

Table of Contents

S.No.	Experiment	Description
1.	Creating Arrays	Creates one and two-dimensional arrays, row/column vectors, and matrices of a given size. Performs arithmetic operations like addition, subtraction, multiplication, and exponentiation. Performs matrix operations like inverse, transpose, rank, and plots.
2.	Matrix Manipulations	Concatenates, indexes, sorts, shifts, reshapes, resizes, and flips about a vertical/horizontal axis.
3.	Relational and Logical Operations	Creates arrays X and Y of a given size (1 x N) and performs relational and logical operations like >, <, ==, <=, >=, ~=, ~, &.
4.	Random Sequences and Plots	Generates random sequences using rand() and randn() functions and plots them.
5.	Evaluating Expressions and Plots	Evaluates a given expression and rounds it to the nearest integer value using Round, Floor, Ceil, and Fix functions. Generates and plots trigonometric functions like sin(t), cos(t), tan(t), sec(t), cosec(t), and cot(t) for a given duration 't'. Generates and plots logarithmic and other functions like log(A), log10(A), square root of A, and real nth root of A. Creates a vector X with elements, $X_n = (-1)^{n+1}/(2n-1)$ and adds up 100 elements of the vector X. Plots the functions x , x^3 , $\exp(x^2)$ over the interval $0 < x < 4$.
6.	Generating Sinusoidal Signals	Generates a sinusoidal signal of a given frequency with titling, labeling, adding text, adding legends, and printing text in Greek letters. Plots as multiple and subplot. Time scales the generated signal for different values like 2X, 4X, 0.25X, and 0.0625X.
7.	Solving Differential Equations	Solves first, second, and third-order ordinary differential equations using built-in functions and plots.
8.	Input Scripts	Writes brief scripts starting each script with a request for input (using input) to evaluate the function h(T) using if-else statements, where $h(T) = (T - 10)$ for $0 < T < 100$ and $h(T) = (0.45 T + 900)$ for $T > 100$.
9.	Generating Square Waves	Generates a square wave from the sum of sine waves of certain amplitude and frequencies.
10.	Basic 2D and 3D Plots	Generates parametric space curves, polygons with vertices, 3D contour lines, pie charts, and bar charts.

Experiment 1 : Creating Arrays

```
% Array Addition
a = [1 2 3 4];
b = [5 6 7 8];

c = a + b;
d = a - b;

% Matrix Addition and Multiplication

A = [1 2 3; 4 5 6; 7 8 9; ];
B = [ 10 11 12; 13 14 15; 16 17 19 ];

C = A + B;
D = A * B;

% Transpose a Matrix

E = A';

% Rank of a Matrix

F = rank(A);

plot(c)

plot(d)

plot(C)

plot(D)

plot(E)
```

Experiment 2 : Matrix Manipulations

```
A = [1 2 3; 4 5 6; 7 8 9;];
```

```
B = [10 11 12; 13 14 15; 16 17 18;];
```

```
C = [A;B];
```

```
C = cat(dim, A, B);
```

Experiment 3 : Relational and Logical Operations

```
% Add up the values of the elements (Check with sum)
X = [1, 2, 3, 4, 5];
total_sum = sum(X);

% Compute the Running Sum (Check with sum), where Running Sum for element j
= the sum of the elements from 1 to j, inclusive.

running_sum = cumsum(X);

% Generate a random sequence using rand() function
seq = rand(1, 100); % generates 1x100 array of random values between 0 and 1

% Plot the sequence
plot(seq)
title('Random Sequence Generated Using rand() Function')
xlabel('Index')
ylabel('Value')

% Generate a random sequence using randn() function
seq = randn(1, 100); % generates 1x100 array of random values from a normal
distribution with mean 0 and standard deviation 1

% Plot the sequence
plot(seq)
title('Random Sequence Generated Using randn() Function')
xlabel('Index')
ylabel('Value')
```

