GEBZE TECHNICAL UNIVERSITY
ENGINEERING FACULTY
ELECTRONICS ENGINEERING
DEPARTMENT

ELEC - 361
Analog Communication Systems
Project

Adı – Soyadı	BÜNYAMİN BERAT GEZER
Numarası	210102002061

2) Simulation Result

1) For a step the results

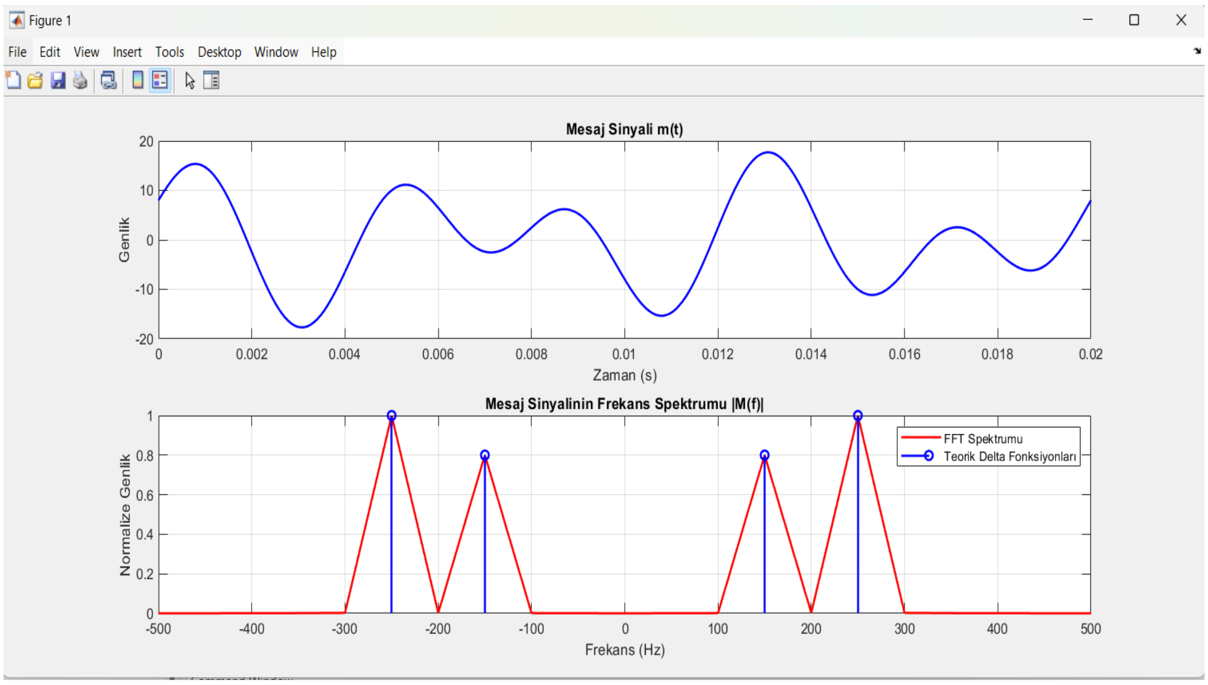

Figure 1 : Message signal for one period and the magnitude spectrum.

2) For b step the results (DSB-SC-AM)

- step b-i results:

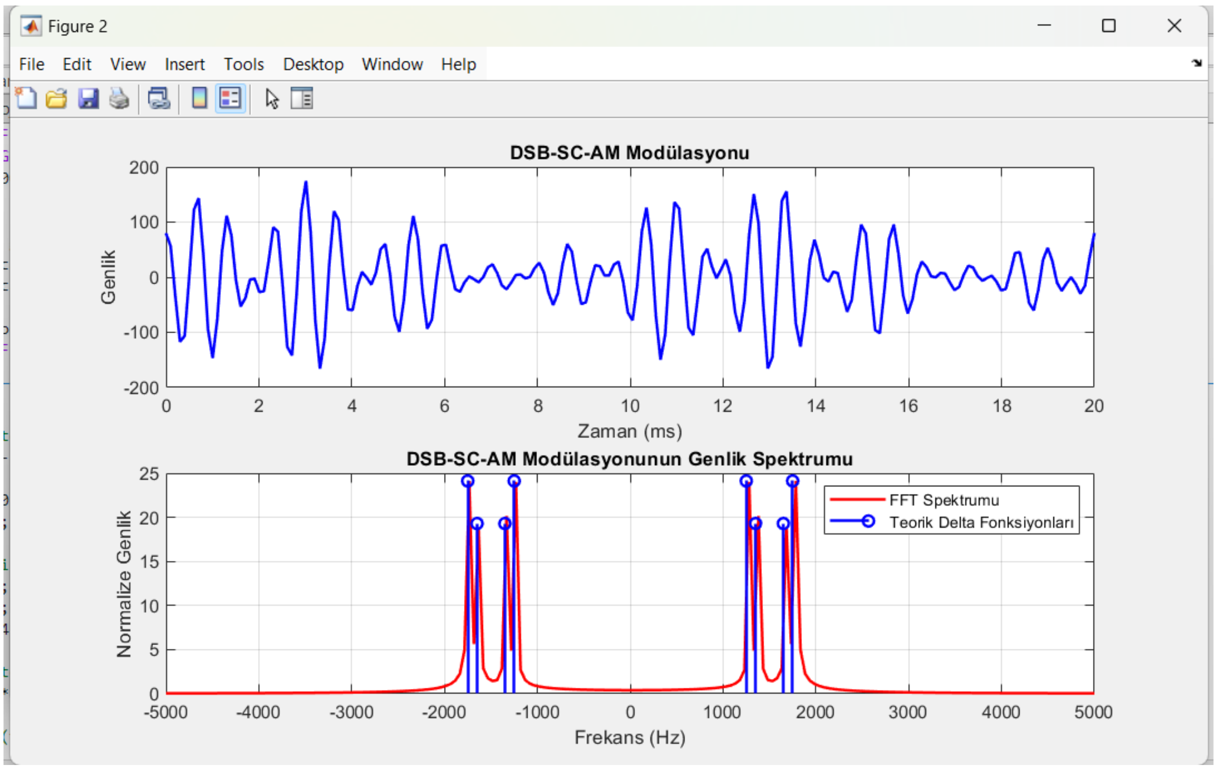

Figure 2 : The modulated signal and its spectrum.

