Simulation & Modeling

Instructor : Ms. Qudsia Yousaf

Class : BSSE – Semester V-A

qudsia.yousaf@cs.uol.edu.pk

Lecture No. 8

Introduction to Matlab

Key Objectives of today's lecture

- MATLAB:–
  - Features
  - Basic Programs

MAT (Matrix)

- MATLAB is a programming language developed by MathWorks.
- It started out as a matrix programming language where linear algebra programming was simple.
- MATLAB is a fourth-generation high-level programming language and interactive environment for numerical computation, visualization and programming
- It has numerous built-in commands and math functions that help you in mathematical calculations, generating plots, and performing numerical methods

Prerequisites

- A little knowledge of any computer programming and understand concepts like variables, constants, expression, statements, etc.
- Already knowledge work in a high-level programming language like C, C++ or Java will be a plus.

MATLAB's Power of Computational Mathematics

- MATLAB is used in every aspect of computational mathematics.
- Following are some commonly used mathematical calculations where it is used most commonly
  - Dealing with Matrices and Arrays
  - 2-D and 3-D Plotting and graphics
  - Linear Algebra
  - Algebraic Equations
  - Non-linear Functions
  - Statistics
  - Data Analysis
  - Calculus and Differential Equations
  - Numerical Calculations
  - Integration
  - Transforms
  - Curve Fitting
  - Various other special functions

Features of MATLAB

- Following are the basic features of MATLAB
  - It is a high-level language for numerical computation, visualization and application development.
  - It also provides an interactive environment for iterative exploration, design and problem solving.
  - It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
  - It provides built-in graphics for visualizing data and tools for creating custom plots.
  - It provides tools for building applications with custom graphical interfaces.
  - It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

Uses of MATLAB

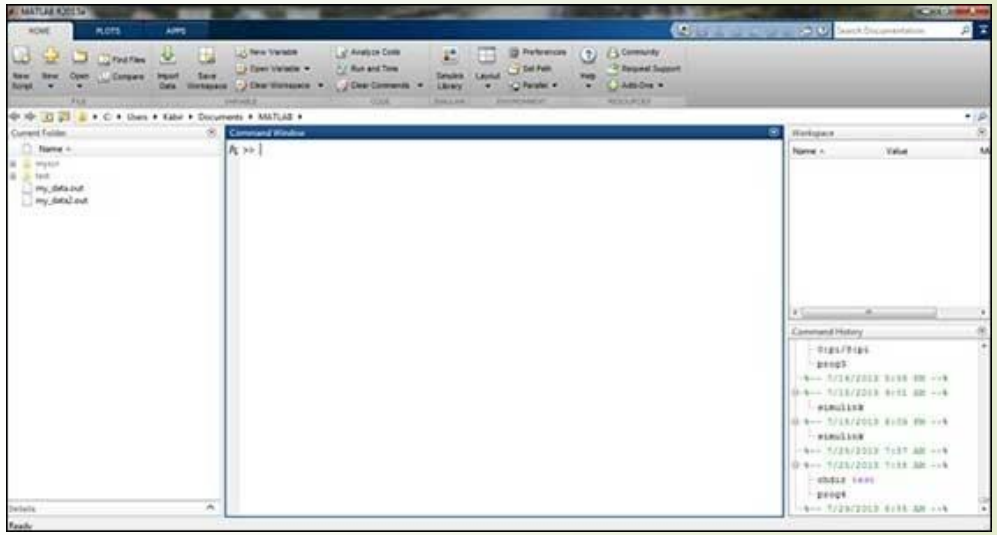
- MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams.
- It is used in a range of applications including
  - Signal Processing and Communications
  - Image and Video Processing
  - Control Systems
  - Test and Measurement
  - Computational Finance
  - Computational Biology

Download MATLAB

- Official link
  - https://www.mathworks.com/downloads/web_downloads/
- Crack version
  - https://www.youtube.com/watch?v=1i6dbTNyYIc
  - https://drive.google.com/drive/folders/13Pm0KOqqCVn99_YLeC8IK10z0P2RI6OL

