KONGUNADU

ARTS AND SCIENCE COLLEGE

(Autonomous)

Re-accredited by NAAC with 'A+' Grade (4th Cycle)

College of Excellence (UGC)

29th Rank among Colleges in NIRF 2023

G. N. Mills (Post), Coimbatore - 641 029

www.kongunaducollege.ac.in

DEPARTMENT OF MATHEMATICS

Supported by DBT STAR College Scheme

LAB MANUAL

B. Sc. MATHEMATICS

MATLAB

2022-2023

INTRODUCTION TO MATLAB

In this introduction we will describe how MATLAB handles simple numerical expressions and mathematical formulas.

The name **MATLAB** stands for **MATrix LABoratory.** MATLAB was written originally to provide easy access to matrix software developed by the **LINPACK** *(linear system package)* and **EISPACK***(Eigen system package)* projects.

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment it has sophisticated data structures, contains built-in editing and debugging tools, and supports *object-oriented programming*. These factors make MATLAB an excellent tool for teaching and research.

MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide.

It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

In addition to the MATLAB documentation which is mostly available on-line.

Here's a brief introduction to Matlab's key features and concepts:

1. **Interactive Environment:** Matlab provides an interactive environment where you can write and execute commands line by line. *It features a command window known as the Matlab Console*, where you can enter commands and see the results immediately.
2. **Numeric Computations:** Matlab excels at performing numerical computations. It supports various mathematical operations and functions for matrix manipulations, linear algebra, numerical integration, optimization, and more.
3. **Data Visualization:** Matlab offers a wide range of built-in functions and tools for creating 2D and 3D plots, graphs, and visualizations. You can generate plots to analyze and present data in a visually appealing manner.
4. **Programming Language:** Matlab is also a programming language that supports control flow structures like loops *(for, while),* conditional statements *(if-else),* and

functions. It allows you to write scripts and functions to automate tasks and perform complex calculations.

5. **Extensive Function Library:** Matlab provides an extensive library of built-in functions and toolboxes, covering areas such as signal processing, image processing, control systems, optimization, statistics, and more. These libraries make it easier to perform specific tasks without writing everything from scratch.
6. .**File I/O and Data Import/Export:** Matlab enables you to read data from various file formats (such as CSV, Excel, text files) and import it into Matlab for analysis. You can also export results and data to different file formats.
7. **Simulink:** Simulink is a powerful companion tool to Matlab that allows you to model, simulate, and analyze dynamic systems using block diagrams. It is commonly used in control systems, signal processing, and other domains.

Matlab's syntax is relatively straightforward and uses English-like commands. It supports both procedural and object-oriented programming styles. The official Matlab documentation, tutorials, and online resources provide extensive guidance for learning and using Matlab effectively.

Overall, Matlab is a versatile software environment widely used by researchers, engineers, and scientists for a broad range of computational tasks, analysis, and simulations.

Starting MATLAB

After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the **MATLAB desktop** appears. The desktop is a window that contains other windows.

The major tools within or accessible from the desktop are:

