MATLAB /EC254 R20

II/IV B.TECH – I SEM-R20

LABORATORY MANUAL

PREPARED BY

MRS.E.SUNEETHA , ASSISTANT PROFESSOR

DEPARTMENT OF E.C.E

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

BAPATLA WOMEN'S ENGINEERING COLLEGE ::BAPATLA
(AFFILIATED TO ACHARYA NAGARJUNA UNIVERSITY)

BAPATLA,BAPATLA DISTRICT,ANDHRA PRADESH,INDIA, PIN:522101

INSTITUTE VISION AND MISSION

VISION

Bapatla Women's Engineering College has been established with forethought to organize the growing needs of higher technical education for WOMEN, in order to provide them a competetive edge and a unique niche in the society.

MISSION

- Our mission is to impart the quality education at par with global standards to the students from all over India and in particular those from the local and rural areas.
- We continuously try to maintain high standards so as to make them technologically competent and ethically strong individuals who shall be able to improve the quality of life and economy of our country.

VISION

To produce highly professional and socially responsible women engineers through world class quality technical education and research to meet industry and Society needs in the field of Electronics and Communication Engineering.

MISSION

M1- To inculcate quality education for women through innovative learning practices with the use of modern tools.

M2- To impart professional ethics and moral values through co-curricular and extra-curricular activities as individual and team member.

M3- Empowering graduates through industrial trainings/on-job training in collaboration with industries with-a-view to life-long learning.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

1. To develop and implementation of Electronics and Communication systems by imparting technical knowledge, professional leadership and communication skills as an individual and team in a multi-disciplinary stream.

2. To train Graduates in solving complex problems, analyzing, designing and imparting research-based knowledge by using modern scientific tools to develop least cost-effective and innovative products.

3. To produce women graduates through life-long learning, who ensure ethical and moral behavior.

PROGRAM SPECIFIC OUTCOMES (PSOs)

1. To understand and analyze basic concepts of electronics and communication in various areas like signal processing, VLSI, embedded systems, communications, digital and analog devices.

2. Ability to design and develop computing systems using concepts of mathematics, electronics engineering and other related inter disciplines to asses and resolve societal, environmental, health safety and cultural issues with professional ethics.

BAPATLA WOMEN'S ENGINEERING COLLEGE::BAPATLA

AN ISO 9001:2015 CERTIFIED INSTITUTION

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Programme Outcomes

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

- **PO2: Problem analysis:** Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- **PO3: Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- **PO4: Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- **PO5: Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools prediction and modeling to complex engineering activities with an understanding of the limitations.
- **PO6: The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- **PO7: Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and the need for sustainable development.
- **PO8: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- **PO9: Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- **PO10: Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and receive clear instructions.
- **PO11: Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in team, to manage projects and in multidisciplinary environments.
- **PO12: Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ACHARYA NAGARJUNA UNIVERSITY
NAGARJUNA NAGAR, GUNTUR

COURSE STRUCTURE

Course Details			Scheme of Instruction			Scheme of Examination			
Course	Code/ Regulation	Lab	L	T	P	Maxi Internal Marks	Maxi External Marks	Total Marks	Credits
II/IV B. Tech ECE	**EC 254/R20**	**MATLAB**	**0**	**0**	**3**	**30**	**70**	**100**	**2**

List of Programs

1. Write a MATLAB program to find the greatest of three numbers? Use Nested if, else if ladder.
2. Write a MATLAB program to read marks of a student and print sum, average and display the grade.
3. Write a MATLAB program to count the digits of a number. Use for loop
4. Write a MATLAB program to check whether a given number is perfect or not. Use do-while condition.
5. Write a MATLAB program to check whether a given number is strong or not. Use while loop.
6. Write a MATLAB program to check whether the given number is Armstrong or not. Use for loop
7. Write a MATLAB program to check whether the given number is palindrome or not. Use for loop
8. Write a MATLAB program to find Fibonacci series up to the given number. Use while loop.
9. Write a MATLAB program to print the result of the series 1+x^2/2+x^3/3+……+x^n/n.
10. Write a MATLAB program to read two matrices and find their sum, difference and product.
11. Write a MATLAB program to find the transpose of a matrix.
12. Write a MATLAB program to print upper and lower triangle of a matrix.
13. Write a MATLAB program on creating simple plots
14. Write a MATLAB program to read an image, perform different operations on image and display the resulting images

Note: A minimum of 10 programs have to be performed and recorded by the candidate to attain eligibility for university practical examination

BAPATLA WOMEN'S ENGINEERING COLLEGE::BAPATLA

AN ISO 9001:2015 CERTIFIED INSTITUTION

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Sub/Code – MATLAB /EC254 R20

COURSE OBJECTIVES

Students will try to learn:

C254.1. To Understand the basic knowledge of Matlab

C254.2. To analyze both theory and practical execution and improving their programming skills in technical computing environment

C254.3. To demonstrate the graphical features in MATLAB for various applications

C254.1. To Understand the Basics of Matlab

BAPATLA WOMEN'S ENGINEERING COLLEGE::BAPATLA

AN ISO 9001:2015 CERTIFIED INSTITUTION

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Sub/Code – MATLAB /EC214 R20

COURSE OUTCOMES

After completion of the subject/course, Students will be able to

CO No.	Course Outcome Statement	Bloom's Taxonomy Level
C254.1	Understand the basics of MATLAB	Understand
C254.2	Analyze theory and practical execution simultaneously for improving the programming skills	Apply
C254.3	Learn graphic features of MATLAB and to use this feature effectively in the various applications.	Create
C254.4	Tabulate results and Analyze	Evaluate

BAPATLA WOMEN'S ENGINEERING COLLEGE::BAPATLA

AN ISO 9001:2015 CERTIFIED INSTITUTION

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Sub/Code – MATLAB /EC 254 R20

COURSE OBJECTIVES MAPPING WITH PEOs

C.OBJ	PEO1	PEO2	PEO3
C254.1	3	2	2
C254. 2	3	-	2
C254.3	3	2	2
C254.4	3	2	2

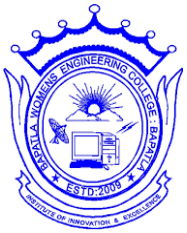
Mapping of COs with POs and PSOs

Course out come	Program Specific Outcomes (PSOs)		Program Outcomes(Pos)											
	PSO 1	PSO 2	PO1	PO2	PO3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
C254.1	*1*	*2*	*3*	*1*	*1*	*3*	*3*	-	-	-	*2*	*3*	*3*	-
C254.2	*2*	*2*	*2*	*2*	*3*	*2*	*2*	-	-	-	-	-	-	-
C254.3	*2*	*2*	*1*	*2*	*3*	-	*2*	-	-	-	-	-	-	-
C254.4	*2*	*1*	*3*	*2*	*1*	*3*	*2*	-	-	*2*	*3*	*3*	*2*	*3*
AVG	*1.75*	*1.75*	*2.25*	*1.75*	*2.0*	*2.0*	*2.25*	-	-	*.5*	*1.25*	*1.5*	*1.25*	*.75*

Correlation Levels:---1: Low 2: Medium 3: Strong, '-' No correlation

List of Programs

S.No	Program Name	Page. No
1	Write a MATLAB program to find the greatest of three numbers? Use Nested if, else if ladder.	43-44
2	Write a MATLAB program to count the digits of a number. Use for loop	45-46
3	Write a MATLAB program to check whether a given number is perfect or not. Use do-while condition.	47-48
4	Write a MATLAB program to check whether a given number is strong or not. Use while loop.	49-50
5	Write a MATLAB program to check whether the given number is Armstrong or not. Use for loop	51-52
6	Write a MATLAB program to check whether the given number is palindrome or not. Use for loop	53-54
7	Write a MATLAB program to find Fibonacci series up to the given number. Use while loop.	55-56
8	Write a MATLAB program to print the result of the series 1+x^2/2+x^3/3+……+x^n/n.	57
9	Write a MATLAB program to read two matrices and find their sum, difference and product.	58-59
10	Write a MATLAB program to find the transpose of a matrix	60-61
11	Write a MATLAB program to print upper and lower triangle of a matrix.	62-63
12	Write a MATLAB program on creating simple plots	64-68
13	Write a MATLAB program to read an image, perform different operations on image and display the resulting images	69-72



FUNDAMENTALS OF MATLAB

1. MATLAB FUNDAMENTALS

INTRODUCTION

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. Typical uses include the following:

- Math and computation
- Algorithm development
- Data acquisition
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows formulating solutions to many technical computing problems, especially those involving matrix representations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or FORTRAN. 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. Today, MATLAB engines incorporate the LAPACK (Linear Algebra Package) and BLAS (Basic Linear Algebra Subprograms) libraries, constituting the state of the art in software for matrix computation.

MATLAB is a powerful computing system for handling the calculations involved in scientific and engineering problems. The name MATLAB stands for MATrix LABoratory, because the system was designed to make matrix computations particularly easy. If you don‘t know what a matrix is, don‘t worry—we will look at them in detail later. This book assumes that you have never used a computer before to do the sort of scientific calculations that MATLAB handles. You will, however, need to be able to find your way around a computer keyboard and the operating system running on your computer (e.g. Windows or UNIX). The only other computer-related skill you will need is some very basic text editing.

One of the many things you will like about MATLAB (and which distinguishes it from many other computer programming systems, such as C++ and Java) is that you can use it interactively. This means you type some commands at the special MATLAB prompt, and get the answers immediately. The problems solved in this way can be very simple, like finding a square root, or they can be much more complicated, like finding the solution to a system of differential equations. For many technical problems you have to enter only one or two commands, and you get the answers at once. MATLAB does most of the work for you. There are two essential requirements for successful MATLAB programming.

If you give MATLAB a garbage instruction, you will get a garbage result. With experience, you will see that with MATLAB you can design, develop and implement computational and graphical tools to do relatively complex science and engineering problems. Of course, some of the necessary experience required to tap the full potential of MATLAB is your continuing education both before and after graduation. With MATLAB you will be able to adjust the look, modify the way you interact with MATLAB, and develop a toolbox of your own that helps you solve problems that are of interest to you. In other words, you can, with significant experience, customize your MATLAB working environment. As you learn the basics of MATLAB and, for that matter, any other computer tool, remember that computer applications do nothing randomly. Hence, as you use MATLAB, observe and study all responses from the command-line operations that you implement. You need to learn what this tool does and what it doesn‘t do. To begin an investigation into the capabilities of MATLAB, we will do relatively simple problems. Initially, we must do problems to which we know the answers because we are evaluating the tool and its capabilities. This is the first step in the process of learning how to use any tool. As you learn about MATLAB, you are also going to learn about computer programming.

You need to do this for two reasons: (1) To create your own computational tools. (2) To appreciate the difficulties involved in the design of efficient, robust and accurate computational and graphical tools (i.e. computer programs). In the rest of this chapter we will look at some simple examples for you to try out. Don‘t bother about understanding exactly what is happening. The understanding will come in later chapters when we look at the details. It is very important for you to exercise the MATLAB tool to learn by hands-on experience how it works. Once you have examined a few of the basic rules in this chapter, you will be prepared to master many of the basic rules presented in the next chapter. This will help you go on to solve more interesting and substantial problems. Finally, in the last section of this chapter you will take a quick tour of the MATLAB desktop.

