Digital Signal Processing Quiz 1

Problem 1: Complex Representation of Sinusoids

We are given a voltage signal:

$$v(t) = 2.5\cos(2\pi ft - \pi/4)$$

where: - Amplitude $A = 2.5$, - Frequency $f = 1$ kHz, - Phase shift $\phi = 2\pi ft - \pi/4$.

(a) Show that $v(t) = \Re\{2.5e^{j\phi}\}$

Using Euler's formula:

$$e^{j\theta} = \cos\theta + j\sin\theta$$

we express $v(t)$ in complex exponential form:

$$\cos(\phi) = \Re\{e^{j\phi}\}$$

Multiplying by the amplitude:

$$v(t) = 2.5\Re\{e^{j\phi}\} = \Re\{2.5e^{j\phi}\}$$

(b) Show that $v(t) = \Im\{2.5e^{j(\phi-\pi/2)}\}$

Using the property:

$$\cos\theta = \sin(\theta + \pi/2)$$

we rewrite:

$$v(t) = 2.5\cos(\phi) = 2.5\sin(\phi + \pi/2) = \Im\{2.5e^{j(\phi-\pi/2)}\}$$

—

Problem 2: Sampling a Sine Wave

(a) Show that $\text{kHz}^{-1} = 1$ millisecond

Since f and T are inversely related:

$$T = \frac{1}{f}$$

For $f = 1$ kHz:

$$T = \frac{1}{1000} = 1 \text{ ms}$$

(b) If the period is 5 ns, find the frequency

Given:

$$T = 5 \text{ ns} = 5 \times 10^{-9} \text{ s}$$

$$f = \frac{1}{T} = \frac{1}{5 \times 10^{-9}} = 200 \text{ MHz}$$

(c) How many samples per period for $f = 5$ kHz, $f_s = 50$ kHz?

Samples per period:

$$\frac{f_s}{f} = \frac{50,000}{5,000} = 10 \text{ samples per period}$$

(d) If $\Delta t = 1/f_s = 0.002$ ms, how many samples per period?

Convert:

$$\Delta t = 0.002 \text{ ms} = 2 \times 10^{-6} \text{ s}$$

Period:

$$T = \frac{1}{5,000} = 2 \times 10^{-4} \text{ s}$$

Samples per period:

$$\frac{T}{\Delta t} = \frac{2 \times 10^{-4}}{2 \times 10^{-6}} = 100$$

—

Problem 3: Digitizing Voltages

(a) Voltage range per step

The range is 0 to 2.56 V and divided into 256 steps:

$$\Delta V = \frac{2.56}{256} = 0.01 \text{ V}$$

(b) What power of 2 gives 256?

$$2^8 = 256$$

Thus, 8 bits are used.

(c) If we double the number of bits, what is the new ΔV?

New steps:

$$2^{16} = 65536$$

New ΔV:

$$\frac{2.56}{65536} = 0.00003906 \text{ V} = 39.06 \text{ µV}$$

—

Problem 4: Sinusoidal Signal with Noise

We consider a signal:

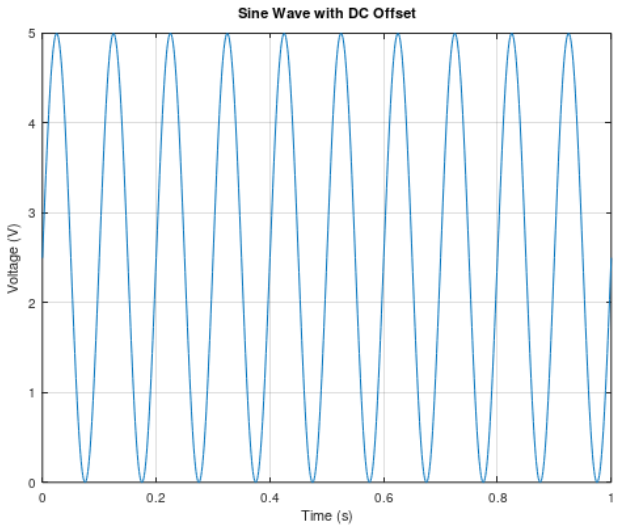
$$s(t) = 2.5 \sin(2\pi f t) + 2.5$$

where the amplitude is 2.5 V, the frequency is $f = 10$ Hz, and the sampling interval is $\Delta t = 1$ ms.