• The Command Window

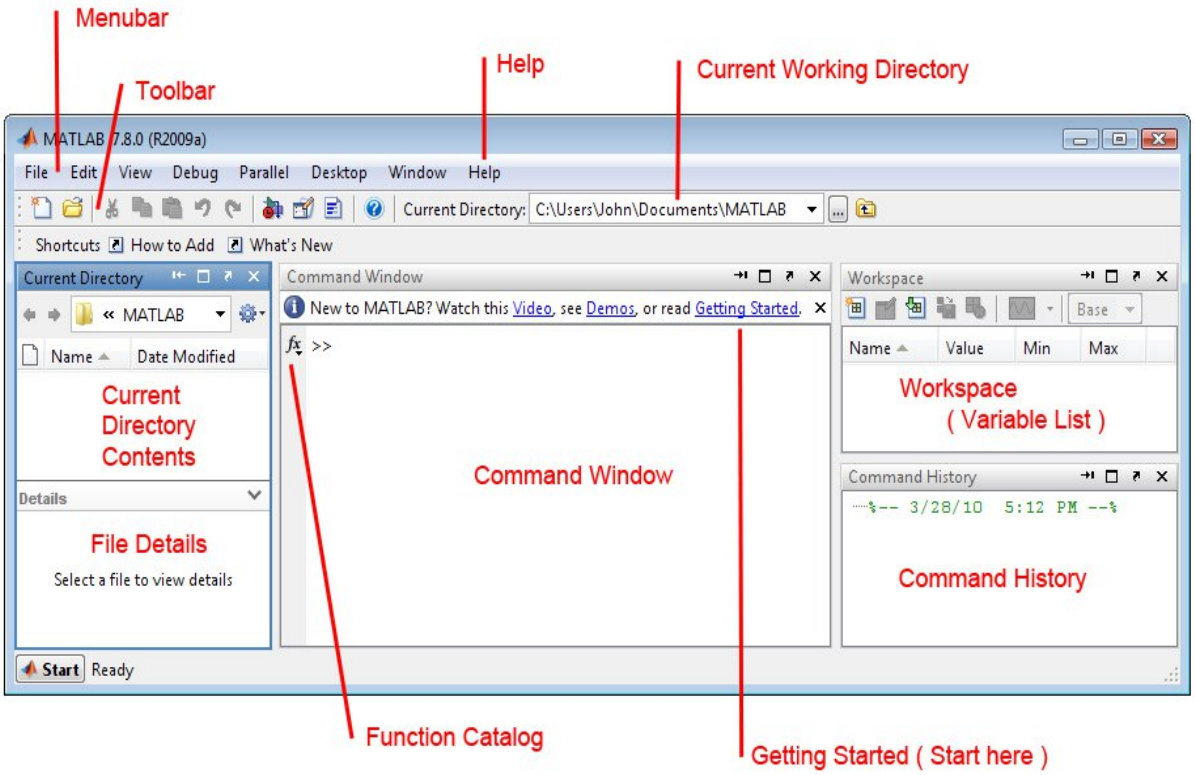
• The Command History

• The Workspace

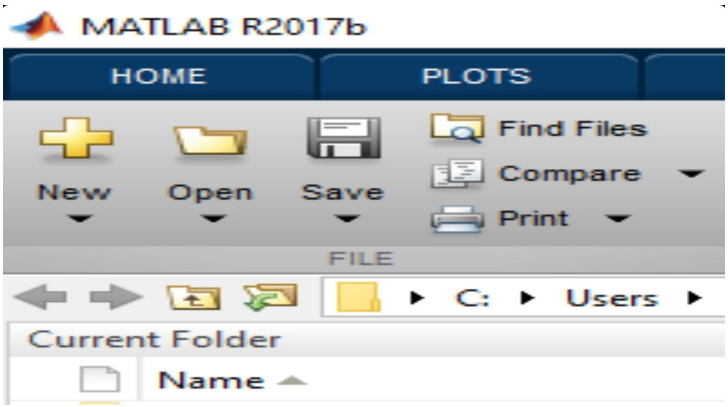
• The Current Directory

• The Help Browser

The MATLAB Work Environment

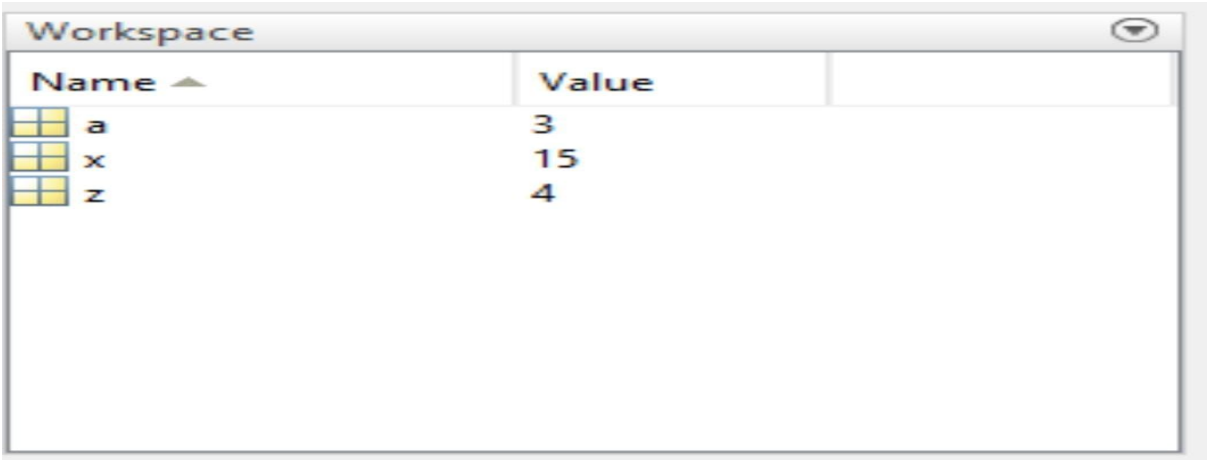


Current Folder - This panel allows you to access the project folders and files.

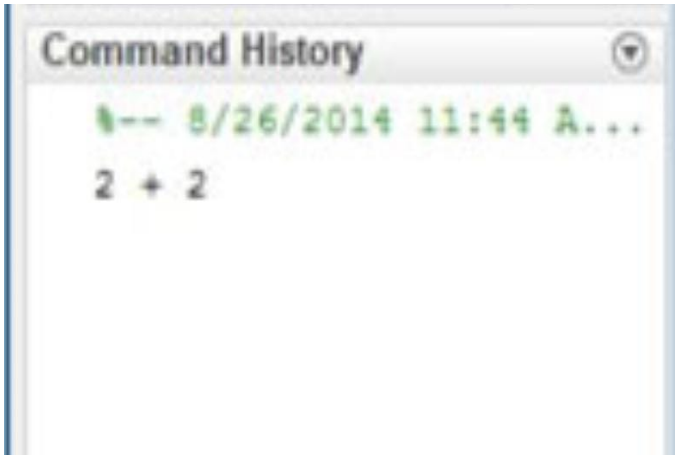


Command Window - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).

Workspace - Theworkspace shows all the variables created and/or imported from files.



Command History - This panel shows or rerun commands that are entered at the command line.



You are now faced with the MATLAB desktop on your computer, which contains the prompt (>>) in the Command Window. Usually, there are 2 types of prompt:

>>For full version

EDU>For educational version

Note:1. To simplify the notation, we will use this prompt, >>, as a standard prompt sign, though our MATLAB version is for educational purpose.

2. MATLAB adds variable to the workspace and displays the result in the Command Window.

Using MATLAB as a calculator

As an example of a simple interactive calculation, just type the expression you want to evaluate. Let's start at the very beginning.

For example, let's suppose you want to calculate the expression, 1 + 2 × 3. You type it at the prompt command (>>) as follows,

```
>> 1+2*3

ans = 7
```

You will have noticed that if you do not specify an output variable, MATLAB uses a default variable **ans,** short for **answer**, to store the results of the current calculation.

Note that the variable ans is created (or overwritten, if it is already existed). To avoid this, you may assign a value to a variable or output argument name. For example,

```
>> x = 1+2*3

x = 7
```

will result in x being given the value 1 + 2 × 3 = 7. This variable name can always be used to refer to the results of the previous computations. Therefore, computing 4x will result in

```
>> 4*x
```

ans = 28.0000

Table 1.1 gives the partial list of arithmetic operators.

Table 1.1: *Basic arithmetic operators*

Symbol	Operation	Example
+	Addition	2 + 3
-	Subtraction	2 - 3
*	Multiplication	2 * 3
/	Division	2 / 3

Creating MATLAB variables

MATLAB variables are created with an assignment statement. The syntax of variable assignment is

variable name = a value (or an expression)

For example,

>> x = expression

where ***expression*** is a combination of numerical values, mathematical operators, variables, and function calls.