In university environments, MATLAB is the standard computational tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the computational tool of choice for research, development, and analysis. MATLAB is complemented by a family of application specific solutions called toolboxes. The Image Processing Toolbox is a collection of MATLAB functions (called M-functions or M-files) that extend the capability of the MATLAB environment for the solution of digital image processing problems. Other toolboxes that sometimes are used to complement IPT are the Signal Processing, Neural Network, Fuzzy Logic, and Wavelet Toolboxes.

MATLAB DESKTOP

In order to use MATLAB it must either be installed on your computer, or you must have access to a network where it is available. The MATLAB desktop is the main MATLAB application window. As Fig. 1.1 shows, the desktop contains five sub windows: the Command Window, the Workspace Browser, the Current Directory Window, the Command History Window, and one or more Figure Windows, which are shown only when the user displays a graphics.

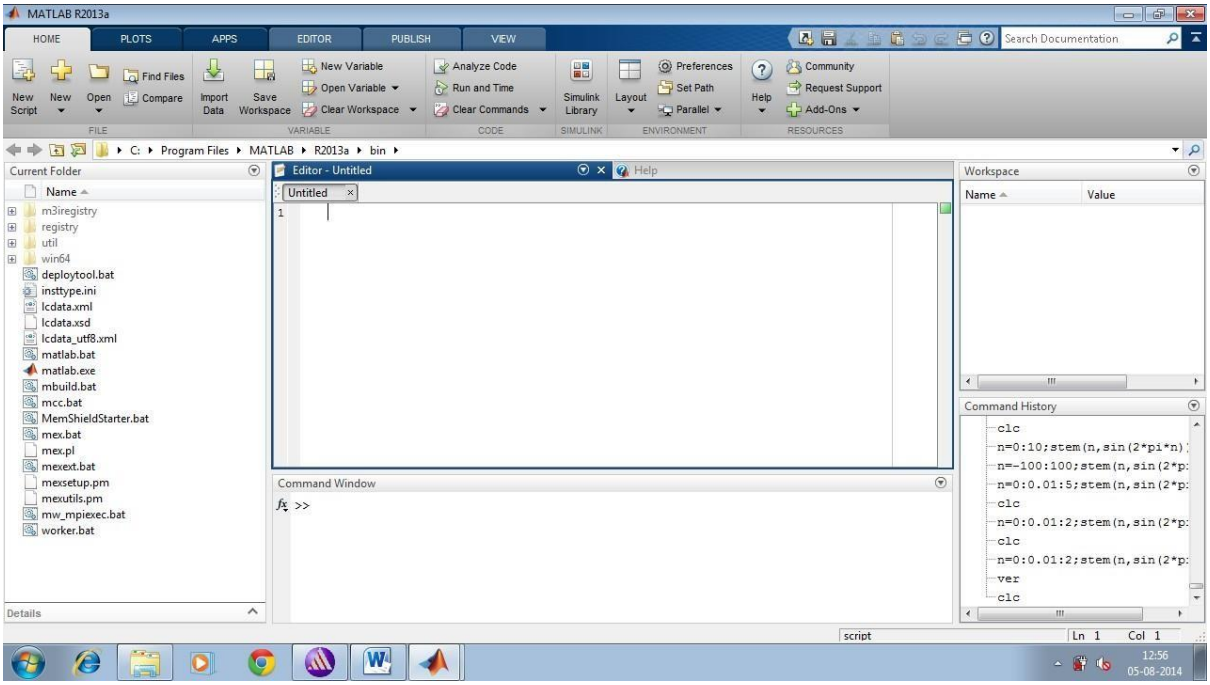


Figure 1.1. MATLAB Version: 8.1.0.604 (R2013a)

To start MATLAB from Windows, double-click the MATLAB icon on your Windows desktop. To start it from a UNIX platform, type matlab at the operating system prompt. When MATLAB starts, the MATLAB desktop opens as shown in Figure 1.1. The window in the desktop that concerns us for this chapter is the Command Window, where the special prompt appears. This prompt means that MATLAB is waiting for a command. You can quit MATLAB at any time with one of the following:

1) Select Exit MATLAB from the desktop File menu.

2) Enter quit or exit at the Command Window prompt.

COMMAND WINDOW

The area in which all commands are entered for using MATLAB on as interactive basis. The Command Window is where the user types MATLAB commands and expressions at the prompt (>>) and where the outputs of those commands are displayed. MATLAB defines the workspace as the set of variables that the user creates in a work session. The Workspace Browser shows these variables and some information about them. Double-clicking on a variable in the Workspace Browser launches the Array Editor, which can be used to obtain information and in some instances edit certain properties of the variable. Command window executes the code line by line basis.

Do not click on the close box in the top right corner of the MATLAB desktop. This does not allow MATLAB to terminate properly and, on rare occasions, may cause problems with your computer operating software (the new author corrupted a graphics utility when doing color graphics by clicking the red x!). Once you have started MATLAB, try the following exercises in the Command Window. If necessary, make the Command Window active by clicking anywhere inside its border. Since we have experience doing arithmetic, we want to examine if MATLAB does this correctly. When applying computer tools this is a required step to gain confidence in the tool and in our ability to use the tool.

(a) Type 2+3 after the >> prompt, followed by Enter, i.e. press the Enter

key, as indicated by <Enter>, below:

>> 2+3 <Enter>

Commands are only carried out when you press Enter. The answer in this case is, of course,. The following Figure 1.2. With figure window in the desktop that concerns us for this chapter in the Command Window,

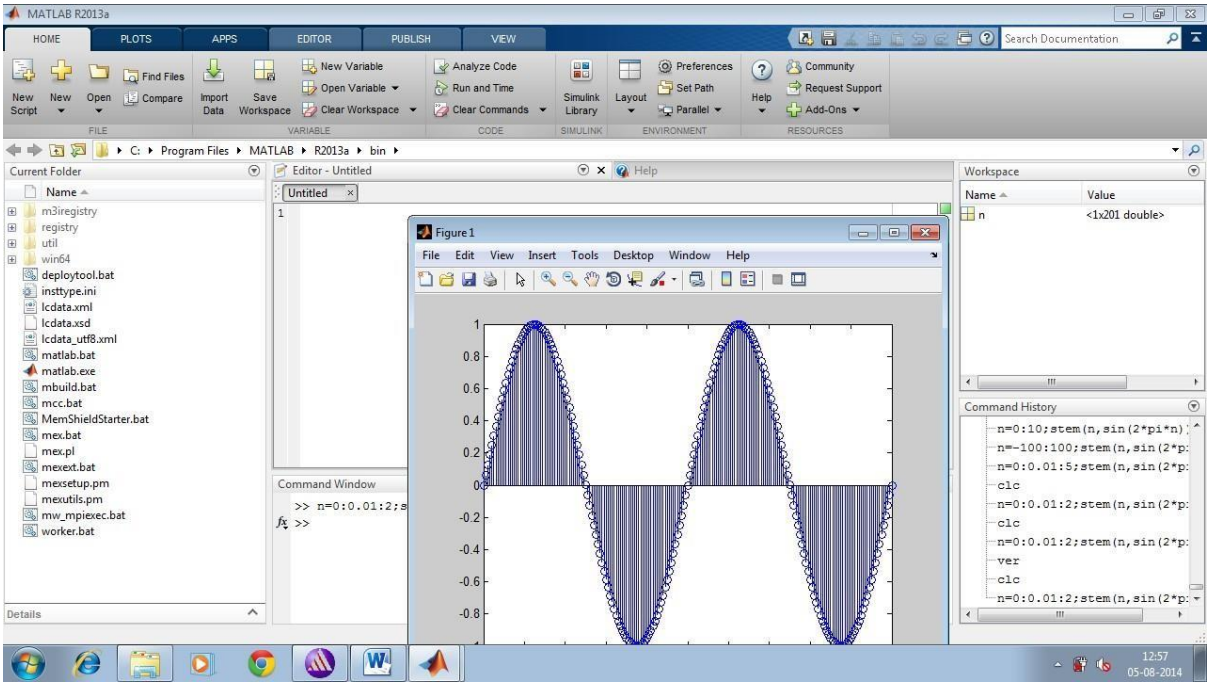


Figure 1.2.The window in the desktop that concerns all Sub Windows

CURRENT DIRECTORY

Provides a list of the MATLAB programs available in the given directory. The Current Directory tab above the Workspace tab shows the contents of the current directory, whose path is shown in the Current Directory Window. For example, in the Windows operating system the path might be as follows:\MATLAB\Work, indicating that directory ―Work‖ is a subdirectory of the main directory ―MATLAB,‖ which is installed in drive C. Clicking on the arrow in the Current Directory Window shows a list of recently used paths. Clicking on the button to the right of the window allows the user to change the current directory MATLAB uses a search path to find M-files and other MATLAB-related files, which are organized in directories in the computer file system. Any file run in MATLAB must reside in the current directory or in a directory that is on the search path. By default, the files supplied with MATLAB and Math Works toolboxes are included in the search path. The easiest way to see which directories are on the search path, or to add or modify a search path, is to select Set Path from the File menu on the desktop, and then use the Set Path dialog box. It is good practice to add any commonly used directories to the search path to avoid repeatedly having the change the current directory.

COMMAND HISTORY

The Command History Window contains a record of the commands a user has entered in the Command Window, including both current and previous MATLAB sessions. Previously entered MATLAB commands can be selected and re-executed from the Command History Window by right-clicking on command or sequence of commands. This action launches a menu from which to select various options in addition to executing the commands. This is a useful feature when experimenting with various commands in a work session.

MATLAB EDITOR

The MATLAB editor is both a text editor specialized for creating M-files and a graphical MATLAB debugger. The editor can appear in a window by itself, or it can be a subwindow in the desktop. M-files are denoted by the extension .m, as in sinwavegen.m. The MATLAB editor window has numerous pull-down menus for tasks such as saving, viewing, and debugging files. Because it performs some simple checks and also uses color to differentiate between various elements of code, this text editor is recommended as the tool of choice for writing and editing M-functions. To open the editor, type edit at the prompt in the Command Window. Similarly, typing edit filename at the prompt opens the M-file filename.min an editor window, ready for editing. As noted earlier, the file must be in the current directory, or in a directory in the search path.