- step b-ii results:

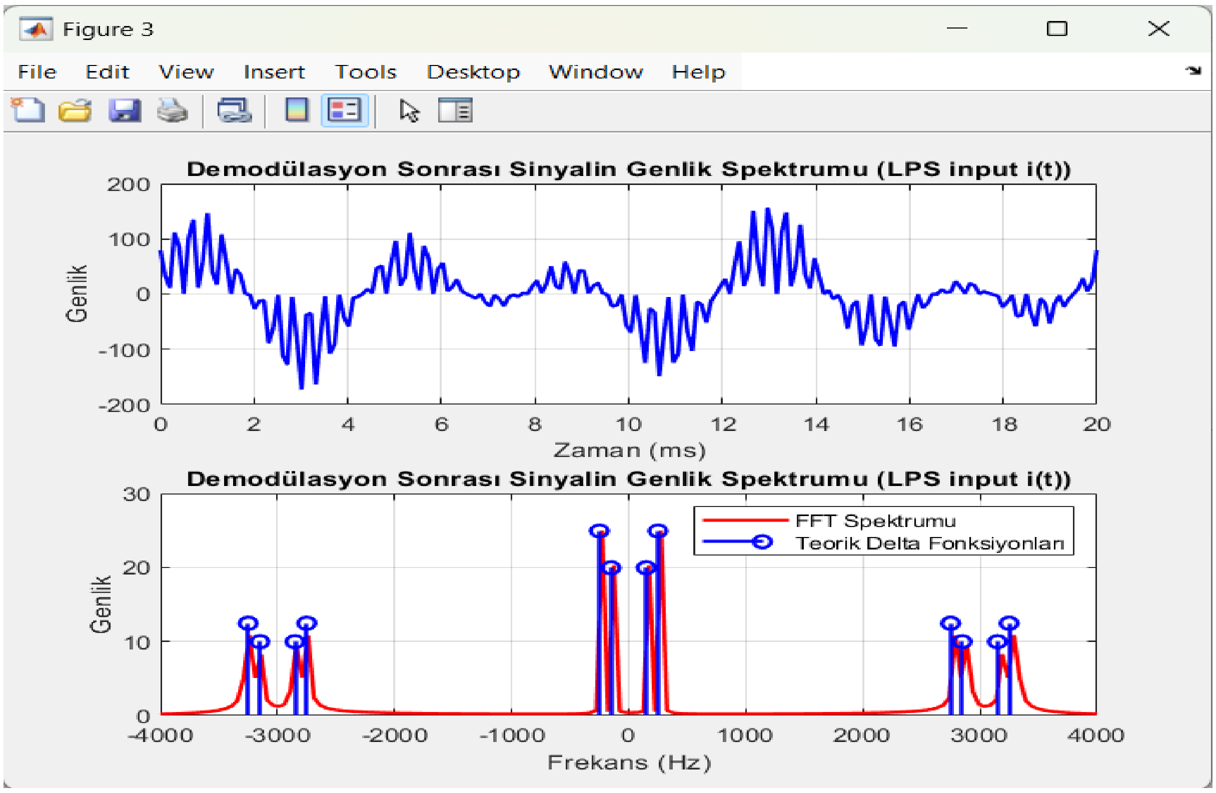

Figure 3: Amplitude and Signal Spectrum After Demodulation (LPS input i(t)).

- step b-iii results:

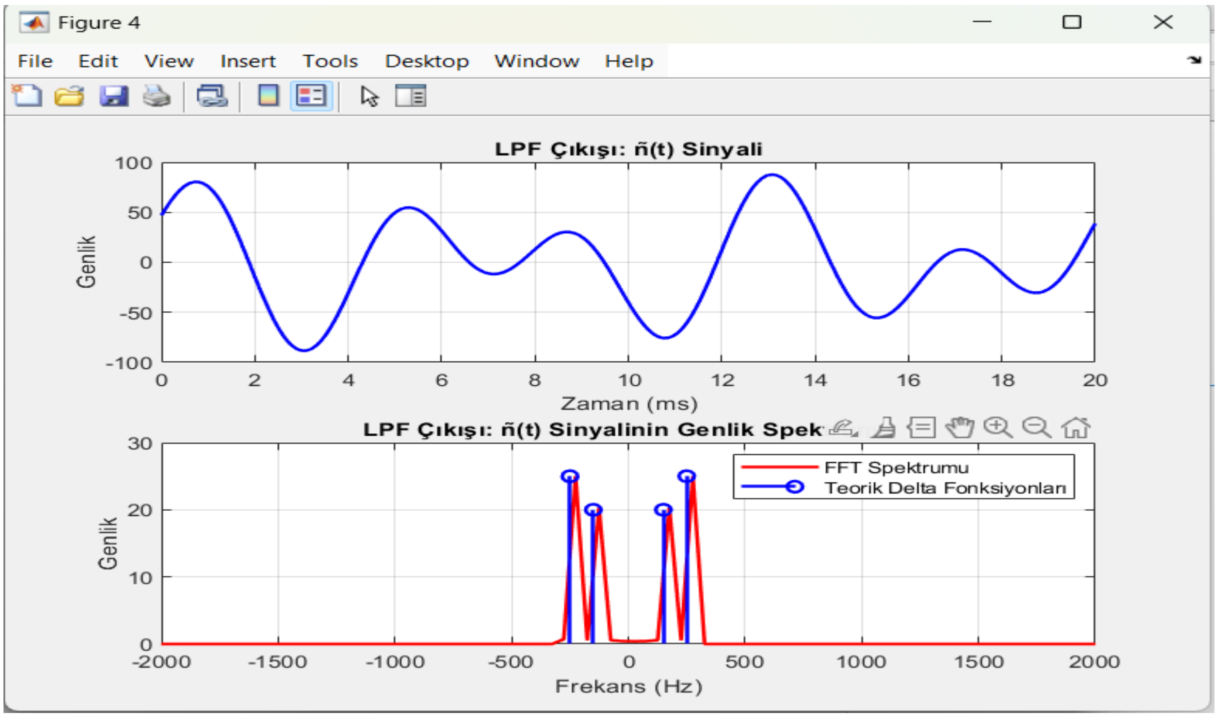

Figure 4: LPF Output: ñ(t) Signal and Its Amplitude Spectrum.