On other words, expression can involve:

• manual entry

• built-in functions

• user-defined functions

Overwriting variable

Once a variable has been created, it can be reassigned. In addition, if you do not wish to see the intermediate results, you can suppress the numerical output by putting a semicolon (**;**) at the end of the line.

Then the sequence of commands looks like this:

>> t = 5;

>> t = t+1

t = 6

Error messages

If we enter an expression incorrectly, *MATLAB* will return an**error message**. For example, in the following, we left out the multiplication sign, *, in the following expression

>> x = 10;

>> 5x

??? 5x

Error: Unexpected MATLAB expression.

Making corrections

To make corrections, we can, of course retype the expressions. But if the expression is lengthy, we make more mistakes by typing a second time. A previously typed command can be recalled with the up-arrow key ↑. When the command is displayed at the command prompt, it can be modified if needed and executed.

Controlling the hierarchy of operations or precedence

Let's consider the previous arithmetic operation, but now we will include parentheses.

For example, 1 + 2 × 3 will become (1 + 2) × 3

>> (1+2)*3

ans = 9

and, from previous example

>> 1+2*3

ans = 7

By adding parentheses, these two expressions give different results: 9 and 7.

Managing the workspace

The contents of the workspace persist between the executions of separate commands. Therefore, it is possible for the results of one problem to have an effect on the next one. To avoid this possibility, it is a good idea to issue a clear command at the start of each new independent calculation.

>>clear

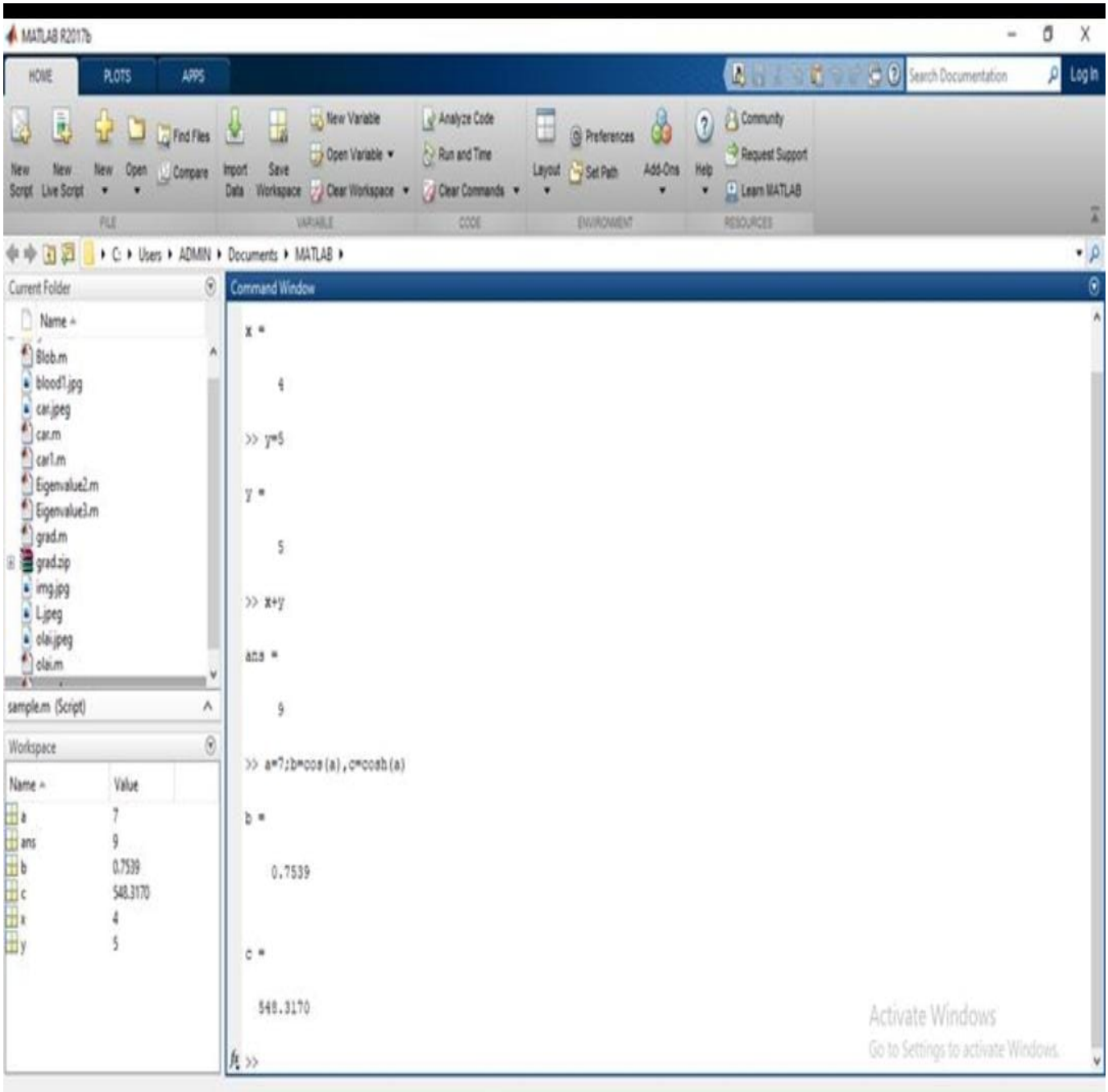
The command **clear** or **clear all** removes all variables from the workspace. This frees up system memory.