GETTING HELP

One of the most valuable resources available in the MATLAB program is **help**. Help file for assistance when required. The principal way to get help is to use the MATLAB Help Browser, opened as a separate window either by clicking on the question mark symbol **(?)** on the desktop toolbar, or by typing Help browser at the prompt in the Command Window. The Help Browser is a Web browser integrated into the MATLAB desktop that displays Hypertext Markup Language (HTML) documents. The Help Browser consists of two panes, the help navigator pane, used to find information, and the display pane, used to view the information. Self-explanatory tabs on the navigator pane are used to perform a search. For example, help on a specific function is obtained by selecting the Search tab, selecting Function Name as the Search Type, and then typing in the function name in the Search for field. It is good practice to open the Help Browser at the beginning of a MATLAB session to have helped readily available during code development or other MATLAB task. Another way to obtain help for a specific function is by typing doc Followed by the function name at the command prompt. For example, typing doc format displays documentation for the function called format in the display pane of the Help Browser. This command opens the browser if it is not already open‘s-functions have two types of information that can be displayed by the user. The first is called the H1 line, which contains the function name and alone-line description. The second is a block of explanation called the Help text block (these are discussed in detail in Section 2.10.1). Typing help at the prompt followed by a function name displays both the H1 line and the Help text for that function in the Command Window. Occasionally, this information can be more up to date than the information in the Help browser because it is extracted directly from the documentation of the M-function in question. Typing look for followed by a keyword displays all the H1 lines that contain that keyword. This function is useful when looking for a particular topic without knowing the names of applicable functions. For example, typing look for edge at the prompt displays all the H1 lines containing that keyword. Because theH1 line contains the function name, it then becomes possible to look at specific functions using the other help methods. Typing look for edge all at the prompt displays the H1 line of all functions that contain the word edge in either the H1 line or the Help text block. Words that contain the characters edge also are detected. For example, the H1 line of a function containing the word pledge in the H1 line or Help text would also be displayed. It is

common MATLAB terminology to use the term help page when referring to the information about an M-function displayed by any of the preceding approaches, excluding look for. It is highly recommended that the reader become familiar with all these methods for obtaining information because in the following chapters we often give only representative syntax forms for MATLAB and IPT functions. This is necessary either because of space limitations or to avoid deviating from a particular discussion more than is absolutely necessary. In these cases we simply introduce the syntax required to execute the function in the form required at that point. By being comfortable with online search methods, the reader can then explore a function of interest in more detail with little effort. Finally, the Math Works' Web site mentioned in Section contains a large database of help material, contributed functions, and other resources that should be utilized when the online documentation contains insufficient information about a desired topic.

SAVING AND RETRIEVING A WORK SESSION

There are several ways to save and load an entire work session (the contents of the Workspace Browser) or selected workspace variables in MATLAB. The simplest is as follows. To save the entire workspace, simply right-click on any blank space in the Workspace Browser window and select Save Workspace As from the menu that appears. This opens a directory window that allows naming the file and selecting any folder in the system in which to save it. Then simply click Save. To save a selected variable from the Workspace, select the variable with a left click and then right-click on the highlighted area. Then select Save Selections from the menu that appears. This again opens a window from which a folder can be selected to save the variable. To select multiple variables, use shift click or control-click in the familiar manner, and then use the procedure just described for a single variable. All files are saved in double-precision, binary format with the extension .mat. These saved files commonly are referred to as MAT-files. For example, a session named, say, mywork_2003_02_10, would appear as the MAT-file mywork_2003_02_10.mat when saved. Similarly, a saved image called final image (which is a single variable in the workspace) will appear when saved as final_image.mat. To load saved workspaces and/or variables, left-click on the folder icon on the toolbar of the Workspace Browser window. This causes a window to open from which a folder containing the MAT-files of interest can be selected. Double-clicking on a selected MAT-file or selecting Open causes the contents of the file to be restored in the Workspace Browser window.

WORKSPACE BROWSER WINDOW

The list of variables used in current working session. It is possible to achieve the same results described in the preceding paragraphs by typing save and load at the prompt, with the appropriate file names and path information. This approach is not as convenient, but it is used when formats other than those available in the menu method are required. The following commands are useful for program initialization, Debugging and Workspace manipulation.

1. Clc - clear command window
2. Clear - clear workspace contents
3. Whos - display workspace contents
4. What - display files in current folder
5. Pause - pause program execution

PROGRAMMING WITH M- FILES

M- FILE: Mat file is a computer program that can be written saved and activated as many times as desired. Script file called an m- file which is only a text file that contains which is only a text file that contains each line of the file as through you typed then at the command prompt. Command window programming based on an interactive basis. This pattern will continue throughout the text since most commands will be readily observed Command window is best for fast results and when procedure may not be needed repeatedly few lines of codes are involved. Computational procedure may need to be repeated for more than one set of input data or so and code consists of relatively long list of commands. MATLAB based computer program for performing a particular task on-file script is called M-file.

1. Creating
2. Saving
3. Setting
4. Running Program

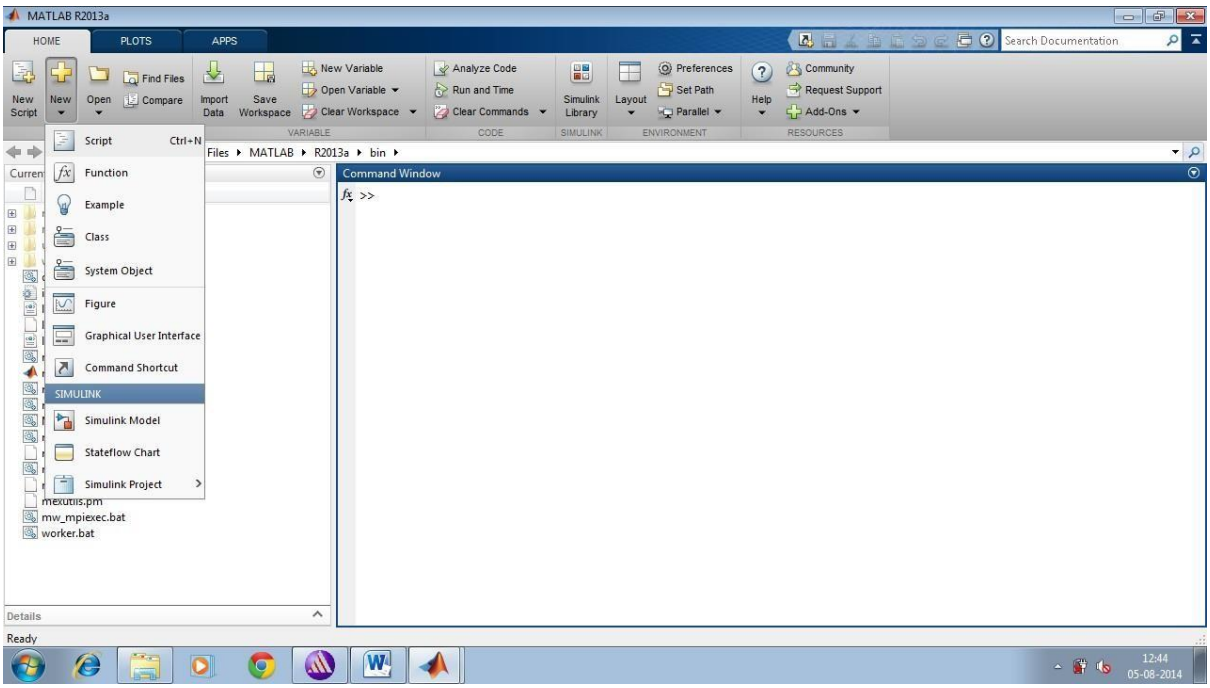


Figure 1.3.The window in the desktop that concerns with programming

TOOL BOXES

It is also features a family of application specific solutions called TOOL BOXES. The following tool boxes are available in MATLAB Version: 8.1.0.604 (R2013a)

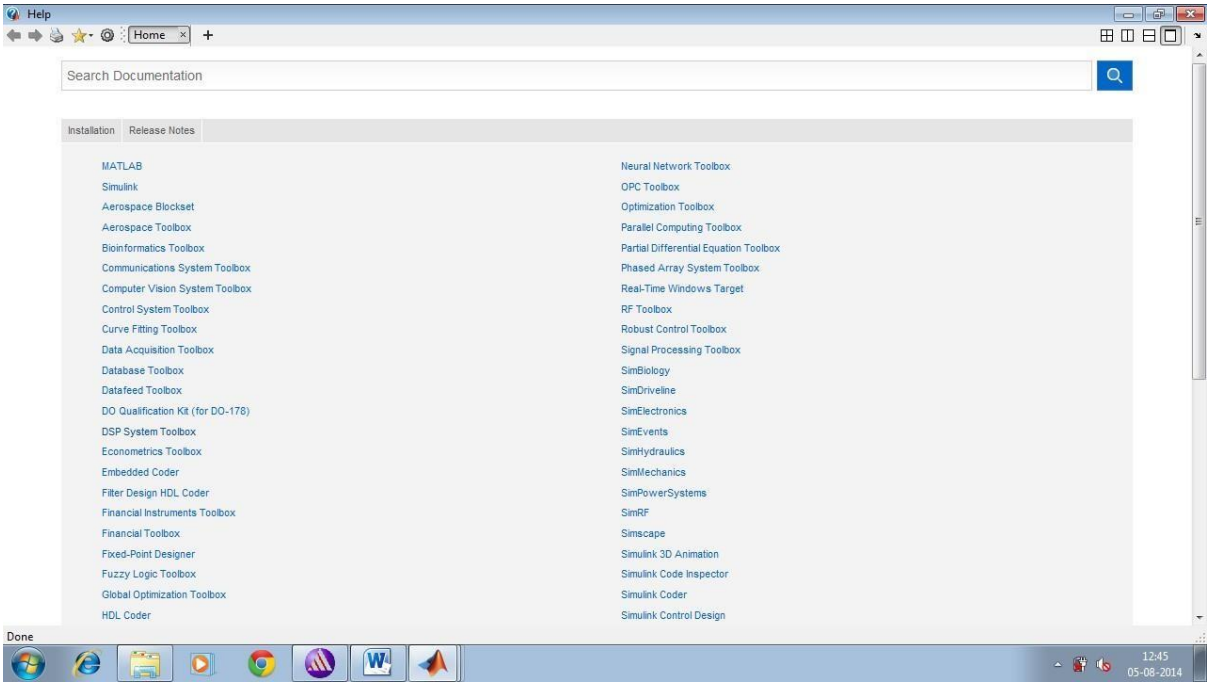


Figure 1.4.The window in the desktop that concerns all tool boxes

The following tool box in MATLAB Version: 8.1.0.604 (R2013a) is Digital Signal Processing. This tool box having available built in functions are shown below.,

