

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING



ADVANCED COLLEGE
OF ENGINEERING AND MANAGEMENT

Data Communication
Lab No. 1 to Lab No. 4

COMPILED LAB REPORT

Submitted By:

Aviskar Poudel
Roll No: ACE079BCT013

Submitted To:

Department of Electronics and
Computer Engineering

July 19, 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
```

```

3 A = [1 2; 3 4];
4 B = [5 6; 7 8];
5
6 matrix_add = A + B;
7 matrix_sub = A - B;
8 matrix_mul = A * B;
9
10
11 disp('Matrix Addition:');
12 disp(matrix_add);
13
14 disp('Matrix Subtraction:');
15 disp(matrix_sub);
16
17 disp('Matrix Multiplication:');
18 disp(matrix_mul);

```

Listing 1.3: Basic Trigonometric Operations

```

1
2 theta_deg = 30;
3
4
5 theta_rad = deg2rad(theta_deg);
6
7
8 sine_val = sin(theta_rad);
9 cosine_val = cos(theta_rad);
10
11
12 disp(['sin(', num2str(theta_deg), ') = ', num2str(sine_val)
13      ']);
14 disp(['cos(', num2str(theta_deg), ') = ', num2str(cosine_val)
15      ']);

```

Listing 1.4: Basic Complex Number Operations

```

1 z = 3 + 4i;
2
3 real_part = real(z);
4 imag_part = imag(z);
5 conj_val = conj(z);
6 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

```
1  a = [4,7];
2  b = [10,20];
3  x = [1,2];
4  y = [1,5];
5
6  c = [3,5];
7  d = [7,10];
8
9
10 subplot(3,1,1);
11 plot(a,b,'--black','LineWidth',1)
12 title('Linedraw1/Aviskar_Poudel/013');
13 legend('Dotted_Line')
14
15 subplot(3,1,2);
16 plot(x,y,'-blue','LineWidth',2)
17 title('Linedraw2/Aviskar_Poudel/013');
18 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  $y = a \sin(2\pi f t)$ ;
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 axis
   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')

```

```

30 legend('Sine_Wave')
31
32 subplot(4,2,4)
33 stem(t,yy,'-blue')
34 hold on
35 plot(t,zeros(size(t)),'-black')
36 title('Lab1/Aviskar_Poudel/CosineWaveDiscrete')
37 legend('Cosine_Wave')
38
39
40 subplot(4,2,5)
41 plot(t,yy,'-blue')
42 hold on
43 plot(t,y,'-red')
44 hold on
45 plot(t,zeros(size(t)),'-black')
46 hold on
47 title('Lab1/Aviskar_Poudel/sineandcosinecombined')
48 legend('Cosine_Wave', 'Sine_Wave')
49
50
51 subplot(4,2,6)
52 stem(t,yy,'-blue')
53 hold on
54 stem(t,y,'-red')
55 hold on
56 plot(t,zeros(size(t)),'-black')
57 hold on
58 title('Lab1/Aviskar_Poudel/sineandcosinecombined')
59 legend('Cosine_Wave', 'Sine_Wave')
60
61 grid on

```

Listing 2.3: All waves in same figure

```

1      %plot all 1 to 5 qn in same graph
2  a = 5;
3  f = 5;
4  t= 0:0.01:1;
5  y = a*sin(2*pi*f*t);
6  yy = a*cos(2*pi*f*t);
7

