TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

ADVANCED COLLEGE
OF ENGINEERING AND MANAGEMENT

Data Communication

Lab No. 1 to Lab No. 5

COMPILED LAB REPORT

Submitted By:

Aviskar Poudel
Roll No: ACE079BCT013

Submitted To:

Department of Electronics and
Computer Engineering

July 27, 2025

LAB NO. 1

1.1 THEORY

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. MATLAB operates primarily on matrices, making it highly efficient for handling complex mathematical computations. It also provides excellent support for plotting and simulating data, which is essential in engineering analysis.

Data communication involves the transmission of data between devices or systems through various transmission media. It often requires analysis and processing of signals, encoding/decoding data, studying error detection and correction schemes, and simulating communication channels. MATLAB helps us to visualize the different signals and modify those signals.

1.2 SOURCE CODE

Listing 1.1: Basic Arithmetic

```
1
2  a = 15;
3  b = 4;
4  addition = a + b;
5  subtraction = a - b;
6  multiplication = a * b;
7  division = a / b;
8  remainder = mod(a, b);
9
10 disp(['Addition: ', num2str(addition)]);
11 disp(['Subtraction: ', num2str(subtraction)]);
12 disp(['Multiplication: ', num2str(multiplication)]);
13 disp(['Division: ', num2str(division)]);
14 disp(['Remainder: ', num2str(remainder)]);
```

Listing 1.2: Basic Matrix Arithmetic

```
1
2
```

```
A = [1 2; 3 4];
B = [5 6; 7 8];

matrix_add = A + B;
matrix_sub = A - B;
matrix_mul = A * B;


disp('Matrix Addition:');
disp(matrix_add);

disp('Matrix Subtraction:');
disp(matrix_sub);

disp('Matrix Multiplication:');
disp(matrix_mul);
```

Listing 1.3: Basic Trigonometric Operations

```

theta_deg = 30;


theta_rad = deg2rad(theta_deg);


sine_val = sin(theta_rad);
cosine_val = cos(theta_rad);


disp(['sin(', num2str(theta_deg), ') = ', num2str(sine_val)
   ]);
disp(['cos(', num2str(theta_deg), ') = ', num2str(cosine_val
   )]);
```

Listing 1.4: Basic Complex Number Operations

```
z = 3 + 4i;

real_part = real(z);
imag_part = imag(z);
conj_val = conj(z);
abs_val = abs(z);
```

```
7
8  disp(['Real part: ', num2str(real_part)]);
9  disp(['Imaginary part: ', num2str(imag_part)]);
10 disp(['Absolute value: ', num2str(abs_val)]);
11 disp(['Conjugate: ', num2str(conj_val)]);
```

1.3 OUTPUT

```
>> addsub
Addition: 19
Subtraction: 11
Multiplication: 60
Division: 3.75
Remainder: 3
```

Figure 1.1: Basic Arithmetic

```
Matrix Addition:
     6     8
    10    12

Matrix Subtraction:
    -4    -4
    -4    -4

Matrix Multiplication:
    19    22
    43    50
```

Figure 1.2: Basic Matrix Arithmetic

```
>> trigonometry
sin(30°) = 0.5
cos(30°) = 0.86603
```

Figure 1.3: Basic Trigonometric Operation

```
>> complex
Real part: 3
Imaginary part: 4
Absolute value: 5
Conjugate: 3-4i
```

Figure 1.4: Basic Complex Number Operation

LAB NO. 2

2.1 THEORY

A line plot in MATLAB is a basic two-dimensional graph that connects data points with straight lines. It is commonly used to visualize mathematical functions, data trends, or relationships between variables.

Trigonometric wave plots, such as sine and cosine waves, demonstrate periodic behavior and are essential in understanding signals and waveforms in data communication. MATLAB's built-in `sin` and `cos` functions allow easy plotting of these signals over a range of values, helping visualize amplitude, frequency, and phase characteristics.

MATLAB uses commands like `plot` for continuous line graphs and `stem` for discrete plots.

2.2 SOURCE CODE

Listing 2.1: Plotting a line in MATLAB

```
a = [4,7];
b = [10,20];
x = [1,2];
y = [1,5];

c = [3,5];
d = [7,10];


subplot(3,1,1);
plot (a,b,'--black', 'LineWidth', 1)
title('Linedraw1/Aviskar␣Poudel/013');
legend('Dotted␣Line')

subplot(3,1,2);
plot (x,y,'-blue', 'LineWidth',2)
title('Linedraw2/Aviskar␣Poudel/013');
legend('Line')
```

```
19
20
21 subplot(3,1,3);
22 stem(c,d,'-red')
23 title('Linedraw3/Aviskar␣Poudel/013');
24 legend('Discrete␣Line')
25
26 grid on;
```

Listing 2.2: Plotting different waves (both continuous and discrete) in MATLAB

```
1  % parameters for wave: frequency, time, amplitude
2  % we know that y = asin2 pie f t;
3  % for t, it determines gap, and is defined as (starting:gap:
       ending)
4
5  a = 5;
6  f = 3;
7  t= 0:0.01:1;
8  y = a*sin(2*pi*f*t);
9  yy = a*cos(2*pi*f*t);
10
11 subplot(4,2,1)
12 plot(t,y,'-red')
13 hold on
14 plot(t,zeros(size(t)),'-black') % this is to make an x asix
       line in graph
15 title('Lab1/Aviskar␣Poudel/SineWaveContinuous')
16 legend('Sine␣Wave')
17
18 subplot(4,2,2)
19 plot(t,yy,'-blue')
20 hold on
21 plot(t,zeros(size(t)),'-black')
22 title('Lab1/Aviskar␣Poudel/CosineWaveContinuous')
23 legend('Cosine␣Wave')
24
25 subplot(4,2,3)
26 stem(t,y,'-red')
27 hold on
28 plot(t,zeros(size(t)),'-black')
29 title('Lab1/Aviskar␣Poudel/SineWaveDiscrete')
```

```
legend('Sine Wave')

subplot(4,2,4)
stem(t,yy,'-blue')
hold on
plot(t,zeros(size(t)),'-black')
title('Lab1/Aviskar Poudel/CosineWaveDiscrete')
legend('Cosine Wave')


subplot(4,2,5)
plot(t,yy,'-blue')
hold on
plot(t,y,'-red')
hold on
plot(t,zeros(size(t)),'-black')
hold on
title('Lab1/Aviskar Poudel/sineandcosinecombined')
legend('Cosine Wave', 'Sine Wave')


subplot(4,2,6)
stem(t,yy,'-blue')
hold on
stem(t,y,'-red')
hold on
plot(t,zeros(size(t)),'-black')
hold on
title('Lab1/Aviskar Poudel/sineandcosinecombined')
legend('Cosine Wave', 'Sine Wave')

grid on
```

Listing 2.3: All waves in same figure

```
    %plot all 1 to 5 qn in same graph
a = 5;
f = 5;
t= 0:0.01:1;
y = a*sin(2*pi*f*t);
yy = a*cos(2*pi*f*t);

```

```
8
9  plot(t,y,'-red')
10 hold on
11 plot(t,zeros(size(t)),'-black')
12 hold on
13 plot(t,yy,'-blue')
14 hold on
15 stem(t,y,'-blue')
16 hold on
17 stem(t,yy,'-blue')
18 hold on
19
20 title('Lab1/Aviskar␣Poudel/All␣Signals␣Combined')
```

