DSP lab 07
Name: Basil khowaja bk08432

Task 1)

Code:

```
clear; close all; clc;

N = 3

fc = 10;

omgc = 2*pi*fc;

k = 0:(N-1);

poles = omgc * exp(1i*(pi/2 + (2*k+1)*pi/(2*N)));

disp('Computed Left-Half-Plane Poles:');

for i = 1:length(poles)

   if abs(imag(poles(i))) < 1e-6

       % Treat as a real pole

       disp(['Pole ', num2str(i), ': Real = ',
num2str(real(poles(i))), ' (Real Pole)']);

   else

       disp(['Pole ', num2str(i), ': Real = ',
num2str(real(poles(i))), ', Imag = ',
num2str(imag(poles(i)))]);

   end

end
```

```
figure('Name','Butterworth Filter
Poles','NumberTitle','off');

scatter(real(poles), imag(poles), 100, 'filled', 'k');

hold on; grid on; axis equal;

xline(0, '--b'); yline(0, '--b');

xlabel('Real Part'); ylabel('Imaginary Part');

title(['Poles in the LHP for N = ', num2str(N)]);
```

```
denom = poly(poles);

gain  = omgc^N;

sys   = tf(gain, real(denom));
```

```
f = -50:0.1:50;

omega = 2*pi*f;

s = 1i * omega;

H = gain ./ polyval(denom, s);

Hmag = abs(H);

Hphs = angle(H);
```

```
figure('Name','Butterworth Lowpass Filter Frequency
Response','NumberTitle','off');
```

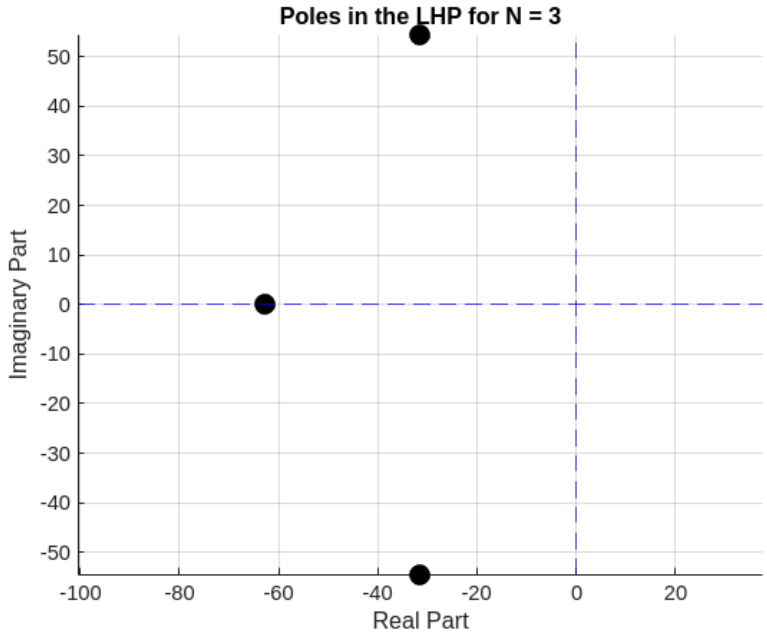
```
subplot(2,1,1);

plot(f, Hmag, 'b', 'LineWidth', 1.5);

xlabel('Frequency (Hz)'); ylabel('|H(f)|');
```

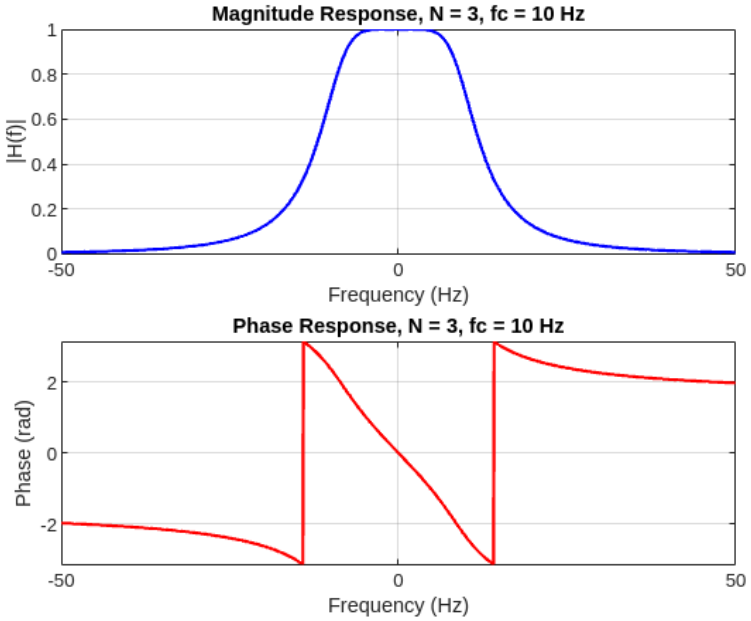
```
title(['Magnitude Response, N = ', num2str(N), ', fc = ',
num2str(fc), ' Hz']);
grid on;

subplot(2,1,2);
plot(f, Hphs, 'r', 'LineWidth', 1.5);
xlabel('Frequency (Hz)'); ylabel('Phase (rad)');
title(['Phase Response, N = ', num2str(N), ', fc = ',
num2str(fc), ' Hz']);
grid on;
axis([-50 50 -pi pi]);

t_imp = linspace(0, 1, 1000);
[h, t_out] = impulse(sys, t_imp);

figure('Name','Impulse Response of Butterworth
LPF','NumberTitle','off');
plot(t_out, h, 'm', 'LineWidth', 1.5);
xlabel('Time (s)'); ylabel('h(t)');
title(['Impulse Response, N = ', num2str(N), ', fc = ',
num2str(fc), ' Hz']);
grid on;

disp('Generic Butterworth filter design complete.');

function [mag, phs] = spectrums(zrs, pls, gain, omega)
```

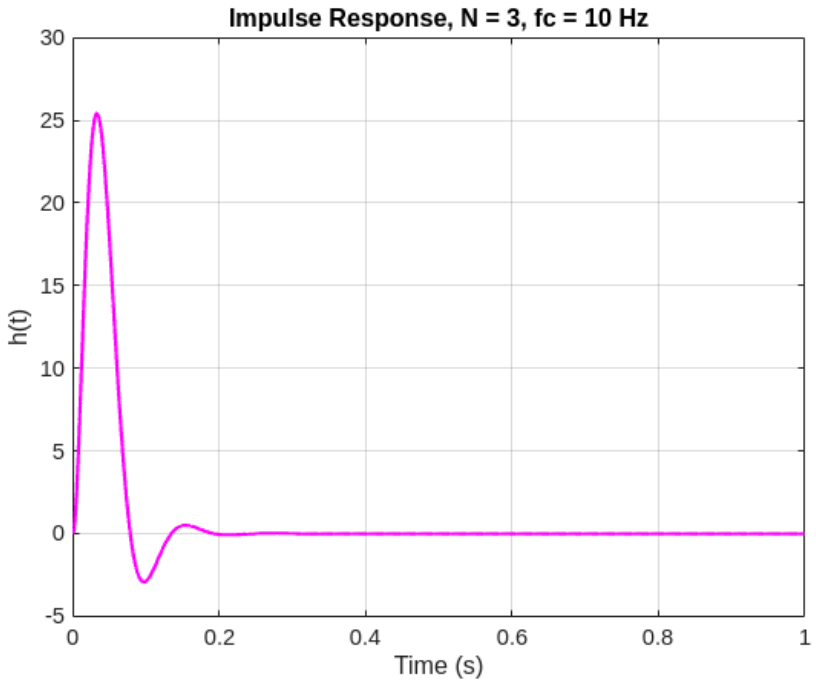
```
    nz = length(zrs);
    np = length(pls);
    nomg = length(omega);
    s = 1j*omega;
    mag = ones(1, nomg);
    phs = zeros(1, nomg);

    for n = 1:nz
        mag = mag .* abs(s - zrs(n));
        phs = phs + angle(s - zrs(n));
    end

    for n = 1:np
        mag = mag ./ abs(s - pls(n));
        phs = phs - angle(s - pls(n));
    end
    mag = mag * gain;
    phs = wrapToPi(phs);
end
```