(a) Octave Code for Generating and Plotting the Signal

The following Octave code generates and plots $s(t)$:

```
t = 0:0.001:1; % Time from 0 to 1 sec, step 1 ms
f = 10;        % Frequency 10 Hz
s = 2.5 * sin(2 * pi * f * t) + 2.5; % Signal
plot(t, s);
xlabel('Time (s)');
ylabel('Voltage (V)');
title('Sine Wave with DC Offset');
grid on;
saveas(gcf, 'sine_wave.png'); % Save plot
```



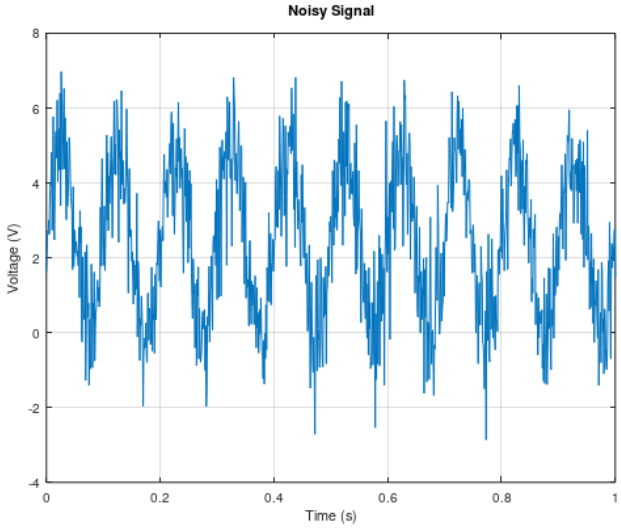
(b) Generate and Add Random Noise

We generate noise with mean 0 and standard deviation 1:

```
n = randn(size(t)); % Noise with mean 0, std 1
z = s + n; % Signal + Noise
```

(c) Plot the Signal with Noise

```
figure;
plot(t, z);
xlabel('Time (s)');
ylabel('Voltage (V)');
title('Noisy Signal');
grid on;
saveas(gcf, 'noisy_signal.png'); % Save plot
```



(d) Compute the Signal-to-Noise Ratio (SNR)

The SNR is calculated as:

$$\text{SNR} = 10 \log_{10} \left(\frac{\text{Power of signal}}{\text{Power of noise}} \right)$$

Octave code to compute SNR:

```
signal_power = mean(s.^2);
noise_power = mean(n.^2);
SNR = 10 * log10(signal_power / noise_power);
disp(['SNR: ', num2str(SNR), ' dB']);
```

Computed result:

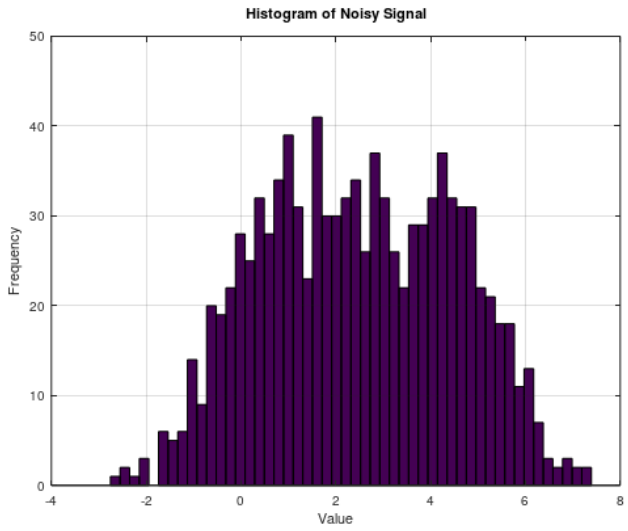
$$\text{SNR} = 9.905 \text{ dB}$$

(e) Histogram of Noisy Signal

To visualize the distribution of noise values, we generate a histogram:

```
figure;
hist(z, 50);
xlabel('Value');
ylabel('Frequency');
title('Histogram of Noisy Signal');
```

```
grid on;
saveas(gcf, 'histogram.png'); % Save plot
```



—

Problem 5: High-Pass RC Filter

$$R(f) = \frac{j\omega\tau}{1 + j\omega\tau}$$

where $\tau = RC$.