3)For c step the results (DSB-LC-AM)

- step c-i results:

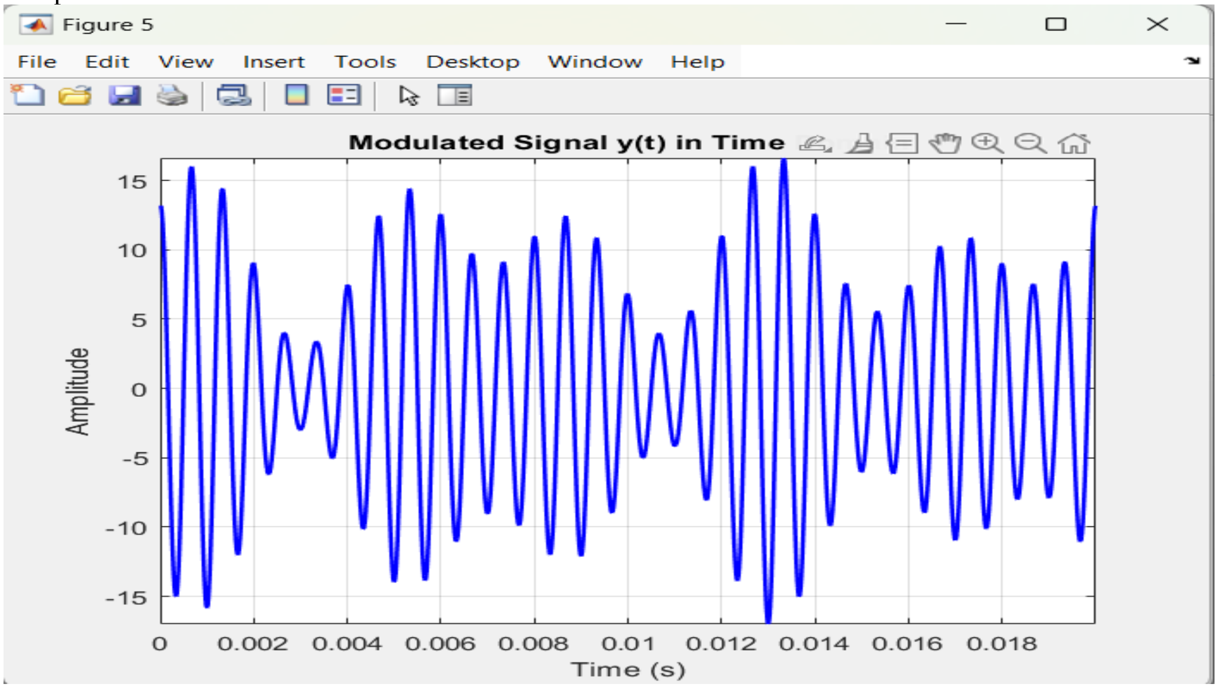

Figure 5: Modulated signal y(t)

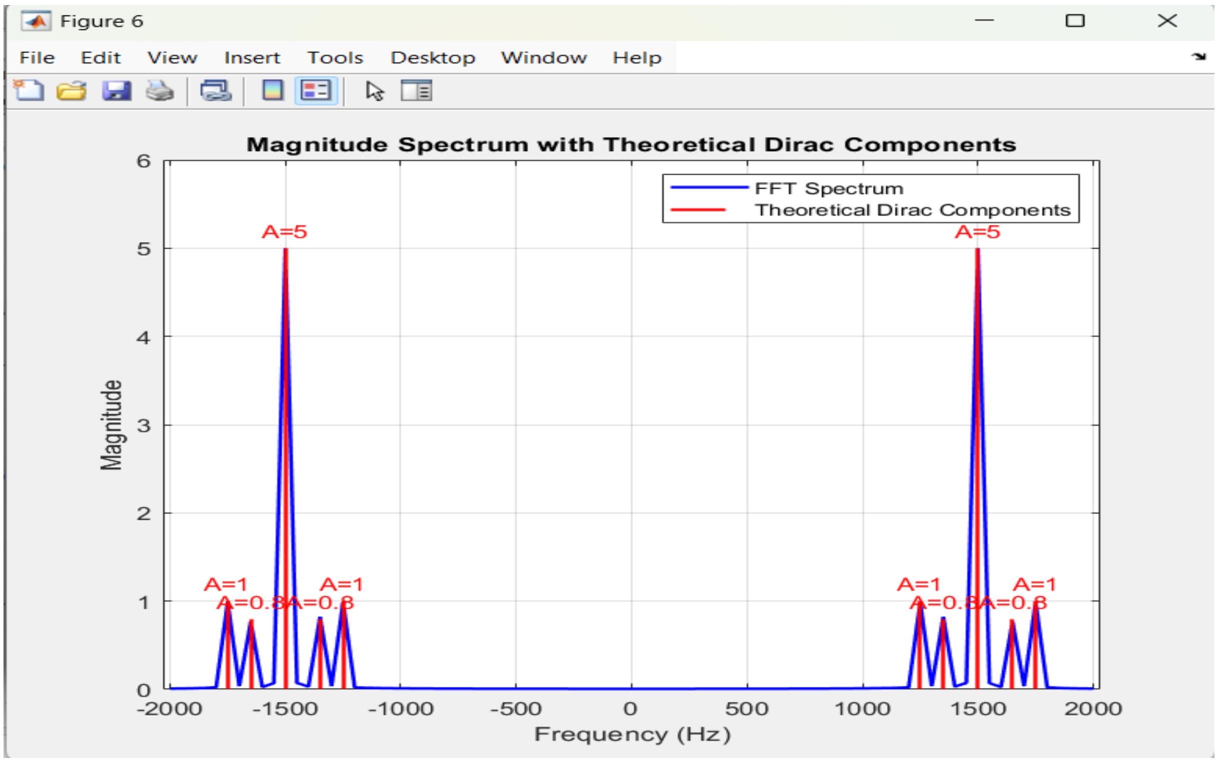

Figure 6: Modulated Signal y(t) its Amplitude Spectrum.