```

```

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

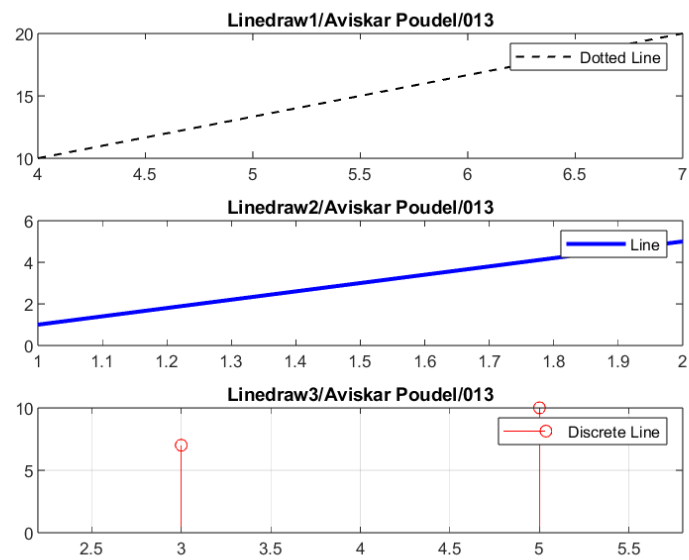


Figure 2.1: Lines

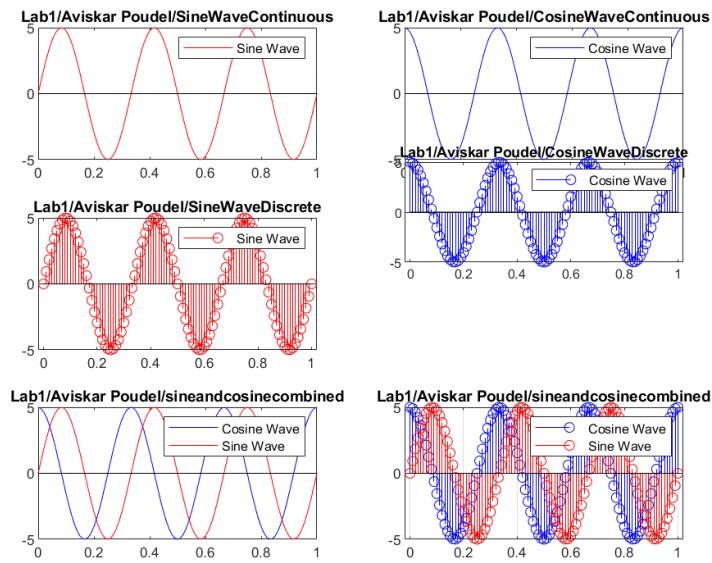


Figure 2.2: Different waves (Discrete and Continuous)

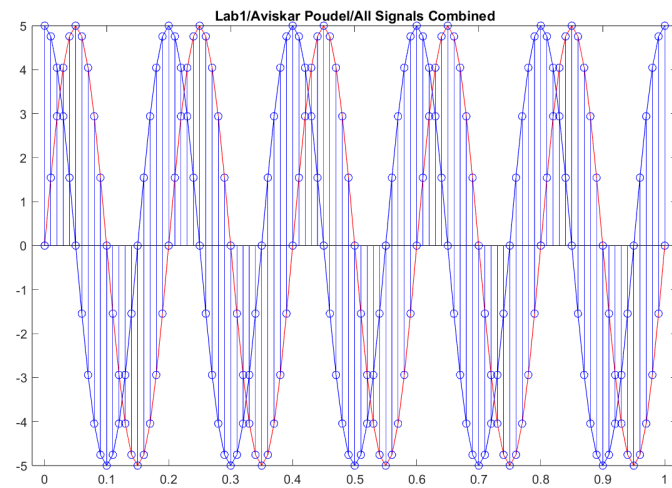


Figure 2.3: All in one figure

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')
```

```

8 legend('Square_Sine')
9 title('Lab3/Aviskar_Poudel/Square_SineWave')
10
11 subplot(2,1,2)
12 plot(t,c,'-red')
13 legend('Square_Cosine')
14 title('Lab3/Aviskar_Poudel/Square_Cosine_Wave')

```

Listing 3.3: Signum Function

```

1 x = zeros(size(t));
2
3 t = -2:0.1:2;
4
5 for i = 1:length(t)
6     if t(i) > 0
7         x(i) = 1;
8     elseif t(i) == 0
9         x(i) = 0;
10    else
11        x(i) = -1;
12    end
13 end
14 subplot(2,1,1)
15 plot(t,x,'-red')
16 legend("Signum function")
17 title('Lab3/Aviskar_Poudel/Ramp-Continuous')
18
19 subplot(2,1,2)
20 stem(t,x,'-red')
21 legend("Ramp function")
22 title('Lab3/Aviskar_Poudel/Ramp-Discrete')

```

Listing 3.4: Ramp Function

```

1
2 x = zeros(size(t));
3
4 t = -2:0.1:2;
5 for i = 1:length(t)
6     if t(i) >= 0
7         x(i) = t(i);
8     else

```

```

9         x(i) = 0;
10     end
11 end
12 subplot(2,1,1)
13 plot (t,x, '-red')
14 legend("Ramp function")
15 title('Lab3/Aviskar_Poudel/Ramp-Continuous')
16
17 subplot(2,1,2)
18 stem(t,x, '-red')
19 legend("Ramp function")
20 title('Lab3/Aviskar_Poudel/Ramp-Discrete')

```

Listing 3.5: Noise Wave

```

1     a = 5;
2     f = 3;
3     t = 0:0.01:1;
4     s = a*sin(2*pi*f*t);
5     c = a*cos(2*pi*f*t);
6     x = rand(1, length(t));
7     z1 = x + s;
8     z2 = x + c;
9     subplot(3,1,1)
10    plot(t,x)
11    legend('Noise')
12    title('Lab3/Aviskar_Poudel/Distorted_Wave')
13
14    subplot(3,1,2)
15    plot(t, z1)
16    legend('Distorted_Sine')
17    title('Lab3/Aviskar_Poudel/Distorted_Sine_Wave')
18
19    subplot(3,1,3)
20    plot(t,z2)
21    legend('Distorted_Cosine')
22    title('Lab3/Aviskar_Poudel/Distorted_Cosine_Wave')