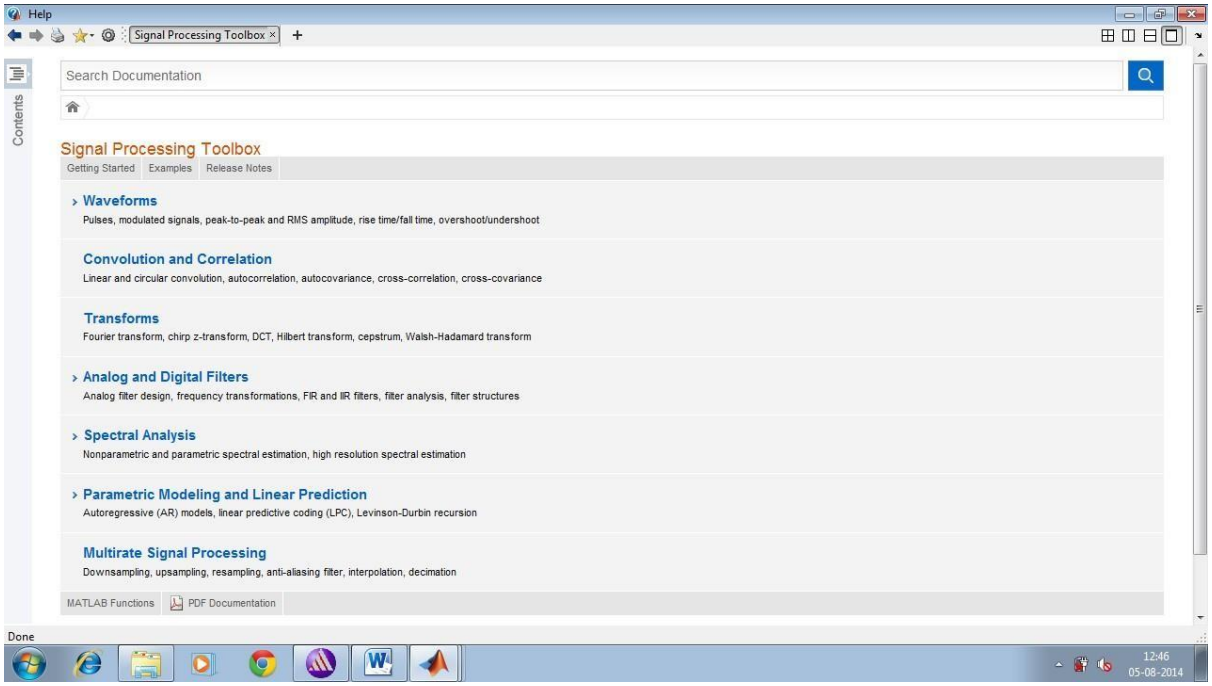


Figure 1.5.The window in the desktop that concerns all Sub Windows

2. MATLAB DATATYPES

All variables and constants in a MATLAB program must have an associated ―type‖. The type specifies the set or ranges of values that the object can take on.

double	Double-precision, floating-point numbers in the approximate range $\pm 10^{308}$(8 bytes per element).
Single	Single-precision, floating-point numbers with values in the approximate range $\pm 10^{38}$ (4 bytes per element).
Uint8	Unsigned 8-bit integers in the range [0,255] (1 byte per element).
Uint16	Unsigned 16-bit integers in the range [0, 65535] (2 bytes per element).
Uint32	Unsigned 32-bit integers in the range [0, 4294967295] (4 bytes per element).
int8	signed 8-bit integers in the range [-128,127] (1 byte per element).
Int16	signed 16-bit integers in the range [-32768, 32767] (2 bytes per element).
Int32	signed 32-bit integers in the range [-2147483648, 2147483647] (4 byte per element).
Char	characters (2 bytes per element).
Logical	values are 0 or 1 (1 byte per element).

MATLAB has 10 fundamental data types or classes. They are given as

Note:

*The default numeric data type is **double precision**.

*All MATLAB computations are done in **double precisions**.

*However before mathematical operations can be performed on such as int8, vint8, single, etc and the data types must be converted to double precision using double function.

3. MATLAB OPERATORS

AIRTHMETIC OPERATORS:

The evaluation of expressions is achieved by means of arithmetic operators. The basic arithmetic operators are given below. Operators operate on operands a and b.

Operator	Operation	Expression
+	Addition	a+b
-	Subtraction	a-b
*	Multiplication	a*b
/	Right division	a/b
\	Left division	a\b
^	power	a^b

Precedence of Operators:

MATLAB has strict rules about several operations may be combined in one expression are performed. In such cases isprecedence of rules. The precedence rules for the operations are given below

Precedence	Operator
1	Parentheses (round brackets)
2	Power, left/right
3	Division
4	Multiplication
5	Addition and Subtraction

Note:

*Parentheses (round brackets) have the highest precedence.

*The difference between round and square brackets is round brackets are used to alter the precedence of operators and to denote subscripts, while square brackets are used to create vectors.

Example:

The following expression a/b *c is evaluated as (a/b) * c but not as a / (b*c).

1+ (2*3)

1 + (2/4)

2 ^ (1+2) / 3

DOT OPERATORS

Arithmetic operations performed on arrays are operats element by element basis.

Operator	Description
.*	Multiplication
. /	Right Division
**. **	Left Division
. ^	Array Power

Example:

Suppose two arrays are given as x= [1 2 3] and y=[4 9 3] Then expression x.*y results in following way

[x(1)*y(1) x(2)*y(2) x(3)*y(3)] = [4 18 9]

Note:

*When array operations are applied to two vectors, both vectors must be the same size.

RELATIONAL OPERATORS

MATLAB allows us to use as arithmetic expression for condition. Condition is usually a logical expression i.e an expression containing a relational operator and which is either true or false. The relational operators are shown in below.

Relational operator	**Meaning**
<	Less than
<=	Less than or Equal
= =	Equal
~=	Not Equal
>	Greater than
>=	Greater than or Equal

Note:*If condition is true statement is executed but if condition is false nothing will happens. Condition may be a vector or matrix in which case it is use true only if all of its elements are non zero. Example:

x > = 0 ;

a ~ = 0;

b ^2 < 4 * a* c ;

if x = = 0 disp („ x equals to zero „) ;end

TRANSPOSE OPERATOR

The transpose operators are used to transpose the array of vector. The transpose operator is apostrophe („) is used to change a row vector into column vector.

Example: suppose X= [1 2 3] then transpose of X is given as

$$X„ = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

Note :

*This transpose operator has the highest priority.

COLON OPERATORS

Colon operator (**:**) is used to generate vectors / arrays with elements increasing or decreasing by regular increments or decrements. This colon operator has the lowest priority.

Example :

If **n = 0:5** then n will be incremented by 1

n = [0 1 2 3 4 5]

LOGICAL OPERATORS

Boolean expressions combine‘s individual boolean terms using MATLAB‘s built in functions. Logical operators such as AND, OR, NOT are used for logical (decision) operations. Logical operators are

Logical Operator	Meaning
&	AND
|	OR
~	NOT

Note :

* In logical operations we have to assign either logical _0 _or logical _1 _to an array.

Example: If A = [1 0 1 0] and B = [0 1 1 1] then

A & B = [1 & 0 0 & 1 1 & 1 0 & 1] =>[0 0 1 0]

A | B = [1 | 0 0 | 1 1 | 1 0 | 1]=> [1 1 1 1]

~ A = [0 1 0 1];

~ B = [1 0 0 0];

4. EXPRESSIONS & STATEMENTS

An **expression** is a formula consisting of variables, numbers, operators and function names.

Example:

2*pi gives the response as **ans** 6.2832

Hence MATLAB uses the function **ans** (which stands for answer) to return the last expression to be evaluated but not assigned to a variable.

MATLAB **statements** are frequently designed with assignment operator (=) is given below

Variable = Expression;

Example:

S = a * x^2 + b*x +c ;

Hence the value of the expression on the right side is assigned to the variable s on the left.

Note:

* The object on the left hand side of the assignment must be a variable name.

5. MATLAB OPERATIONS

MATHEMATICAL OPERATIONS:

MATLAB is highly vector zed programming language which means that whenever possible operation apply to entire arrays on an element by element basis.

Expression	Description
y = x+a	Scalar Addition
y = x-a	Scalar Subtraction
y = x*a	Scalar Multiplication
y = x/a	Scalar Division
y = x ^ a	Scalar Power

Where y = x +a;

x= array = x(i, j)

a= scalar

y = array = y(i,j)

Example:

Suppose x= [1 2 3] and a= 5; then y = [1+5 2+5 3+5]; = > [6 7 8];

ARRAY OPERATIONS:

MATRIX is a two dimension set of numbers arranged in rows and columns. Another name of matrix is array. Suppose two arrays x,y and the result z is also an array.

Expression	Description
Z= x+y	Array Addition
Z= x-y	Array Subtraction
Z= x.*y	Array Multiplication
Z=x./y	Array Division
Z=x.^y	Array ppower

Example:

Z = x+ y means Z (i,j) = x (i,j) + y(i,j)

If x= [1 2 3]; y= [1 2 3]; then

Z= x + y = [1+1 2+2 3+3] = [2 4 6];

Z= x – y = [1-1 2-2 3-3] = [0 0 0] ;

Z= x .* y = [1*1 2*2 3*3] = [1 4 9];

Z= x ./ y = [1/1 2/2 3/3] = [1 1 1];

Z= x .^ y = [1^1 2^2 3^3] = [1 4 2];

MATRIX OPERATIONS:

When performing linear algebra of two matrices are compatible for multiplication if the number of columns of first matrix equals the number of rows in second matrix.

C = A * B;

Let a, b are two column vectors with _m‘ elements where _m‘ is the number of rows of a.

C = a‟ * b;

C = a * b‟;

Precedence	Operators
1	( ) i.e Parantheses
2	^, .^, _, .‘ i.e Power and Transpose
3	+ Unary plus - Unary minus ~ Not
4	/ Right division \ Left division * Multiplication **.*** Array Multiplication **./** Array Right division **.** Array Left division
5	+ Addition - Subtraction
6	**:** Colon
7	> Greater than < Less than >= Greater than or Equal < = Less than or Equal = = Equal ~ = Not Equal
8	& AND | OR

OPERATOR PRECEDENCE FOR ALL OPERATORS

6. VARIABLES & VARIABLE INITIALIZATION

INTRODUCTION

The basic variable is a matrix or an array. MATLAB variables include arrays (scalars, vectors, matrices) which can be real or complex strings. Hence MATLAB operates on all of it's elements. A variable name consists only of letters, digits and underscores and must start with a letter. Only 31 characters are significant.MATLAB is case sensitive by default. All variables created during a session remain in the work space until it is removed with clear. The command **"who "**lists the variables in workspace. **"whos"** also gives their sizes.

- MATLAB refers all variables as arrays, whether they are scalars (single valued arrays) , vectors (1-D arrays) or matrices (2-D arrays).
- String is a collection of characters. String contents are enclosed in _Single Quotes _while the elements of an array are enclosed in _Square Brackets'.
- For scalar Square Brackets are optional.
- Vectors and matrices are entered in Square Brackets. Elements are seperated by spaces or commas. Rows are separated by semicolons.
- Complex numbers may be represented using special variables i and j which stand for the unit imaginary number i.e square root of -1 .
- Like high level programming languages, MATLAB generally ignores spaces and line brackets and these may be provided as desired for readability.
- MATLAB comments began with **% symbol.**

Examples:

1. **Q**= pi = 3.141592
2. **S** = _ This is a String'
3. **A** = 4.39
4. j = sqrt (-1);
5. b= 2.1 + j * 3.7

COLUMN VECTOR: The column vector is an (N×1) matrix or array with vertical arrangement of numbers

$$X = [x_{i1}] \quad i = 1,2,\text{- - - - } N = \begin{matrix} X_{11} \\ X_{21} \\ = \\ = \\ X_{n1} \end{matrix}$$

ROW VECTOR

The row vector is an (1×N) matrix or array with horizontal arrangement of numbers.

$Y = [\, y_{1j} \,]_{\,j = 1 - - - - M} = [\, y_{11} \;\; y_{12} \;\; y_{13} \text{---------} y_{1M}]$

GENERAL MATRIX

General matrix with order N×M is given as

$A = [a_{ij}]_{\,i= 1,2 - - - - N,;\; j = 1,2, \text{------} M;}$

$a_{11} \quad a_{12} \text{----------} a_{1M}$

$a_{21} \text{---} a_{22} \; a_{2M}$

$a_{N1} \quad a_{N2} - - - - a_{NM}$ N× M

Examples:

1).a = [3] ; Scalar

2).x = [1 2 3] ; Row vector

3).y = [1 ; 2 ; 3] ; Column vector

$$y = \begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

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

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

❖ An element of a vector is referred to by a subscript in round brackets.