- step c-ii results:

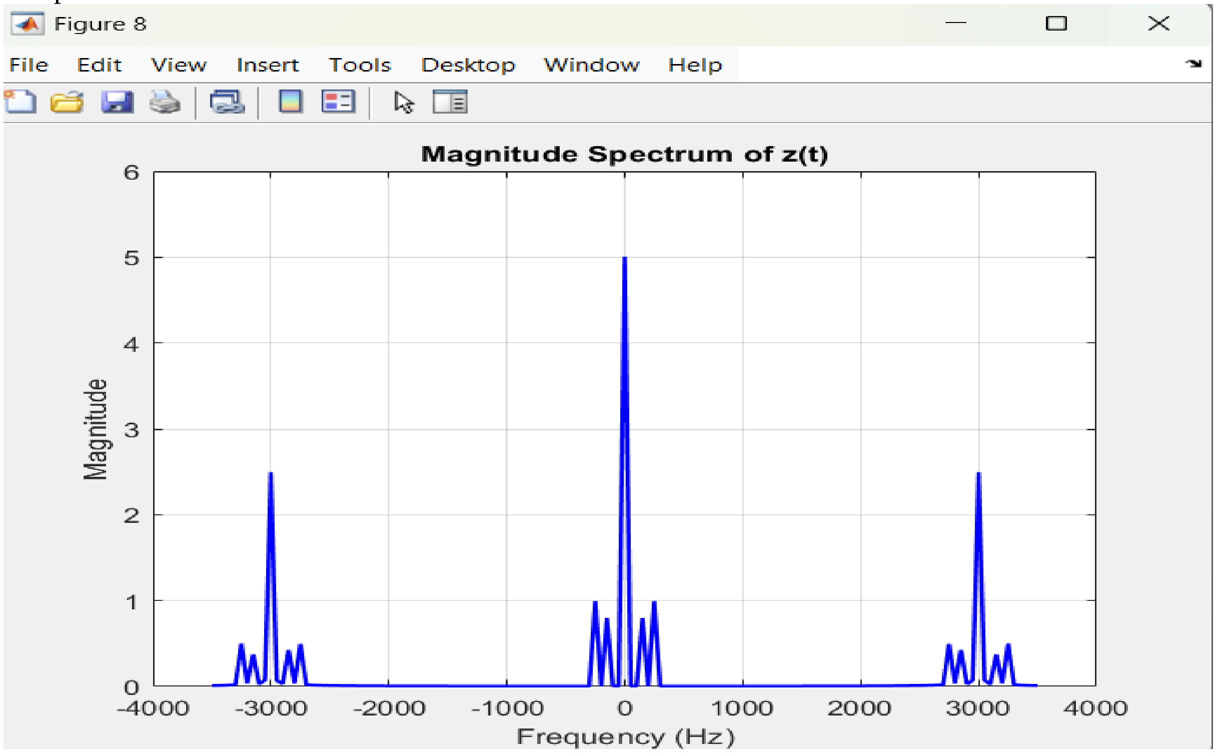

Figure 7: Demodulated Signal z(t) its Amplitude Spectrum.

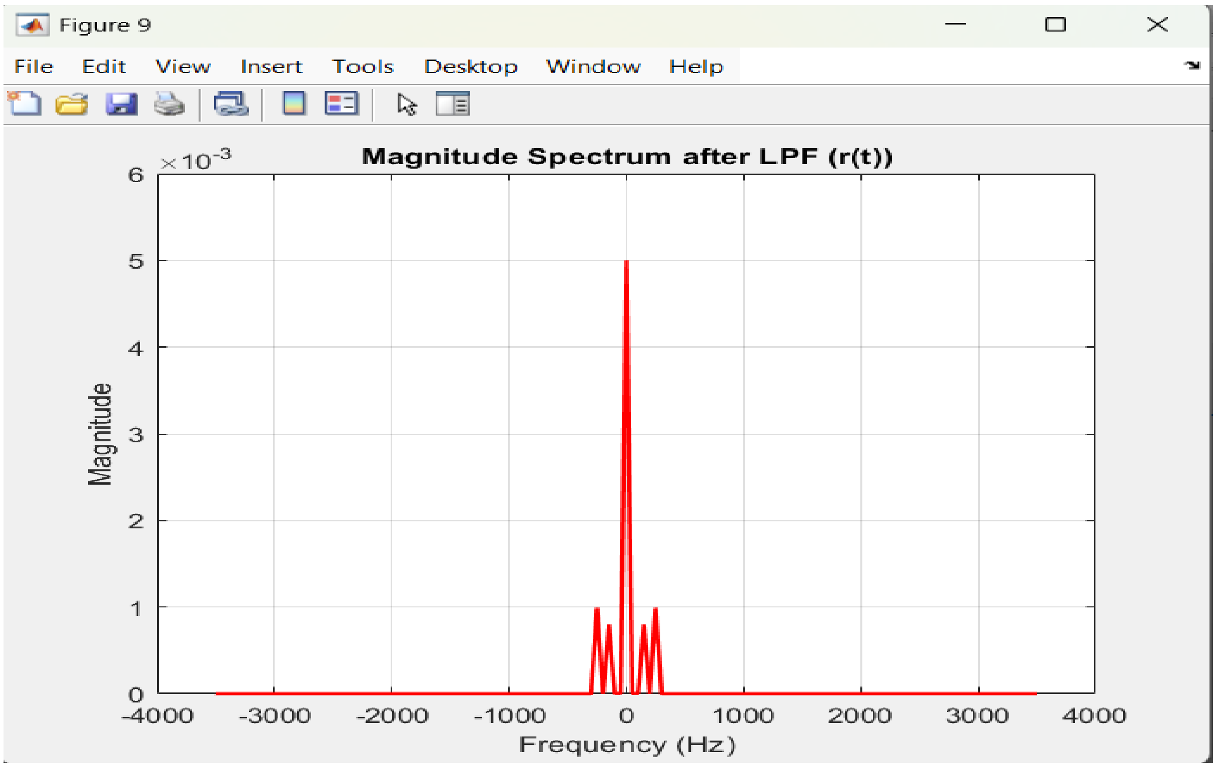

Figure 8: After signal z(t) passes LPF , Signal r(t) its Amplitude Spectrum.