```

Listing 3.6: Impulse Function

```

1
2     t = -2:0.1:2;
3     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

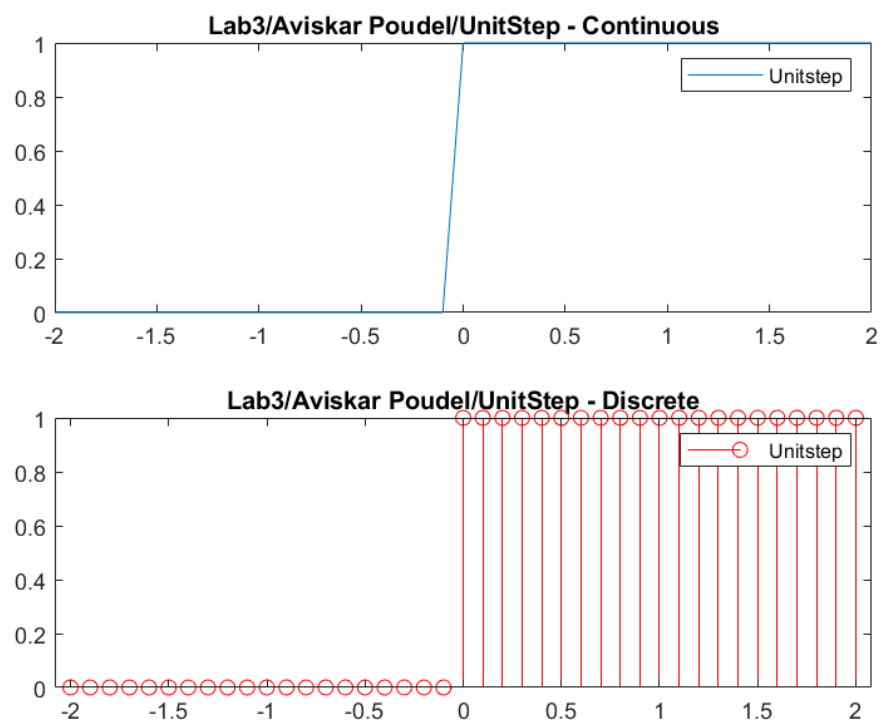


Figure 3.1: Unit Step Function

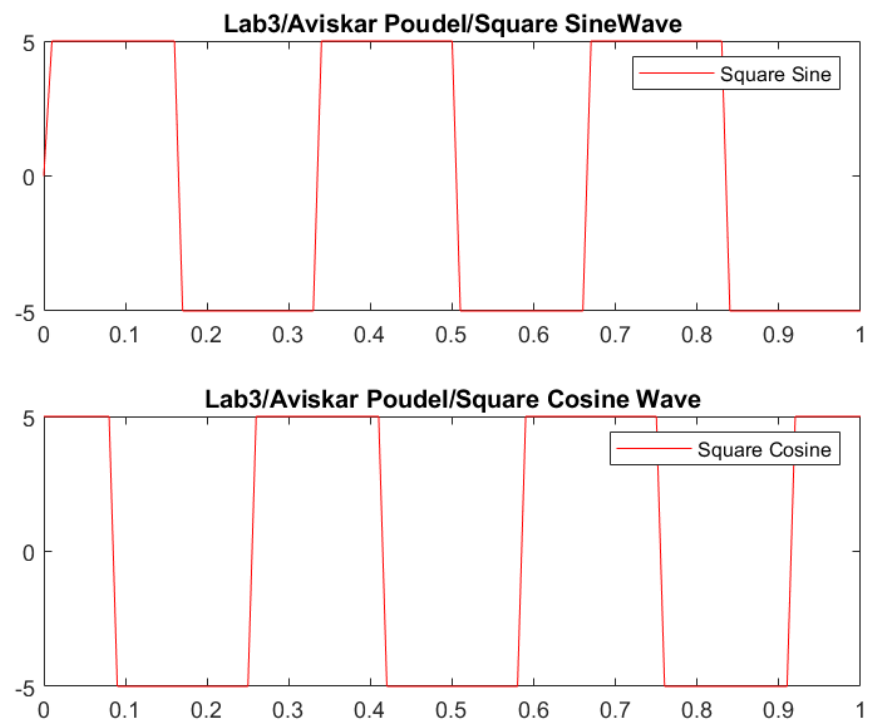


Figure 3.2: Square Wave

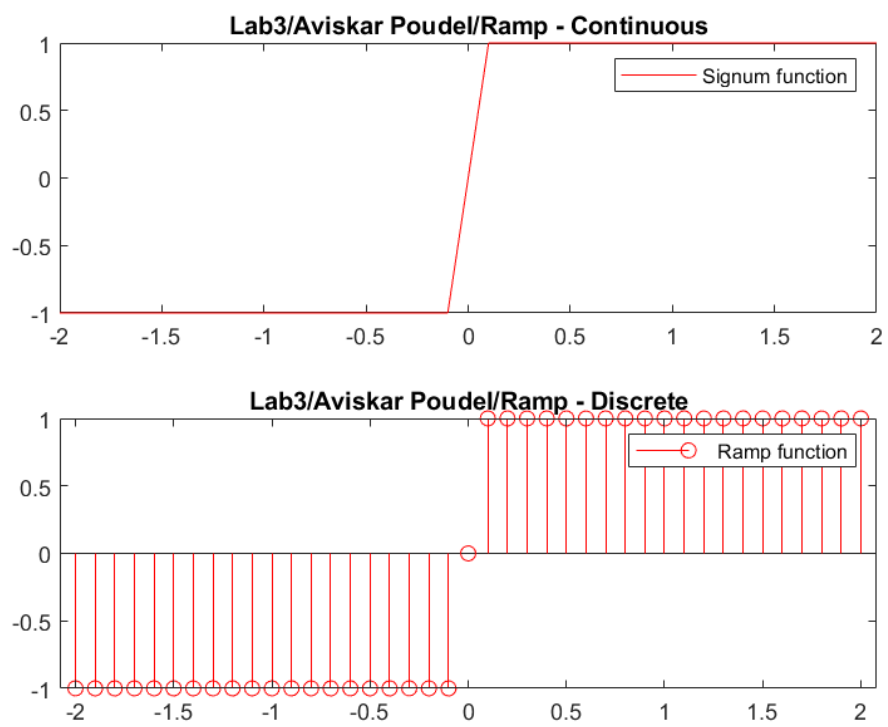