Experiment 4 : Random Sequences and Plots

% Trigonometric Functions - sin(t), cos(t), tan(t), sec(t), cosec(t) and cot(t) for a given duration, 't'.

```
x = 5.7;
rounded1 = round(x);
rounded2 = floor(x);
rounded3 = ceil(x);
rounded4 = fix(x);

t = 0:0.01:2*pi;
y1 = sin(t);
y2 = cos(t);
y3 = tan(t);
y4 = sec(t);
y5 = csc(t);
y6 = cot(t);

subplot(2,3,1);
plot(t, y1);
title('Sine Function');
xlabel('t');
ylabel('sin(t)');

subplot(2,3,2);
plot(t, y2);
title('Cosine Function');
xlabel('t');
ylabel('cos(t)');

subplot(2,3,3);
plot(t, y3);
title('Tangent Function');
xlabel('t');
ylabel('tan(t)');

subplot(2,3,4);
plot(t, y4);
title('Secant Function');
```

```

xlabel('t');
ylabel('sec(t)');

subplot(2,3,5);
plot(t, y5);
title('Cosecant Function');
xlabel('t');
ylabel('csc(t)');

subplot(2,3,6);
plot(t, y6);
title('Cotangent Function');
xlabel('t');
ylabel('cot(t)');

% Logarithmic and other Functions – log(A), log10(A), Square root of A, Real
nth root of A.

A = 1:0.1:10; % sample points from 1 to 10 with 0.1 interval
y1 = log(A);
y2 = log10(A);
y3 = sqrt(A);
y4 = nthroot(A, 3); % 3rd root of A

subplot(2,2,1);
plot(A, y1);
title('Natural Logarithm');
xlabel('A');
ylabel('log(A)');

subplot(2,2,2);
plot(A, y2);
title('Base 10 Logarithm');
xlabel('A');
ylabel('log_{10}(A)');

subplot(2,2,3);
plot(A, y3);
title('Square Root');
xlabel('A');
ylabel('\sqrt{A}');

```

```
subplot(2,2,4);  
plot(A, y4);  
title('Cubic Root');  
xlabel('A');  
ylabel('\sqrt[3]{A}');
```

Experiment 5 : Evaluating Expressions and Plots

% Creating a vector X with elements $X_n = (-1)^{(n+1)}/(2n-1)$ and adding up 100 elements of the vector X

```
N = 100;
X = zeros(1, N);

for n = 1:N
    X(n) = (-1)^(n+1)/(2*n-1);
end

sum_X = sum(X(1:N));
```

% Plotting the functions x, x^3, exp(x), and exp(x^2) over the interval 0 < x < 4

```
x = 0:0.01:4;
y1 = x;
y2 = x.^3;
y3 = exp(x);
y4 = exp(x.^2);
```

```
subplot(2,2,1);
plot(x, y1);
title('x');
xlabel('x');
ylabel('y');
```

```
subplot(2,2,2);
plot(x, y2);
title('x^3');
xlabel('x');
ylabel('y');
```

```
subplot(2,2,3);
plot(x, y3);
title('exp(x)');
xlabel('x');
ylabel('y');
```



```
subplot(2,2,4);  
plot(x, y4);  
title('exp(x^2)');  
xlabel('x');  
ylabel('y');
```

Experiment 6 : Generating Sinusoidal Signals

```
% Define the frequency of the signal
freq = 2;

% Define the time duration of the signal
t = 0:0.01:2;

% Generate the sinusoidal signal
x = sin(2*pi*freq*t);

% Plot the signal
plot(t,x);

% Add labels and title
xlabel('Time (s)');
ylabel('Amplitude');
title('Sinusoidal Signal');

% Add legends
legend('Signal');

% Add text
text(1.5,0.5,'Signal Frequency = 2 Hz');

% Print Greek letters
text(1.5,-0.5,'\omega = 2\pi f');

% Generate multiple plots
figure;
subplot(2,1,1);
plot(t,x);
xlabel('Time (s)');
ylabel('Amplitude');
title('Sinusoidal Signal');

subplot(2,1,2);
plot(t,sin(2*pi*4*t));
xlabel('Time (s)');
ylabel('Amplitude');
title('Sinusoidal Signal with 4 Hz frequency');
```