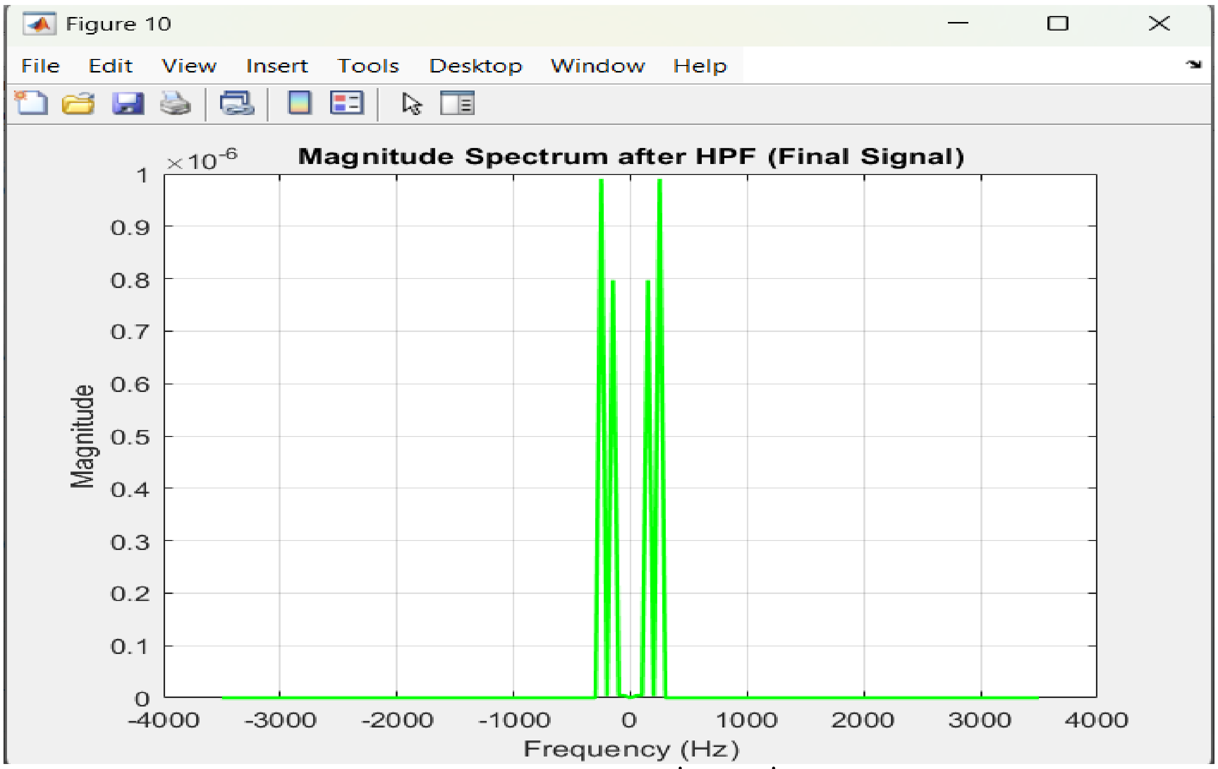

Figure 9: After signal r(t) passes HPF, Signal FİNAL SİGNAL its Amplitude Spectrum.

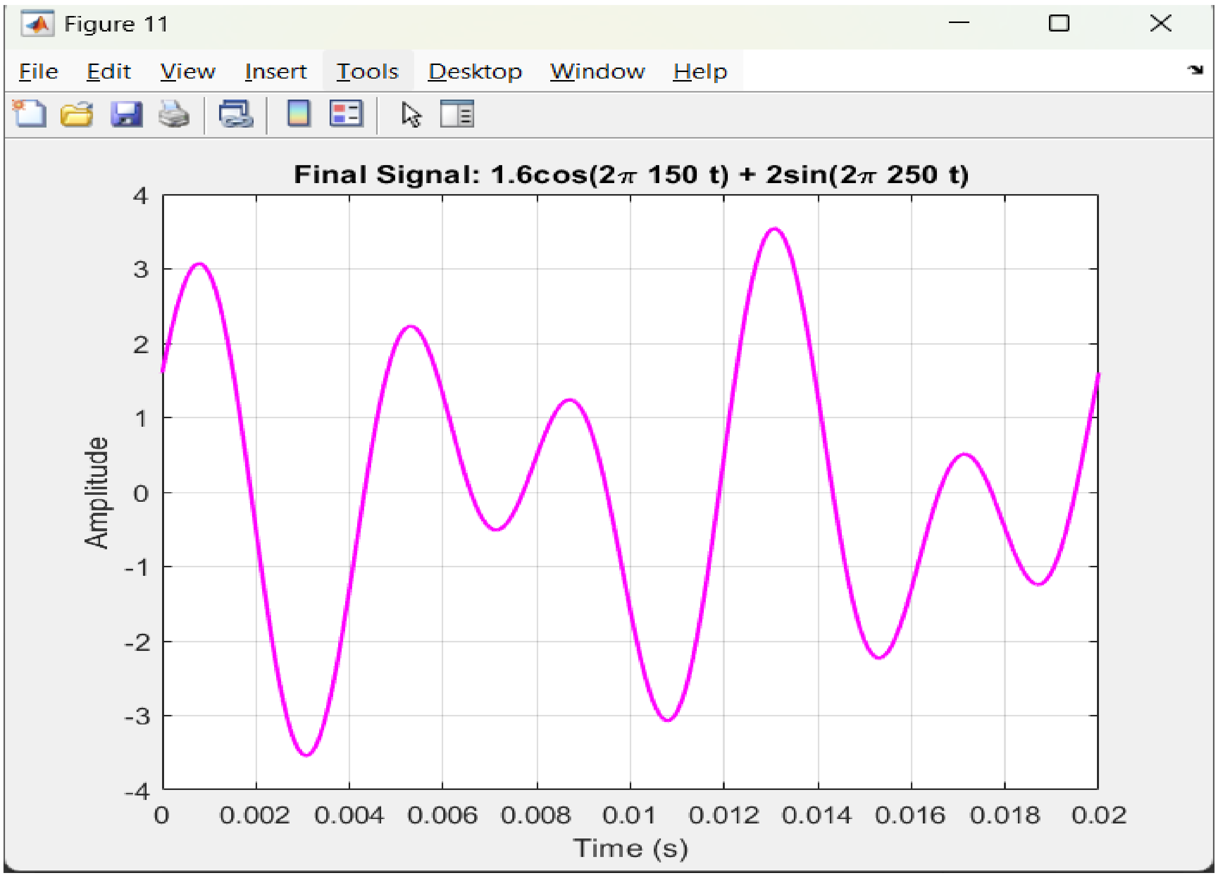

Figure 10: Final signal representation in the time domain.