X = [1 2 3] ;

C = X (3) ; gives the element 3 of array X.

A(1,2) ; gives the element 2 of first row.

Q = Z‘ ; gives the transposed version of Z.

Note :

Semicolon operator is used in two ways.

1 . When used within square brackets it is a delimiter that separates rows of an array.

2 . When used at the end of an assignment statement it suppresses the presenting the value of the variable and no semicolon error occurred.

ELEMENTARY MATRICES

MATLAB provides many useful functions to create matrices. These include

1. **Zeros (M, N) :** This generates M × N matrix with all elements are zeros.

   Ex : A = zeros (1, 5)
   A = [0 0 0 0 0]
2. **Ones (M, N) :** This function generates M × N matrix with all elements are ones.

   Ex : B = ones (1, 4) ; B = [1 1 1 1];

3. **eye (N) :** This function generates N× N identity matrix.

   Ex : C = eye (3) = $\begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$

4. **rand (M , N) :** This function generates M× N matrix with uniformly distributed \. . \ . pseudo random numbers in (0, 1)
   Ex : D = rand (1 , 3) = [0.0150 0.3061 0.9882]

5. **Rands (M, N) :** This function generates M × N matrix with normally distributed pseudo random numbers.

Ex : E = rands (1, 3) = [-0.9838 0.5294 0.0732]

VARIABLE INITIALIZATION

MATLAB variables can be initialized using the colon operators and built in functions. The colon operator is powerful operator that is used in 2 ways.

1. When used with two or three operands it indicates a range of values starting with the first operating and ending with last operand.
2. When used for all the values of the subscript.

MATLAB built in functions for Vector initializations:

1. **X = m : n** ----------> Which generates the vector of elements incremented from **m** to **n** with incrimination value of 1.

   X = m: n gives X = [m m+1 m+2 ----------n]

2. **X = m: i: n** ------ > Which generates the vector of elements incremented from **m** to **n** . With incrementation value of i.

   X = **m : i : n** = [m m+i m+2i ----------n]

3. **C = X [p : q]** -----> Which generates the elements **p** through **q** of x .

4. **Z = linspace (a, b, n)** > Which generates n values linearly spaced over (a,b).

   Ex : Z = linspace (1 , 2, 5) = [1 1.2500 1.5000 1.7500 2]

BUILT IN FUNCTIONS FOR MATRIX MANIPULATIONS

1. **Length (X) :** Which generates the total number of elemnts in vector X.

   X = [1 2 3 -1 0];

   y = length (x) ; gives 5 as the answer.

2. **Reshape (A) :** This function is used to change the size of matrix. Size gives the size of matrix or array.

3. **fliplr:** Which is used to change the matrix elements from left to right direction . or right to left direction.

4.

X = [1 2 3 4 5]

Z= fliplr (X) = [5 4 3 2 1]

5. **flipud :** Which is used to change the matrix elements from top to bottom direction or bottom to top direction.

$$X = \begin{matrix}1\\2\\3\end{matrix} \; ; \; Z = \text{flipud}(X) = \begin{matrix}1\\2\\3\end{matrix}$$

6. **C = A (: , k) :** Which gives the column k of matrix A.

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

$$C = A(:,2) = \begin{matrix}2\\5\\8\end{matrix}$$

7. **C = A (k, :):** Which gives the row k of matrix A.

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

C = A(2 , :) = [4 5 6]

8. **Tril :** Which gives the lower triangular of the matrix.

$$X = \begin{matrix}1&2&3\\4&5&6\\7&8&9\end{matrix} \; ; \quad \text{tril}(X) = \begin{matrix}1&0&0\\4&5&0\\7&8&9\end{matrix}$$

.

9. **Triu :** Which gives the upper triangular of the matrix.

$$X = \begin{matrix}1&2&3\\4&5&6\\7&8&9\end{matrix} \; ; \quad \text{tril}(X) = \begin{matrix}1&2&3\\0&5&6\\0&0&9\end{matrix}$$

MATLAB supports the following special variables and constants .

Name	Meaning
ans	Most recent answer.
eps	Accuracy of floating-point precision.
i,j	The imaginary unit √-1.
Inf	Infinity.
NaN	Undefined numerical result (not a number).
pi	The number π

7. BRANCHING & LOOPS

BRANCHING STATEMENTS

In some applications MATLAB has to make decision which is fundamental to all. These statements uses relational operator.

If statement:

The if constant which is fundamental to all computing languages is the basis of such decision making.

Syntax for if statement:

If expression-1/condition

Statement;

End;

Note:

- In the above syntax condition|expression-1 is alogical expression be an expreesion containing a relational operator, and which is either true/false
- MATLAB allows us to use as arithmetic expression for condition
- If the expression evaluates to 0.it is regarding as false; any other value is true
- If the condition is true, statement is executes but it condition is false nothing happens
- Condition may be a vector/ matrix in which case it is true only it all its elements are non zero.

If-else statement

The syntax for if-else statement is given below

If condition_1;

Statements 1;

Else

Ststements 2;

End;

Note:

(i) statement-1 and statement-2 represents one are more statements

(ii) if condition is true statements are executed ,but if condition is false; statement B are executed.

(Iii) The else part is optional

Nested –if /Else if/Ladder:

The following if construct can be used to implement program branching.

Syntax for else-if statement:

```
if  condition1/expression1
        Statement A;
else if condition 2
        Statement B;
else if condition 3
        Statement C;.
        .
else
        statement E
end;
```

Procedure:

(i) Condition is tested .if it is true statement A are executed; MATLAB then moves to the next statement after end.

(ii) If condition is false MATLAB checks condition 2.if it is true, statement B are executed, followed by the statement after end;

(iii) In this way all the conditions are tested until a true condition is found as same as a true condition is found, no further else-if‘s are examined and MATLAB jumps off the loader

(iv) If none of the conditions are true, statements E after else are executed

(v) You should arrange the logic so that not more than one of the condition is true

(vi) There can be any number of else if but at most only one else

(vii) Else if must be written as one word

Switch/case/otherwise statement*:*

The switch statement executes certain statements based on the value of a variable or expression. To implement a menu of options the following switch statement is of term used.

Syntax of switch statement

switch choice

case value-1;

statements1;

case value-2;

statements-2;

case value-3;

statement-3;

…….

otherwise

statement-k;

end

Note:

Here choice is an integer valued variable or expression. If choice is equal to value-k,the program control is transferred to statement K; otherwise it is transferred to the otherwise statement. The otherwise case is optional.

WHILE LOOP :

Many numerical computations can be done iteratively using loops. The most general loop uses while statement as follows.

While expression

Statement % loop body

End

Here the loop body a block of statements is executed repeatedly while the expression is true. Typically the loop body is changes the value of variable is an expression so that eventually the expression becomes false and loop terminates the body of a while loop is executed zero or more times depends on initial value of the expression.

FOR LOOP :

For k= m: p: n

Statement;

End

The first time through the loop the loop body is executed with k=m. The second time the loop body is executed with k= m+p and this process continues while k is less than or equal to n.

```
%program for for loop
a=2;
x=a/2;
for i=1:6;
x=(x+a/x)/2;
disp(x);
end;
```

output:

```
1.5000    1.4167    1.4142    1.4142    1.4142    1.4142
```

8. GRAPHS

The MATLAB environment offers a variety of data plotting functions plus a set of GUI tools to create, and modify graphic displays. The GUI tools afford most of the control over graphic properties and options that typed commands such as annotate, get, and set provide.

A figure is a MATLAB window that contains graphic displays (usually data plots) and UI components. You create figures explicitly with the figure function, and implicitly whenever you plot graphics and no figure is active. By default, figure windows are resizable and include pull-down menus and toolbars.

A plot is any graphic display you can create within a figure window. Plots can display tabular data, geometric objects, surface and image objects, and annotations such as titles, legends, and colorbars. Figures can contain any number of plots. Each plot is created within a 2-D or a 3-D data space called an axes. You can explicitly create axes with the axes or subplot functions.

A graph is a plot of data within a 2-D or 3-D axes. Most plots made with MATLAB functions and GUIs are therefore graphs. When you graph a one-dimensional variable (e.g., rand(100,1)), the indices of the data vector (in this case 1:100) become assigned as x values, and plots the data vector as y values. Some types of graphs can display more than one variable at a time, others cannot.The contents and varieties of figures, plots and graphs that MATLAB can make are explained in the following sections.

plot Linear plot.

plot(X,Y) plots vector Y versus vector X. If X or Y is a matrix, then the vector is plotted versus the rows or columns of the matrix,whichever line up. If X is a scalar and Y is a vector, disconnectedline objects are created and plotted as discrete points vertically at X.

plot(Y) plots the columns of Y versus their index.If Y is complex, plot(Y) is equivalent to plot(real(Y),imag(Y)).In all other uses of plot, the imaginary part is ignored.

Various line types, plot symbols and colors may be obtained with **plot(X,Y,S)** where S is a character string made from one element from any or all the following 3 columns:

b	blue	. point	- solid
g	green	o circle	: dotted
r	red	x x-mark	-. dashdot
c	cyan	+ plus	-- dashed
m	magenta	* star	(none) no lin
y	yellow	s square	
k	black	d diamond	
w	white	v triangle (down)	
		^ triangle (up)	
		<triangle (left)	
		>triangle (right)	
		p pentagram	
		h hexagram	

For example, **plot(X,Y,'c+:')** plots a **cyan dotted line** with a **plus** at each data point; **plot(X,Y,'bd')** plots **blue diamond** at each data point but does not draw any line.

Many types of MATLAB® functions are available for displaying vector data as line plots, as well as functions for annotating and printing these graphs. The following table summarizes the functions that produce basic line plots. These functions differ in the way they scale the plot's axes. Each accepts input in the form of vectors or matrices and automatically scales the axes to accommodate the data. **Axes are automatically scaled and drawn to include the minimum and maximum data points.**

We can override this with

Axis ([Xmin Xmax Ymin Ymax])

Saving to printing graphs:To obtain a hard copy of graph type point in the command window after the graph appears in single window. The figure can also be sowed into a specified file in pot scripts (ps) or encapsulated post script(eps)

Syntax:

point-d.device type_options file name

Script	Function name
ps	black and white post script
psc	color post script
ps2	level2 black and white post script
psc2	level 2 color post script
eps	black and white eps
epsc	color eps
eps2	level2 black and white eps
epsc2	level2 color eps

MATLAB can also graphic file in the following popular formats among others;

Dill-saves a file in adobe illustrator format

Djpeg-saves a file as a jpeg image

Dtiff-saves a file as a compressed tiff image

9. INPUT&OUTPUT

INPUT

To prompt the user for input from the key board and display output on the console screen, the following built in MATLAB functions can be used.

```
X=input(_prompt‘);        %get key board input using prompt string prompt
Y=printf(_format,list‘);  %display the list of values format string prompt
S=sprint(_format,list);   %send formatted output to string S.
```

Note:

***T**hearguments prompt and format string variables or string constants, enclosed in quotes.

*For input function, arrays can be entered from the key board if they are enclosed in square brackets, and strings can be entered if they are enclosed in single quotes.

