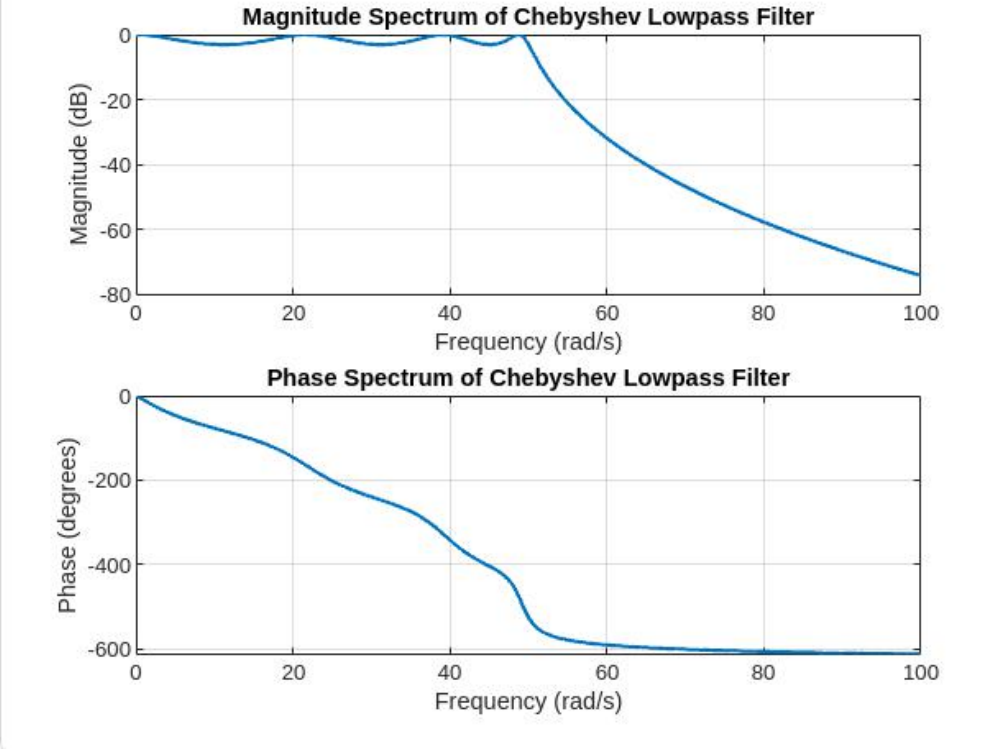
code:

```
omega1 = 50;
Rp = 3;
N = 7;


epsilon = sqrt(10^(Rp/10) - 1);
fprintf('Computed ripple factor epsilon = %f\n', epsilon);


beta = asinh(1/epsilon) / N;
p_all = [];


for k = 0:(2*N - 1)
alpha = (2*k + 1) * pi / (2 * N);
pk = 1i * omega1 * (cos(alpha)*cosh(beta) - 1i*sin(alpha)*sinh(beta));
p_all = [p_all; pk];
end


p = p_all(real(p_all) < 0);
if length(p) ~= N
error('Unexpected number of stable poles. Check the calculations.');
end


K = prod(-p);
den = poly(p);
num = K;


w = linspace(0, 2*omega1, 1000);
H = freqs(num, den, w);


figure;


subplot(2,1,1);
plot(w, 20*log10(abs(H)), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');
```

```
title('Magnitude Spectrum of Chebyshev Lowpass Filter');
grid on;


subplot(2,1,2);
plot(w, unwrap(angle(H)) * (180/pi), 'LineWidth', 1.5);
xlabel('Frequency (rad/s)');
ylabel('Phase (degrees)');
title('Phase Spectrum of Chebyshev Lowpass Filter');
grid on;
```

output:

```
Computed ripple factor epsilon = 0.997628
```



Observation:

In this task, we designed and analyzed a 7th-order Chebyshev Type I lowpass filter with a passband ripple of 3 dB and plotted its magnitude and phase responses. The purpose of this filter is to allow frequencies up to 50 rad/s to pass with minimal attenuation while strongly attenuating higher frequencies. The computed ripple factor ($\epsilon \approx 0.9976$\epsilon \approx 0.9976$\epsilon \approx 0.9976$) determines the passband behavior, ensuring that the gain fluctuates slightly before rolling off sharply into the stopband.