2.3 OUTPUT

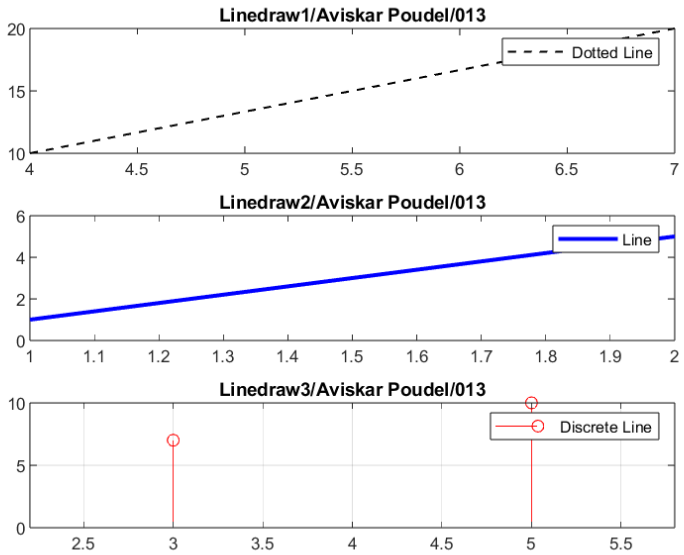

Figure 2.1: Lines

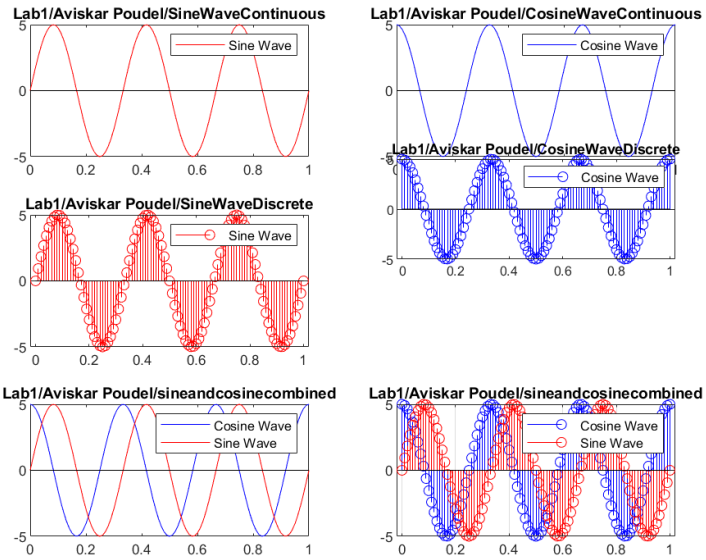

Figure 2.2: Different waves (Discrete and Continuous)

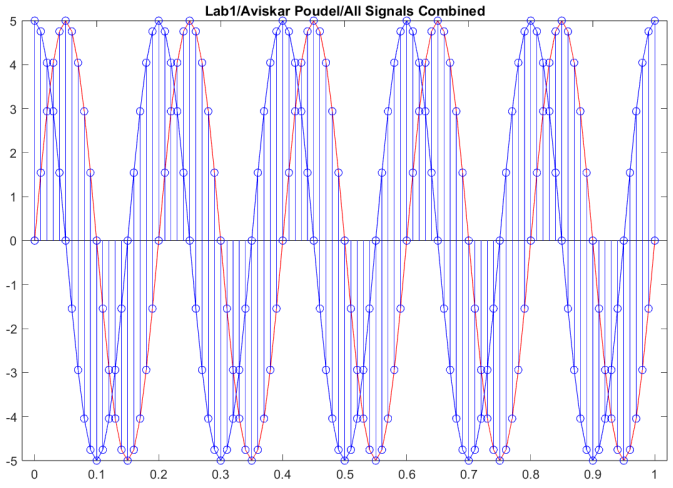

Figure 2.3: All in one figure

LAB NO. 3

3.1 THEORY

Basic signal functions are essential in data communication and signal processing. The unit step function models signals turning on at a specific time. The square wave is a periodic signal switching between two levels, commonly used in digital systems. The signum function indicates the sign of a value. The ramp function represents a linearly increasing signal starting at zero. The impulse function models an ideal instantaneous spike used to analyze system responses. Noise refers to random unwanted signals that affect communication quality.

3.2 SOURCE CODE

Listing 3.1: Unit Step Function

```
1
2  t = -2:0.1:2;
3  unitstep = t>=0;
4
5  subplot(2,1,1)
6  plot(t,unitstep)
7  legend("Unitstep")
8  title('Lab3/Aviskar␣Poudel/UnitStep␣-␣Continuous')
9
10
11 subplot(2,1,2)
12 stem(t,unitstep, '-red')
13 legend("Unitstep")
14 title('Lab3/Aviskar␣Poudel/UnitStep␣-␣Discrete')
```

Listing 3.2: Square Function

```
1      a = 5;
2  f = 3;
3  t = 0:0.01:1;
4  s = a*sign(sin(2*pi*f*t));
5  c = a*sign(cos(2*pi*f*t));
6  subplot(2,1,1)
7  plot(t, s, '-red')
```

```
legend('Square Sine')
title('Lab3/Aviskar Poudel/Square SineWave')

subplot(2,1,2)
plot(t,c,'-red')
legend('Square Cosine')
title('Lab3/Aviskar Poudel/Square Cosine Wave')
```

Listing 3.3: Signum Function

```
    x = zeros(size(t));

t = -2:0.1:2;

for i = 1:length(t)
    if t(i) > 0
        x(i) = 1;
    elseif t(i) == 0
        x(i) = 0;
    else
        x(i) = -1;
    end
end
subplot(2,1,1)
plot (t,x, '-red')
legend("Signum function")
title('Lab3/Aviskar Poudel/Ramp - Continuous')

subplot(2,1,2)
stem(t,x, '-red')
legend("Ramp function")
title('Lab3/Aviskar Poudel/Ramp - Discrete')
```

Listing 3.4: Ramp Function

```

x = zeros(size(t));

t = -2:0.1:2;
for i = 1:length(t)
    if t(i)>=0
        x(i) = t(i);
    else
```

```
        x(i) = 0;
    end
end
subplot(2,1,1)
plot (t,x, '-red')
legend("Ramp function")
title('Lab3/Aviskar Poudel/Ramp - Continuous')

subplot(2,1,2)
stem(t,x, '-red')
legend("Ramp function")
title('Lab3/Aviskar Poudel/Ramp - Discrete')
```

Listing 3.5: Noise Wave

```
    a = 5;
f = 3;
t = 0:0.01:1;
s = a*sin(2*pi*f*t);
c = a*cos(2*pi*f*t);
x = rand(1, length(t));
z1 = x + s;
z2 = x + c;
subplot(3,1,1)
plot(t,x)
legend('Noise')
title('Lab3/Aviskar Poudel/Distorted Wave')

subplot(3,1,2)
plot(t, z1)
legend('Distorted  Sine')
title('Lab3/Aviskar Poudel/Distorted Sine Wave')

subplot(3,1,3)
plot(t,z2)
legend('Distorted Cosine')
title('Lab3/Aviskar Poudel/Distorted Cosine Wave')
```