MATLAB Environment

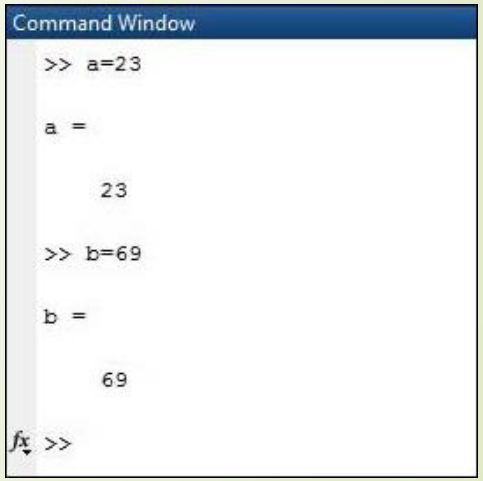
- MATLAB development IDE can be launched from the icon created on the desktop.
- The main working window in MATLAB is called the desktop.
- When MATLAB is started, the desktop appears in its default layout



MATLAB Environment (cont.)

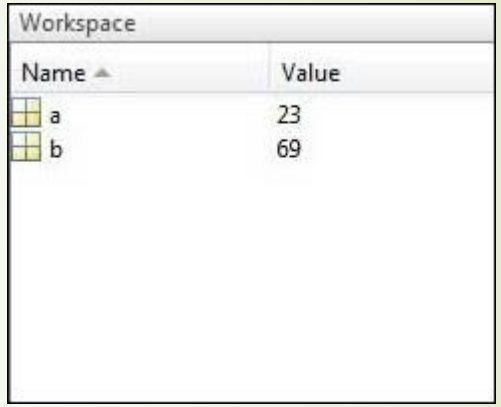
- The desktop has the following panels
  - **Current Folder**
    - This panel allows you to access the project folders and files.

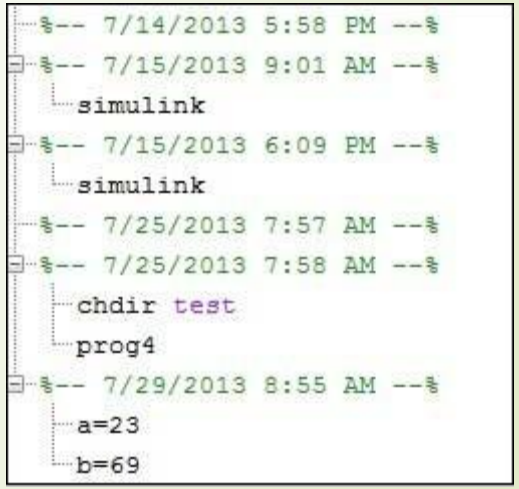
  - **Command Window**
    - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>)



MATLAB Environment (cont.)

- **Workspace**
  - The workspace shows all the variables created and/or imported from files.
- **Command History**
  - This panel shows or return commands that are entered at the command line.





MATLAB - Basic Syntax

- MATLAB environment behaves like a super-complex calculator. You can enter commands at the >> command prompt.
- MATLAB is an interpreted environment. In other words, you give a command and MATLAB executes it right away.
  - Type a valid expression, for example,
    - 5 + 5, ans = 10
    - 3 ^ 2 % 3 raised to the power of 2
    - sin(pi /2) % sine of angle 90 degree
    - x = 3;y = x + 5 %(;) represents end of statement

Commands for Managing a Session

Command	Purpose
clc	Clears command window.
clear	Removes variables from memory.
exist	Checks for existence of file or variable.
help	Searches for a help topic.
quit	Stops MATLAB.
who	Lists current variables.
whos	Lists current variables (long display).