Figure 3.3: Signum Function

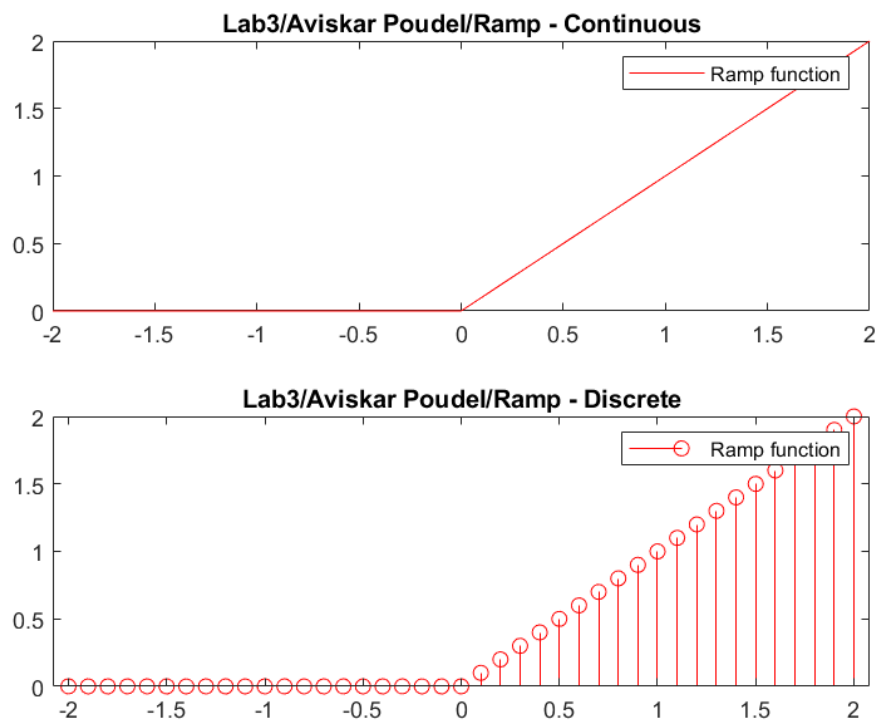


Figure 3.4: Ramp Function

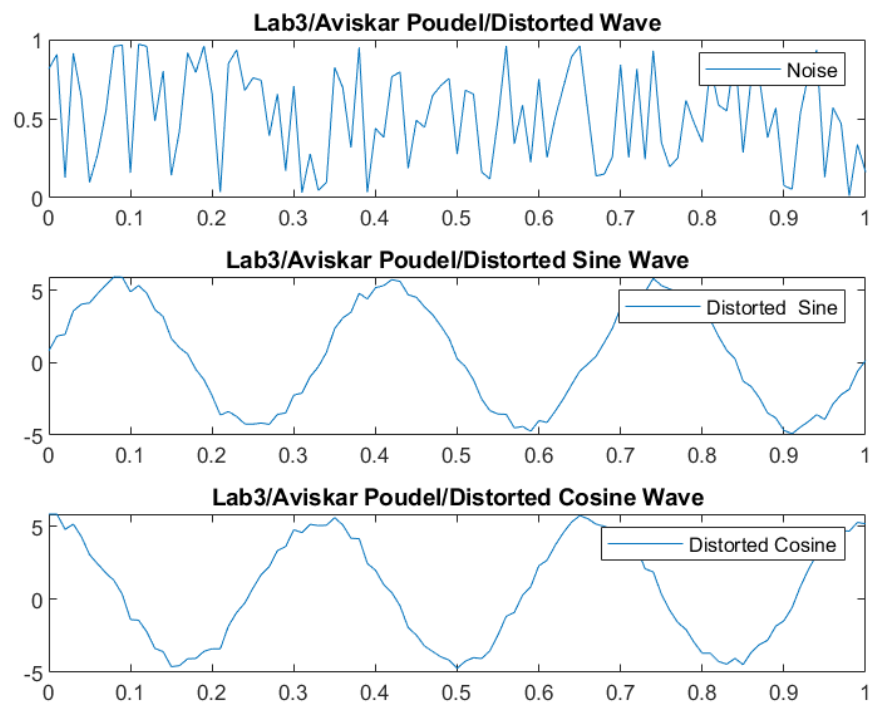


Figure 3.5: Noise Wave

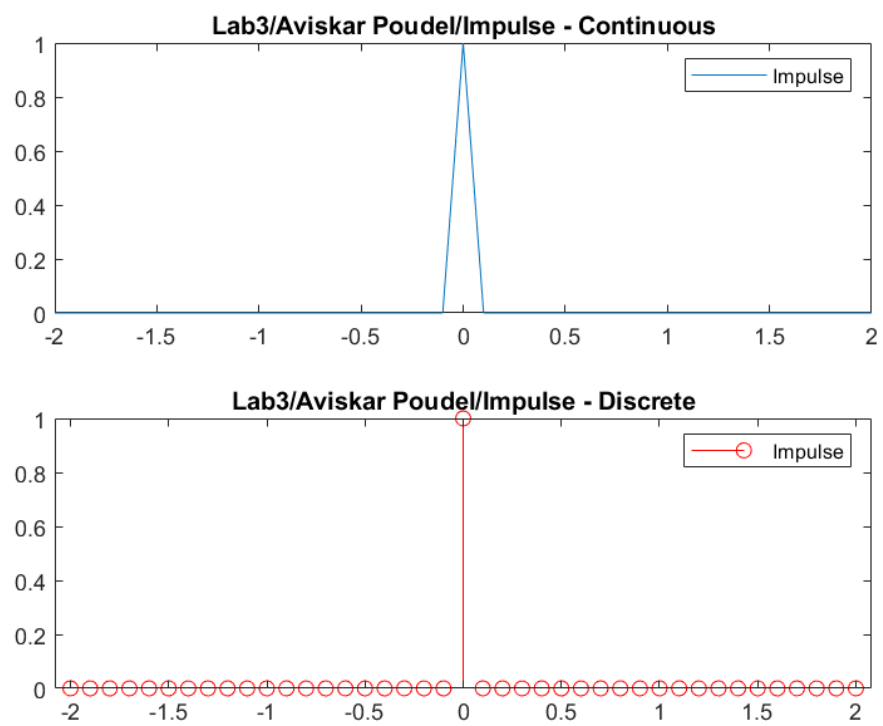


Figure 3.6: Impulse Function

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