Listing 3.6: Impulse Function

```

t = -2:0.1:2;
impulse = t == 0;
```

```
4
5  subplot(2,1,1)
6  plot(t,impulse)
7  legend("Impulse")
8  title('Lab3/Aviskar␣Poudel/Impulse␣-␣Continuous')
9
10 subplot(2,1,2)
11 stem(t,impulse, '-red')
12 legend("Impulse")
13 title('Lab3/Aviskar␣Poudel/Impulse␣-␣Discrete')
```

3.3 OUTPUT

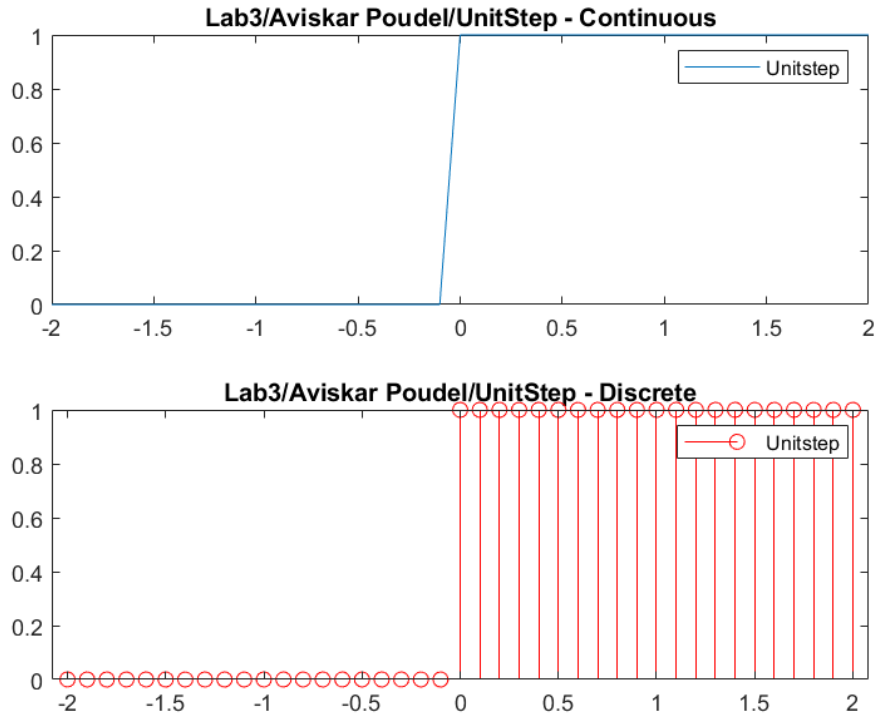

Figure 3.1: Unit Step Function

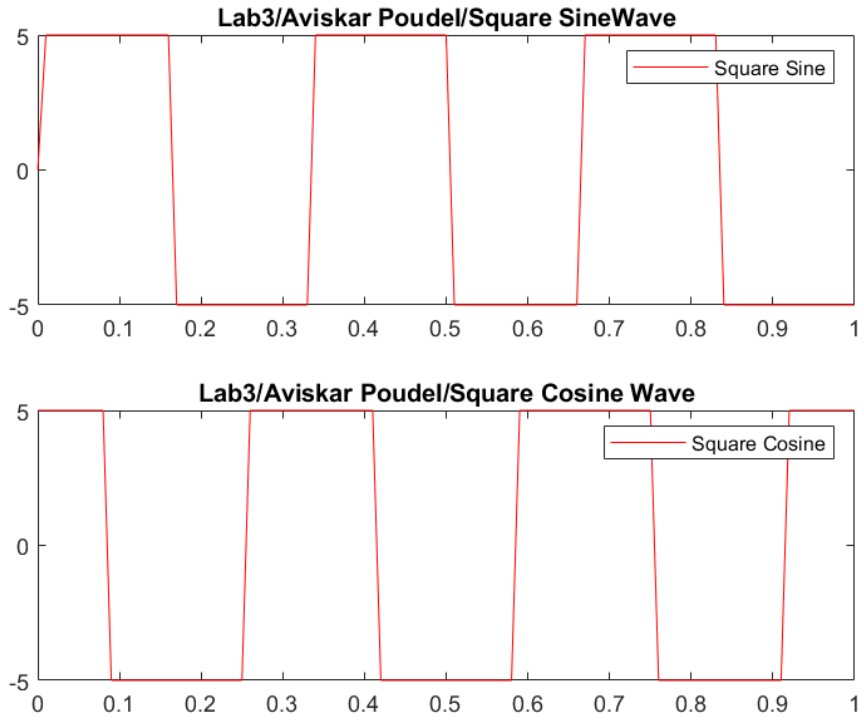

Figure 3.2: Square Wave

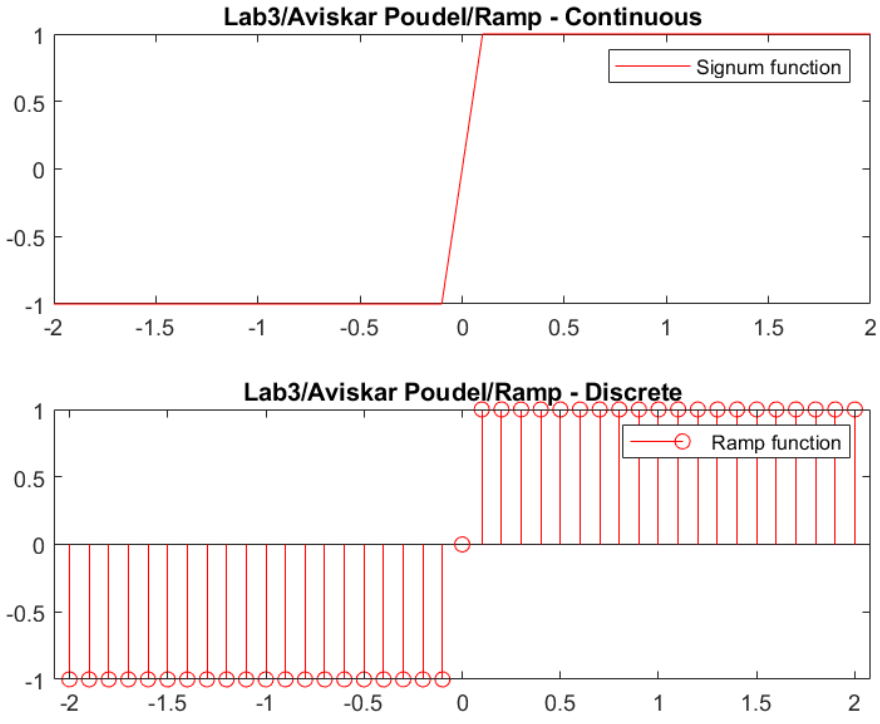

Figure 3.3: Signum Function

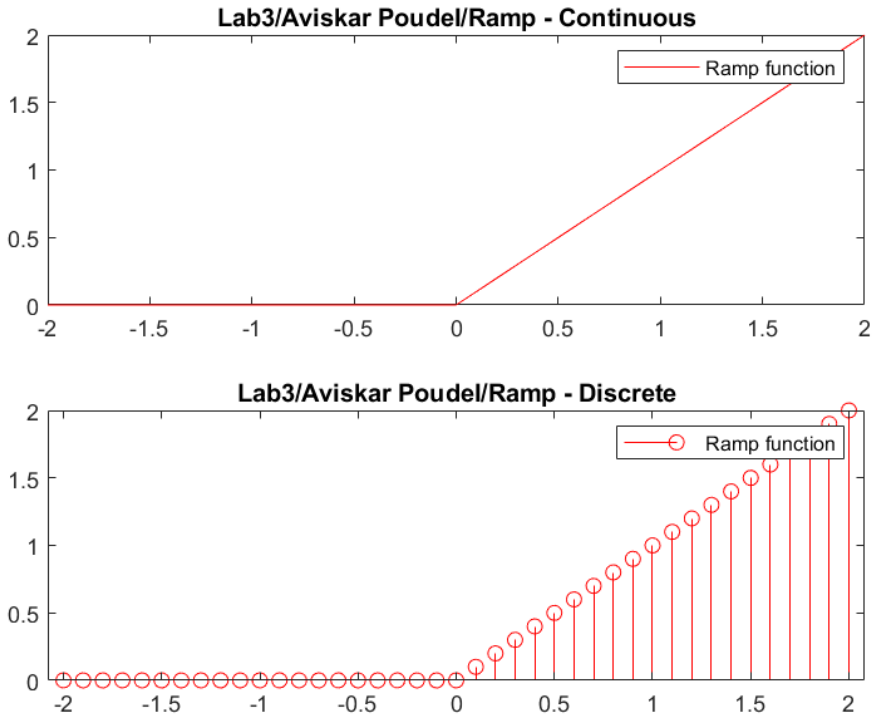

Figure 3.4: Ramp Function

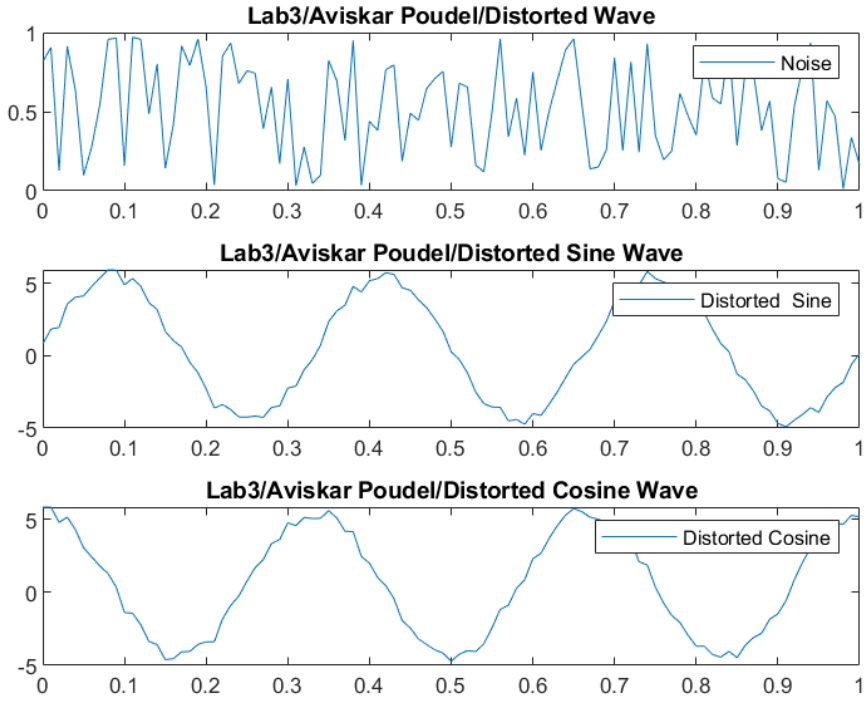

Figure 3.5: Noise Wave

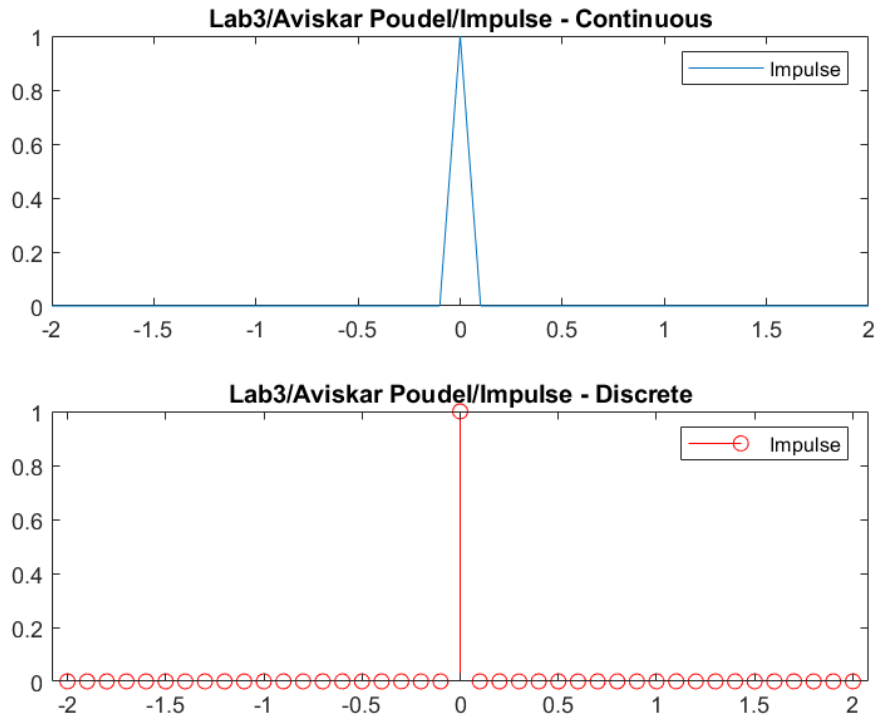

Figure 3.6: Impulse Function

LAB NO. 4

4.1 THEORY

Modulation is the process of varying one or more properties of a high-frequency carrier signal (such as amplitude, frequency, or phase) in accordance with a low-frequency message signal. It enables efficient transmission of data over long distances and through different media. The main types of modulation are:

- **Amplitude Modulation (AM)**: Varies the amplitude of the carrier signal in proportion to the message signal.
- **Frequency Modulation (FM)**: Varies the frequency of the carrier according to the message signal.
- **Phase Modulation (PM)**: Varies the phase of the carrier in relation to the message signal.

Modulation is essential in data communication as it helps to adapt the signal to the characteristics of the transmission medium, allows multiplexing of multiple signals, and improves resistance to noise and interference.

A function $f(t)$ is called **even** if it satisfies

$$f(-t) = f(t) \quad \text{for all } t$$

Its graph is symmetric about the y-axis. Examples include $\cos(t)$ and t^2.

A function $f(t)$ is called **odd** if it satisfies

$$f(-t) = -f(t) \quad \text{for all } t$$

Its graph is symmetric about the origin. Examples include $\sin(t)$ and t^3.

Any function can be expressed as the sum of its even and odd parts, which is useful in signal analysis, particularly in Fourier series and transforms.