Experiment 7 : Solving Differential Equations

```
% First Order

% Define the differential equation
dydt = @(t,y) -2*y + 1;

% Define the time duration and initial condition
tspan = [0 10];
y0 = 0;

% Solve the differential equation
[t,y] = ode45(dydt,tspan,y0);

% Plot the solution
plot(t,y);
xlabel('Time');
ylabel('y');
title('Solution to First-Order Differential Equation');

% % % % % % % % % % % % % % % %

% Second Order

% Define the differential equation
dy2dt2 = @(t,y) -2*y(1) - 0.5*y(2) + 1;
dy1dt = @(t,y) y(2);
dydt = @(t,y) [dy1dt(t,y); dy2dt2(t,y)];

% Define the time duration and initial condition
tspan = [0 10];
y0 = [0; 0];

% Solve the differential equation
[t,y] = ode45(dydt,tspan,y0);

% Plot the solution
plot(t,y(:,1));
xlabel('Time');
ylabel('y');
title('Solution to Second-Order Differential Equation');
```

% % % % % % % % % % % % % % % %

% Third Order

% Define the differential equation

dy3dt3 = @(t,y) -2*y(1) - 0.5*y(2) + 1;

dy2dt = @(t,y) y(3);

dy1dt = @(t,y) y(2);

dydt = @(t,y) [dy1dt(t,y); dy2dt(t,y); dy3dt3(t,y)];

% Define the time duration and initial condition

tspan = [0 10];

y0 = [0; 0; 0];

% Solve the differential equation

[t,y] = ode45(dydt,tspan,y0);

% Plot the solution

plot(t,y(:,1));

xlabel('Time');

ylabel('y');

title('Solution to Third-Order Differential Equation');

Experiment 8 : Input Scripts

```
% Request user input for T value
T = input("Enter a value for T between 0 and 100: ");

% Check if T is within the valid range
if T < 0 || T > 100
    disp("Invalid input. T must be between 0 and 100.")
else
    % Evaluate the function h(T)
    h = T - 10;
    disp("h(T) = " + h);
end

% Request user input for T value
T = input("Enter a value for T: ");

% Check if T is greater than 100
if T > 100
    % Evaluate the function h(T)
    h = 0.45*T + 900;
    disp("h(T) = " + h);
else
    disp("Invalid input. T must be greater than 100.")
end
```

Experiment 9 : Generating Square Waves

```
% Define the frequency and amplitude of the sine waves
f1 = 100; % frequency of first sine wave
A1 = 1; % amplitude of first sine wave
f2 = 300; % frequency of second sine wave
A2 = 0.5; % amplitude of second sine wave

% Define the time axis and sampling rate
Fs = 10000; % sampling rate
t = 0:1/Fs:1; % time axis

% Generate the sine waves
s1 = A1*sin(2*pi*f1*t);
s2 = A2*sin(2*pi*f2*t);

% Add the sine waves together to create a square wave
square_wave = s1 + s2;
square_wave(square_wave >= 0) = 1;
square_wave(square_wave < 0) = -1;

% Plot the square wave
plot(t, square_wave);
ylim([-1.5 1.5]);
xlabel('Time (s)');
ylabel('Amplitude');
title('Square Wave from Sum of Sine Waves');
```

Experiment 10 : Basic 2D and 3D Plots

```
% Parametric Space Curve

% Define the parameter t
t = linspace(0, 2*pi, 1000);

% Define the x, y, and z coordinates as functions of t
x = cos(t);
y = sin(t);
z = t;

% Plot the curve in 3D
plot3(x, y, z, 'LineWidth', 2);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('Parametric Space Curve');


% Polygon with Vertices

% Define the vertices of the polygon
x = [0 1 1 0];
y = [0 0 1 1];

% Plot the polygon
patch(x, y, 'r');
axis equal;
xlabel('X');
ylabel('Y');
title('Polygon with Vertices');


% 3D Contour Lines

% Define the function to plot
[X,Y,Z] = peaks(25);

% Plot the contour lines
```

```
contour3(X,Y,Z,20);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('3D Contour Lines');

% Pie Chart

% Define the data to plot
data = [20 30 50];

% Plot the pie chart
pie(data);
legend({'Slice 1', 'Slice 2', 'Slice 3'});
title('Pie Chart');

% Bar Chart

% Define the data to plot
data = [10 20 30 40 50];

% Plot the bar chart
bar(data);
xlabel('X');
ylabel('Y');
title('Bar Chart');
```