```
1      t = -10:0.1:10;    % time vector
2
3  % sinc function: sin(x)/x
4  sinc_func = sin(t)./t;
5  sinc_func(t==0) = 1;    % define sinc(0)=1 manually
6
7  % Plot continuous
8  subplot(2,1,1)
9  plot(t, sinc_func)
10 legend("Sinc")
11 title('Lab4/Aviskar_Poudel/Sinc-Continuous')
12 xlabel('t')
13 ylabel('sinc(t)')
14
15 % Plot discrete
16 subplot(2,1,2)
17 stem(t, sinc_func, 'r')
18 legend("Sinc")
19 title('Lab4/Aviskar_Poudel/Sinc-Discrete')
20 xlabel('t')
21 ylabel('sinc(t)')
```

Listing 4.2: Exponentially Decaying Function

```
1  t = 0:0.1:10;    % time vector
2
3  % exponential decay function, e.g. e^{-t}
4  exp_decay = 5 * exp(-t);
5
6  % Plot continuous
7  subplot(2,1,1)
8  plot(t, exp_decay)
9  legend("Exp Decay")
10 title('Lab4/Aviskar_Poudel/Exponential_Decay-Continuous')
11 xlabel('t')
12 ylabel('e^{-t}')
13
14 % Plot discrete
```

```

15 subplot(2,1,2)
16 stem(t, exp_decay, 'r')
17 legend("Exp Decay")
18 title('Lab4/Aviskar_Poudel/Exponential_Decay_Discrete')
19 xlabel('t')
20 ylabel('e^{-t}')

```

Listing 4.3: Even and Odd function

```

1 t = -2:0.01:2;
2 %x = t;
3
4 for i = 1:length(t)
5     if t(i)>=0
6         x(i) = 2;
7     else
8         x(i) = 0;
9     end
10 end
11
12 % Compute even and odd parts
13 xe = (x + fliplr(x)) / 2;
14
15
16 xo = (x - fliplr(x)) / 2;
17
18 % Plot original
19 figure
20 subplot(3,2,1)
21 plot(t, x)
22 title('Lab4/Aviskar_Poudel/Original_Function')
23 xlabel('t')
24 ylabel('f(t)')
25
26 subplot(3,2,2)
27 stem(t, x, 'r')
28 title('Lab4/Aviskar_Poudel/Original_Function_Discrete')
29 xlabel('t')
30 ylabel('f(t)')
31
32 % Plot even part
33 subplot(3,2,3)

```

```

34 plot(t, xe)
35 title('Lab4/Aviskar_Poudel/Even_Function')
36 xlabel('t')
37 ylabel('f_e(t)')
38
39 subplot(3,2,4)
40 stem(t, xe, 'r')
41 title('Lab4/Aviskar_Poudel/Even_Function_Discrete')
42 xlabel('t')
43 ylabel('f_e(t)')
44
45 % Plot odd part
46 subplot(3,2,5)
47 plot(t, xo)
48 title('Lab4/Aviskar_Poudel/Odd_Function')
49 xlabel('t')
50 ylabel('f_o(t)')
51
52 subplot(3,2,6)
53 stem(t, xo, 'r')
54 title('Lab4/Aviskar_Poudel/Odd_Function_Discrete')
55 xlabel('t')
56 ylabel('f_o(t)')

```

Listing 4.4: Phase Modulation