The formatted print function behaves like the corresponding function in language C.For each variable or expression in the comma-separated by list, there is a format specified in format string indicating where the next value is to be placed and the format used to display it. The following are examples of format specifiers or place holders.

Format specifier	List
%d	Decimal point
%f	Fixed point
%e	Scientific notation real
%g	General real
%s	String
%x	Hexadecimal

*A special symbol \n is used for a new line.

*The sprintf function is identical to the fprintf function, except it sends the output to a string s instead of the console screen.

OUTPUT

There are two straight forward ways of getting output from MATLAB.

(1). By entering a variable name, assignment or expression on the command window, window line, without a semicolon.

(2). With a **disp** statement.

The syntax of disp statement is **disp (variable)**

Example:

```
X=2;

disp(_the answer is----_);

disp(X)
```

Note:

*****When you use disp the variable name is not displayed and you don‘t get a line feed before the value is displayed as you enter a variable name on the command window line without a semicolon.

*We can disply more than one number on a line as disp([x y z])

MATLAB PROGRAMS

1. Greatest of three numbers

Aim: To develop MATLAB program for finding the greatest of three numbers using nested-if and else-if ladder.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: Consider three numbers for a,b,c. And then compare the numbers with each other. First compare the number with a. if a is greater than b, then compare a with c. if b is greater than c, print a is greatest. Repeat the same procedure for the other two numbers.

Program

```
%%%%-------------Greatest of three numbers ----------------- %%%%%%%%%%%%
%%-------------------if condition-------------------------%%
clc
clear all
close all
a=input('enter the first number');
b=input('enter the second number');
c=input('enter the third number');
if (a>=b) && (a>=c)
    fprintf('Greatest number is %d/n',a)
end
if (b>=a) && (b>=c)
    fprintf('Greatest number is %d', b)
end
if (c>=a) && (c>=b)
    fprintf('Greatest number is %d', c)
end
%%-------------------------------------------------------------------%%

%%-------------------else - if condition-------------------------%%
clc
clear all
close all
a=input('enter the first number\n');
b=input('enter the second number\n');
c=input('enter the third number\n');
if (a>=b) && (a>=c)
    fprintf('Greatest number is %d\n',a)
else if(b>=a) && (b>=c)
 fprintf('Greatest number is %d\n',b)
    else
         fprintf('Greatest number is %d\n',c)
    end
end
%%%%-------------------------------------------------------------------%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter the first number

6

enter the second number

9

enter the third number

8

Greatest number is 9

Result:

Greatest of three numbers is obtained using if condition and else-if ladder and verified theoretically and practically using MATLAB.

Viva Questions

1. Give syntax for nested if condition
2. Write the syntax for else-if ladder
3. When 5 numbers are to be compared, what is the condition to find the greatest number

2. Counting digits of number

AIM: To develop MATLAB program for counting digits of a given number using while loop

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY:

To find the count of number of digits of a given number. Consider a number. Use mod command dividing by 10. Check for the remainder in every case and repeat the procedure until x=0. The count value is updated by the value of sum

PROGRAM

```
%%%%%----------------Count of digits-----------------------%%%%%%%
clc
clear all
close all
x=input('enter number');
% disp(x)
sum=0;
while x>0
    t=mod(x,10);
    sum=sum+1;
    x=(x-t)/10;
end
fprintf('No.of digits is\t %d',sum)
%%-------------------------------------------------------------------%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

OUTPUT

enter number
56

No.of digits is 2

Result:

The count of digits of a given number is obtained using while loop, and verified theoretically and practically using MATLAB.

Viva Questions

1. How do you count digits of a number?
2. What is the function of “mod” in MATLAB
3. Which numbers are called counting numbers

3. Perfect Number

AIM: To develop MATLAB program for checking the number is perfect number or not.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY:

The number is called perfect number if the sum of its divisors is equal to the original number. Consider a number. Perform mod operation to check for the remainder in order to obtain the divisors of the number. Obtain the sum of divisors for every iteration. Now compare with original number whether it is equal or not. If it is equal, the number is perfect number. If not equal it is not a perfect number.

Program

```
%%%%----------------Perfect Number-----------------------%%%%%%%
clc
clear all
close all
x=input('enter number\n');
n=1;
sum=0;
while n<x
    x1=mod(x,n);
    if (x1==0)
        sum=sum+n;
    end
    n=n+1;
end
if sum==x
    fprintf('%d is a perfect number',x)
else
    fprintf('%d is not a perfect number',x)
end
%%%-------------------------------------------------------------------------%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter number

28

28 is a perfect number

Result

The given number is checked whether it is perfect number or not and verified theoretically and practically using MATLAB.

Viva Questions

1. Define perfect number.
2. Write the first three perfect number.
3. What is the use of perfect numbers
4. Is zero a perfect number?

4. Strong Number

AIM: To develop MATLAB program for checking whether the given number is strong number or not.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: The number is called strong number if the sum of the factorial of digits is equal to original number. Consider a number and convert into string first to obtain the length of string. For this length of loop, obtain factorial in each iteration. After that perform the sum of factorials in each iteration and check whether it is equal to original number or not. If it is equal to original number, it is called strong number. If not it is not a strong number.

PROGRAM

```
%%%%%----------------Factorial-----------------------%%%%%%%
clc
clear all
close all
x=input(' enter number');
xx=num2str(x);
l=max(size(xx));
sum=0;
for i=1:l
    a=xx(i);
    aa=str2double(a);
    a1=factorial(aa);
    sum=sum+a1;
end
if sum==x
    fprintf('%d is strong number',x)
else
    fprintf('%d is Not strong number',x)
end
%%-------------------------------------------------------------------------%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter number 145

145 is strong number

Result

The given number is checked whether it is strong number or not and verified theoretically and practically using MATLAB.

Viva Questions

1. Define strong number.
2. Give any 3 strong numbers?

5. Armstrong Number

AIM: To develop MATLAB program for checking whether the given number is Armstrong number or not.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: The number is called Armstrong number if the sum of the cubes of digits is equal to original number. Consider a number and convert into string first to obtain the length of string. For this length of loop, obtain cube of individual digit in each iteration. After that perform the sum of cubes in each iteration and check whether it is equal to original number or not. If it is equal to original number, it is called Armstrong number. If not it is not a strong number.

PROGRAM

```
%%%%%----------------Armstrong Number-----------------------%%%%%%%
clc
clear all
close all
x=input('enter number');
xx=num2str(x);
l=max(size(xx));
sum=0;
for i=1:l
    a=xx(i);
    a1=str2num(a);
    a=a1^3;
    sum=sum+a;
end
if sum==x
    fprintf('%d is armstrong number',x)
else
    fprintf('%d is not armstrong number',x)
end
%%-------------------------------------------------------------------%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter number153

153 is armstrong number

Result

The given number is checked whether it is Armstrong number or not and verified theoretically and practically using MATLAB.

Viva Questions

1. Define Armstrong Number?
2. List some examples of Armstrong numbers?
3. Is 307 an Armstrong number?

6. Palindrome Number

AIM: To develop MATLAB program for checking whether the given number is palindrome number or not.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: The number is called palindrome number if the number remains same when the digits are reversed. Consider a number. Reverse the digits of a number and perform mod operation to check for remainder. Repeat the same procedure until the value is equal to 0. If the assigned reverse variable is equal to the original number, then it is called palindrome.

PROGRAM

```
%%%%----------------Palindrome-----------------------%%%%%%%
clc
clear all
close all
x=input('enter number\n');
x1=x;
rev=0;
while x>0
    xx=mod(x,10);
    rev=(rev*10)+xx;
    x=floor(x/10);
end
if (rev==x1)
    fprintf('%d is a palindrome',x1);
else
    fprintf('%d is not a palindrome',x1)
end
%%%-------------------------------------------------------------------------%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

OUTPUT

```
enter number
121
121 is a palindrome
```

Result

The given number is checked whether it is palindrome number or not and verified theoretically and practically using MATLAB.

Viva Questions

1. Define Palindrome
2. What are examples of Palindrome?
3. What is the logic of palindrome number?
4. What is the smallest palindrome number?
5. What is the largest palindrome number?

7. Fibonacci Series

AIM: To develop MATLAB program for obtaining Fibonacci series up to the given range

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: The Fibonacci sequence is a type of series where each number is the sum of the two that precede it. Generally, it starts from 0 and 1. Both 0 and 1 are initialized with variables and in each iteration summation with the preceding number is performed. The procedure is repeated up to the range of given number.

PROGRAM

```
%%%%----------------Fibonocci series-----------------------%%%%%%%
clc
clear all
close all
x=input('enter number\n');
s=0;
a=1;
b=1;
disp('fibonocci series upto given range is')
fprintf('\n %d %d',s,a);
while b<x
    b=s+a;
    if (b<=x)
        fprintf('%d',b)
        s=a;
        a=b;
    else
        break
    end
end
%%%-------------------------------------------------------------------------%%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter number

5

fibonocci series up to given range is

0 11235

Result

Fibonacci series up to the given range is obtained and verified theoretically and practically using MATLAB.

Viva Questions

1. What is meant by Fibonacci series?
2. What is the advantage of Fibonacci sequence?
3. What is the Fibonacci of 10?
4. What is the formula for Fibonacci sequence?

8. Exponential Series

AIM: To develop MATLAB program for printing the result of exponential series 1+x^2/2+x^3/3+……+x^n/n

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: The result of exponential series 1+x^2/2+x^3/3+……+x^n/n is obtained by initializing both the values of x and n. the value of x and the term in n. The series is evaluated using both the values of x and n.

PROGRAM

```
%%%%-------------Exxponential series ----------------- %%%%%%%%%%%%%
clc
clear all
close all
x=input('enter the value of x');
n=input('enter number of terms');
sum=0;
fprintf('the result of series\t');
for i=1:n-1
    sum=sum+power(x,i);
    fprintf('%d^%d %d\t', x,i,i);
end
sum=sum+(power(x,n)/n);
%%%%-------------------------------------------------------------------%%%%%%%%%%%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Result

The result of exponential series 1+x^2/2+x^3/3+……+x^n/n is obtained and verified theoretically and practically using MATLAB.

Viva Questions

1. What is meant by exponential series?
2. What is an exponential rule?
3. What type of sequence is exponential sequence?

9. Matrix operations

AIM: To develop MATLAB program for printing the result of exponential series 1+x^2/2+x^3/3+……+x^n/n

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: Initially two matrices are considered. The sum, difference and product of two matrices is obtained by performing addition, subtraction and multiplication of two matrices in element by element form.