Output for N=3: (odd case)

```
Pole 1: Real = -31.4159, Imag = 54.414
Pole 2: Real = -62.8319 (Real Pole)
Pole 3: Real = -31.4159, Imag = -54.414
```

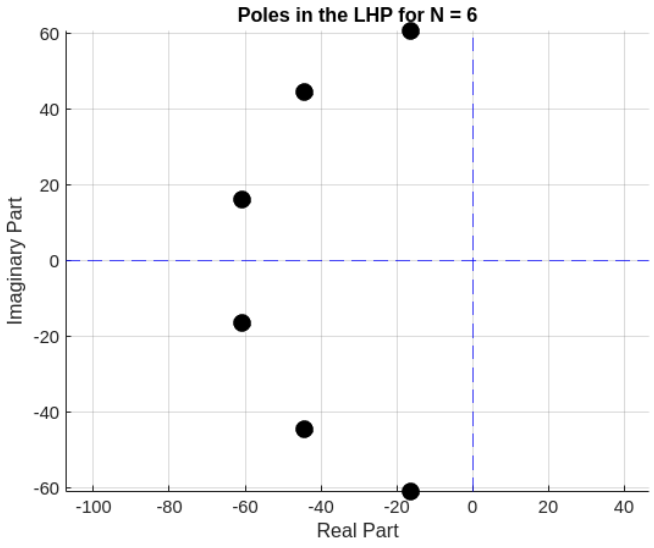


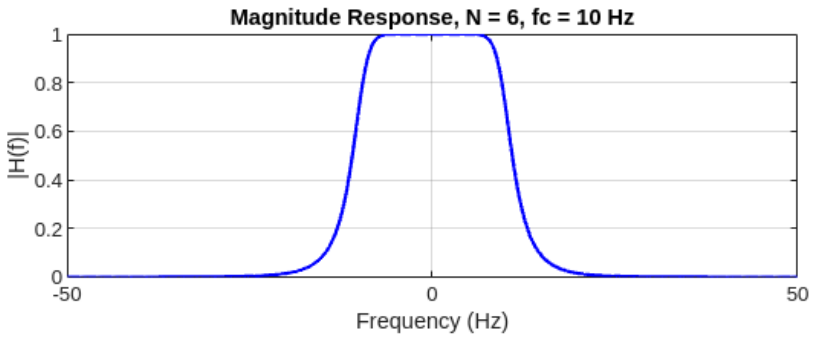


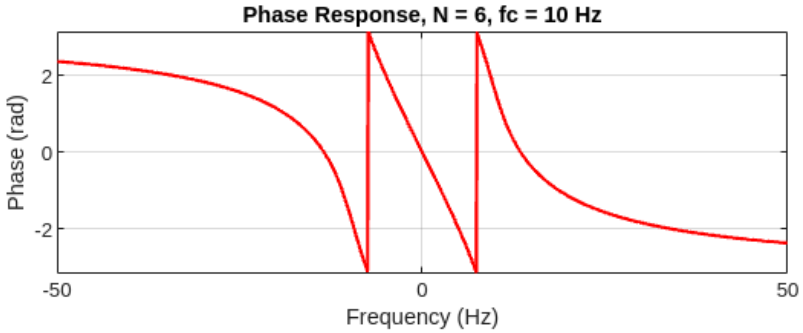


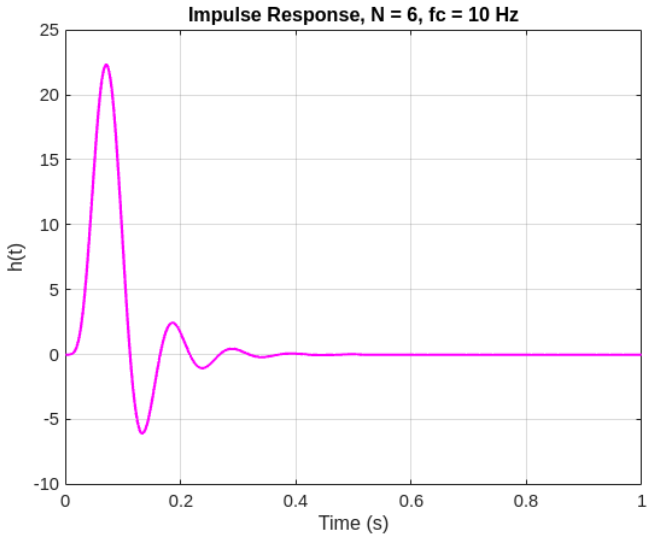
For N=6 (even case)

```
Computed Left-Half-Plane Poles:

Pole 1: Real = -16.2621, Imag = 60.6909
Pole 2: Real = -44.4288, Imag = 44.4288
Pole 3: Real = -60.6909, Imag = 16.2621
Pole 4: Real = -60.6909, Imag = -16.2621
Pole 5: Real = -44.4288, Imag = -44.4288
Pole 6: Real = -16.2621, Imag = -60.6909
```




Magnitude Response, N = 6, fc = 10 Hz
|H(f)|
1
0.8
0.6
0.4
0.2
0
-50
0
50
Frequency (Hz)


Phase Response, N = 6, fc = 10 Hz
Phase (rad)
2
0
-2
-50
0
50
Frequency (Hz)


Impulse Response, N = 6, fc = 10 Hz
h(t)
25
20
15
10
5
0
-5
-10
0
0.2
0.4
0.6
0.8
1
Time (s)

Observation:

When N=3 (odd), the filter has exactly one real pole on the negative real axis and a pair of complex-conjugate poles in the left-half plane. This real pole shows that odd order Butterworth filters always have one purely real solution. having a real pole means there is a direct “single exponential” component in the filter’s time-domain response. Meanwhile, the complex-conjugate poles indicate damped oscillations. For N=6 (even) all poles are in complex-conjugate pairs none of them are purely real. 6 number of poles leads to a steeper roll-off in the frequency domain but can also increase oscillatory behavior in the impulse response. In real-world analog filter implementations, these poles correspond to the physical reactive elements (capacitors/inductors or op-amp RC stages) needed to achieve the desired attenuation characteristic. A higher-order filter requires more stages and yields a sharper transition from passband to stopband.

Looking at the magnitude responses for both N=3 and N=6 (with the same cutoff frequency fc=10 Hz we see that the filter with N=3 has a somewhat gentler roll-off beyond 10 Hz. The passband is near unity gain up to about 10 Hz then it gradually attenuates signals above this frequency. But the N=6 filter has a noticeably sharper drop-off beyond 10 Hz, meaning it more aggressively suppresses higher-frequency components. In real applications, a sharper roll-off can be advantageous if you need to isolate a narrow passband and reject unwanted higher frequencies. However, it also tends to introduce more time-domain ringing or overshoot (discussed below).

The phase curves for N=6 have more abrupt jumps; this is because each additional pair of poles contributes a more rapid phase shift around the cutoff. In real terms, a large phase shift implies time-domain delay and potential distortion for signals with significant content around the cutoff. Designers must sometimes balance a steep magnitude response with the phase distortion it introduces.

The impulse responses show how each filter responds to a sudden “spike” input. For N=3 , we see a damped waveform that initially peaks, then oscillates slightly, and decays back to zero. This relatively short settling time means the filter introduces moderate ringing. When N=6 , the filter has more oscillations and a slightly higher initial peak, reflecting the higher-order poles. In a practical sense, this indicates that while the N=6 filter provides better frequency selectivity

(sharper roll-off), it also rings more in the time domain, which can momentarily distort signals during rapid transitions. Thus, higher-order filters are more selective but can introduce more overshoot and longer settling times. Engineers choose the filter order based on how quickly they need to attenuate unwanted frequencies versus how much time-domain distortion (ringing, overshoot) they can tolerate.