```

1 am = 5;
2 fm = 3;
3 t = -1:0.001:1;
4
5 msg = am*cos(2*pi*fm*t);
6
7 subplot(3,1,1)
8 plot(t, msg, '-red')
9 hold on
10 plot(t, zeros(size(t)), '-black')
11 title('Lab4/Aviskar_Poudel/Cos_Msg_Signal')
12 legend('Cos_Msg_Signal')
13
14 ac = 10;
15 fc = 50;
16

```

```

17 carrier = ac*cos(2*pi*fc*t);
18
19 subplot(3,1,2)
20 plot(t, carrier, '-red')
21 hold on
22 plot(t, zeros(size(t)), '-black')
23 title('Lab4/Aviskar_Poudel/Carrier_Signal')
24 legend('Cos_Carrier_Signal')
25
26 kp = 1;
27 beta = kp * am;
28
29 PM = ac*cos(2*pi*fc*t + beta*cos(2*pi*fm*t));
30
31 subplot(3,1,3)
32 plot(t, PM, '-red')
33 hold on
34 plot(t, zeros(size(t)), '-black')
35 title('Lab4/Aviskar_Poudel/Phase_Mod_Signal')
36 legend('Phase_Mod_Signal')

```

Listing 4.5: Frequency Modulation

```

1
2 am = 5;
3 fm = 3;
4 t = -1:0.001:1;
5
6 msg = am*cos(2*pi*fm*t);
7
8 subplot(3,1,1)
9 plot(t, msg, '-red')
10 hold on
11 plot(t, zeros(size(t)), '-black')
12 title('Lab4/Aviskar_Poudel/Cos_Msg_Signal')
13 legend('Cos_Msg_Signal')
14
15 ac = 10;
16 fc = 50;
17
18 carrier = ac*cos(2*pi*fc*t);
19

```

```

20 subplot(3,1,2)
21 plot(t, carrier, '-red')
22 hold on
23 plot(t, zeros(size(t)), '-black')
24 title('Lab4/Aviskar_Poudel/Carrier_Signal')
25 legend('Cos_Carrier_Signal')
26
27 kf = 5;
28 beta = (kf*am)/fm;
29
30 FM = ac*cos(2*pi*fc*t + beta*sin(2*pi*fm*t));
31
32 subplot(3,1,3)
33 plot(t, FM, '-red')
34 hold on
35 plot(t, zeros(size(t)), '-black')
36 title('Lab4/Aviskar_Poudel/Freq_Mod_Signal')
37 legend('Freq_Mod_Signal')

```

Listing 4.6: Amplitude Modulation

```

1 t = -1:0.001:1;
2
3 a = 5;
4 f = 5;
5 y = a*sin(2*pi*f*t);
6
7 subplot(3,1,1)
8 plot(t, y, '-red')
9 hold on
10 plot(t, zeros(size(t)), '-black')
11 title('Lab4/Aviskar_Poudel/SineMessageSignal')
12 legend('Sine_Wave')
13
14 hf = 50;
15 a2 = 5;
16 carrier = a2*sin(2*pi*hf*t);
17
18 subplot(3,1,2)
19 plot(t, carrier, '-red')
20 hold on
21 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