PROGRAM

```
%%-------Matrix operations--------- %%%%%
clc
clear all
close all
a=input('enter first matrix\n');
b=input('enter second matrix\n');
c=a+b;
d=a-b;
e=a*b;
disp(c)
disp(d)
disp(e)
%%-------------------------------------------------------%%%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

enter first matrix
[9 8 7;6 5 4;3 2 1]
enter second matrix
[1 1 1;1 1 1;1 1 1]

```
    10     9     8
     7     6     5
     4     3     2
```

```
 8   7   6
 5   4   3
 2   1   0

24  24  24
15  15  15
 6   6   6
```

Result

The sum, difference and product of two matrices is obtained and verified theoretically and practically using MATLAB

Viva Questions

1. What is a square matrix?
2. What is symmetric property of matrices
3. What is the rule to make product of matrices?
4. Represent sum and difference of a matrix.

10. Matrix Transpose

AIM: To develop MATLAB program for printing the result of exponential series 1+x^2/2+x^3/3+……+x^n/n

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: Consider a matrix form. Define the size of matrix with respect to number of rows and columns. For the total size of rows and columns interchange every element of row to column and column to row. Repeat the procedure for the total elements of a matrix. And obtain the transpose of a matrix.

PROGRAM

```
%%-------Matrix Transpose--------- %%%%%
    A = [1 2 3 5;4 5 6 7; 8 9 10 77;4 5 66 8];
    [m,n] = size(A) ;
    B = A ;
    for i = 1:m
          for j = 1:n
                B(i,j) = A(j,i) ;
          end
    end
    disp(B)
%%-----------------------------------------------------%%%%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.
6. Observe the resultant output in Command window\figure window.

Output

```
disp(A)
   1    2    3    5
   4    5    6    7
   8    9   10   77
   4    5   66    8
```

```
disp(B)

    1     4     8     4
    2     5     9     5
    3     6    10    66
    5     7    77     8
```

Result

The transpose of a matrix is obtained and verified theoretically and practically using MATLAB

Viva Questions

1. What is a transpose of matrix?
2. Define condition for transpose
3. What are properties of matrix?
4. What is the Eigen value of a transpose?

11.Upper and Lower Triangle of a Matrix

AIM: To develop MATLAB program for printing the upper and lower triangle of a matrix

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

THEORY: Consider certain number of rows and columns of a matrix. And assign the elements that are all greater than the diagonal elements to print the upper triangle and vie-versa to print the lower triangle of a matrix.

PROGRAM

```
%%%%----------------Upper  triangle-----------------------%%%%%%%
clc
close all
y=[];
for i=1:5
    for j=1:5
        if(i>=j)
           y(i,j)=1;
        else
            y(i,j)=0;
        end
    end
end
disp(y)
%%-------------------------------------------------------------------------%%

%%%%---------------------- Lower triangle-----------------%%%%%%
y=[];
for i=1:5
    for j=1:5
        if(i<=j)
           y(i,j)=1;
        else
            y(i,j)=0;
        end
    end
end
disp(y)
%%-------------------------------------------------------------------------%%
```

Procedure

1. Click on the MATLAB icon. The command window will be opened.
2. Go to tool bar and open new script
3. Save the program with .m extension.
4. Debug the program.
5. Correct the errors and run the program.

6. Observe the resultant output in Command window\figure window.

Output

```
1   0   0   0   0
1   1   0   0   0
1   1   1   0   0
1   1   1   1   0
1   1   1   1   1

1   1   1   1   1
0   1   1   1   1
0   0   1   1   1
0   0   0   1   1
0   0   0   0   1
```

Result

The upper and lower triangle of a matrix are obtained and verified theoretically and practically using MATLAB

Viva Questions

1. Define upper triangle of matrix
2. Define Lower triangle of matrix
3. What is a square matrix?
4. Is upper and lower triangle symmetric?
5. Does inverse exist for upper (or lower) triangle matrix?

12. Creating Simple Plots

AIM: To develop MATLAB program for generating simple plots

APPARATUS REQUIRED: PC

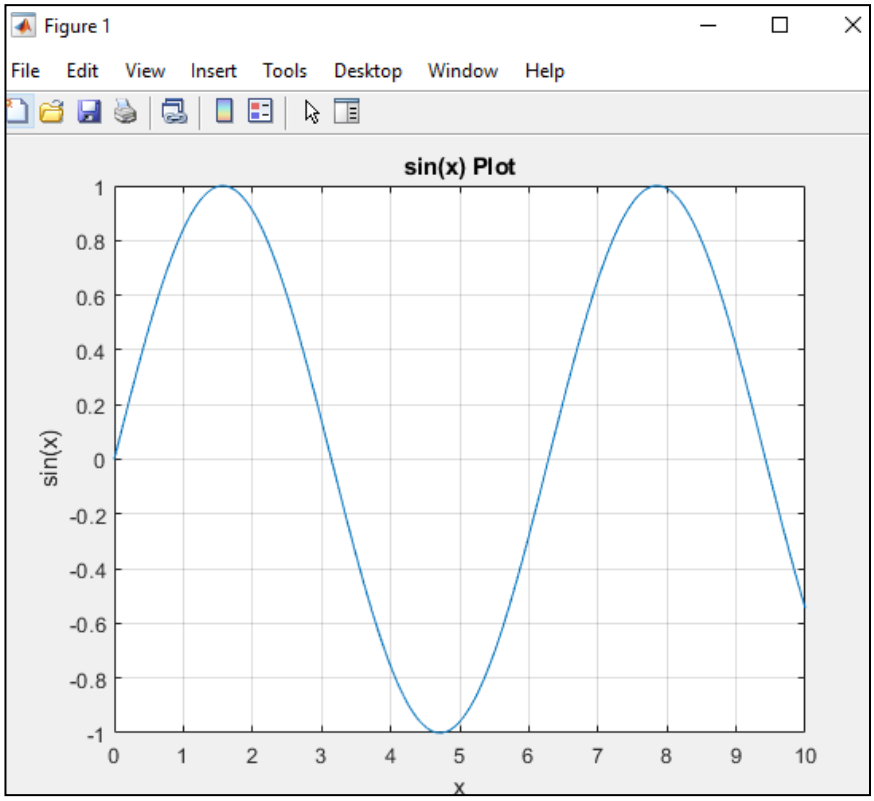
SOFTWARE USED: MATLAB

THEORY: Generate the sequence range for unit step, sine wave, impulse, and ramp. Read the time index of a discrete time signal and then plot the resultant signals. xlabel, ylabel, and title should be indicated. The pictorial representation of all the signals is obtained according to the given range.

PROGRAM

```
%%%%%----------------Simple Plots-----------------------%%%%%%%

%%%%%----------------Sine Plot-----------------------%%%%%%%
clc
clear all
close all
x=[0:0.01:10];
y=sin(x);
plot(x,y)
xlabel('x')
ylabel('sin(x)')
title('sin(x) Plot')
grid on
%%-------------------------------------------------------------------------%%
```

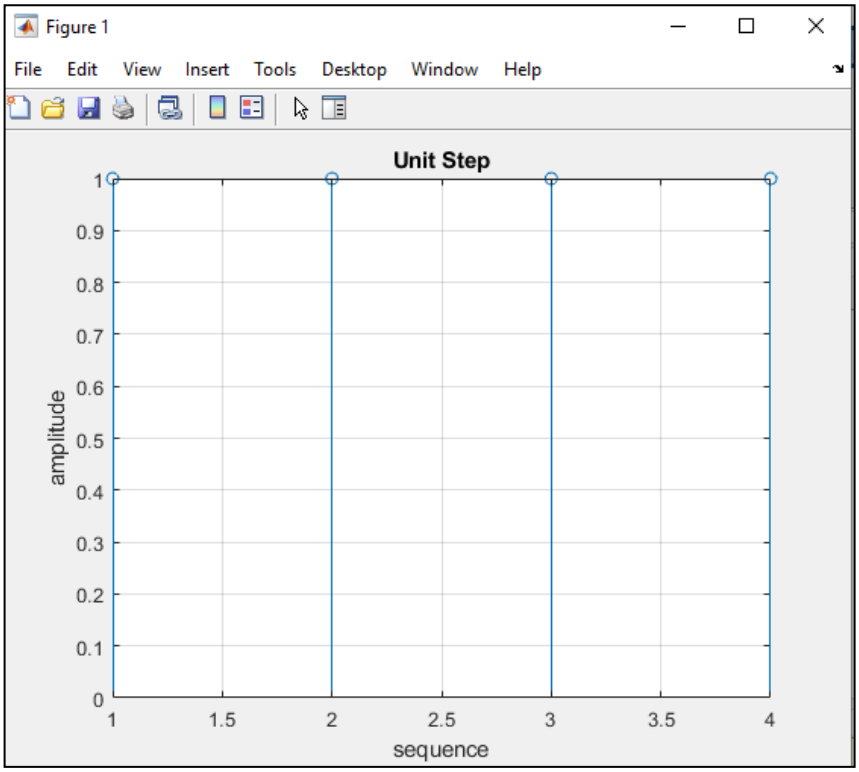
OUTPUT



```
%%%%----------------Unit Step-----------------------%%%%%%%
clc
clear all
close all
n=input('enter the sequence');
t=1:n;
u=ones(1,n);
stem(t,u)
xlabel('sequence')
ylabel('amplitude')
grid on
title('Unit Step')
%%-------------------------------------------------------------------------%%
```

OUTPUT

```
Enter sequence
4
```

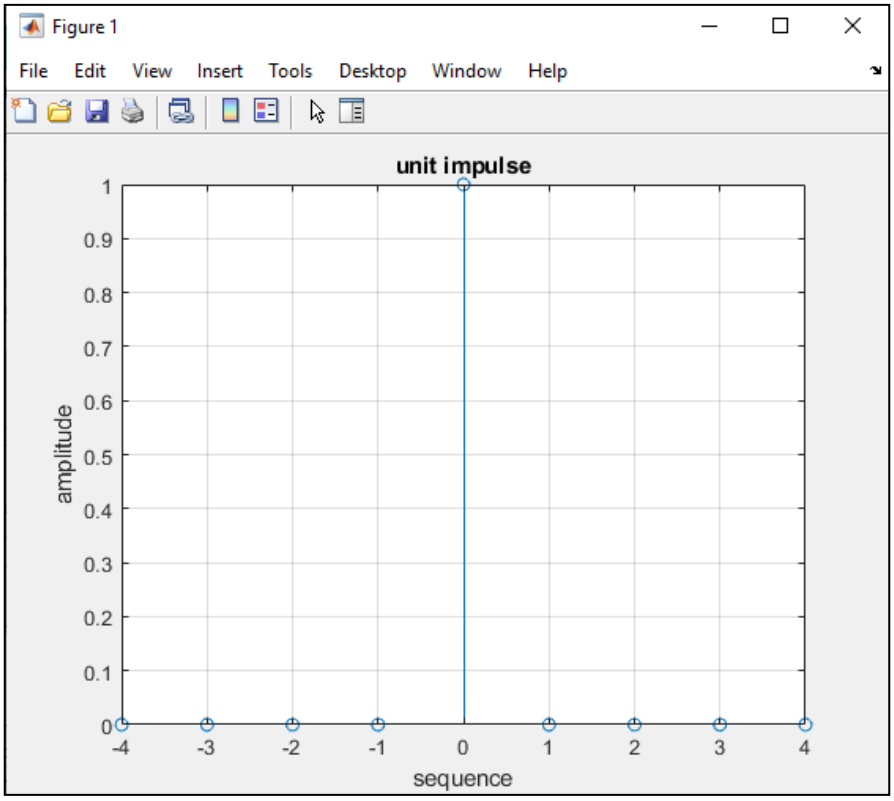



```
%%%%----------------Unit Impulse-----------------------%%%%%%%
clc
clear all
close all
N=input('enter sequence');
n=-N:1:N;
x=[zeros(1,N),1,zeros(1,N)];
stem(n,x)
xlabel('sequence')
ylabel('amplitude')
title('unit impulse')
```