4.2 SOURCE CODE

Listing 4.1: Sinc Function

```
        t = -10:0.1:10;    % time vector

% sinc function: sin(x)/x
sinc_func = sin(t)./t;
sinc_func(t==0) = 1;     % define sinc(0)=1 manually

% Plot continuous
subplot(2,1,1)
plot(t, sinc_func)
legend("Sinc")
title('Lab4/Aviskar␣Poudel/Sinc␣-␣Continuous')
xlabel('t')
ylabel('sinc(t)')

% Plot discrete
subplot(2,1,2)
stem(t, sinc_func, 'r')
legend("Sinc")
title('Lab4/Aviskar␣Poudel/Sinc␣-␣Discrete')
xlabel('t')
ylabel('sinc(t)')
```

Listing 4.2: Exponentially Decaying Function

```
t = 0:0.1:10;     % time vector

% exponential decay function, e.g. e^{-t}
exp_decay = 5 * exp(-t);

% Plot continuous
subplot(2,1,1)
plot(t, exp_decay)
legend("Exp Decay")
title('Lab4/Aviskar␣Poudel/Exponential␣Decay␣-␣Continuous')
xlabel('t')
ylabel('e^{-t}')

% Plot discrete
```

```
subplot(2,1,2)
stem(t, exp_decay, 'r')
legend("Exp Decay")
title('Lab4/Aviskar Poudel/Exponential Decay - Discrete')
xlabel('t')
ylabel('e^{-t}')
```

Listing 4.3: Even and Odd function

```
t = -2:0.01:2;
%x = t;

for i = 1:length(t)
    if t(i)>=0
        x(i) = 2;
    else
        x(i) = 0;
    end
end

% Compute even and odd parts
xe = (x + fliplr(x)) / 2;


xo  = (x - fliplr(x)) / 2;

% Plot original
figure
subplot(3,2,1)
plot(t, x)
title('Lab4/Aviskar Poudel/Original Function')
xlabel('t')
ylabel('f(t)')

subplot(3,2,2)
stem(t, x, 'r')
title('Lab4/Aviskar Poudel/Original Function Discrete')
xlabel('t')
ylabel('f(t)')

% Plot even part
subplot(3,2,3)
```

```
plot(t, xe)
title('Lab4/Aviskar␣Poudel/Even␣Function')
xlabel('t')
ylabel('f_e(t)')

subplot(3,2,4)
stem(t, xe, 'r')
title('Lab4/Aviskar␣Poudel/Even␣Function␣Discrete')
xlabel('t')
ylabel('f_e(t)')

% Plot odd part
subplot(3,2,5)
plot(t, xo)
title('Lab4/Aviskar␣Poudel/Odd␣Function')
xlabel('t')
ylabel('f_o(t)')

subplot(3,2,6)
stem(t, xo, 'r')
title('Lab4/Aviskar␣Poudel/Odd␣Function␣Discrete')
xlabel('t')
ylabel('f_o(t)')
```

Listing 4.4: Phase Modulation

```
am = 5;
fm = 3;
t = -1:0.001:1;

msg = am*cos(2*pi*fm*t);

subplot(3,1,1)
plot(t, msg, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Cos␣Msg␣Signal')
legend('Cos␣Msg␣Signal')

ac = 10;
fc = 50;

```

```
carrier = ac*cos(2*pi*fc*t);

subplot(3,1,2)
plot(t, carrier, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Carrier␣Signal')
legend('Cos␣Carrier␣Signal')

kp = 1;
beta = kp * am;

PM = ac*cos(2*pi*fc*t + beta*cos(2*pi*fm*t));

subplot(3,1,3)
plot(t, PM, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Phase␣Mod␣Signal')
legend('Phase␣Mod␣Signal')
```

Listing 4.5: Frequency Modulation

```

am = 5;
fm = 3;
t = -1:0.001:1;

msg = am*cos(2*pi*fm*t);

subplot(3,1,1)
plot(t, msg, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Cos␣Msg␣Signal')
legend('Cos␣Msg␣Signal')

ac = 10;
fc = 50;

carrier = ac*cos(2*pi*fc*t);

```

```
subplot(3,1,2)
plot(t, carrier, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Carrier␣Signal')
legend('Cos␣Carrier␣Signal')

kf = 5;
beta = (kf*am)/fm;

FM = ac*cos(2*pi*fc*t + beta*sin(2*pi*fm*t));

subplot(3,1,3)
plot(t, FM, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/Freq␣Mod␣Signal')
legend('Freq␣Mod␣Signal')
```

Listing 4.6: Amplitude Modulation

```
t = -1:0.001:1;

a = 5;
f = 5;
y = a*sin(2*pi*f*t);

subplot(3,1,1)
plot(t, y, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab4/Aviskar␣Poudel/SineMessageSignal')
legend('Sine␣Wave')

hf = 50;
a2 = 5;
carrier = a2*sin(2*pi*hf*t);

subplot(3,1,2)
plot(t, carrier, '-red')
hold on
plot(t, zeros(size(t)), '-black')
```

```
22 title('Lab4/Aviskar␣Poudel/CarrierSignal')
23 legend('Carrier␣Wave')
24
25 am_signal = (1 + (y/a)) .* carrier;
26
27 subplot(3,1,3)
28 plot(t, am_signal, '-red')
29 hold on
30 plot(t, zeros(size(t)), '-black')
31 title('Lab4/Aviskar␣Poudel/AM␣Signal')
32 legend('AM␣Signal')
```