(a) Magnitude of $R(f)$

$$|R(f)| = \frac{\omega\tau}{\sqrt{1 + (\omega\tau)^2}}$$

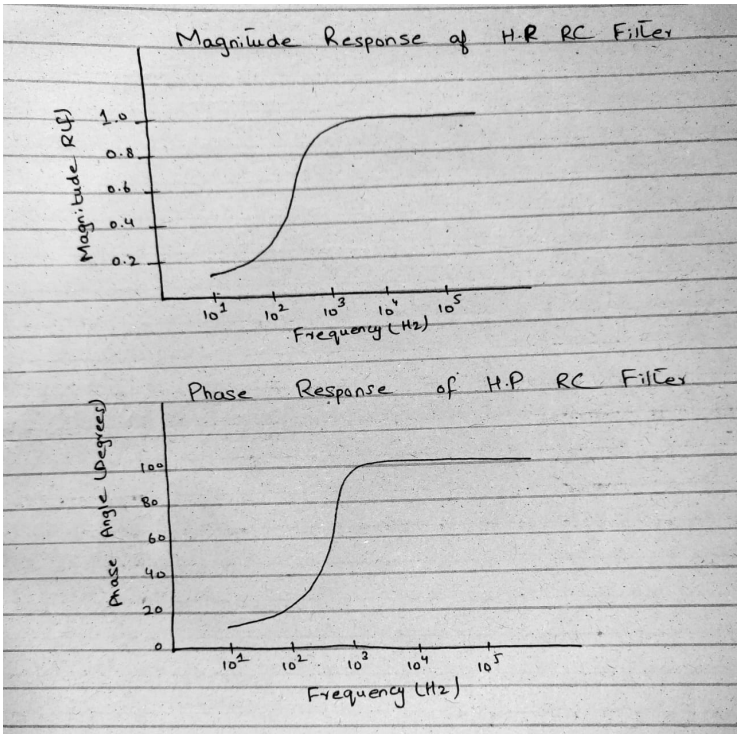
(b) Phase Angle

$$\theta = \tan^{-1}(\omega\tau)$$

(c) Graph the Magnitude and Phase Angle versus Frequency

The magnitude and phase response of the high-pass RC filter are plotted as a function of frequency. The magnitude response shows that low frequencies are

attenuated while high frequencies pass through. The phase response shifts from 0° at low frequencies to 90° at high frequencies.



(d) Compute Filtered Amplitude for $f = 0.5$ kHz

```
% Given values
R = 1e3;  % 1 k
C = 1e-6; % 1 F
f = 0.5e3; % 0.5 kHz
A = 1;
tau = R * C;
omega = 2 * pi * f;
Rf = (omega * tau) / sqrt(1 + (omega * tau)^2);
filtered_amplitude = A * Rf;
filtered_amplitude_squared = filtered_amplitude^2;

% Display results
disp(['Filtered Amplitude: ', num2str(filtered_amplitude)]);
disp(['(A(f)R(f))^2: ', num2str(filtered_amplitude_squared)]);
```

Computed result:

$$\text{Filtered Amplitude} = 0.95289$$

$$(A(f)R(f))^2 = 0.908$$

Problem 6: Aliasing and Sampled Frequency Calculation

The sampled frequency f_s' is determined by aliasing, using the formula:

$$f_s' = |f - Nf_s|$$

where:

- f is the original signal frequency,
- f_s is the sampling rate,
- $N = \text{round}(f/f_s)$ ensures we find the correct aliasing component.