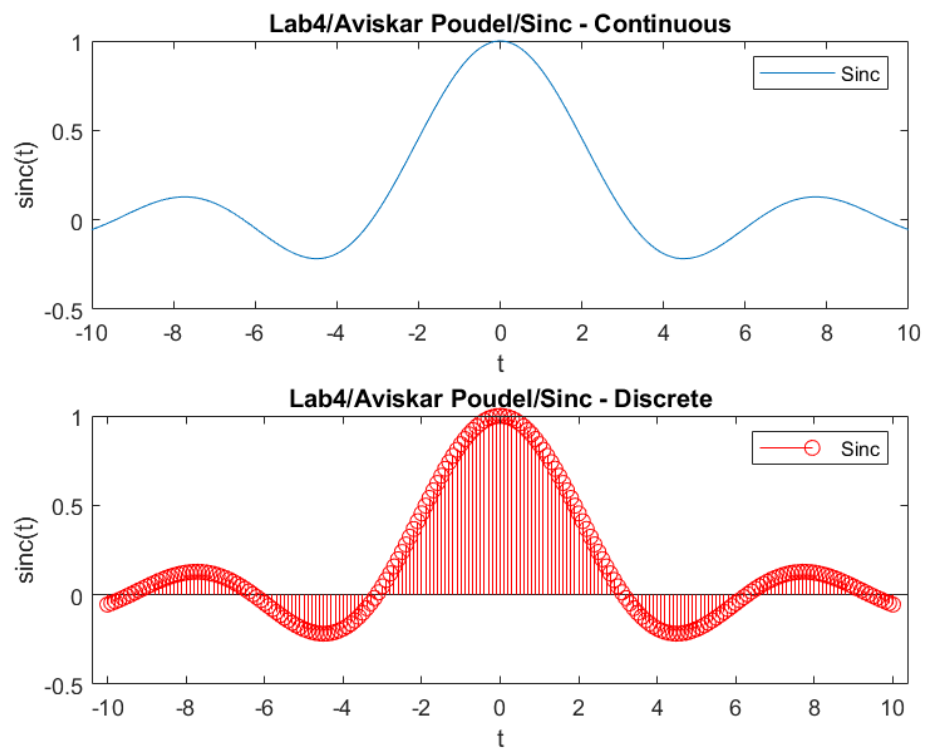


Figure 4.1: Sinc Function

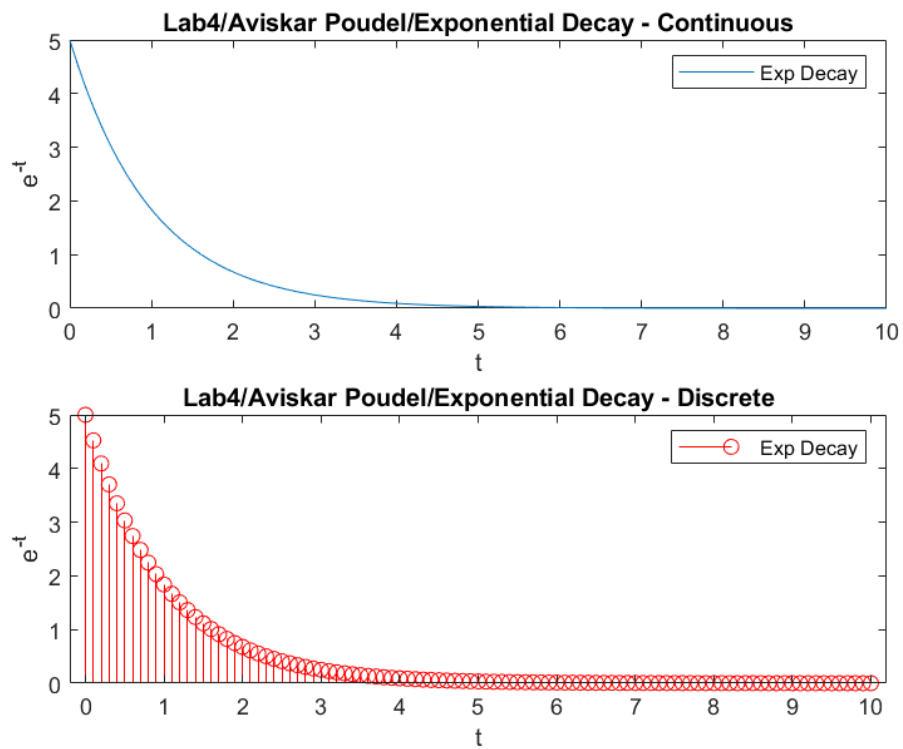


Figure 4.2: Exponentially Decaying Function

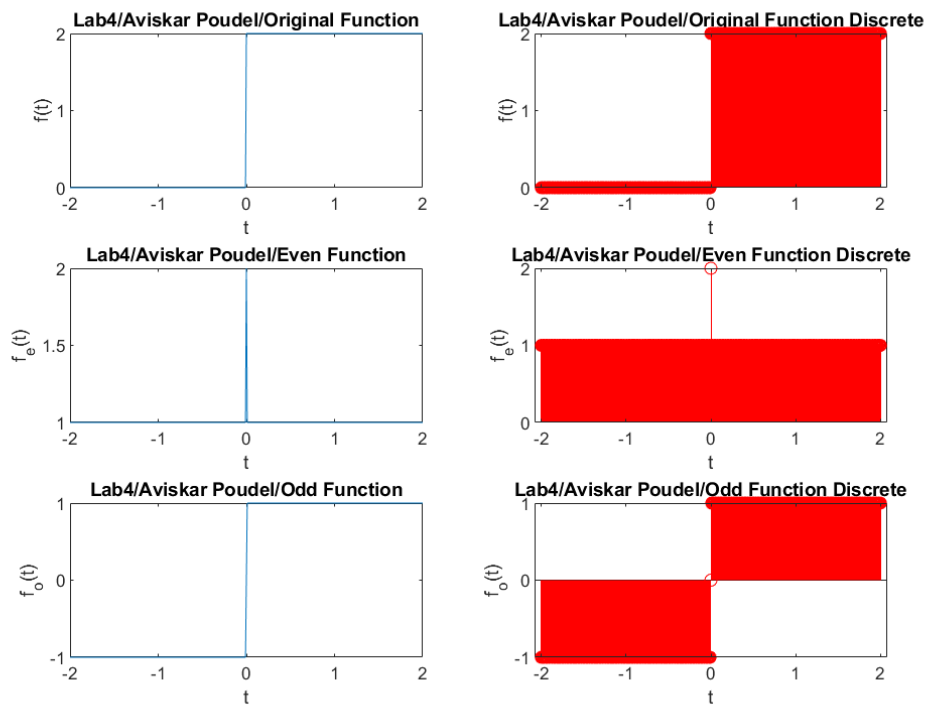


Figure 4.3: Even Odd Function