4.3 OUTPUT

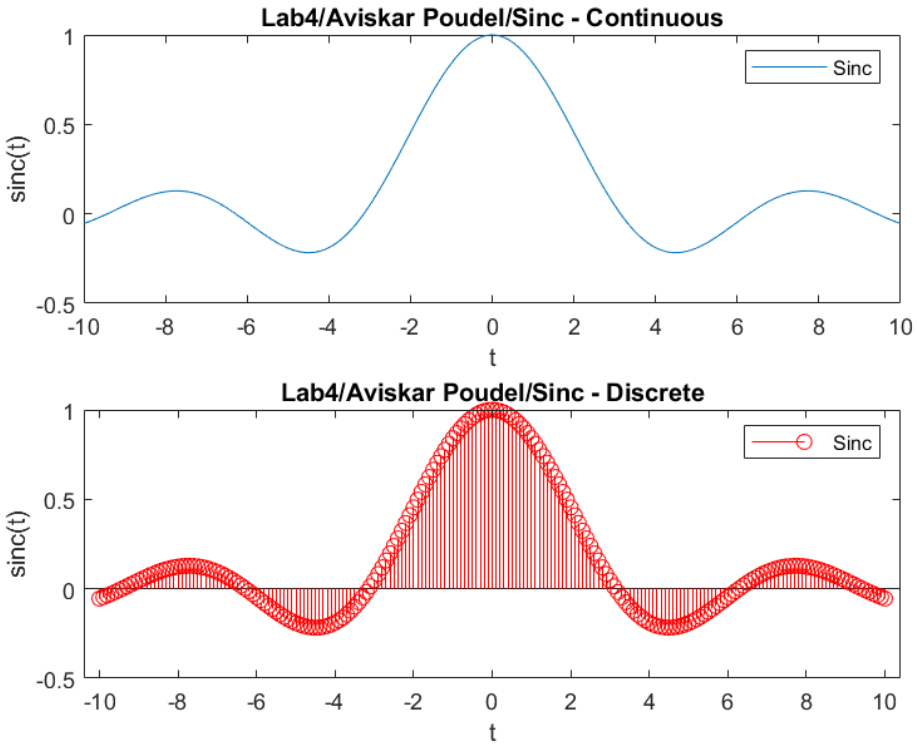

Figure 4.1: Sinc Function

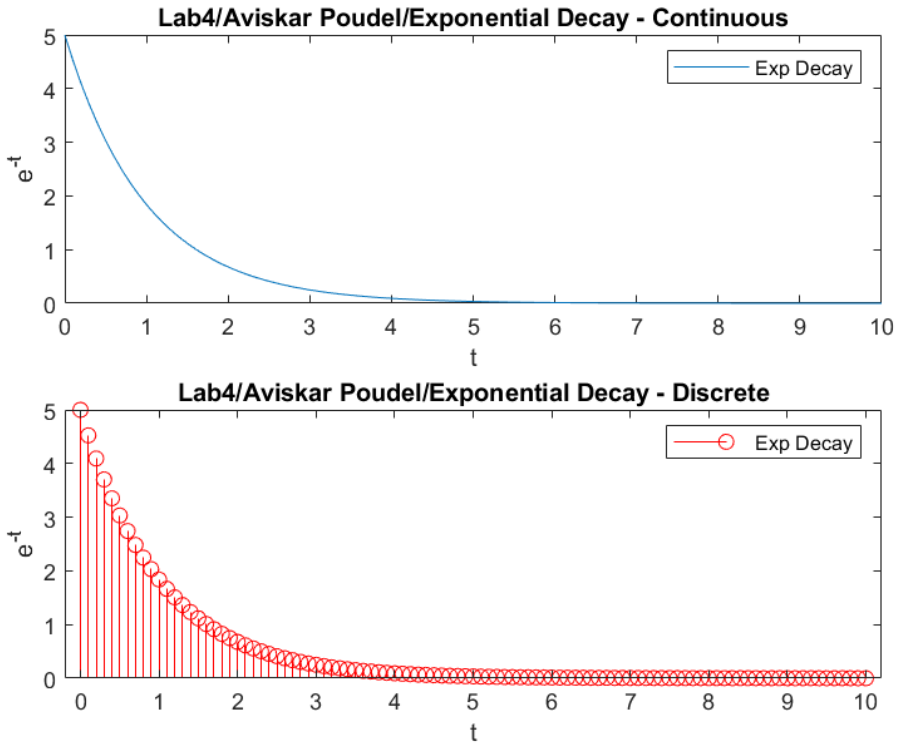

Figure 4.2: Exponentially Decaying Function

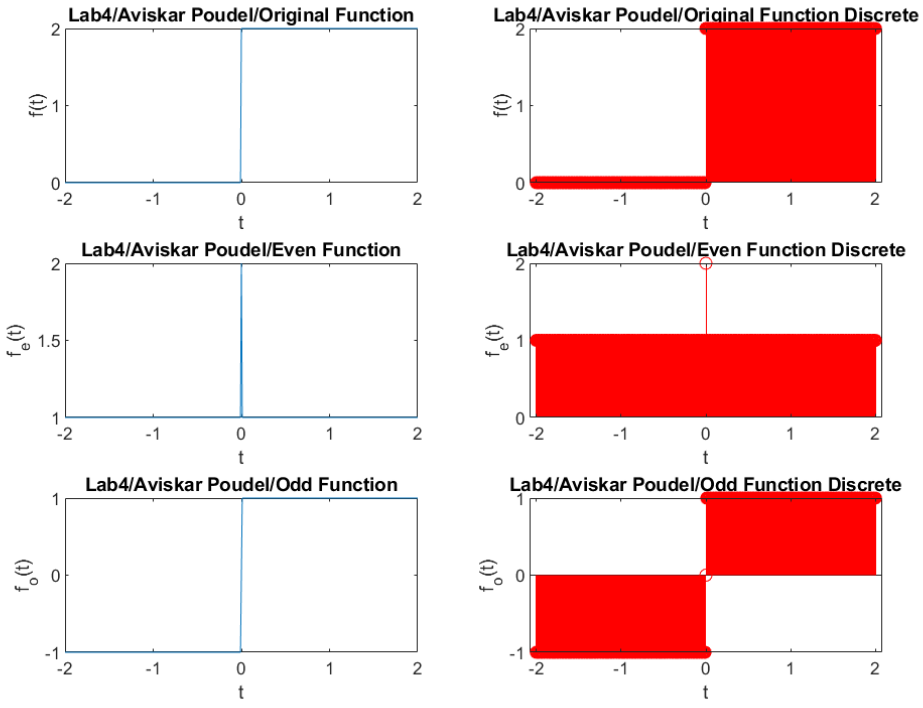

Figure 4.3: Even Odd Function

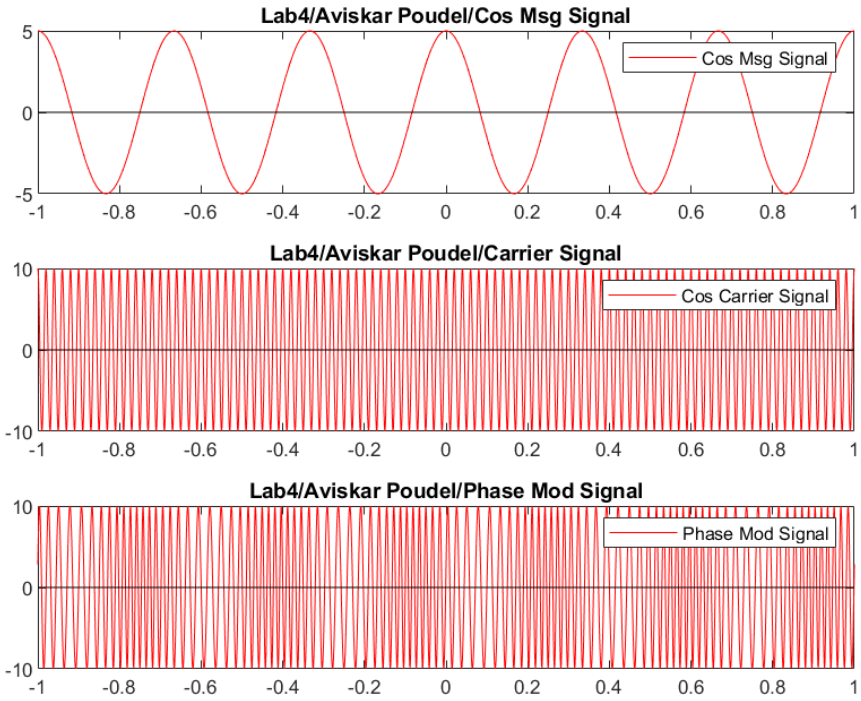

Figure 4.4: Phase Modulation

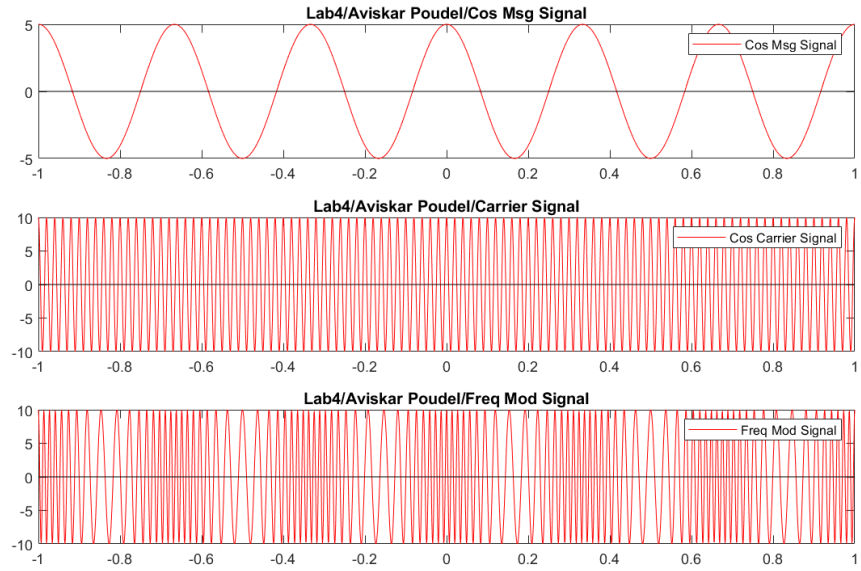

Figure 4.5: Frequency Modulation

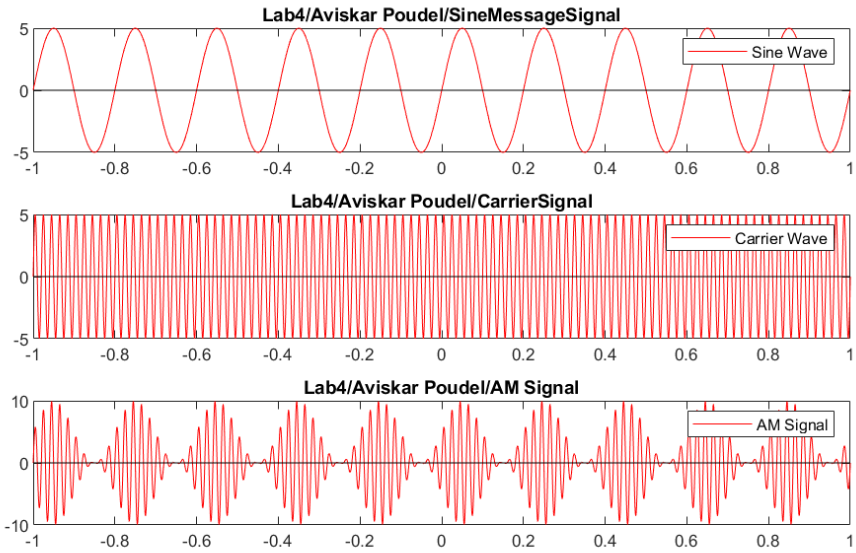

Figure 4.6: Amplitude Modulation

LAB NO. 5

5.1 THEORY

Digital modulation is the process of altering the characteristics of a high-frequency carrier signal (such as amplitude, frequency, or phase) based on a digital message signal, typically composed of discrete levels such as 0 and 1 (or -1 and +1). It is widely used in digital communication systems to transmit binary data over analog channels.

The key types of digital modulation demonstrated in this lab are:

- **Amplitude Shift Keying (ASK)**: A form of amplitude modulation where the carrier is transmitted with one amplitude for logic '1' and a different (often zero) amplitude for logic '0'. This allows binary data to control the presence or strength of the carrier.
- **Frequency Shift Keying (FSK)**: The frequency of the carrier is shifted between two predefined values depending on whether the digital message signal is high or low. For example, a high-frequency carrier may represent logic '1', and a low-frequency carrier represents logic '0'.
- **Phase Modulation (PM)**: The phase of the carrier is varied in proportion to the digital message signal. For binary messages, this may result in a phase shift of $+\theta$ for logic '1' and $-\theta$ for logic '0'. This is the basis of Binary Phase Shift Keying (BPSK).

Digital modulation offers advantages such as higher noise immunity, efficient bandwidth usage, and ease of signal processing. These techniques are foundational in modern communication systems including wireless networks, modems, and satellite communication.

In signal analysis, digital message signals like square waves are typically odd functions due to their symmetry about the origin, much like the sine function:

$$f(-t) = -f(t)$$

Understanding the symmetry and properties of these signals helps in their representation and analysis using tools like the Fourier Transform.

Overall, this lab emphasizes how fundamental modulation principles can be extended to digital signaling, enabling the effective transmission and encoding of binary information onto carrier signals.

5.2 SOURCE CODE

Listing 5.1: Frequency Modulation

```
amp = 5;
fp = 4;
fc1 = 50;
fc2 = 100;
t = 0:0.001:1;


c1 = (amp/2) * sin(2*pi*fc1*t);
c2 = (amp/2) * sin(2*pi*fc2*t);

m = amp * sign(sin(2*pi*fp*t));

fsk_signal = zeros(size(t));
for i = 1:length(t)
    if m(i) > 0
        fsk_signal(i) = (amp/2) * sin(2*pi*fc2*t(i));
    else
        fsk_signal(i) = (amp/2) * sin(2*pi*fc1*t(i));
    end
end


subplot(4,1,1);
plot(t, c1, 'b');
grid on;
xlabel('Time');
ylabel('Amplitude');
title('Lab5/Aviskar␣Poudel/Carrier␣Signal␣1␣(Low␣Freq)');

subplot(4,1,2);
plot(t, c2, 'r');
grid on;
```

```
xlabel('Time');
ylabel('Amplitude');
title('Lab5/Aviskar␣Poudel/Carrier␣Signal␣2␣(High␣Freq)');

subplot(4,1,3);
plot(t, m, 'k');
grid on;
xlabel('Time');
ylabel('Amplitude');
title('Lab5/Aviskar␣Poudel/Message␣Signal␣(Digital␣Square␣
   Wave)');

subplot(4,1,4);
plot(t, fsk_signal, 'm');
grid on;
xlabel('Time');
ylabel('Amplitude');
title('Lab5/Aviskar␣Poudel/FSK␣Modulated␣Signal');
```

Listing 5.2: Amplitude Modulation

```
t = -1:0.001:1;

a = 5;
f = 5;
y_dig = a*sign(sin(2*pi*f*t));

subplot(3,1,1)
plot(t, y_dig, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab5/Aviskar␣Poudel/Sine-DigitalMessageSignal')
legend('Sine␣Wave')

hf = 50;
a2 = 5;
carrier = a2*sin(2*pi*hf*t);

subplot(3,1,2)
plot(t, carrier, '-red')
hold on
plot(t, zeros(size(t)), '-black')
```

```
title('Lab5/Aviskar Poudel/CarrierSignal')
legend('Carrier Wave')

am_signal = (1 + (y_dig/a)) .* carrier;

subplot(3,1,3)
plot(t, am_signal, '-red')
hold on
plot(t, zeros(size(t)), '-black')
title('Lab5/Aviskar Poudel/AM Digital Signal')
legend('AM Signal')
```