(a) Given $f_s = 10$ kHz, $f = 2.5$ kHz

$$N = \text{round}\left(\frac{2.5}{10}\right) = 0$$

$$f_s' = |2.5 - (0 \times 10)|$$

$$f_s' = |2.5 - 0| = 2.5 \text{ kHz}$$

Answer: 2.5 kHz

(b) Given $f_s = 10$ kHz, $f = 5$ kHz

$$N = \text{round}\left(\frac{5}{10}\right) = 0$$

$$f_s' = |5 - (0 \times 10)|$$

$$f_s' = |5 - 0| = 5 \text{ kHz}$$

Answer: 5 kHz

(c) Given $f_s = 10$ kHz, $f = 15$ kHz

$$N = \text{round}\left(\frac{15}{10}\right) = 1$$

$$f_s' = |15 - (1 \times 10)|$$

$$f_s' = |15 - 10| = 5 \text{ kHz}$$

Answer: 5 kHz

(d) Given $f_s = 10$ kHz, $f = 20$ kHz

$$N = \text{round}\left(\frac{20}{10}\right) = 2$$

$$f_s' = |20 - (2 \times 10)|$$

$$f_s' = |20 - 20| = 0 \text{ kHz}$$

Answer: 0 kHz (aliasing makes it appear as DC)

—

Problem 7: System Transformation

The system S shifts the signal by half a period:

$$S[s(t)] = s(t - T/2)$$

where $T = 1/f$.

(a) Compute $S[s(t)]$

Given $s(t) = 2\sin(2\pi ft)$:

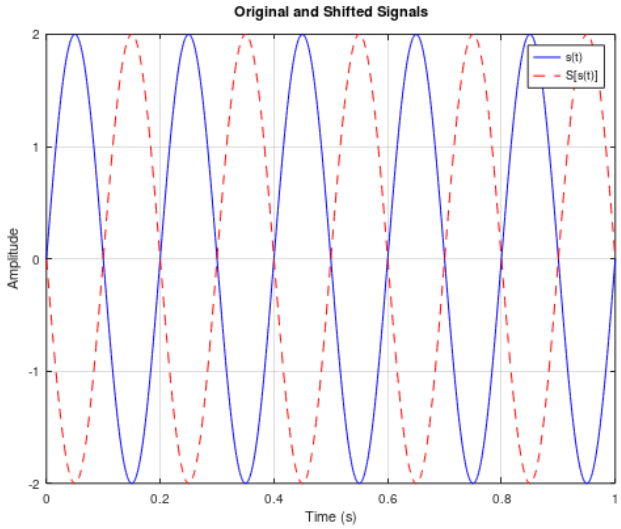
$$S[s(t)] = 2\sin(2\pi ft - \pi) = -2\sin(2\pi ft)$$

(b) Graph Input and Output

```
% Define time and frequency
t = 0:0.001:1; % 1 second, step 1 ms
f = 5; % 5 Hz sine wave

% Original signal
s = 2 * sin(2 * pi * f * t);

% Shifted signal
```

```
S_s = -s; % Equivalent to s(t - T/2)

% Plot both
figure;
plot(t, s, 'b', 'DisplayName', 's(t)'); hold on;
plot(t, S_s, 'r--', 'DisplayName', 'S[s(t)]');
xlabel('Time (s)');
ylabel('Amplitude');
title('Original and Shifted Signals');
legend;
grid on;
```

The input and output are phase-inverted sine waves.



(c) Compute $s(t) + S[s(t)]$

$$s(t) + S[s(t)] = 2\sin(2\pi ft) - 2\sin(2\pi ft) = 0$$

This confirms that the system is anti-symmetric.

—

Problem 8: System Properties

We analyze whether the system is linear or nonlinear.

(a) Given $y[n] = S(x[n]) = -x[n-1]$

This operation is a shift and negation, which is linear, as it satisfies:

$$S(x_1 + x_2) = -(x_1[n-1] + x_2[n-1]) = S(x_1) + S(x_2)$$
$$S(ax) = -ax[n-1] = aS(x)$$

Thus, this system is linear.

(b) Given $y[n] = S(x[n]) = (x[n])^2$

This operation is nonlinear, as:

$$S(ax) = (ax[n])^2 = a^2 x[n]^2 \neq aS(x)$$

Thus, this system is nonlinear.

(c) Answering the Linear vs. Nonlinear Question

- (a) Linear - (b) Nonlinear

Problem 9: Even or Odd Functions

A function f(t) is classified as: - Even if f(t) = f(t). - Odd if $f(-t) = -f(t)$.