Entering multiple statements per line

It is possible to enter multiple statements per line. Use commas (,) or semicolons (;) to enter more than one statement at once. Commas (,) allow multiple statements per line without suppressing output.

```
>> a=7; b=cos(a), c=cosh(a)

 b =  0.6570

c = 548.3170
```

Example:



Getting help

To view the online documentation, select **MATLAB Help** from **Help menu** or MATLAB Help directly in the Command Window. The preferred method is to use the Help Browser.

The Help Browser can be started by selecting the ?icon from the desktop toolbar. On the other hand, information about any command is available by typing

>>**help Command**

Another way to get help is to use the **lookfor** command. The lookfor command differs from the help command.

The help command searches for an exact function name match, while the lookfor command searches the quick summary information in each function for a match.

For example, suppose that we were looking for a function to take the inverse of a matrix. Since MATLAB does not have a function named inverse, the command help inverse will produce nothing. On the other hand, the command lookfor inverse will produce detailed information, which includes the function of interest, inv.

>>**lookfor inverse**

Note : At this particular time of our study, it is important to emphasize one main point. Because MATLAB is a huge program; it is impossible to cover all the details of each function one by one. However, we will give you information how to get help. Here are some examples:

• Use on-line help to request info on a specific function

>>**help sqrt**

• In the current version (MATLAB version 7), the doc function opens the on-line version of the help manual. This is very helpful for more complex commands.

>>**doc plot**

• Use lookfor to find functions by keywords. The general form is

>>lookforFunctionName

EXPERIMENT: 1

1.1 OBJECTIVES

Maxima and minima for two variable functions using MATLAB.

For example: $f = x^4 + y^4 - 4xy + 1$

1.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

1.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

1.4 PROGRAM

```
clc;
clear;
close all;
fprintf('Finding min and max for the given function \n');

xMin = -2;
xMax = 2;
yMin = -2;
yMax = 2;
numPoints = 200;
xv = linspace(xMin, xMax, numPoints);
yv = linspace(yMin, yMax, numPoints);
[x, y] = meshgrid(xv, yv);

fprintf('Creating function.\n');
f = x.^4 + y.^4 - 4.*x.*y + 1;
fprintf('f=x^4+y^4-4xy+1\n');

maxValue = max(abs(f(:)));
minValue = min(abs(f(:)));
fprintf('The max of f = %f.\nThe min of f = %f.\n', maxValue,
minValue);
```

1.5 OUTPUT

```
Finding min and max for the given function
Creating function.
f=x^4+y^4-4xy+1
The max of f = 49.000000.
The min of f = 0.000144.
```

EXPERIMENT: 2

2.1 OBJECTIVES

Area of the region bounded by the curve using MATLAB.