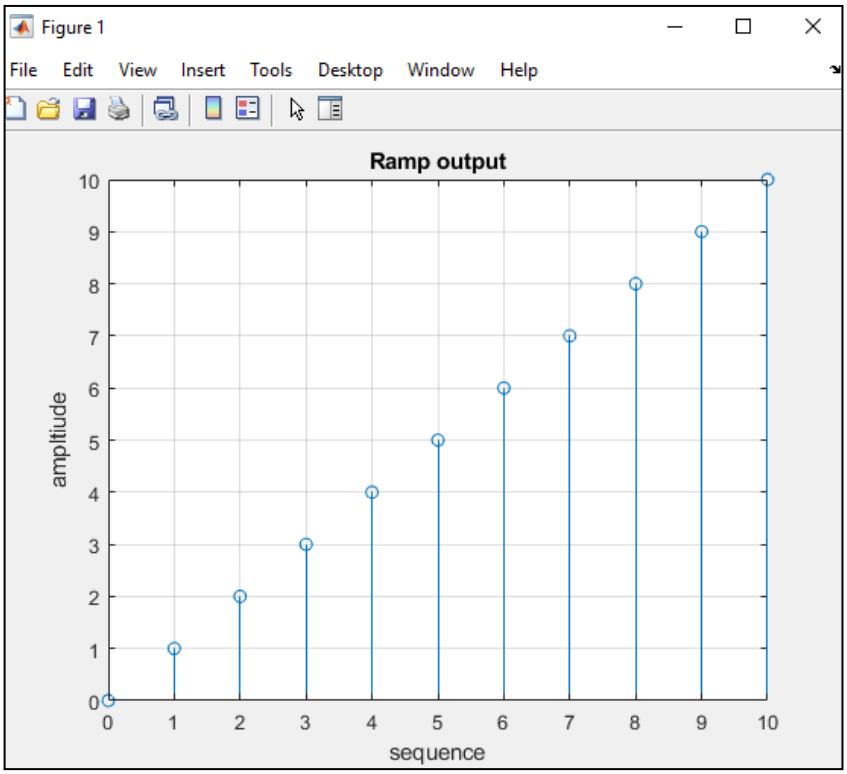
```
grid on
%%%%%-------------------------------------------------------%%%%%%%
```

OUTPUT

```
Enter sequence
4
```




```
%%%%%----------------Unit Ramp-------------------------%%%%%%%
clc
clear all
close all
n=0:1:10;
stem(n,n)
xlabel('sequence')
ylabel('ampltiude')
title('Ramp output')
grid on
%%%-------------------------------------------------%%%%%%%%%%%%
```



Procedure

13. Click on the MATLAB icon. The command window will be opened.

14. Go to tool bar and open new script

15. Save the program with .m extension.

16. Debug the program.

17. Correct the errors and run the program.

18. Observe the resultant output in Command window\figure window.

Result

Simple plots- unit step, unit impulse, unit ramp, and sine wave are verified and observed practically using MATLAB

Viva Questions

1. Define plot
2. What is a discrete time signal?
3. What is unit step signal
4. What is a ramp signal
5. What is an impulse

13. Image Operations

AIM: To develop MATLAB program for reading an image, performing different image operations and displaying the resulting images.

APPARATUS REQUIRED: PC

SOFTWARE USED: MATLAB

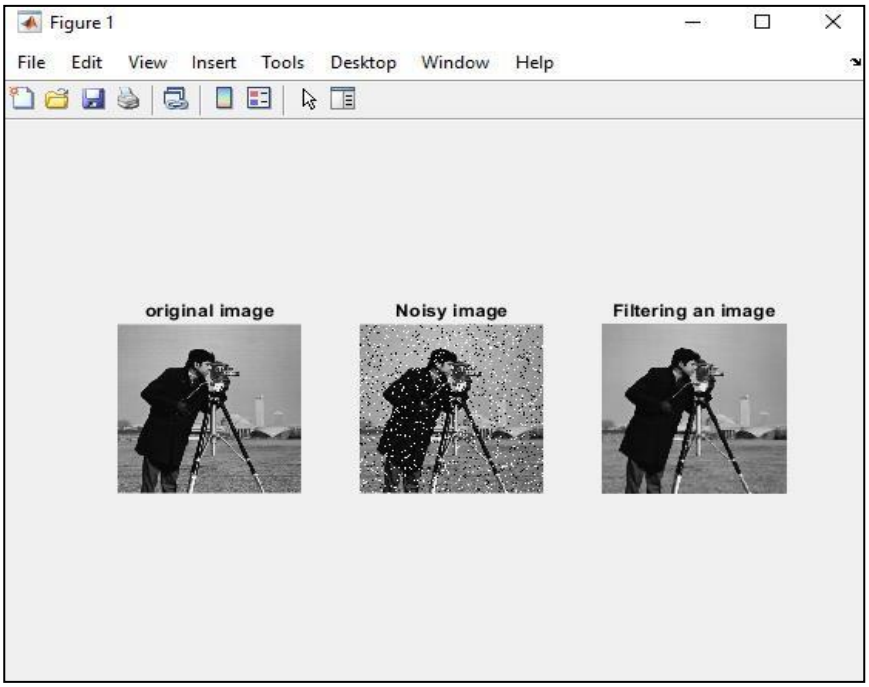
THEORY: Generate the sequence range for unit step, sine wave, impulse, and ramp. Read the time index of a discrete time signal and then plot the resultant signals. xlabel, ylabel, and title should be indicated. The pictorial representation of all the signals is obtained according to the given range.

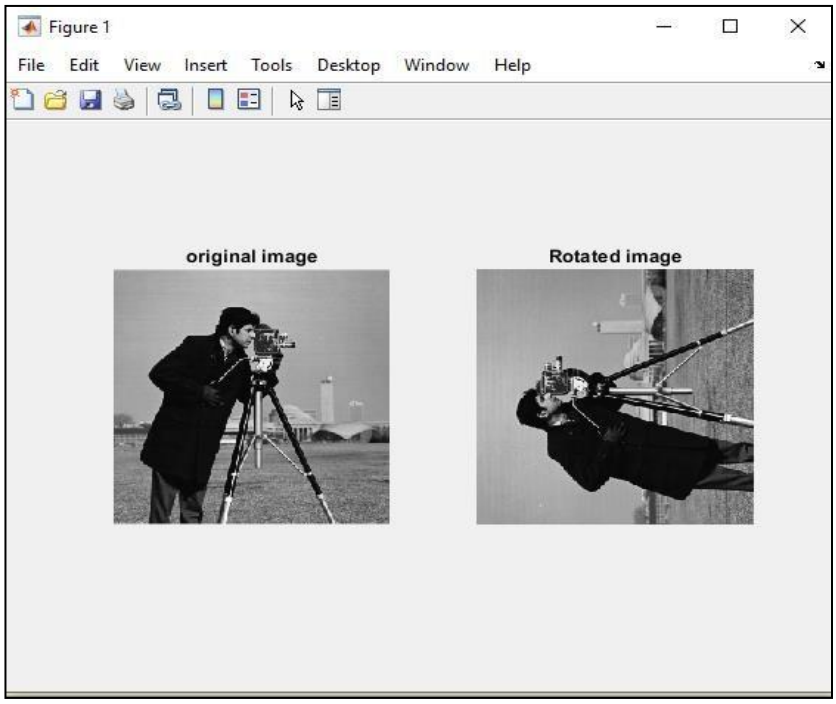
PROGRAM

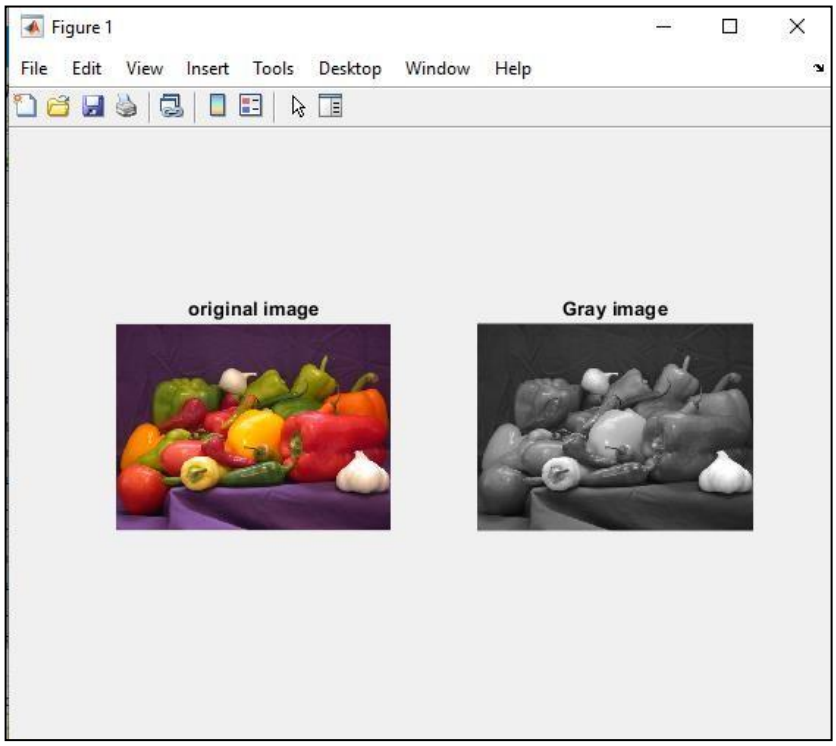
```
%%%%----------------------- Image operations-----------------%%%%%%%
%%%%----------------------- Filtering Image-----------------%%%%%%%

clc
clear all
close all
a=imread('cameraman.tif');
subplot(131)
imshow(a)
title('original image')
I1=imnoise(a,'salt & pepper',0.1);
subplot(132)
imshow(I1)
title('Noisy image')
b=medfilt2(I1);
subplot(133)
imshow(b)
title('Filtering an image')
%%%%----------------------- ------------------------------------%%%%%%%

%%%%----------------------- Rotating Image-----------------%%%%%%%
clc
clear all
close all
a=imread('cameraman.tif');
subplot(121)
imshow(a)
title('original image')
aa=imrotate(a,90,'bilinear','crop');
subplot(122)
imshow(aa)
title('Rotated image')
%%%%----------------------- ------------------------------------%%%%%%%

%%%%----------------- Color Image to Gray Image ------------ %%%%%%%
clc
clear all
close all
a=imread('peppers.png');
subplot(121)
imshow(a)
```

```
title('original image')
a1=rgb2gray(a);
subplot(122)
imshow(a1)
title('Gray image')
%%%%---------------------- ------------------------------------%%%%%%
```

Simulation Result







Result

Different image operations are performed and the resulting images are displayed verified practically using MATLAB

Viva Questions

1. Define an image
2. What are different image formats
3. What is a gray-level image
4. Define resolution of an image
5. Define filtering of an image