(a) Function: $\cos(2\pi ft)$

Checking symmetry:

$$\cos(-2\pi ft) = \cos(2\pi ft)$$

Since $\cos(x)$ is an even function, we conclude:

$\cos(2\pi ft)$ is even.

(b) Function: $\exp(-(t/\sigma)^2)$

$$\exp(-(-t/\sigma)^2) = \exp(-(t/\sigma)^2)$$

Since the squared term remains unchanged, we conclude:

$\exp(-(t/\sigma)^2)$ is even.

(c) Function: $\exp(-\alpha t)$

$$\exp(-\alpha(-t)) = \exp(\alpha t)$$

Since $f(-t) \neq f(t)$ and $f(-t) \neq -f(t)$, the function is:

Neither even nor odd.

(d) Function: $at^2 + bt + c$

- at^2 is even since $(-t)^2 = t^2$. - bt is odd since $b(-t) = -bt$. - c (constant) is even.

Since the function contains both even and odd terms, it is:

Neither even nor odd.

—

Problem 10: Fourier Transform Properties

The Fourier Transform operator $\mathcal{F}$ has the following properties:

(a) Homogeneity

$$\mathcal{F}\{ax(t)\} = a\mathcal{F}\{x(t)\}$$

Since Fourier Transform is a linear operation, the homogeneity property holds.

(b) Additivity

$$\mathcal{F}\{x_1(t) + x_2(t)\} = \mathcal{F}\{x_1(t)\} + \mathcal{F}\{x_2(t)\}$$

Since Fourier Transform is linear, the additivity property holds.

(c) Shift-Invariance

$$\mathcal{F}\{x(t - t_0)\} = X(f)e^{-j2\pi f t_0}$$

A time shift introduces a phase shift in the frequency domain, proving that the Fourier transform is shift-invariant up to a complex exponential factor.

—

Problem 11: Fourier Transform of the Dirac Delta Function

The Dirac delta function $\delta(t - t_0)$ satisfies the sifting property:

$$f(t_0) = \int_{-\infty}^{\infty} f(t)\delta(t - t_0)dt$$

This means that multiplying any function by $\delta(t-t_0)$ and integrating simply evaluates the function at t_0.

(a) Fourier Transform of $a\delta(t - t_0)$

The Fourier Transform of a function $x(t)$ is defined as:

$$X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt$$

Applying this to $a\delta(t - t_0)$:

$$X(f) = \int_{-\infty}^{\infty} a\delta(t - t_0)e^{-j2\pi ft}dt$$

Using the sifting property:

$$X(f) = ae^{-j2\pi ft_0}$$

Thus, the Fourier Transform of $a\delta(t - t_0)$ is:

$$X(f) = ae^{-j2\pi ft_0}$$

(b) Magnitude of the Result

The magnitude of a complex function $X(f)$ is:

$$|X(f)| = |ae^{-j2\pi ft_0}|$$

Since $e^{-j2\pi ft_0}$ has unit magnitude:

$$|X(f)| = |a|$$

(c) Phase Angle of the Result

The phase angle $\theta(f)$ of a complex number $X(f)$ is:

$$\theta(f) = \arg(ae^{-j2\pi ft_0})$$

$$\theta(f) = \arg(a) - 2\pi ft_0$$

Thus, the phase angle of $X(f)$ is:

$$\theta(f) = -2\pi ft_0 + \arg(a)$$

—

Problem 12: Inverse Fourier Transform

We use the Fourier transform property:

$$\mathcal{F}^{-1}\{\delta(f - f_0)\} = e^{j2\pi f_0 t}$$

(a) Given:

$$F(f) = \frac{a}{2}(\delta(f - f_0) + \delta(f + f_0))$$

Applying the inverse Fourier Transform:

$$x(t) = \frac{a}{2}\left(e^{j2\pi f_0 t} + e^{-j2\pi f_0 t}\right)$$

Using Euler's identity:

$$e^{j\theta} + e^{-j\theta} = 2\cos(\theta)$$

$$x(t) = a\cos(2\pi f_0 t)$$

(b) Given:

$$F(f) = \frac{a}{2j}(\delta(f - f_0) - \delta(f + f_0))$$

Applying the inverse Fourier Transform:

$$x(t) = \frac{a}{2j}\left(e^{j2\pi f_0 t} - e^{-j2\pi f_0 t}\right)$$

Using Euler's identity:

$$e^{j\theta} - e^{-j\theta} = 2j\sin(\theta)$$

$$x(t) = a\sin(2\pi f_0 t)$$

—

Problem 13: Amplitude Modulation

(a) Express as Complex Exponentials

Using Euler's formula:

$$\cos(2\pi f t) = \frac{e^{j2\pi f t} + e^{-j2\pi f t}}{2}$$

$$A\cos(2\pi f_{LO} t) = A\frac{e^{j2\pi f_{LO} t} + e^{-j2\pi f_{LO} t}}{2}$$

$$\frac{m}{A}\cos(2\pi f_A t) = \frac{m}{A}\frac{e^{j2\pi f_A t} + e^{-j2\pi f_A t}}{2}$$

(b) Multiply the Two Functions

$$(A\cos(2\pi f_{LO}t))\left(\frac{m}{A}\cos(2\pi f_A t)\right)$$

Using:

$$\cos(x)\cos(y) = \frac{1}{2}[\cos(x-y)+\cos(x+y)]$$

$$A\cos(2\pi f_{LO}t)\times\frac{m}{A}\cos(2\pi f_A t) = \frac{m}{2}[\cos(2\pi(f_{LO}-f_A)t)+\cos(2\pi(f_{LO}+f_A)t)]$$

Thus, the resulting signal contains two frequencies:

$$f_{LO}-f_A, \quad f_{LO}+f_A$$

—

1 Code Projects

1.1 Echo Effect in Audio Signals

This project generates an echo effect in an audio signal. The sampling frequency is 20 kHz, and the echo occurs every 0.25 seconds, with each echo being half the amplitude of the previous one. Finally, a sine tone is convolved with the echo response to produce the final output.

Octave Code for Echo Effect

```
% Sampling parameters
fs = 20000;  % 20 kHz sampling rate
duration = 2;  % 2 seconds
N = fs * duration;  % Total number of samples

% (a) Create delta function [n]
delta = zeros(1, N);
delta(1) = 1;  % Impulse at n=0

% (b) Create echoes every 0.25 sec
echo_interval = 0.25 * fs;  % 5000 samples
num_echoes = floor(N / echo_interval);  % Number of echoes

% (c) Decaying echoes with half the amplitude
echo_signal = zeros(1, N);
for i = 1:num_echoes
    echo_signal(1 + i * echo_interval) = 0.5^i;  % Half amplitude each echo
end
```

```
% (d) Generate sine tone
t = (0:N-1) / fs;  % Time vector
f_sine = 500;  % Frequency of sine wave (500 Hz)
sine_wave = sin(2 * pi * f_sine * t);
sine_wave(N/4:end) = 0;  % Zeroing after 1/4 of the duration

% (e) Convolve sine wave with echo response
output_signal = conv(sine_wave, echo_signal, 'same');

% Play the result
sound(output_signal, fs);

% Save the result as a WAV file
audiowrite('echo_output.wav', output_signal, fs);
```

Download the Output Audio

Click the link below to download the generated echo audio file.

Download Echo Audio Files (MEGA Link)

—

1.2 Amplitude Modulation and Filtering

This project creates an amplitude-modulated (AM) signal with added noise. A bandpass filter is applied to eliminate unwanted noise. The spectrum is analyzed using the FFT function.