For example: f = x^3 +x^2-3*x +15

2.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

2.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

2.4 PROGRAM

```
clc;
clear;
close all;
fprintf('Finding area under the given cure \n');
syms x;
f = x^3 +x^2-3*x +15;

fprintf('Given funciton is f = x^3 +x^2-3*x +15 \n');
fprintf('Given limit: 1 to 2 \n');
a=int(f,1,2);
fprintf('Area: '), disp(double(a));
```

2.5 OUTPUT

```
Finding area under the given cure
Given function is f = x^3 - 2*x +5
Given limit: 1 to 2
Area:    16.5833
```

EXPERIMENT: 3

3.1 OBJECTIVES

Solving system of differential equations using MATLAB.

i) Solve the first-order differential equation $\frac{dy}{dt} = ay$.

ii) Solve the second-order differential equation $\frac{dy^2}{dt^2} = ay$.

iii) Solve the first-order differential equation $\frac{dy}{dt} = ay$ with the initial condition $y(0) = 5$.

iv) Solve the second-order differential equation $\frac{dy^2}{dt^2} = a^2 y$ with the initial condition $y(0) = b$ and $y'(0) = 1$.

v) Solve the system of differential equations $\frac{dy}{dt} = z$ and $\frac{dz}{dt} = -y$.

3.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

3.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

3.4 PROGRAM

i) Solve the first-order differential equation $\frac{dy}{dt} = ay$.

```
syms y(t) a
eqn = diff(y,t) == a*y;
S = dsolve(eqn)
```

ii) Solve the second-order differential equation $\frac{dy^2}{dt^2} = ay$.

```
syms y(t) a
eqn = diff(y,t,2) == a*y;
ySol(t) = dsolve(eqn)
```

iii) Solve the first-order differential equation $\frac{dy}{dt} = ay$ with the initial condition $y(0) = 5$.

```
syms y(t) a
eqn = diff(y,t) == a*y;
cond = y(0) == 5;
ySol(t) = dsolve(eqn,cond)
```

iv) Solve the second-order differential equation $\frac{dy^2}{dt^2} = a^2 y$ with the initial condition $y(0) = b$ and $y'(0) = 1$.

```
syms y(t) a b
eqn = diff(y,t,2) == a^2*y;
Dy = diff(y,t);
cond = [y(0)==b, Dy(0)==1];
ySol(t) = dsolve(eqn,cond)
```

v) Solve the system of differential equations $\frac{dy}{dt} = z$ and $\frac{dz}{dt} = -y$.

```
syms y(t) z(t)
eqns = [diff(y,t) == z, diff(z,t) == -y];
S = dsolve(eqns)
```

3.5 OUTPUT

i) Solve the first-order differential equation $\frac{dy}{dt} = ay$.

```
S = C1*exp(a*t)
```

ii) Solve the second-order differential equation $\frac{dy^2}{dt^2} = ay$.

```
ySol(t) =  C1*exp(-a^(1/2)*t) + C2*exp(a^(1/2)*t)
```

iii) Solve the first-order differential equation $\frac{dy}{dt} = ay$ with the initial condition $y(0) = 5$.

```
ySol(t) = 5*exp(a*t)
```

iv) Solve the second-order differential equation $\frac{dy^2}{dt^2} = a^2 y$ with the initial condition $y(0) = b$ and $y'(0) = 1$.

```
ySol(t) =(exp(a*t)*(a*b + 1))/(2*a) + (exp(-a*t)*(a*b -
1))/(2*a)
```

v) Solve the system of differential equations $\frac{dy}{dt} = z$ and $\frac{dz}{dt} = -y$.

```
S =
  struct with fields:
    z: C2*cos(t) - C1*sin(t)
    y: C1*cos(t) + C2*sin(t)
```