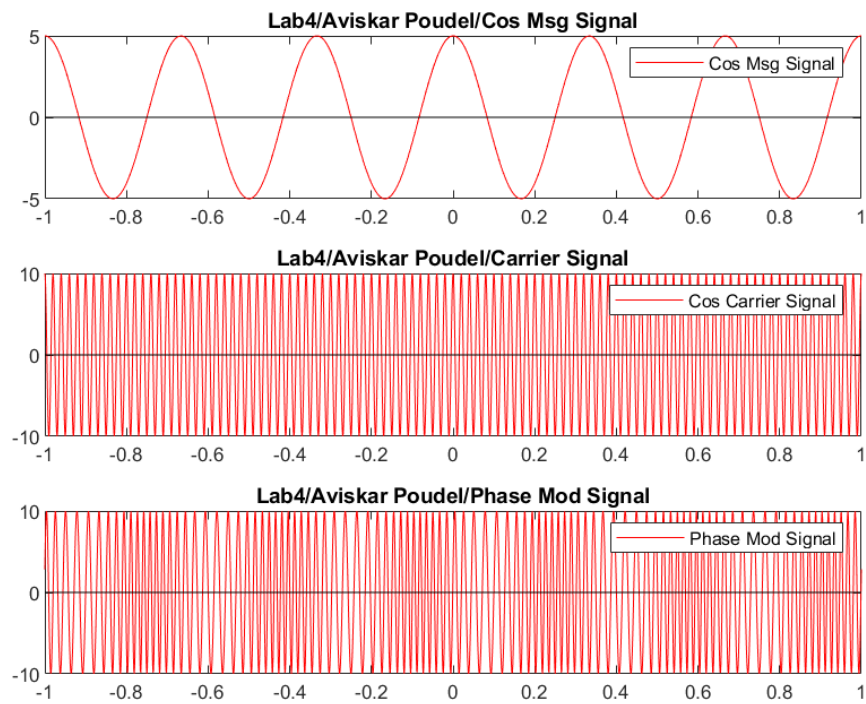


Figure 4.4: Phase Modulation

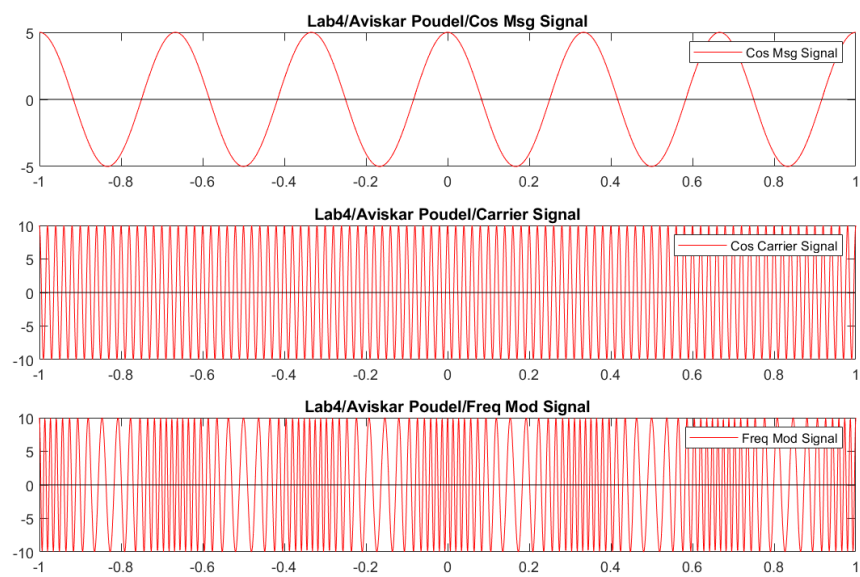


Figure 4.5: Frequency Modulation

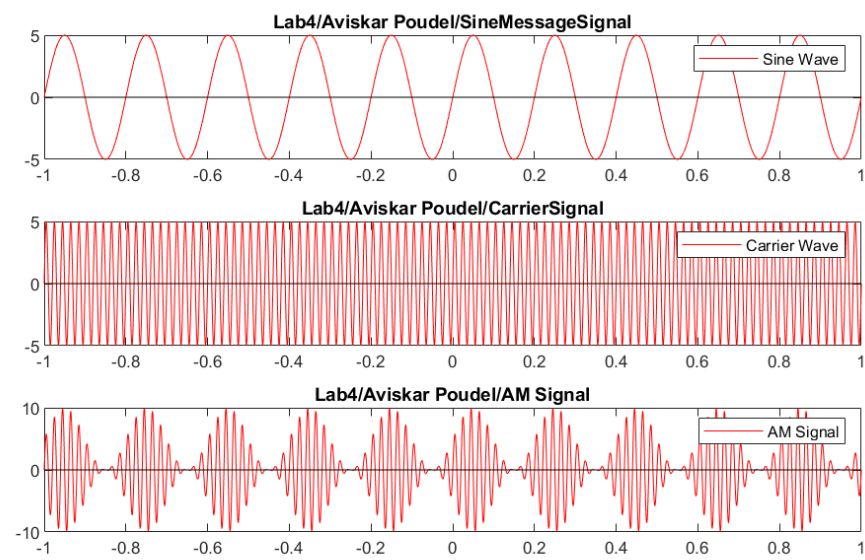


Figure 4.6: Amplitude Modulation