MATLAB - M-Files

- MATLAB allows writing two kinds of program files −
- **Scripts**
  - script files are program files with **.m extension**.
  - In these files, you write series of commands, which you want to execute together. Scripts do not accept inputs and do not return any outputs.
  - They operate on data in the workspace.
- **Functions**
  - functions files are also program files with **.m extension**.
  - Functions can accept inputs and return outputs.
  - Internal variables are local to the function.
- You can use the MATLAB editor or any other text editor to create your **.m** files.

Example of .m file in MATLAB

```
NoOfStudents = 6000;
TeachingStaff = 150;
NonTeachingStaff = 20;
Total = NoOfStudents + TeachingStaff ...
 + NonTeachingStaff;
disp(Total);
```

Data Types Available in MATLAB

Sr.No.	Data Type & Description
1	**int8** 8-bit signed integer
2	**uint8** 8-bit unsigned integer
3	**int16** 16-bit signed integer
4	**uint16** 16-bit unsigned integer
5	**int32** 32-bit signed integer
6	**uint32** 32-bit unsigned integer
7	**int64** 64-bit signed integer
8	**uint64** 64-bit unsigned integer
9	**single** single precision numerical data
10	**double** double precision numerical data
11	**logical** logical values of 1 or 0, represent true and false respectively
12	**char** character data (strings are stored as vector of characters)

MATLAB - Operators

- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations
- MATLAB is designed to operate primarily on whole matrices and arrays
- Therefore, operators in MATLAB work both on scalar and non-scalar data
- MATLAB allows the following types of elementary operations
  - Arithmetic Operators
  - Relational Operators
  - Logical Operators

Arithmetic Operators

Sr.No.	Operator & Description
1	**+** Addition or unary plus. A+B adds the values stored in variables A and B. A and B must have the same size, unless one is a scalar. A scalar can be added to a matrix of any size.
2	**-** Subtraction or unary minus. A-B subtracts the value of B from A. A and B must have the same size, unless one is a scalar. A scalar can be subtracted from a matrix of any size.
3	***** Matrix multiplication. C = A*B is the linear algebraic product of the matrices A and B. More precisely,
4	**.*** Array multiplication. A.*B is the element-by-element product of the arrays A and B. A and B must have the same size, unless one of them is a scalar.
5	**/** Slash or matrix right division. B/A is roughly the same as B*inv(A). More precisely, B/A = (A'\B')'.
6	**./** Array right division. A./B is the matrix with elements A(i,j)/B(i,j). A and B must have the same size, unless one of them is a scalar.

Arithmetic Operators (cont.)

9	**^** Matrix power. X^p is X to the power p, if p is a scalar. If p is an integer, the power is computed by repeated squaring. If the integer is negative, X is inverted first. For other values of p, the calculation involves eigenvalues and eigenvectors, such that if [V,D] = eig(X), then X^p = V*D.^p/V.
10	**.^** Array power. A.^B is the matrix with elements A(i,j) to the B(i,j) power. A and B must have the same size, unless one of them is a scalar.
11	**'** Matrix transpose. A' is the linear algebraic transpose of A. For complex matrices, this is the complex conjugate transpose.
12	**.'** Array transpose. A.' is the array transpose of A. For complex matrices, this does not involve conjugation.

Relational Operators

Sr.No.	Operator & Description
1	**<** Less than
2	**<=** Less than or equal to
3	**>** Greater than
4	**>=** Greater than or equal to
5	**==** Equal to
6	**~=** Not equal to

Logical Operators

Sr. No.	Operator / Symbol
1	**&** AND operator
2	| OR operator
3	**~** NOT operator

Decision Operators (if…end)

```
if <expression>
    % statement(s) will execute if the boolean
expression is true
    <statements>
end
```

Decision Operators (if…else…end)

```
if <expression>
     % statement(s) will execute if the boolean
expression is true
     <statement(s)>
else <statement(s)>
     % statement(s) will execute if the boolean
expression is false
end
```