Overall, these plots demonstrate the classic Butterworth filter trade-off: as N increases, the passband remains maximally flat, but the transition band becomes steeper, and the impulse response becomes more oscillatory. In real-world terms, an odd-order filter always has one real pole, which slightly reduces the total oscillatory nature compared to an even-order filter of the same order. Meanwhile, the even-order filter exhibits no real poles and achieves a steeper slope but at the cost of more complex transient behavior.

Task 2)

Code:

```
clear; close all; clc;
```

```
fc    = 10;
omgc  = 2*pi*fc;
f     = -25:0.05:25;
omega = 2*pi*f;
```

```
Nodd = [3, 5, 7, 9];
lineStyles = {'-','--',':','-.'};
figure('Name','Butterworth LPF Response (Odd
Orders)','NumberTitle','off');
```

```
subplot(2,1,1); hold on;
title('Magnitude Response (Odd Orders)');
```

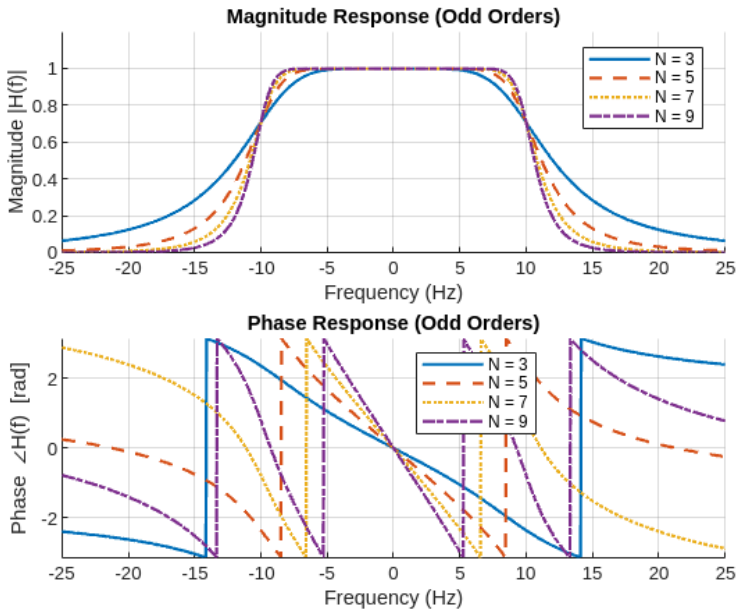
```
xlabel('Frequency (Hz)'); ylabel('Magnitude |H(f)|');
axis([-25,25,0,1.2]);
grid on;

subplot(2,1,2); hold on;
title('Phase Response (Odd Orders)');
xlabel('Frequency (Hz)'); ylabel('Phase \angleH(f) [rad]');
axis([-25,25,-pi,pi]);
grid on;

legendOdd = cell(length(Nodd),1);

for idx = 1:length(Nodd)
   N = Nodd(idx);
   k_idx = ceil(N/2) : floor(3*N/2);
   pls = omgc * exp(1i * pi * k_idx / N);
   [Hmag, Hphs] = spectrums([], pls, omgc^N, omega);
   subplot(2,1,1);
   plot(f, Hmag, lineStyles{idx}, 'LineWidth',1.5);

   subplot(2,1,2);
   plot(f, Hphs, lineStyles{idx}, 'LineWidth',1.5);

   legendOdd{idx} = ['N = ' num2str(N)];
end

subplot(2,1,1);
legend(legendOdd, 'Location','Best');
```

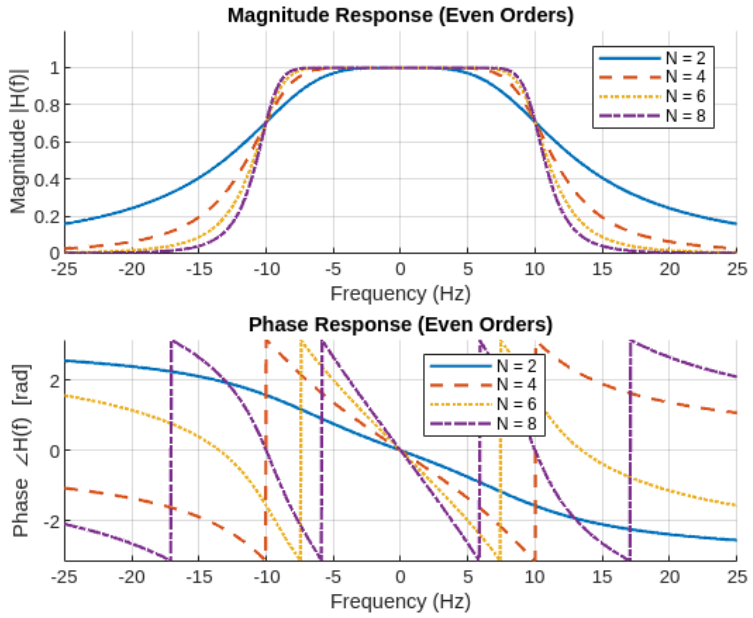
```
subplot(2,1,2);
legend(legendOdd, 'Location','Best');

Neven = [2, 4, 6, 8];
lineStylesEven = {'-','--',':','-.'};
figure('Name','Butterworth LPF Response (Even
Orders)','NumberTitle','off');

subplot(2,1,1); hold on;
title('Magnitude Response (Even Orders)');
xlabel('Frequency (Hz)'); ylabel('Magnitude |H(f)|');
axis([-25,25,0,1.2]);
grid on;

subplot(2,1,2); hold on;
title('Phase Response (Even Orders)');
xlabel('Frequency (Hz)'); ylabel('Phase \angleH(f) [rad]');
axis([-25,25,-pi,pi]);
grid on;

legendEven = cell(length(Neven),1);

for idx = 1:length(Neven)
   N = Neven(idx);

   k_idx = ceil(N/2) : floor((3*N-2)/2);
    pls = omgc * exp(1i * (2*k_idx + 1) * pi/(2*N));
```

```
    [Hmag, Hphs] = spectrums([], pls, omgc^N, omega);


    subplot(2,1,1);
    plot(f, Hmag, lineStylesEven{idx}, 'LineWidth',1.5);


    subplot(2,1,2);
    plot(f, Hphs, lineStylesEven{idx}, 'LineWidth',1.5);


    legendEven{idx} = ['N = ' num2str(N)];
end
```

```
subplot(2,1,1);
legend(legendEven, 'Location','Best');
subplot(2,1,2);
legend(legendEven, 'Location','Best');
```

Output:





Observation:

In case of odd orders, As N increases, we can see that the passband (roughly $|f| \lesssim 10|$ stays near one, showing that the maximally flat nature of Butterworth filters. The change from passband to stopband becomes steeper with higher NNN. Physically, this means a 7th- or 9th-order filter more aggressively attenuates signals above the cutoff frequency, which actually makes it better at isolating desired low frequencies. However, building such a high-order filter in hardware means more stages (e.g., more op-amp RC sections), increasing cost and complexity and also the power required. The phase curves for (N=3,5,7,9) show us a smooth transition from about 0 rad at low frequencies to approximately $-\pi \cdots -2\pi$ at higher frequencies. As N increases, we see more abrupt changes around the cutoff frequency (10 Hz). Physically, a filter's phase shift indicates time-domain delay, signals near the cutoff are delayed more. For large N, the phase transition is more sudden, so signals near the cutoff can experience stronger phase distortion.

In case of even orders A similar trend appears: the magnitude response remains near unity in the passband but drops off faster as N grows. For N=8, the roll-off is quite sharp, meaning the filter strongly suppresses any signal above ±10\pm. In real terms, a higher-order filter can be vital if we need minimal leakage of unwanted frequencies (like in communications or audio crossovers), but each additional order can introduce extra phase distortion and complexity. Similarly, (N=2,4,6,8) produce phase curves with steeper drops near ±10 we can notice multiple discontinuities (wraps) due to the way MATLAB plots phase from −π-\pi−π to π\piπ. In real analog circuits, this phase shift means signals at certain frequencies could be delayed by a significant fraction of a cycle, which can matter in timing-critical applications. A higher-order filter tends to have a faster transition in phase, correlating with a steeper magnitude roll-off.