EXPERIMENT: 4

4.1 OBJECTIVES

Laplace transform of the various mathematical functions using MATLAB.
i) 1/sqrt(x)
ii) exp(-a*t)

4.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

4.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

4.4 PROGRAM

i) 1/sqrt(x)

```
syms x y
f = 1/sqrt(x);
F = laplace(f)
```

ii) exp(-a*t)

```
syms a t y
```

```
f = exp(-a*t);
F = laplace(f)
```

4.5 OUTPUT

i) 1/sqrt(x)

```
F = pi^(1/2)/s^(1/2)
```

ii) exp(-a*t)

```
F = 1/(a + s)
```

EXPERIMENT: 5

5.1 OBJECTIVES

Inverse Laplace transform of the various mathematical functions using MATLAB.

i) 1/s^2

ii) 1/(s-a)^2

5.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

5.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

5.4 PROGRAM

i) 1/s^2

```
syms s
F = 1/s^2;
f = ilaplace(F)
```

ii) 1/(s-a)^2

```
syms a s
F = 1/(s-a)^2;
f = ilaplace(F)
```

5.5 OUTPUT

i) 1/s^2

```
f =  t
```

ii) 1/(s-a)^2

```
f =  t*exp(a*t)
```

EXPERIMENT: 6

6.1 OBJECTIVES

Constructing three dimensional vector field for the given vector using MATLAB.

6.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

6.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

6.4 PROGRAM

i)

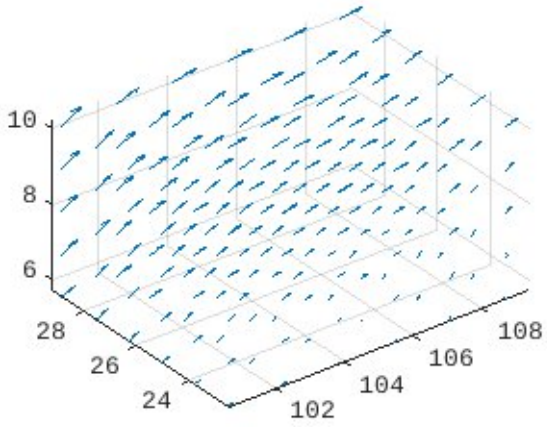
```
load wind
X = x(5:10,20:25,6:10);
Y = y(5:10,20:25,6:10);
Z = z(5:10,20:25,6:10);
U = u(5:10,20:25,6:10);
V = v(5:10,20:25,6:10);
W = w(5:10,20:25,6:10);

quiver3(X,Y,Z,U,V,W)
axis equal
```

ii)

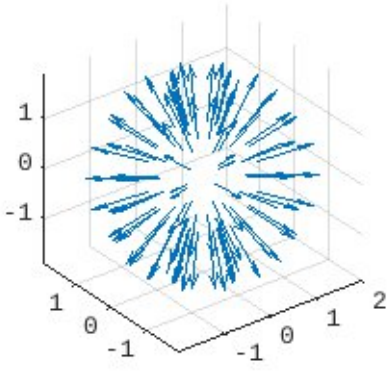
```
[X,Y,Z] = sphere(10);
[U,V,W] = surfnorm(X,Y,Z);
quiver3(X,Y,Z,U,V,W,0)
axis equal
```

6.5 OUTPUT

i)



ii)



EXPERIMENT: 7

7.1 OBJECTIVES

Visualizing sphere using MATLAB.