Decision Operators (if…else…end)

```
if <expression 1>
      % Executes when the boolean expression 1 is true
      if <expression 2>
            % Executes when the boolean expression 2 is true
      end
end
```

Taking Input

- x = input(prompt) – for numerical input
- str = input(prompt,'s') – for character input

Decision Operators (Switch)

- x = input(prompt) – for numerical input
- str = input(prompt,'s') – for character input

```
grade = 'B';
switch(grade)
        case 'A'
                fprintf('Excellent!\n' );
        case 'B'
                fprintf('Well done\n' );
        case 'C'
                fprintf('Well done\n' );
        case 'D'
                fprintf('You passed\n' );
        case 'F'
                fprintf('Better try again\n' );
        otherwise
                fprintf('Invalid grade\n' );
end
```

Loops (while loop)

```
a = 10;
% while loop execution
while( a < 20 )
       fprintf('value of a: %d\n', a);
       a = a + 1;
end
```

Loops (for loop)

```
for a = 10:20
    fprintf('value of a: %d\n', a);
end
```

Vectors

- A vector is a one-dimensional array of numbers. MATLAB allows creating two types of vectors
  - Row vectors

```
r = [7 8 9 10 11]
```

  - Column vectors

```
c = [7
     8
     9
     10
     11]
```

```
c = [7; 8; 9; 10; 11]
```

Matrix

- A matrix is a two-dimensional array of numbers

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

Functions

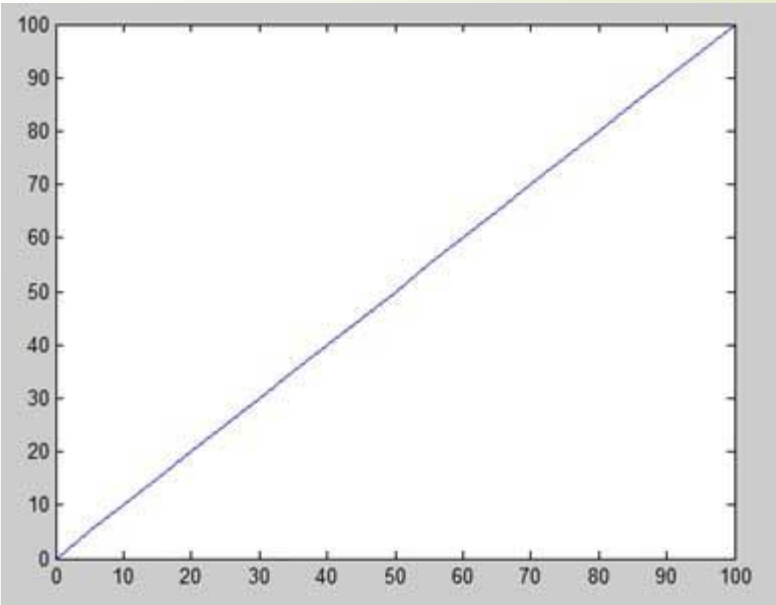
- A function is a group of statements that together perform a task. In MATLAB, functions are defined in separate files.
- The name of the file and of the function should be the same

```
function max = mymax(n1, n2, n3, n4, n5)
        %This function calculates the maximum of the
        % five numbers given as input
        max = n1;
        if(n2 > max) max = n2;
        end
        if(n3 > max) max = n3;
        end
        if(n4 > max) max = n4;
        end
        if(n5 > max) max = n5;
        end
```

Plotting

- To plot the graph of a function, you need to take the following steps
  - Define **x**, by specifying the **range of values** for the variable **x**, for which the function is to be plotted
  - Define the function, **y = f(x)**
  - Call the **plot** command, as **plot(x, y)**

```
x = [0:5:100];
y = x;
plot(x, y)
```