3)Comparison of Analytical and Simulation Results and Conclusions

- The primary objective of this project was to analyze the modulation of a message signal with a carrier signal and its subsequent demodulation using both analytical and simulation methods. Initially, the message signal m(t)=8cos(300πt)+10sin(500πt) was analyzed analytically and visualized in the time and frequency domains for one period using MATLAB. In the analysis of the DSB-SC-AM modulation method, the modulated signal and its spectrum were thoroughly examined. During the demodulation process, when the carrier signal was correctly regenerated, no DC component was observed at the output of the low-pass filter (LPF), and the message signal was successfully recovered.

  For DSB-LC-AM modulation, the process was carried out with a modulation index of μ=0.7The demodulation was performed using the envelope detection method, and the DC component was successfully removed, resulting in the recovery of a clean message signal. In both modulation types, it was observed that the amplitude of the final signal differed from that of the original message signal. However, it was emphasized that this discrepancy can be adjusted by fine-tuning the filter gains. The MATLAB simulation results were shown to be consistent with the analytical calculations, demonstrating the theoretical and practical validity of the modulation and demodulation processes in analog communication systems.

4)Matlab Codes:

```
%% a
% Örnekleme parametreleri
fs = 100000; % Örnekleme frekansı (100 kHz)
T_ortak = 1/50; % Ortak periyot (0.02 s)
t = 0:1/fs:T_ortak; % 1 ortak periyot için zaman aralığı

% Mesaj sinyali
m_t = 8*cos(300*pi*t) + 10*sin(500*pi*t);

% Mesaj sinyalinin Fourier dönüşümü
M_f = fft(m_t);
N = length(M_f);
frequencies = linspace(-fs/2, fs/2, N); % Frekans ekseni
M_f_shifted = fftshift(M_f); % Sıfır frekansı merkeze taşıma
magnitude_spectrum = abs(M_f_shifted) / max(abs(M_f_shifted)); % Normalize

% Genlik spektrumunu teorik bileşenlerle karşılaştırma
theoretical_frequencies = [-250, -150, 150, 250]; % Teorik frekanslar
theoretical_amplitudes = [5, 4, 4, 5]; % Teorik genlikler

% Grafikler
figure;
```

```
% Mesaj sinyalinin zamandaki grafiği
subplot(2,1,1);
plot(t, m_t, 'b', 'LineWidth', 1.5);
title('Mesaj Sinyali m(t)');
xlabel('Zaman (s)');
ylabel('Genlik');
grid on;

% Mesaj sinyalinin spektrumu
subplot(2,1,2);
plot(frequencies, magnitude_spectrum, 'r', 'LineWidth', 1.5); hold on;

% Teorik delta işaretlerini gösterme
stem(theoretical_frequencies, theoretical_amplitudes / max(theoretical_amplitudes), 'b', 'LineWidth',
1.5);
title('Mesaj Sinyalinin Frekans Spektrumu |M(f)|');
xlabel('Frekans (Hz)');
ylabel('Normalize Genlik');
xlim([-500 500]); % Frekans eksenini sınırla
grid on;
legend('FFT Spektrumu', 'Teorik Delta Fonksiyonları');


%% b-i


% Parametreler
Fs = 10000;  % Örnekleme frekansı (10 kHz)
T = 0.02;    % Ortak periyot (20 ms)
N = T * Fs;  % Örnekleme noktası sayısı
t = linspace(0, T, N); % Zaman vektörü
fc = 1500;   % Taşıyıcı frekansı (Hz)

% Mesaj sinyali
m_t = 8*cos(300*pi*t) + 10*sin(500*pi*t);

% Taşıyıcı sinyali
c_t = 10 * cos(2*pi*fc*t);

% DSB-SC-AM Modülasyonu (mesaj * taşıyıcı)
s_t = m_t .* c_t;

% Zaman grafiği - Modüle edilmiş sinyal
figure;
subplot(2,1,1);
plot(t*1000, s_t, 'b', 'LineWidth', 1.5);  % Zamanı milisaniyeye çeviriyoruz
title('DSB-SC-AM Modülasyonu');
xlabel('Zaman (ms)');
ylabel('Genlik');
grid on;

% Fourier Dönüşümü ve Spektrum
S_f = fft(s_t);  % Fourier Dönüşümü
f = linspace(-Fs/2, Fs/2, N);  % Frekans ekseni
S_f_shifted = fftshift(S_f);  % Fourier spektrumu sıfır frekans etrafına kaydır
```

```
% Genlik spektrumu
amplitude_spectrum = abs(S_f_shifted) / N;

% Teorik frekanslar ve genlikler
theoretical_frequencies = [-1750, -1650, -1350, -1250, 1250, 1350, 1650, 1750];  % Teorik frekanslar
theoretical_amplitudes = [25, 20, 20, 25, 25, 20, 20, 25];  % Teorik genlikler

% Genlikleri ölçekleme
max_amplitude = max(amplitude_spectrum);
scaled_amplitude = theoretical_amplitudes / max(theoretical_amplitudes) * max_amplitude;

% Spektrum grafiği - Modüle edilmiş sinyal
subplot(2,1,2);
plot(f, amplitude_spectrum, 'r', 'LineWidth', 1.5); hold on;

% Teorik delta işaretlerini gösterme
stem(theoretical_frequencies, scaled_amplitude, 'b', 'LineWidth', 1.5);
title('DSB-SC-AM Modülasyonunun Genlik Spektrumu');
xlabel('Frekans (Hz)');
ylabel('Normalize Genlik');
xlim([-5000 5000]);  % Frekans eksenini sınırla
grid on;
legend('FFT Spektrumu', 'Teorik Delta Fonksiyonları');

%% b-ii

% Parametreler
Fs = 10000;  % Örnekleme frekansı (10 kHz)
T = 0.02;    % Ortak periyot (20 ms)
N = T * Fs;  % Örnekleme noktası sayısı
t = linspace(0, T, N); % Zaman vektörü