The magnitude spectrum shows that the filter maintains a relatively high gain for frequencies below 50 rad/s, with noticeable ripple in the passband. This ripple is a key characteristic of Chebyshev filters, which trade off a perfectly flat response for a steeper roll-off in the transition region. Unlike Butterworth filters, which have a gradual roll-off, the Chebyshev filter sacrifices smoothness in the passband to achieve faster attenuation beyond the cutoff frequency. As expected, at around 50 rad/s, the magnitude response begins to drop sharply, showing a steep transition into the stopband. This confirms that the filter effectively blocks higher frequencies while keeping the desired low-frequency components intact.

The phase response shows that phase shift increases negatively as frequency increases. This behavior is expected for IIR (Infinite Impulse Response) filters, where higher-order filters introduce greater phase shift due to the pole-based design. The phase does not remain linear, meaning different frequency components experience different time delays as they pass through the filter. This can cause distortion in time-domain signals, particularly in applications where phase relationships between frequency components are crucial, such as in audio or communications systems. The phase plot also shows a steep change near the cutoff frequency, which indicates that frequencies around the transition region experience rapid phase variation.

In real-world applications, this filter acts as a frequency-selective system that allows signals with lower frequencies (below 50 rad/s) to pass while attenuating high-frequency noise or unwanted components. The presence of ripple in the passband means that certain frequencies below 50 rad/s may have slightly amplified or attenuated gains, but the signal integrity remains largely preserved. The steep roll-off ensures that frequencies above 50 rad/s are suppressed effectively, preventing interference from unwanted high-frequency components. However, the nonlinear phase response suggests that if this filter is applied to signals with wide frequency content, the different time delays across frequencies could lead to distortion, which might need correction using phase equalization techniques.

Task 3)

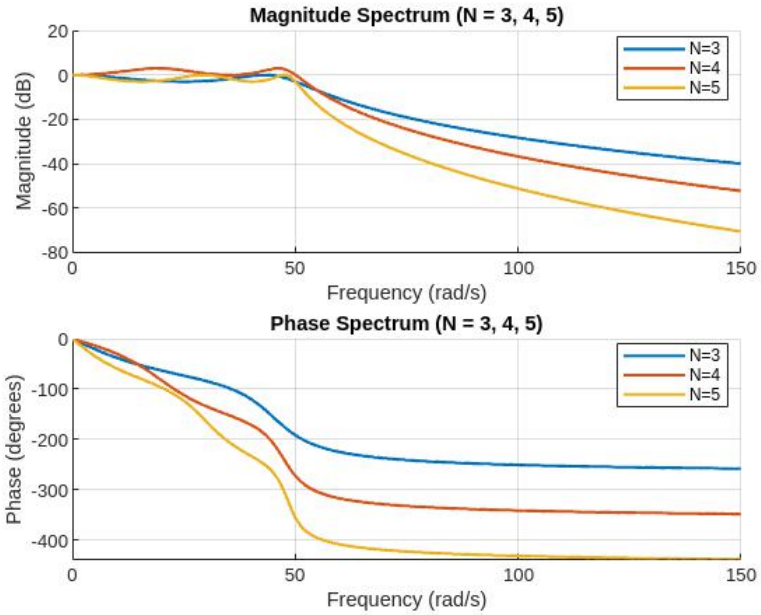
code:

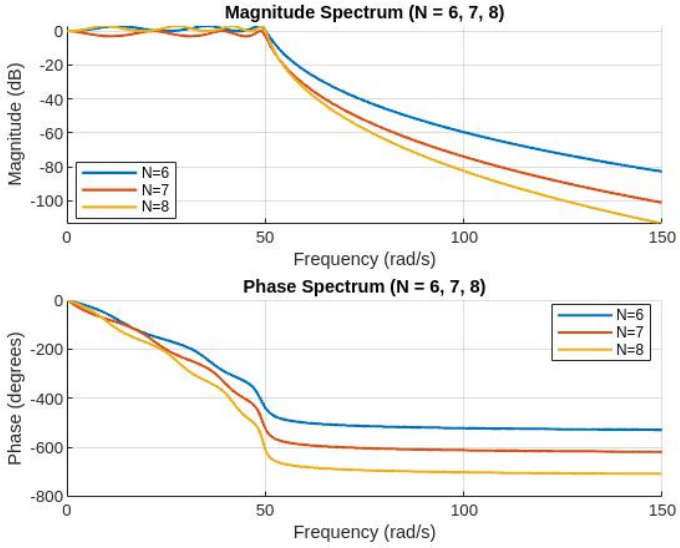
```
clear; clc; close all;
omega1 = 50;
Rp = 3;
epsilon = sqrt(10^(Rp/10) - 1);


w = linspace(0, 3*omega1, 1000);
figure('Name','Chebyshev Comparison: N = 3, 4, 5');


subplot(2,1,1);
hold on; grid on;
title('Magnitude Spectrum (N = 3, 4, 5)');
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');


for N = [3, 4, 5]
beta = asinh(1/epsilon) / N;
p_all = [];
for k = 0:(2*N - 1)
alpha_k = (2*k + 1)*pi/(2*N);
pk = 1i*omega1*(cos(alpha_k)*cosh(beta) - 1i*sin(alpha_k)*sinh(beta));
p_all = [p_all; pk];
end
p = p_all(real(p_all) < 0);
K = prod(-p);
den = poly(p);
num = K;


H = freqs(num, den, w);
plot(w, 20*log10(abs(H)), 'DisplayName', sprintf('N=%d', N), 'LineWidth',1.5);
end


legend('Location','best');


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

```
title('Phase Spectrum (N = 3, 4, 5)');
xlabel('Frequency (rad/s)');
ylabel('Phase (degrees)');


for N = [3, 4, 5]
beta = asinh(1/epsilon) / N;
p_all = [];
for k = 0:(2*N - 1)
alpha_k = (2*k + 1)*pi/(2*N);
pk = 1i*omega1*(cos(alpha_k)*cosh(beta) - 1i*sin(alpha_k)*sinh(beta));
p_all = [p_all; pk];
end
p = p_all(real(p_all) < 0);
K = prod(-p);
den = poly(p);
num = K;
H = freqs(num, den, w);
plot(w, unwrap(angle(H))*180/pi, 'DisplayName', sprintf('N=%d', N),
'LineWidth',1.5);
end


legend('Location','best');


figure('Name','Chebyshev Comparison: N = 6, 7, 8');


subplot(2,1,1);
hold on; grid on;
title('Magnitude Spectrum (N = 6, 7, 8)');
xlabel('Frequency (rad/s)');
ylabel('Magnitude (dB)');


for N = [6, 7, 8]
beta = asinh(1/epsilon) / N;
p_all = [];
for k = 0:(2*N - 1)
alpha_k = (2*k + 1)*pi/(2*N);
pk = 1i*omega1*(cos(alpha_k)*cosh(beta) - 1i*sin(alpha_k)*sinh(beta));
p_all = [p_all; pk];
end
p = p_all(real(p_all) < 0);
K = prod(-p);
den = poly(p);
num = K;
```