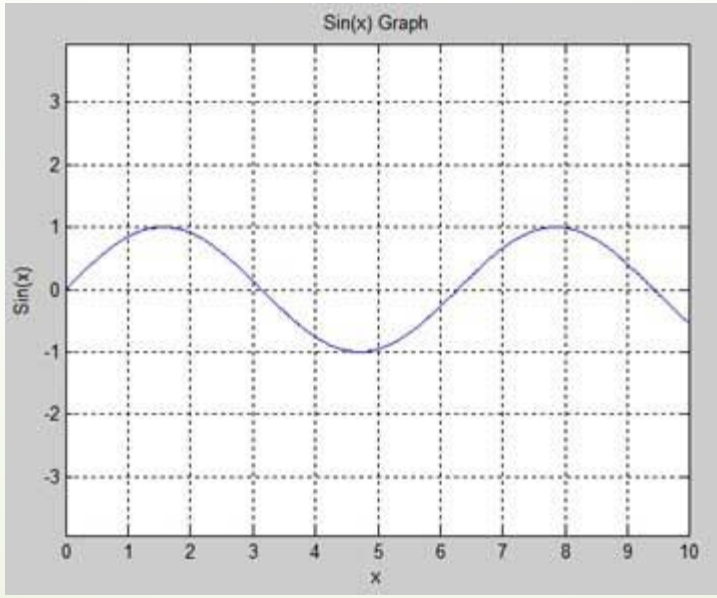


Adding Title, Labels, Grid Lines and Scaling on the Graph

- MATLAB allows you to add title, labels along the x-axis and y-axis, grid lines and also to adjust the axes to spruce up the graph.
  - The **xlabel** and **ylabel** commands generate labels along x-axis and y-axis.
  - The **title** command allows you to put a title on the graph.
  - The **grid on** command allows you to put the grid lines on the graph.
  - The **axis equal** command allows generating the plot with the same scale factors and the spaces on both axes.
  - The **axis square** command generates a square plot.

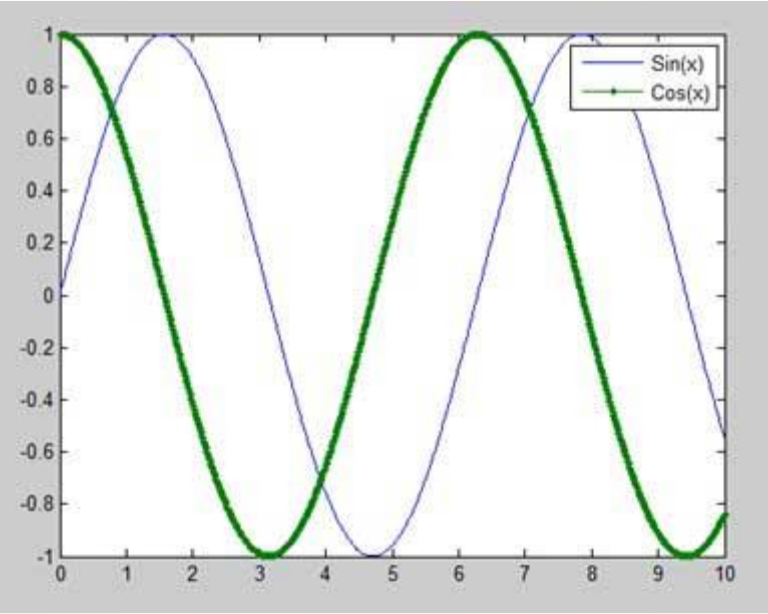
Plotting Example

```
x = [0:0.01:10];
y = sin(x);
plot(x, y), xlabel('x'), ylabel('Sin(x)'), title('Sin(x) Graph'), grid on, axis equal
```



Drawing Multiple Functions on the Same Graph

```
x = [0 : 0.01: 10];
y = sin(x); g = cos(x);
plot(x, y, x, g, '.-'), legend('Sin(x)', 'Cos(x)')
```



Exercise

- Create a function which prompts user to add a specific age
- On the basis of which the programs decides whether the entered age is eligible for casting vote or not
- Call this in command prompt or in another program

THANK YOU !