These graphs show the classic Butterworth filter trade-off: higher orders yield flatter passbands and sharper cutoffs but at the expense of more rapid phase shifts and more hardware. Odd orders have that single real pole; even orders do not. Real engineers choose an order that meets their frequency isolation requirements while keeping phase distortion and hardware complexity within acceptable limits.

Task 3)

Code:

```
clear; close all; clc;
```

```
fc    = 10;
omgc  = 2*pi*fc;
T_end = 2;
dt    = 1e-3;
t     = 0:dt:T_end;
```

```
fs    = 1/dt;
Nt    = length(t);
```

```
x = sin(2*pi*3*t) + 0.5*sin(2*pi*30*t);
Nfft   = 2^nextpow2(Nt);
X_fft  = fft(x, Nfft);
X_shifted = fftshift(X_fft);
X_mag_2sided = abs(X_shifted)/Nt;
f_axis_2sided = (-Nfft/2 : Nfft/2-1) * (fs/Nfft);
```

```
N_values = [3, 4, 5, 6, 7, 11];
```

```
for idx = 1:length(N_values)
   N_order = N_values(idx);
   k = 1:N_order;
   allPoles = omgc * exp(1i*(pi/2 + (2*k - 1)*pi/(2*N_order)));
   LHP_poles = allPoles(real(allPoles) < 0);
   if length(LHP_poles) ~= N_order
       warning(['Expected ' num2str(N_order) ...
                ' LHP poles but found ' num2str(length(LHP_poles))
'.']);
   end
```

```
   disp(['--- Butterworth Poles (N = ' num2str(N_order) ') ---']);
   for p_idx = 1:length(LHP_poles)
       p_val = LHP_poles(p_idx);
       disp(['Pole ' num2str(p_idx) ': Real=' num2str(real(p_val)) ...
             ', Imag=' num2str(imag(p_val))]);
   end
```

```
   figure('Name',['Butterworth Poles (N=',
num2str(N_order),')'],'NumberTitle','off');
   scatter(real(LHP_poles), imag(LHP_poles), 80, 'r', 'filled');
   hold on; grid on; axis equal;
   xline(0, '--k'); yline(0, '--k');
   xlabel('Real Axis'); ylabel('Imag Axis');
   title(['Poles in LHP for N = ', num2str(N_order)]);
   denom = poly(LHP_poles);
   denom = real(denom);
   numer = omgc^N_order;
   sys   = tf(numer, denom);
   [h, t_imp] = impulse(sys, t);
   y = lsim(sys, x, t);
   figure('Name',['Task 3: Butterworth LPF (N=', num2str(N_order),')'],
...
          'NumberTitle','off');
```

```
   subplot(2,2,1);
   plot(t, x, 'b', 'LineWidth',1.5);
   grid on; xlabel('Time (s)'); ylabel('Amplitude');
   title('Composite Signal x(t)');
```

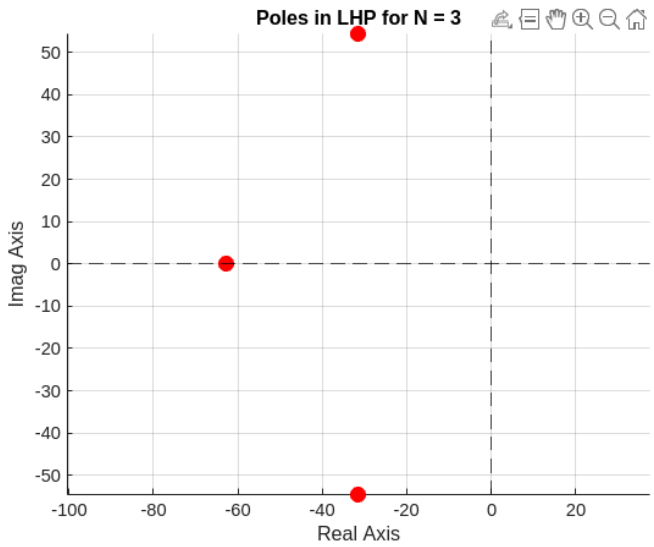
```
   subplot(2,2,2);
   plot(t_imp, h, 'r', 'LineWidth',1.5);
   grid on; xlabel('Time (s)'); ylabel('h(t)');
   title('Impulse Response');
```

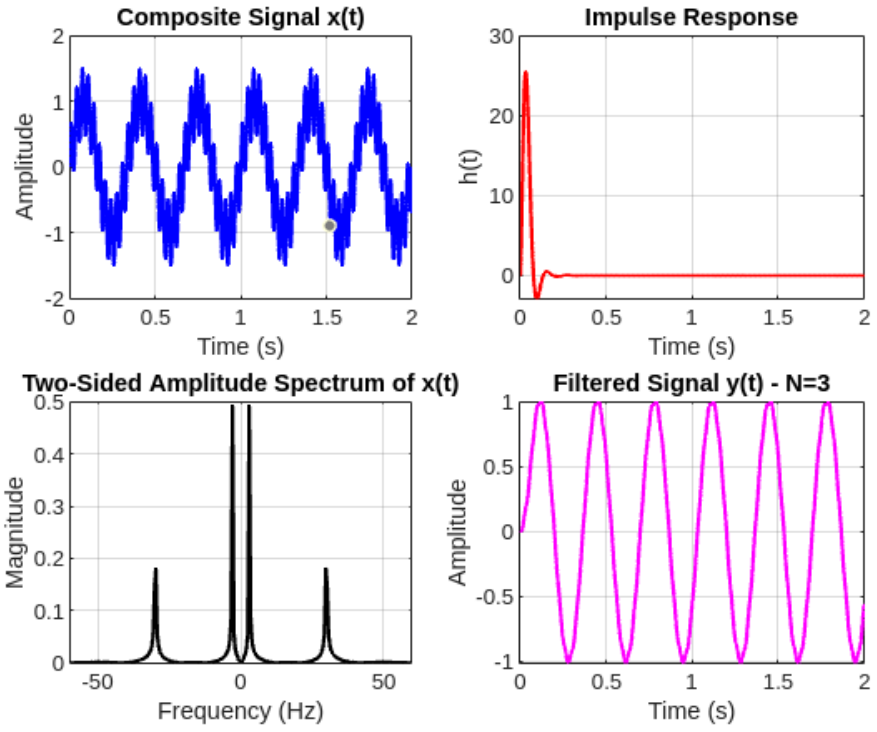
```
   subplot(2,2,3);
   plot(f_axis_2sided, X_mag_2sided, 'k', 'LineWidth',1.5);
```

```
    grid on; xlabel('Frequency (Hz)'); ylabel('Magnitude');
    title('Two-Sided Amplitude Spectrum of x(t)');
    xlim([-60 60]);

    subplot(2,2,4);
    plot(t, y, 'm', 'LineWidth',1.5);
    grid on; xlabel('Time (s)'); ylabel('Amplitude');
    title(['Filtered Signal y(t) - N=' num2str(N_order)]);
end

disp('All Butterworth LPF experiments complete!');
```

Outputs:

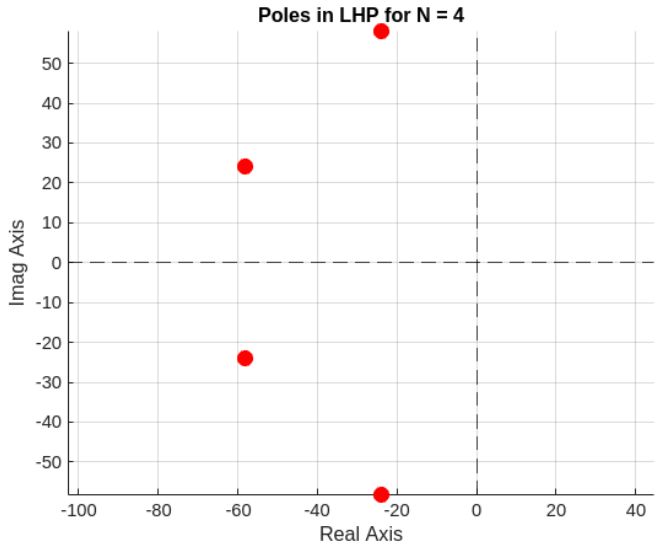
```
--- Butterworth Poles (N = 3) ---
Pole 1: Real=-31.4159, Imag=54.414
Pole 2: Real=-62.8319, Imag=7.6947e-15
Pole 3: Real=-31.4159, Imag=-54.414
```

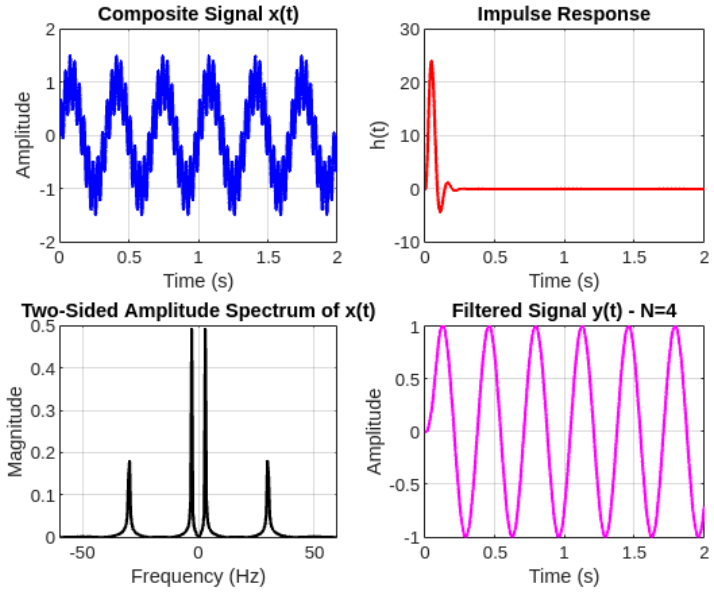




For n=4:

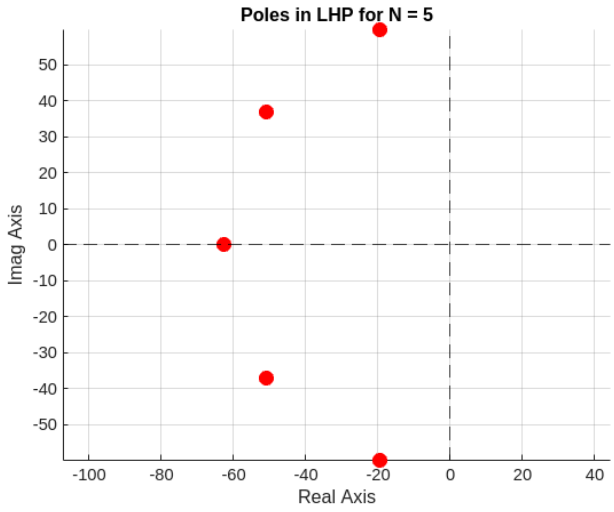
```
--- Butterworth Poles (N = 4) ---
Pole 1: Real=-24.0447, Imag=58.0491
Pole 2: Real=-58.0491, Imag=24.0447
Pole 3: Real=-58.0491, Imag=-24.0447
Pole 4: Real=-24.0447, Imag=-58.0491
```

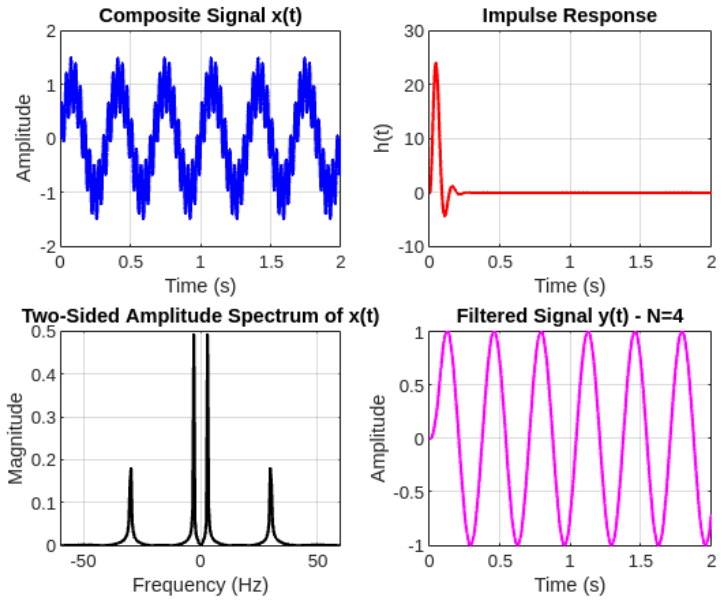




For n=5:

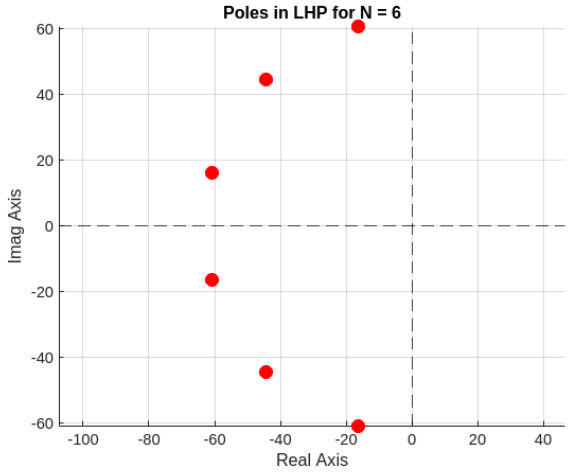
```
--- Butterworth Poles (N = 5) ---
Pole 1: Real=-19.4161, Imag=59.7566
Pole 2: Real=-50.832, Imag=36.9316
Pole 3: Real=-62.8319, Imag=7.6947e-15
Pole 4: Real=-50.832, Imag=-36.9316
Pole 5: Real=-19.4161, Imag=-59.7566
```

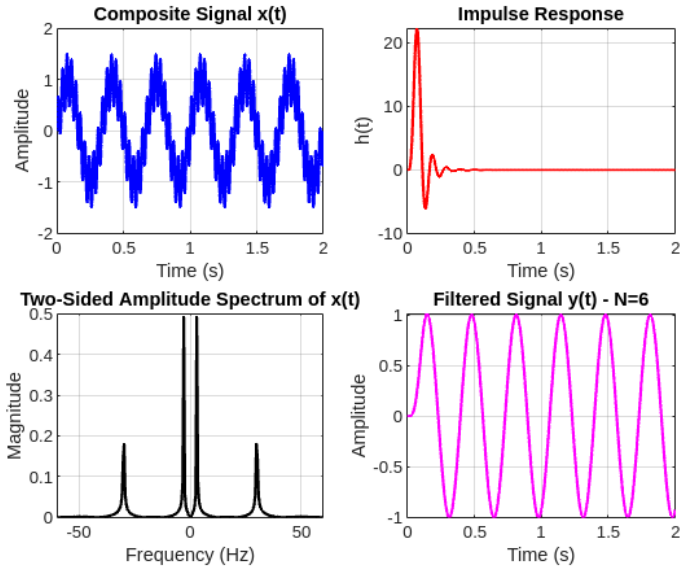




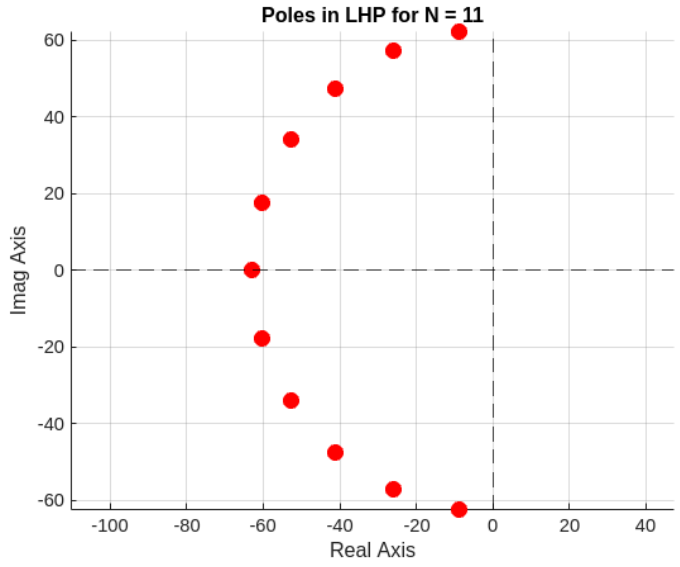
For n=6:

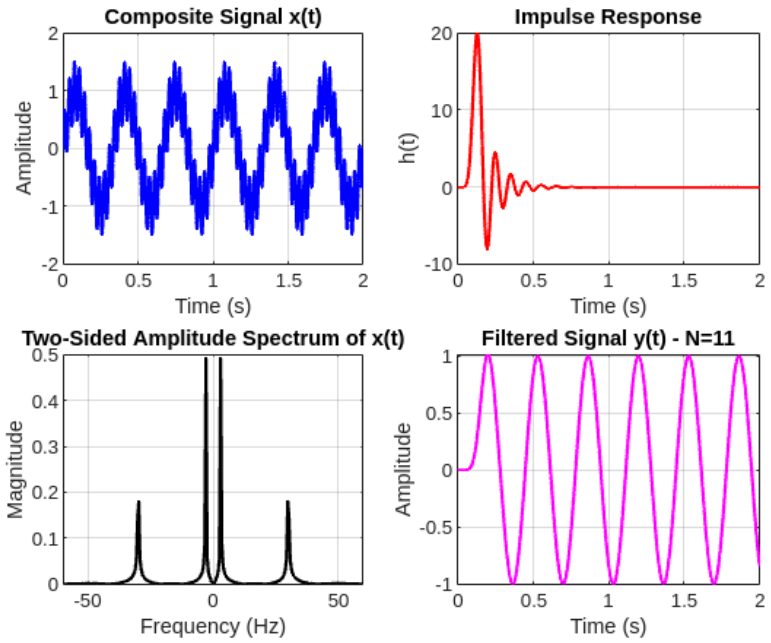
```
--- Butterworth Poles (N = 6) ---
Pole 1: Real=-16.2621, Imag=60.6909
Pole 2: Real=-44.4288, Imag=44.4288
Pole 3: Real=-60.6909, Imag=16.2621
Pole 4: Real=-60.6909, Imag=-16.2621
Pole 5: Real=-44.4288, Imag=-44.4288
Pole 6: Real=-16.2621, Imag=-60.6909
```






```
Pole 1: Real=-8.9419, Imag=62.1923
Pole 2: Real=-26.1013, Imag=57.1539
Pole 3: Real=-41.1461, Imag=47.4851
Pole 4: Real=-52.8575, Imag=33.9695
Pole 5: Real=-60.2867, Imag=17.7018
Pole 6: Real=-62.8319, Imag=7.6947e-15
Pole 7: Real=-60.2867, Imag=-17.7018
Pole 8: Real=-52.8575, Imag=-33.9695
Pole 9: Real=-41.1461, Imag=-47.4851
Pole 10: Real=-26.1013, Imag=-57.1539
Pole 11: Real=-8.9419, Imag=-62.1923
```





Observation:

The results from Task 3 show how a Butterworth lowpass filter of different orders affects a composite signal containing both low and high-frequency components. The first key observation comes from the poles of the transfer function, which are always in the left-half plane, ensuring stability. As the filter order increases, the poles become more densely packed along a semicircle, indicating a sharper frequency roll-off. This behavior is expected because higher-order filters require more poles to achieve a steeper transition between the passband and stopband. In practical terms, this means that increasing the filter order improves frequency selectivity, making it more effective at rejecting unwanted high-frequency signals, but at the cost of added complexity and potential signal distortion in the time domain.

Looking at the impulse response, we see a clear difference between filters of lower and higher orders. Lower-order filters, such as N=3, exhibit a relatively smooth and fast-decaying impulse response, meaning they settle quickly after an impulse input. However, as N increases, the impulse response shows more

oscillations before settling, which is a characteristic of higher-order filters introducing more resonant modes. This effect, often referred to as ringing, occurs because each additional pole contributes a damped oscillatory component to the response. In practical applications, this means that while higher-order filters provide better frequency selectivity, they may introduce transient distortions, which can be undesirable in systems requiring precise time-domain behavior, such as audio processing or real-time control applications.

The amplitude spectrum of the original signal provides insight into what frequencies are present before filtering. The composite signal consists of two frequency components: 3 Hz and 30 Hz. The spectrum clearly shows two peaks at these frequencies, representing the low and high-frequency components, respectively. The Butterworth filter is designed with a cutoff at 10 Hz, meaning that it should allow the 3 Hz signal to pass while significantly attenuating the 30 Hz component. After filtering, the frequency content of the signal changes as expected, with the 30 Hz peak being suppressed while the 3 Hz component remains dominant. The effectiveness of this suppression improves as NNN increases, which confirms that higher-order filters provide stronger attenuation of unwanted frequencies in the stopband.

Examining the time-domain output of the filtered signal further validates this behavior. For lower-order filters, such as N=3 or N=4, the filtered signal retains the general shape of a 3 Hz sinusoid, but some remnants of the 30 Hz component are still visible as minor distortions or high-frequency fluctuations. As N increases to values like N=7 or N=11, the 30 Hz component is almost completely removed, leaving behind a smooth 3 Hz sinusoid. This confirms that a higher-order filter provides better rejection of high-frequency content, ensuring a cleaner output signal. However, as seen in the impulse response, increasing NNN also introduces phase shifts and potential transient distortions, which might not be ideal in applications where phase integrity is crucial.

Overall, the results from Task 3 highlight the classic trade-offs in filter design. A higher-order Butterworth filter effectively removes unwanted high-frequency content, providing better signal clarity, but at the cost of increased complexity and potential signal distortion. In real-world applications, the choice of filter order depends on the specific requirements of the system, whether sharp frequency selectivity is more important than minimal distortion in the time domain. The observations confirm that Butterworth filters are well-suited for applications where

a smooth frequency response is needed with minimal ripple, making them a popular choice in audio processing, biomedical signal filtering, and communication systems where preserving the essential low-frequency components is critical.