```
H = freqs(num, den, w);


plot(w, 20*log10(abs(H)), 'DisplayName', sprintf('N=%d', N), 'LineWidth',1.5);
end
legend('Location','best');


subplot(2,1,2);
hold on; grid on;
title('Phase Spectrum (N = 6, 7, 8)');
xlabel('Frequency (rad/s)');
ylabel('Phase (degrees)');


for N = [6, 7, 8]
beta = asinh(1/epsilon) / N;
p_all = [];
for k = 0:(2*N - 1)
alpha_k = (2*k + 1)*pi/(2*N);
pk = 1i*omega1*(cos(alpha_k)*cosh(beta) - 1i*sin(alpha_k)*sinh(beta));
p_all = [p_all; pk];
end
p = p_all(real(p_all) < 0);
K = prod(-p);
den = poly(p);
num = K;
H = freqs(num, den, w);


plot(w, unwrap(angle(H))*180/pi, 'DisplayName', sprintf('N=%d', N),
'LineWidth',1.5);
end
legend('Location','best');
```

output:





Observation:

In this task, we designed and analyzed Chebyshev Type I lowpass filters for different filter orders (N=3,4,5,6,7,8N=3,4,5,6,7,8N=3,4,5,6,7,8) and compared their magnitude and phase responses. The purpose of this analysis is to observe how increasing the filter order affects the filter's performance in terms of passband behavior, stopband attenuation, and phase characteristics. The key takeaway is that higher-order

filters provide sharper transitions but also introduce greater phase shifts.

The magnitude spectrum shows that as the filter order NNN increases, the filter achieves a steeper roll-off at the cutoff frequency, meaning it attenuates high-frequency components more effectively. For lower-order filters (e.g., N=3N=3N=3), the transition from passband to stopband is more gradual, allowing some higher-frequency components to partially pass through. In contrast, higher-order filters (e.g., N=8N=8N=8) show a much sharper cutoff, meaning that unwanted high-frequency signals are attenuated more aggressively. However, the tradeoff is that as NNN increases, the passband ripple becomes more pronounced. This ripple is a direct result of the Chebyshev filter design, where increased ripple allows for a sharper transition between the passband and stopband.

The phase response indicates that as the order NNN increases, the filter introduces a greater phase shift, meaning that different frequency components experience larger time delays. This is expected because higher-order filters have more complex pole structures, leading to more significant phase variation. The phase curves show that lower-order filters introduce a relatively mild phase shift, whereas higher-order filters cause phase shifts exceeding -800 degrees at the highest frequencies in the plot. This increasing phase shift is important in applications where maintaining signal phase relationships is critical, such as in communications and control systems.

In real-world applications, increasing the filter order makes the filter more selective, allowing only a narrow range of low frequencies to pass while blocking higher-frequency noise or interference more effectively. This is beneficial in applications where high-frequency components are unwanted, such as in audio signal processing (to remove unwanted high-frequency noise) or in data communication systems (to ensure only the desired frequency band is used). However, the increasing phase shift with higher-order filters means that signals passing through the filter may experience time delays, potentially leading to distortion in applications where phase linearity is required. For example, in speech processing or music equalization, excessive phase distortion can alter the timing of different frequency components, making the sound unnatural.

Task 4:

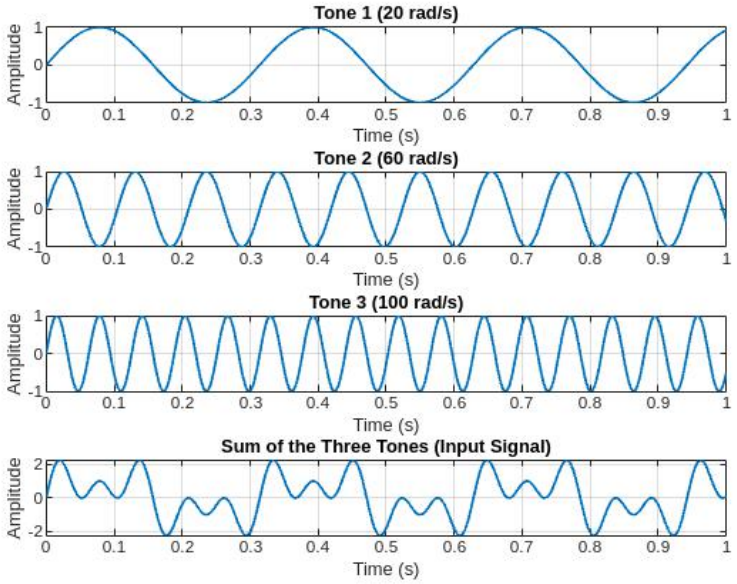
code:

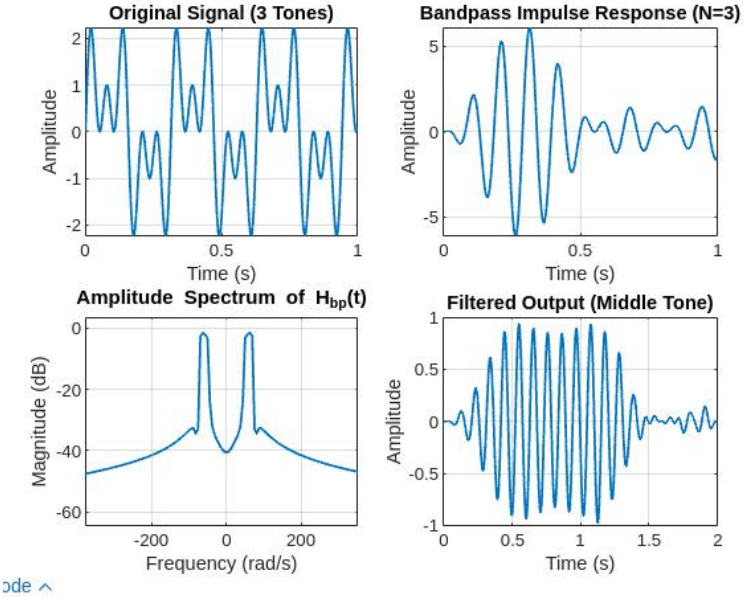
```
clear; clc; close all;