7.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

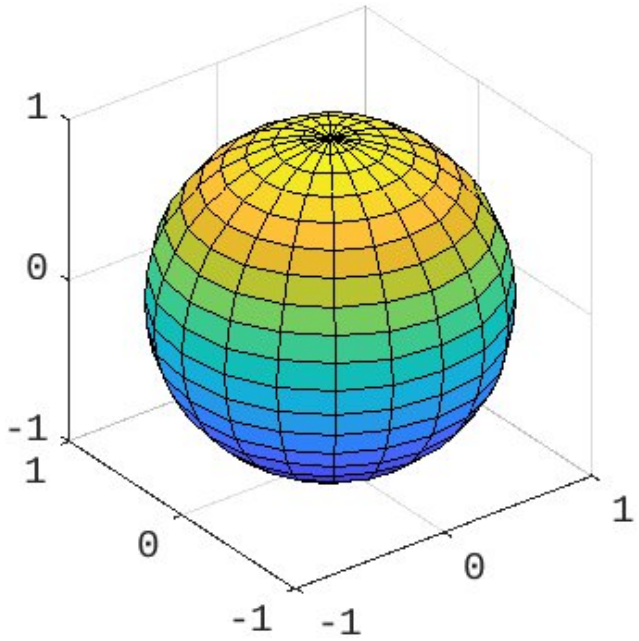
7.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

7.4 PROGRAM

```
sphere
axis equal
```

7.5 OUTPUT



EXPERIMENT: 8

8.1 OBJECTIVES

Visualizing cylinder using MATLAB.

8.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

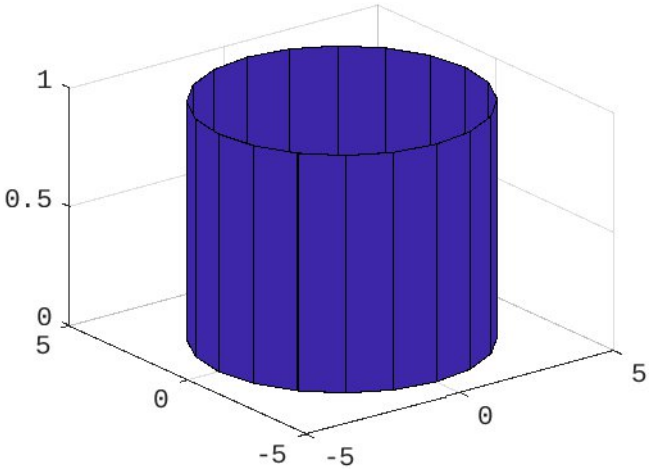
8.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

8.4 PROGRAM

```
r = 4;
[X,Y,Z] = cylinder(r);
surf(X,Y,Z)
```

8.5 OUTPUT



EXPERIMENT: 9

9.1 OBJECTIVES

Plotting 3D curves using MATLAB.

9.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

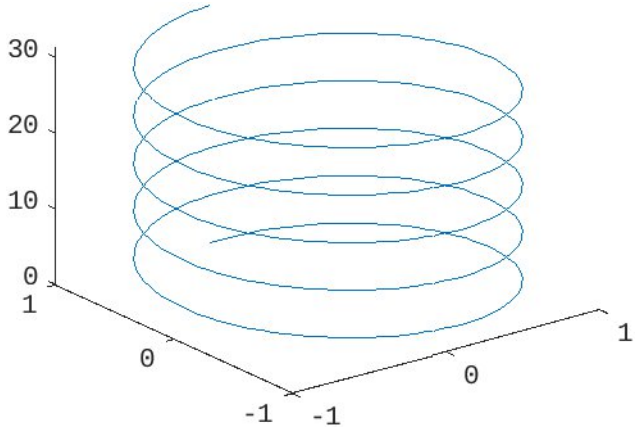
9.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

9.4 PROGRAM

```
t = 0:pi/50:10*pi;
st = sin(t);
ct = cos(t);
plot3(st,ct,t)
```

9.5 OUTPUT



EXPERIMENT: 10

10.1 OBJECTIVES

Plotting 3D surfaces using MATLAB.

10.2 SOFTWARE REQUIRED

1. MATLAB
2. Windows 7/8/10/11

10.3 PROCEDURE

1. Open MATLAB
2. Open new M-file
3. Type the program
4. Save in current directory
5. Compile and Run the program
6. For the output see command window\ Figure window

10.4 PROGRAM

```
[X,Y] = meshgrid(1:0.5:10,1:20);
Z = sin(X) + cos(Y);
surf(X,Y,Z)
```

10.5 OUTPUT