Task 4)

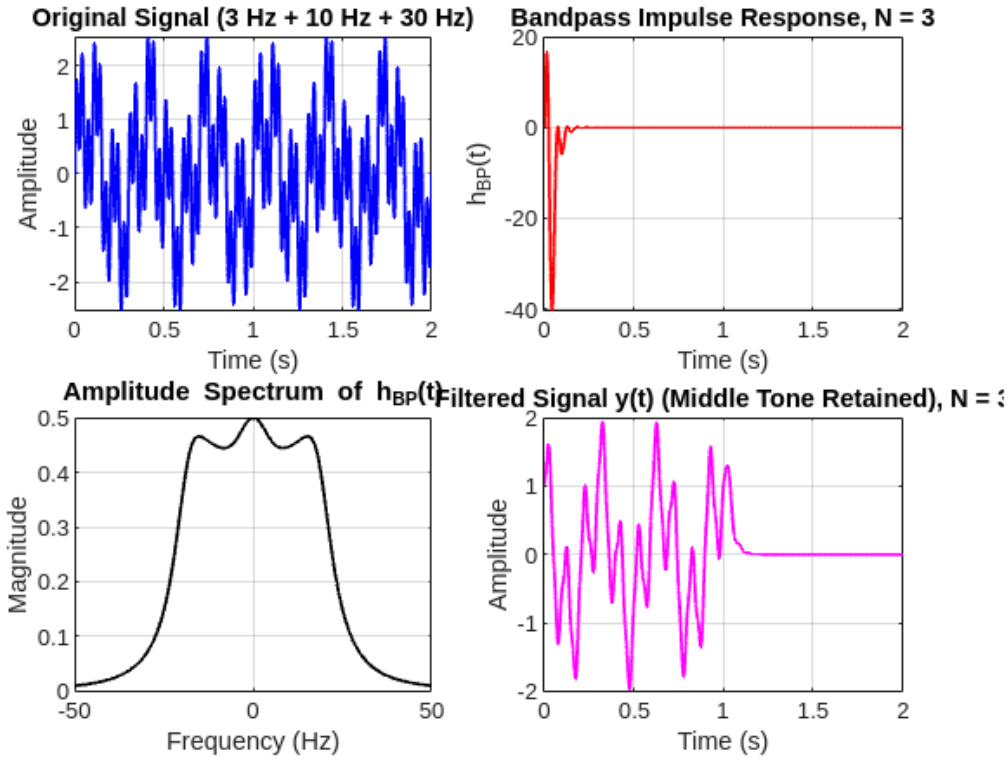
Code:

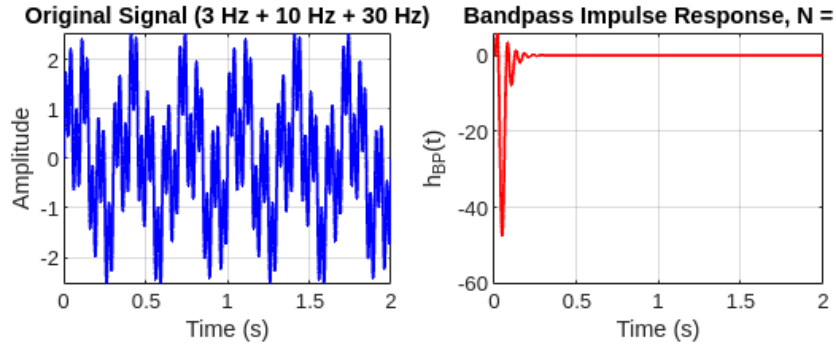
```
clear; close all; clc;

fc    = 10;
omgc  = 2*pi*fc;
f_mid = 10;
omega_mid = 2*pi*f_mid;

T_end = 2;
dt    = 1e-3;
t     = 0:dt:T_end;

fs = 1/dt;
x = sin(2*pi*3*t) + sin(2*pi*10*t) + sin(2*pi*30*t);
N_values = [3, 4, 5, 6, 7];

for idx = 1:length(N_values)
   N = N_values(idx);
   k_vec = 1:N;
   poles_lp = omgc * exp(1i*(pi/2 + (2*k_vec -
1)*pi/(2*N)));
   denom_lp = poly(poles_lp);
   denom_lp = real(denom_lp);
   gain_lp = omgc^N;
   sys_lp = tf(gain_lp, denom_lp);
   [h_lp, t_imp] = impulse(sys_lp, t);
   h_bp = 2 * h_lp .* cos(omega_mid * t_imp);
```

```
    y = conv(x, h_bp, 'same') * dt;
    Nfft = 2^nextpow2(length(h_bp));
    Hbp_fft = fft(h_bp, Nfft);
    Hbp_fft_shifted = fftshift(Hbp_fft);
    f_axis = (-Nfft/2:Nfft/2-1) * (fs/Nfft);
    Hbp_mag = abs(Hbp_fft_shifted) / length(h_bp);
    figure('Name',['Task 4: Bandpass Filter Experiment, N =
', num2str(N)], 'NumberTitle','off');

    subplot(2,2,1);
    plot(t, x, 'b', 'LineWidth',1.2);
    grid on;
    xlabel('Time (s)'); ylabel('Amplitude');
    title('Original Signal (3 Hz + 10 Hz + 30 Hz)');

    subplot(2,2,2);
    plot(t_imp, h_bp, 'r', 'LineWidth',1.2);
    grid on;
    xlabel('Time (s)'); ylabel('h_{BP}(t)');
    title(['Bandpass Impulse Response, N = ', num2str(N)]);

    subplot(2,2,3);
    plot(f_axis, Hbp_mag, 'k', 'LineWidth',1.2);
    grid on;
    xlabel('Frequency (Hz)'); ylabel('Magnitude');
    title('Amplitude Spectrum of h_{BP}(t)');
    xlim([-50 50]);

    subplot(2,2,4);
    plot(t, y, 'm', 'LineWidth',1.2);
    grid on;
    xlabel('Time (s)'); ylabel('Amplitude');
    title(['Filtered Signal y(t) (Middle Tone Retained), N =
', num2str(N)]);
```

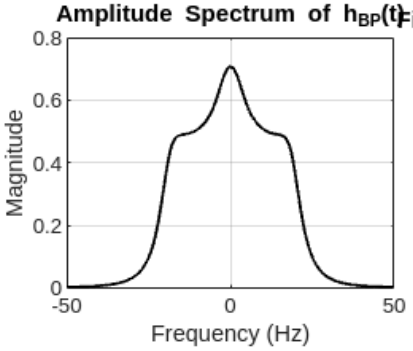
```
    disp(['Task 4: Experiment complete for N = '
num2str(N)]);
end

disp('All experiments for Task 4 complete!');
```

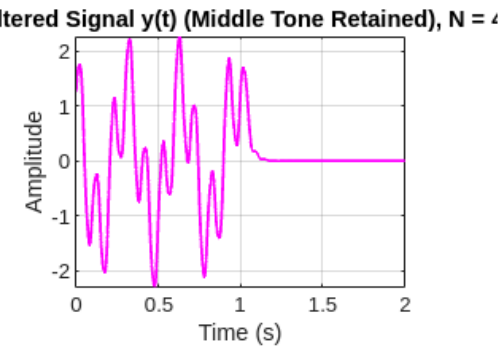
Output:

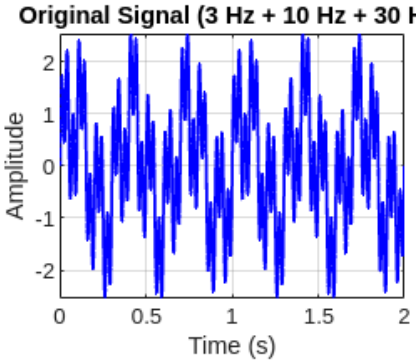


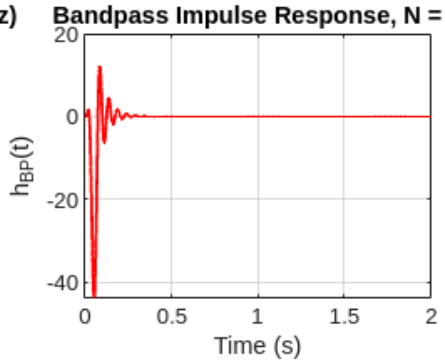

Original Signal (3 Hz + 10 Hz + 30 Hz)
Amplitude
Time (s)

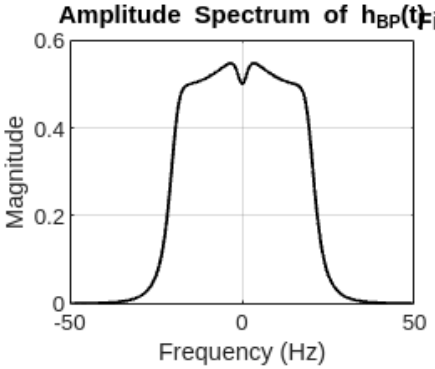
Bandpass Impulse Response, N = 4
$h_{BP}(t)$
Time (s)