dt = 1e-3;
tEnd = 1.0;
t = (0:dt:tEnd).';


tone1 = sin(20*t);
tone2 = sin(60*t);
tone3 = sin(100*t);
x = tone1 + tone2 + tone3;


figure('Name','Individual Tones and Their Sum');
subplot(4,1,1);
plot(t, tone1, 'LineWidth',1.2); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Tone 1 (20 rad/s)');


subplot(4,1,2);
plot(t, tone2, 'LineWidth',1.2); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Tone 2 (60 rad/s)');


subplot(4,1,3);
plot(t, tone3, 'LineWidth',1.2); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Tone 3 (100 rad/s)');


subplot(4,1,4);
plot(t, x, 'LineWidth',1.2); grid on;
xlabel('Time (s)'); ylabel('Amplitude');
title('Sum of the Three Tones (Input Signal)');


omega_mid = 60;
BW = 20;
omega_lp = BW/2;
```

```
Rp = 3;
orders = [3, 5, 7];


for N = orders
figure('Name', sprintf('Chebyshev Bandpass Filter (N=%d)', N));
h_lp = chebyLowpassImpulse(N, omega_lp, Rp, t);
h_bp = 2 * h_lp .* cos(omega_mid * t);


x_col = x(:);
h_bp_col = h_bp(:);
y = conv(x_col, h_bp_col) * dt;
t_y = (0:length(y)-1).' * dt;
subplot(2,2,1);
plot(t, x, 'LineWidth',1.2);
grid on; xlabel('Time (s)'); ylabel('Amplitude');
title('Original Signal (3 Tones)');


subplot(2,2,2);
plot(t, h_bp, 'LineWidth',1.2);
grid on; xlabel('Time (s)'); ylabel('Amplitude');
title(sprintf('Bandpass Impulse Response (N=%d)', N));