Octave Code for AM Signal and Filtering

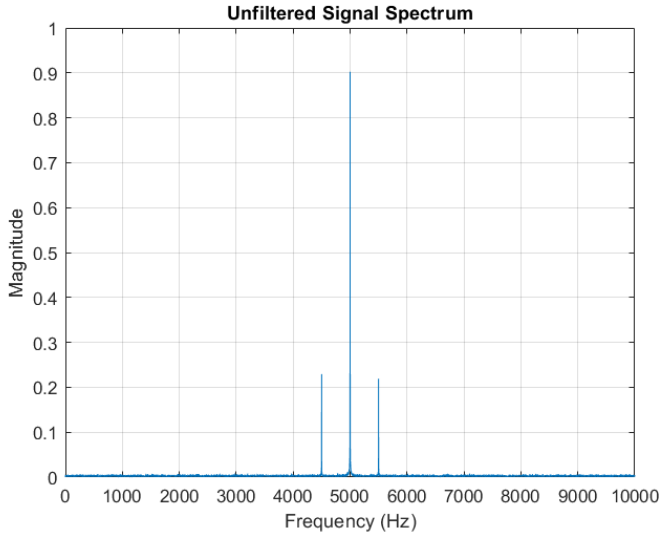
```
% Sampling parameters
fs = 20000;  % 20 kHz sampling rate
t = 0:1/fs:1;  % 1 second duration

% Carrier and modulating frequencies
fc = 5000;  % Carrier frequency (5 kHz)
fm = 500;   % Modulating frequency (500 Hz)
m = 0.5;    % Modulation index

% (1) Generate AM signal
modulating_signal = cos(2 * pi * fm * t);
carrier_signal = cos(2 * pi * fc * t);
am_signal = (1 + m * modulating_signal) .* carrier_signal;

% (2) Add Gaussian noise
noisy_signal = am_signal + 0.2 * randn(size(am_signal));
```

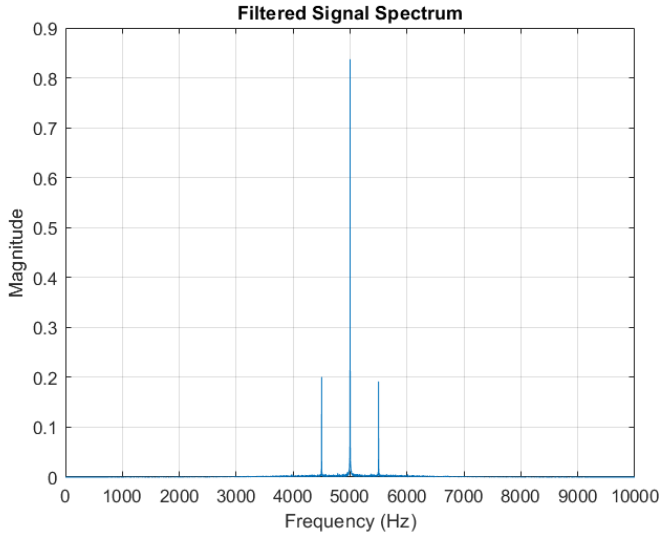
```
% (3) Compute FFT of unfiltered signal
L = length(t);
f = fs * (0:(L/2)) / L;
Y = fft(noisy_signal);
P2 = abs(Y / L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);

% Plot spectrum of unfiltered signal
figure;
plot(f, P1);
title('Unfiltered Signal Spectrum');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
grid on;

% (4) Apply bandpass filter (High-pass + Low-pass)
lowpass_cutoff = 6000;  % 6 kHz cutoff
highpass_cutoff = 4000; % 4 kHz cutoff

% Design filters
[b_low, a_low] = butter(4, lowpass_cutoff/(fs/2), 'low');
[b_high, a_high] = butter(4, highpass_cutoff/(fs/2), 'high');

% Apply filters
filtered_signal = filter(b_low, a_low, noisy_signal);
filtered_signal = filter(b_high, a_high, filtered_signal);

% (5) Compute FFT of filtered signal
Y_filtered = fft(filtered_signal);
P2_filtered = abs(Y_filtered / L);
P1_filtered = P2_filtered(1:L/2+1);
P1_filtered(2:end-1) = 2*P1_filtered(2:end-1);

% Plot spectrum of filtered signal
figure;
plot(f, P1_filtered);
title('Filtered Signal Spectrum');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
grid on;

% Play the filtered sound
sound(filtered_signal, fs);

% Save the filtered audio
```

```
audiowrite('filtered_output.wav', filtered_signal, fs);
```

Spectral Analysis Plots

Below are the spectra of the unfiltered and filtered signals.





Download the Output Audio

Click the link below to download the filtered AM signal file.

Download Filtered Audio Files (MEGA Link)