fc = 1500;   % Taşıyıcı frekansı (Hz)

% Mesaj sinyali
m_t = 8*cos(300*pi*t) + 10*sin(500*pi*t);

% Taşıyıcı sinyali
c_t = 10 * cos(2*pi*fc*t);

% DSB-SC-AM Modülasyonu (mesaj * taşıyıcı)
s_t = m_t .* c_t;

% Demodülasyon: s(t) * cos(2*pi*fc*t)
y_t = s_t .* cos(2*pi*fc*t);  % Demodülasyon işlemi

% Zaman grafiği: Demodülasyon sonrası sinyal
figure;
subplot(2,1,1);
plot(t*1000, y_t, 'b', 'LineWidth', 1.5); % Zamanı ms cinsine çevirdik
title('Demodülasyon Sonrası Sinyalin Genlik Spektrumu (LPS input i(t))');
xlabel('Zaman (ms)');
ylabel('Genlik');
grid on;
```

```
% Fourier Dönüşümü
Y_f = fft(y_t);              % Fourier dönüşümü
f = linspace(-Fs/2, Fs/2, N);  % Frekans ekseni
Y_f_shifted = fftshift(Y_f);   % Spektrumun sıfır frekansa göre merkezlenmesi

% Genlik spektrumu
amplitude_spectrum = abs(Y_f_shifted) / N;

% Teorik frekanslar ve genlikler (manuel verilmiş)
theoretical_frequencies = [-3250, -3150, -2850, -2750, -250, -150, 150, 250, 2750, 2850, 3150, 3250];
theoretical_amplitudes = [12.5, 10, 10, 12.5, 25, 20, 20, 25, 12.5, 10, 10, 12.5];

% Normalize edilen genlik spektrumunu ölçekleme
scaled_theoretical_amplitudes = theoretical_amplitudes / max(theoretical_amplitudes) *
max(amplitude_spectrum);

% Spektrum grafiği: Demodülasyon sonrası sinyal
subplot(2,1,2);
plot(f, amplitude_spectrum, 'r', 'LineWidth', 1.5); hold on;

% Teorik delta fonksiyonlarını ekleme
stem(theoretical_frequencies, scaled_theoretical_amplitudes, 'b', 'LineWidth', 1.5);

title('Demodülasyon Sonrası Sinyalin Genlik Spektrumu (LPS input i(t))');
xlabel('Frekans (Hz)');
ylabel('Genlik');
xlim([-4000 4000]); % Frekans eksenini sınırla
grid on;
legend('FFT Spektrumu', 'Teorik Delta Fonksiyonları');


%% b-iii


% Parametreler
Fs = 10000;  % Örnekleme frekansı (Hz)
T = 0.02;    % Ortak periyot (20 ms)
N = T * Fs;  % Örnekleme noktası sayısı
t = linspace(0, T, N); % Zaman vektörü
fc = 1500;   % Taşıyıcı frekansı (Hz)
cutoff_freq = 300; % Düşük geçiren filtre kesim frekansı (Hz)

% Mesaj sinyali
m_t = 8*cos(300*pi*t) + 10*sin(500*pi*t);

% Taşıyıcı sinyali
c_t = 10 * cos(2*pi*fc*t);

% DSB-SC-AM Modülasyonu (mesaj * taşıyıcı)
s_t = m_t .* c_t;

% Demodülasyon: s(t) * cos(2*pi*fc*t)
y_t = s_t .* cos(2*pi*fc*t);  % Demodülasyon işlemi

% Fourier Dönüşümü: Demodülasyon sonrası sinyal
```

```
Y_f = fft(y_t);              % Fourier dönüşümü
f = linspace(-Fs/2, Fs/2, N);  % Frekans ekseni
Y_f_shifted = fftshift(Y_f);   % Spektrumun sıfır frekansa göre merkezlenmesi

% LPF uygulaması: Frekans penceresi
H_f = abs(f) <= cutoff_freq; % Kesim frekansı altında 1, diğerlerinde 0
Y_f_filtered = Y_f_shifted .* H_f; % Frekans domeninde filtreleme

% Ters Fourier Dönüşümü: Zaman domenine dönüş
Y_f_unshifted = ifftshift(Y_f_filtered); % FFT kaydırmasını geri al
i_t = ifft(Y_f_unshifted, 'symmetric');  % Ters Fourier dönüşümü

% Genlik spektrumu (ölçekleme yapılmadan)
amplitude_spectrum_filtered = abs(Y_f_filtered) / N;

% Frekans ekseninde maksimum değerleri ölçekleme
amplitude_spectrum_filtered = amplitude_spectrum_filtered * max([25, 20]) /
max(amplitude_spectrum_filtered);

% Zaman grafiği: Filtrelenmiş sinyal (ñ(t))
figure;
subplot(2,1,1);
plot(t*1000, i_t, 'b', 'LineWidth', 1.5); % Zamanı ms cinsine çevirdik
title('LPF Çıkışı: ñ(t) Sinyali');
xlabel('Zaman (ms)');
ylabel('Genlik');
grid on;

% Spektrum grafiği: Filtrelenmiş sinyal
subplot(2,1,2);
plot(f, amplitude_spectrum_filtered, 'r', 'LineWidth', 1.5);
title('LPF Çıkışı: ñ(t) Sinyalinin Genlik Spektrumu');
xlabel('Frekans (Hz)');
ylabel('Genlik');
xlim([-2000 2000]); % Frekans eksenini sınırla
grid on;

% Manuel genliklerle teorik karşılaştırma
theoretical_frequencies = [-250, -150, 150, 250]; % Teorik frekanslar
theoretical_amplitudes = [25, 20, 20, 25]; % Teorik genlikler
hold on;
stem(theoretical_frequencies, theoretical_amplitudes, 'b', 'LineWidth', 1.5);
legend('FFT Spektrumu', 'Teorik Delta Fonksiyonları');

%% c-i

% Parameters
t = 0:1e-5:0.02; % Time vector (0 to 20 ms with 10 us steps)
Ac = 10;         % Carrier amplitude
fc = 1500;       % Carrier frequency (Hz)
mu = 0.7;        % Modulation index

% Normalized message signal mn(t)
f1 = 150;        % Frequency of first component (Hz)
f2 = 250;        % Frequency of second component (Hz)
```

```
mnt = 0.454*cos(2*pi*f1*t) + 0.567*sin(2*pi*f2*t);

% Modulated signal (DSB-LC-AM)
yt = Ac * (1 + mu * mnt) .* cos(2*pi*fc*t);

% Plot y(t) (Time-domain representation)
figure;
plot(t, yt, 'b', 'LineWidth', 1.5);
grid on;
title('Modulated Signal y(t) in Time Domain');
xlabel('Time (s)');
ylabel('Amplitude');
ylim([-15 15]); % Adjust y-axis limits for clarity

% Compute magnitude spectrum with scaling
N = length(yt);
Y_f = abs(fftshift(fft(yt)))/(N/2);
Y_f = Y_f / 2; % Apply 1/2 scaling factor