L = length(h_bp_col);
Hf = fft(h_bp_col)*dt;
Hf_shift = fftshift(Hf);
df_rad = 2*pi/(L*dt);
freqAxisShift = (-floor(L/2) : ceil(L/2)-1).' * df_rad;
subplot(2,2,3);
plot(freqAxisShift, 20*log10(abs(Hf_shift)), 'LineWidth',1.2);
grid on; xlabel('Frequency (rad/s)'); ylabel('Magnitude (dB)');
title('Amplitude Spectrum of H_{bp}(t)');


subplot(2,2,4);
plot(t_y, y, 'LineWidth',1.2);
grid on; xlabel('Time (s)'); ylabel('Amplitude');
title('Filtered Output (Middle Tone)');
end


function h_lp = chebyLowpassImpulse(N, W, Rp, t)


epsilon = sqrt(10^(Rp/10) - 1);
```

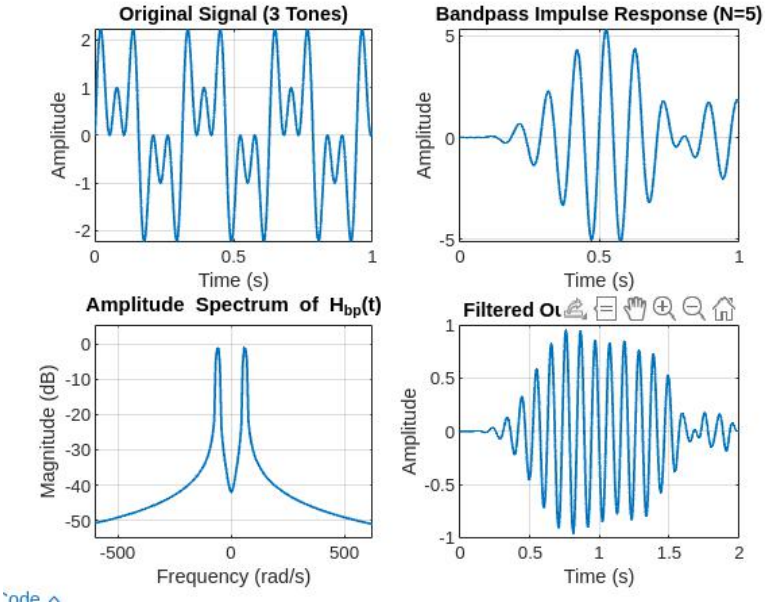
```
beta = asinh(1/epsilon)/N;
p_all = [];
for k = 0:(2*N - 1)
alpha_k = (2*k+1)*pi/(2*N);
pk = 1i*W * (cos(alpha_k)*cosh(beta) - 1i*sin(alpha_k)*sinh(beta));
p_all = [p_all; pk];
end
p = p_all(real(p_all)<0);
if length(p) ~= N
error('Unexpected number of stable poles: check design.');
end