Listing 5.3: Phase Modulation

```
clc;
clear all;
close all;

Amp = 5;
fm = 2;   % Message frequency (binary)
fc = 10;  % Carrier frequency

t = 0:0.001:1;

% Carrier signal
x = Amp * sin(2 * pi * fc * t);
subplot(3, 1, 1);
plot(t, x);
grid on;
xlabel('time');
ylabel('amplitude');
title('Aviskar Poudel/Carrier Signal');
legend('carrier wave');

% Message signal (bipolar square wave using sign function)
y = sign(sin(2 * pi * fm * t));
subplot(3, 1, 2);
plot(t, y);
grid on;
xlabel('time');
ylabel('amplitude');
title('Aviskar Poudel/Message Signal');
```

```
legend('message signal');

% PSK Modulation (invert carrier phase based on message bit)
a = x .* y;
subplot(3, 1, 3);
plot(t, a);
grid on;
xlabel('time');
ylabel('amplitude');
title('Aviskar Poudel/PSK Signal');
legend('PSK signal');
```

5.3 OUTPUT

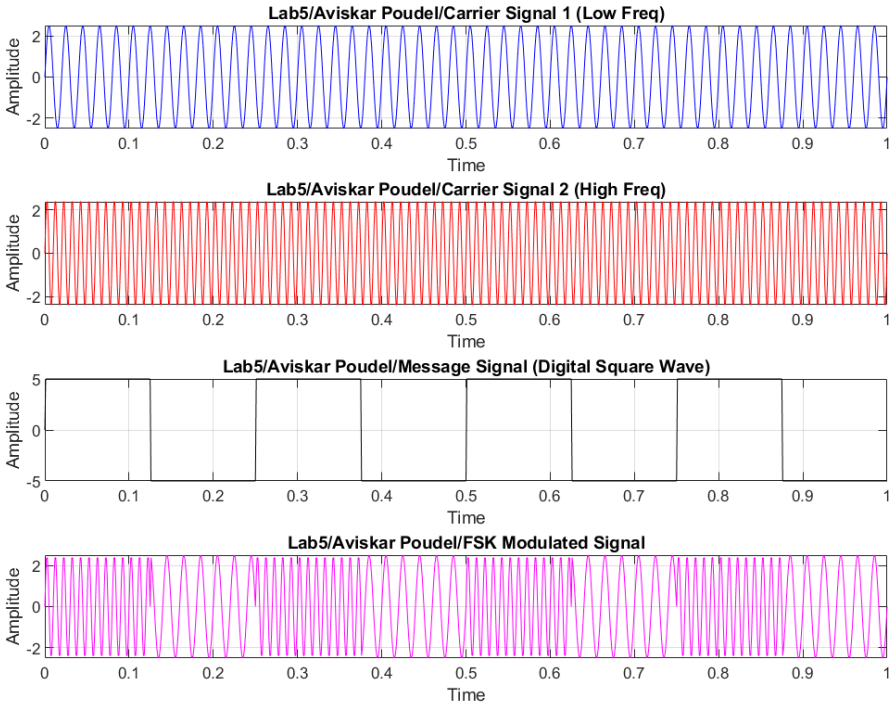

Figure 5.1: Frequency Modulation

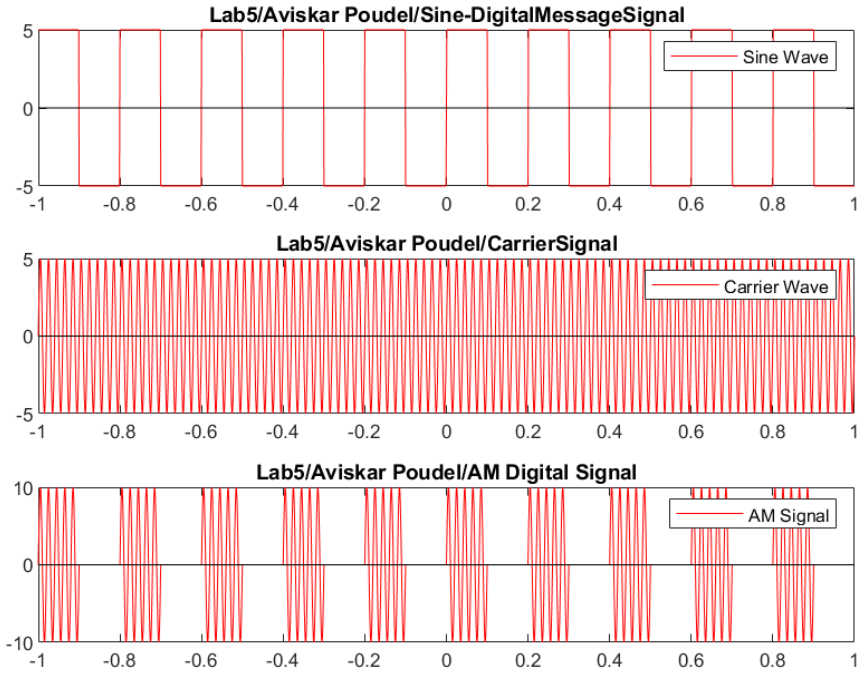

Figure 5.2: Amplitude Modulation

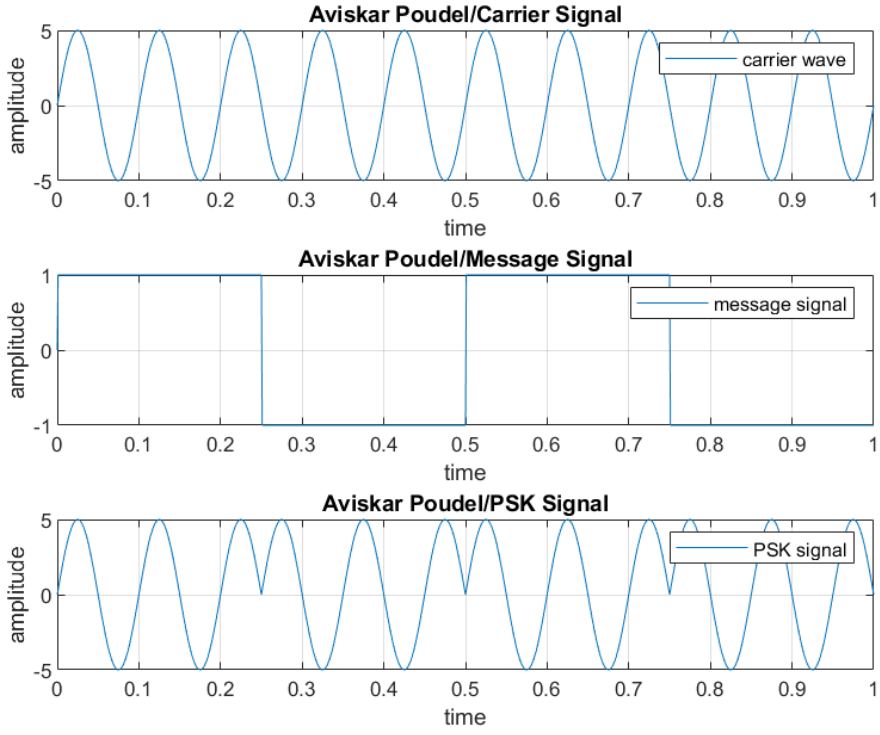

Figure 5.3: Phase Modulation