f = linspace(-1/(2*(t(2)-t(1))), 1/(2*(t(2)-t(1))), N);

% Focus on relevant frequency range
f_range = abs(f) <= 2000; % Limit to [-2000, 2000] Hz
f = f(f_range);
Y_f = Y_f(f_range);

% Theoretical Dirac delta components (manually defined)
theoretical_freqs = [-1750, -1650, -1500, -1350, -1250, 1250, 1350, 1500, 1650, 1750];
theoretical_amps = [1, 0.8, 5, 0.8, 1, 1, 0.8, 5, 0.8, 1];

% Plot the magnitude spectrum
figure;
plot(f, Y_f, 'b', 'LineWidth', 1.5); % FFT spectrum
hold on;
grid on;

% Add theoretical Dirac delta components manually
stem(theoretical_freqs, theoretical_amps, 'r', 'LineWidth', 1.5, 'Marker', 'none'); % Stem plot for Dirac
for i = 1:length(theoretical_freqs)
    text(theoretical_freqs(i), theoretical_amps(i) + 0.2, ...
        ['A=', num2str(theoretical_amps(i))], 'Color', 'red', 'FontSize', 10, ...
        'HorizontalAlignment', 'center');
end

% Labels and title
title('Magnitude Spectrum with Theoretical Dirac Components');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
legend('FFT Spectrum', 'Theoretical Dirac Components');
ylim([0 6]); % Y-axis limit for clarity
hold off;
```

```
%% c-ii
% Time and signal parameters
t = 0:1e-5:0.02;   % Time vector (0 to 20 ms with 10 us steps)
Ac = 10;           % Carrier amplitude
fc = 1500;         % Carrier frequency (Hz)
mu = 0.7;          % Modulation index

% Normalized message signal mn(t)
f1 = 150;          % Frequency of first component (Hz)
f2 = 250;          % Frequency of second component (Hz)
mnt = 0.454*cos(2*pi*f1*t) + 0.567*sin(2*pi*f2*t);

%% Modulated Signal (DSB-LC-AM)
yt = Ac * (1 + mu * mnt) .* cos(2*pi*fc*t);

% Plot y(t) (Time-domain representation)
figure;
plot(t, yt, 'b', 'LineWidth', 1.5);
grid on;
title('Modulated Signal y(t) in Time Domain');
xlabel('Time (s)');
ylabel('Amplitude');
ylim([-15 15]); % Adjust y-axis limits for clarity

%% Step 1: Generate z(t) by multiplying y(t) with cos(2*pi*fc*t)
zt = (yt .* cos(2*pi*fc*t)) / 2; % Apply 1/2 factor here

% Compute magnitude spectrum of z(t)
N = length(zt);
Z_f = abs(fftshift(fft(zt)))/(N/2);
fs = 1 / (t(2) - t(1)); % Sampling frequency
f = linspace(-fs/2, fs/2, N);

% Focus on relevant frequency range [-3500, 3500]
f_range = abs(f) <= 3500; % Frequency range
f = f(f_range);
Z_f = Z_f(f_range);

% Plot magnitude spectrum of z(t)
figure;
plot(f, Z_f, 'b', 'LineWidth', 1.5);
grid on;
title('Magnitude Spectrum of z(t)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');

%% Step 2: Apply LPF to z(t) (Bandwidth = 250 Hz)
lpf_cutoff = 250; % LPF cutoff frequency (Hz)
lpf_mask = abs(f) <= lpf_cutoff; % Mask for LPF
Z_f_lpf = Z_f .* lpf_mask; % Apply LPF in frequency domain

% Transform back to time domain after LPF
rt = ifft(ifftshift(Z_f_lpf), 'symmetric');

% Compute magnitude spectrum of r(t)
```

```
R_f = abs(fftshift(fft(rt)))/(N/2);

% Plot magnitude spectrum after LPF
figure;
plot(f, R_f, 'r', 'LineWidth', 1.5);
grid on;
title('Magnitude Spectrum after LPF (r(t))');
xlabel('Frequency (Hz)');
ylabel('Magnitude');

%% Step 3: Apply HPF to r(t) with 10 Hz cutoff (Remove DC component)
hpf_cutoff = 10; % HPF cutoff frequency (Hz)
hpf_mask = abs(f) > hpf_cutoff; % Mask for HPF
R_f_hpf = R_f .* hpf_mask; % Apply HPF in frequency domain

% Transform back to time domain after HPF
final_signal_hpf = ifft(ifftshift(R_f_hpf), 'symmetric');

% Compute magnitude spectrum after HPF
Final_f_hpf = abs(fftshift(fft(final_signal_hpf)))/(N/2);

% Plot magnitude spectrum after HPF
figure;
plot(f, Final_f_hpf, 'g', 'LineWidth', 1.5);
grid on;
title('Magnitude Spectrum after HPF (Final Signal)');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([-4000 4000]); % Focus on relevant frequency range

% 1.6cos(2*pi*150*t) + 2sin(2*pi*250*t)
A1 = 1.6;           % Amplitude of the first term
A2 = 2;             % Amplitude of the second term

% Compute final signal
final_signal = A1 * cos(2*pi*f1*t) + A2 * sin(2*pi*f2*t);

% Plot final_signal
figure;
plot(t, final_signal, 'm', 'LineWidth', 1.5);
grid on;
title('Final Signal: 1.6cos(2\pi 150 t) + 2sin(2\pi 250 t)');
xlabel('Time (s)');
ylabel('Amplitude');
```

5)References

1. B. P. Lathi, *Modern Digital and Analog Communication Systems*, 4th ed. New York, NY, USA: Oxford University Press, 2009.
2. Course Notes from ELEC361: Analog Communication Systems, Fall 2024.
3. E. Özdemir, "DSB-SC Modulation and Demodulation," YouTube, Available: https://www.youtube.com/watch?v=3wBlB-TFwkQ.
4. Y. Çelik, "AM Modulation Techniques Explained," YouTube, Available: https://www.youtube.com/watch?v=UKYs7ckctc4.
5. T. Aksoy, "MATLAB: AM Modulation in Practice," YouTube, Available: https://www.youtube.com/watch?v=VnkCgH8KimY.
6. *MATLAB R2024b* . Natick, MA, USA: MathWorks, 2024.