K = prod(-p);
den = poly(p);
den = real(den);
num = real(K);
sys_lp = tf(num, den);
hData = impulse(sys_lp, t);
h_lp = squeeze(hData);
end
```

output:

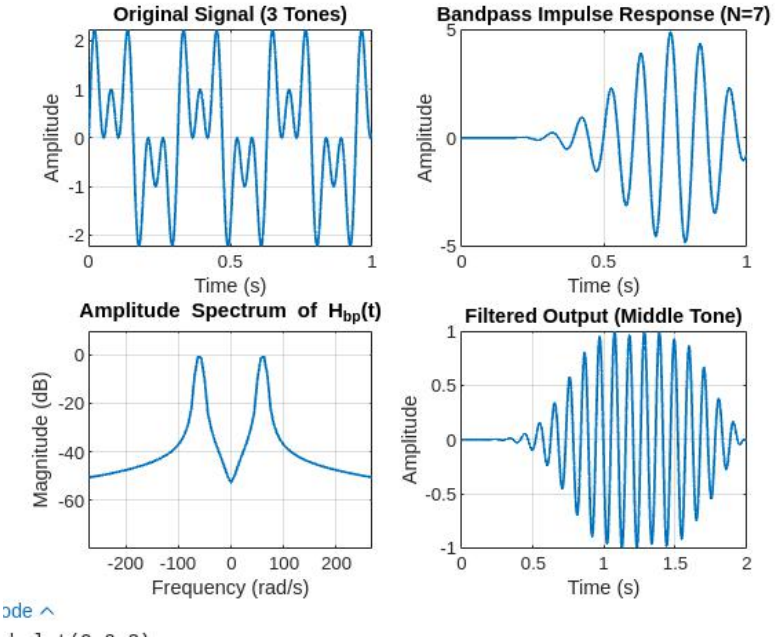



Original Signal (3 Tones)
Amplitude
Time (s)
Bandpass Impulse Response (N=3)
Amplitude
Time (s)
Amplitude Spectrum of $H_{bp}(t)$
Magnitude (dB)
Frequency (rad/s)
Filtered Output (Middle Tone)
Amplitude
Time (s)

ɔde ^


Original Signal (3 Tones)
Amplitude
Time (s)
Bandpass Impulse Response (N=5)
Amplitude
Time (s)
Amplitude Spectrum of $H_{bp}(t)$
Magnitude (dB)
Frequency (rad/s)
Filtered Ou
Amplitude
Time (s)

ɔde ^



Observations:

In this task, we applied Chebyshev bandpass filters of different orders (N=3,5,7N=3,5,7N=3,5,7) to extract the middle tone (60 rad/s) from a signal containing three frequency components: 20 rad/s, 60 rad/s, and 100 rad/s. The objective was to allow the 60 rad/s component to pass while suppressing the lower (20 rad/s) and higher (100 rad/s) tones. The first plot represents the three individual tones and their sum, where the low-frequency tone (20 rad/s) has the longest wavelength, the middle tone (60 rad/s) has a moderate frequency, and the high-frequency tone (100 rad/s) oscillates the fastest. When combined, these three tones form a complex signal due to the superposition of sinusoids, which acts as the input signal for filtering.

To isolate the middle tone, we used Chebyshev bandpass filters designed with different filter orders. The bandpass impulse response for each order shows how the filter reacts to an input signal. For N=3N=3N=3, the impulse response has a broader main lobe with slower decay, allowing a wider range of frequencies to pass. As the order increases to N=5N=5N=5, the response becomes more oscillatory, indicating improved frequency selectivity. For N=7N=7N=7, the main lobe becomes even narrower, ensuring that the filter isolates only the 60 rad/s tone more effectively. This demonstrates that increasing NNN enhances the filter’s ability to focus on a narrower frequency band while rejecting others more efficiently.

The amplitude spectrum of the filter confirms that it is centered around ±60\pm60±60 rad/s, meaning the filter allows these frequencies while attenuating others. For lower-order filters (N=3), the transition from passband to stopband is more gradual, allowing some unwanted frequency components to pass through. In contrast, for higher-order filters (N=7), the transition is sharper, meaning that the low (20 rad/s) and high (100 rad/s) frequencies are significantly suppressed, making the filter more precise. This sharper roll-off ensures better isolation of the desired frequency while minimizing interference from unwanted signals.

The filtered output signal in each case reveals how well the filter extracts the middle tone. For N=3N=3N=3, residual components from the 20 rad/s and 100 rad/s tones remain, indicating that the filter does not fully suppress them. As the filter order increases to N=5N=5N=5, the undesired components are further reduced, and the 60 rad/s tone becomes more dominant. For N=7N=7N=7, the output signal is almost a pure 60 rad/s tone, confirming that the filter has effectively isolated the middle frequency. This result demonstrates that increasing NNN leads to better frequency selectivity, reducing leakage of unwanted tones.

In practical applications, this filtering technique is useful in audio processing, where specific frequencies are enhanced or removed, wireless communications, where only a certain frequency band is required, and biomedical signal processing, such as isolating heartbeat frequencies from noisy ECG data. However, increasing the filter order also introduces greater phase distortion and requires more computational resources, making it essential to balance selectivity and efficiency depending on the application.