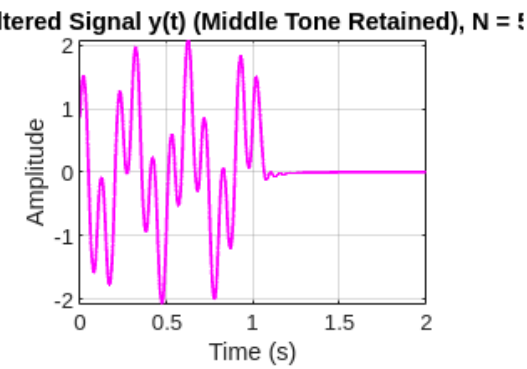

Amplitude Spectrum of $h_{BP}(t)$
Magnitude
Frequency (Hz)

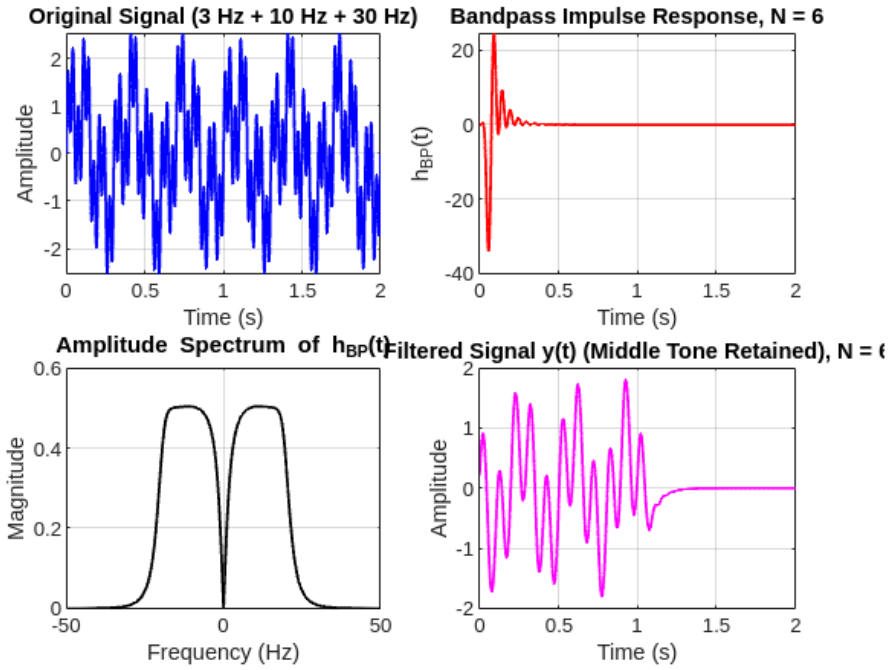

Filtered Signal y(t) (Middle Tone Retained), N = 4
Amplitude
Time (s)

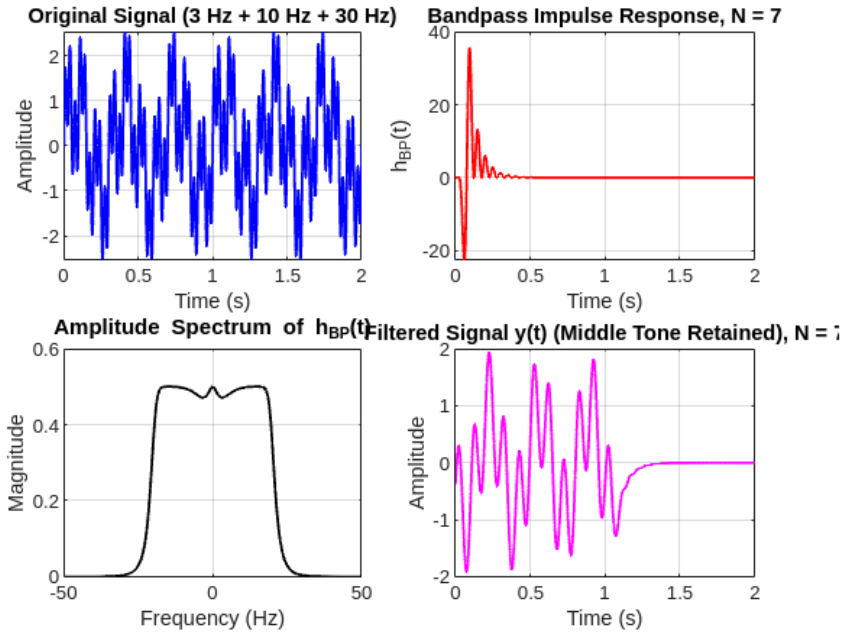

Original Signal (3 Hz + 10 Hz + 30 Hz)
Amplitude
Time (s)


Bandpass Impulse Response, N = 5
$h_{BP}(t)$
Time (s)


Amplitude Spectrum of $h_{BP}(t)$
Magnitude
Frequency (Hz)


Filtered Signal y(t) (Middle Tone Retained), N = 5
Amplitude
Time (s)


Original Signal (3 Hz + 10 Hz + 30 Hz)
Amplitude
Time (s)
Bandpass Impulse Response, N = 6
$h_{BP}(t)$
Time (s)
Amplitude Spectrum of $h_{BP}(t)$
Magnitude
Frequency (Hz)
Filtered Signal y(t) (Middle Tone Retained), N = 6
Amplitude
Time (s)


Original Signal (3 Hz + 10 Hz + 30 Hz)
Amplitude
Time (s)
Bandpass Impulse Response, N = 7
$h_{BP}(t)$
Time (s)
Amplitude Spectrum of $h_{BP}(t)$
Magnitude
Frequency (Hz)
Filtered Signal y(t) (Middle Tone Retained), N = 7
Amplitude
Time (s)

Observations:

In the first plot of each set, we observe the original signal composed of three sinusoidal components at 3 Hz, 10 Hz, and 30 Hz. The waveform appears as a complex oscillation due to the interference of these three frequencies. The presence of low-frequency (3 Hz), mid-frequency (10 Hz), and high-frequency (30 Hz) components results in a signal with visible periodic beats. In practical terms, this type of signal could represent a composite signal carrying different frequency components, such as an audio signal with different tones or an RF signal carrying multiple modulated signals. The goal of the bandpass filter is to isolate only the 10 Hz component while suppressing the other two.

The second plot in each set displays the bandpass impulse response which is derived from the lowpass impulse response via modulation with a cosine function at the center frequency (10 Hz). The impulse response initially has a high amplitude but gradually decays over time. As the filter order NNN increases, the impulse response exhibits more oscillations before settling. This phenomenon reflects the trade-off in filter design: higher-order filters provide a sharper frequency response but introduce longer transient effects. In real-world applications, this means that a higher-order bandpass filter would require more time to stabilize when filtering real-time signals, potentially causing distortion in applications requiring rapid transitions.

The third plot illustrates the amplitude spectrum of the bandpass filter, showing the frequencies that the filter retains and rejects. We can see that the peak of the spectrum is centered around 10 Hz, with symmetric peaks at -10 Hz due to the Fourier transform properties of real-valued signals. The width of the passband is determined by the filter order: higher orders create a sharper, more selective bandpass filter, while lower orders allow more leakage of unwanted frequencies. This means that for real-world filtering applications, a higher-order filter would be used in scenarios where precise frequency isolation is necessary (e.g., medical signal processing), whereas a lower-order filter might be sufficient for general audio or RF applications.

The fourth plot in each set shows the filtered signal y(t)y(t)y(t), which should ideally retain only the 10 Hz component while eliminating the 3 Hz and 30 Hz components. In the cases where NNN is low, we see slight distortions, indicating

some leakage from the suppressed frequencies. As NNN increases, the output signal becomes more sinusoidal, resembling a clean 10 Hz tone with minimal interference from the original 3 Hz and 30 Hz components. This demonstrates that a higher-order bandpass filter better isolates the target frequency but at the cost of a longer impulse response. In practical applications, such as communications or biomedical signal processing, selecting the right filter order is crucial to balancing frequency selectivity with system responsiveness.