ENGINEERING MATHEMATICS-I
MATLAB

Department of Science and Humanities

Introduction to MATLAB

- MATLAB is a programming language developed by MathWorks.
- MATLAB stands for **MAT**rix **LAB**oratory.
- MATLAB is a program for doing numerical computation. It was originally designed for solving linear algebra type problems using matrices.
- While other programming languages mostly work with numbers one at a time, MATLAB is designed to operate primarily on whole matrices and arrays.

Introduction to MATLAB, Continued…

- Using MATLAB, an image (or any other data like sound, etc.) can be converted to a matrix and then various operations can be performed on it to get the desired results and values.
- 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 in mathematical calculations, generating plots, and performing numerical methods.

MATLAB's Power of Computational Mathematics

- MATLAB is used in every fact of computational mathematics. Following are some commonly used mathematical calculations where MATLAB is used:

- Dealing with Matrices and Arrays
- 2-D and 3-D Plotting and graphics
- Linear Algebra
- Algebraic Equations
- Statistics

MATLAB's Power of Computational Mathematics, Continued…

- Data Analysis
- Calculus and Differential Equations
- Numerical Calculations
- Integration
- Transforms
- Curve Fitting
- Special Functions

Uses of MATLAB

- MATLAB is widely used as a computational tool in Science and Engineering covering the fields of Physics, Chemistry, Mathematics, and all engineering streams.
- It is used in a range of applications including:

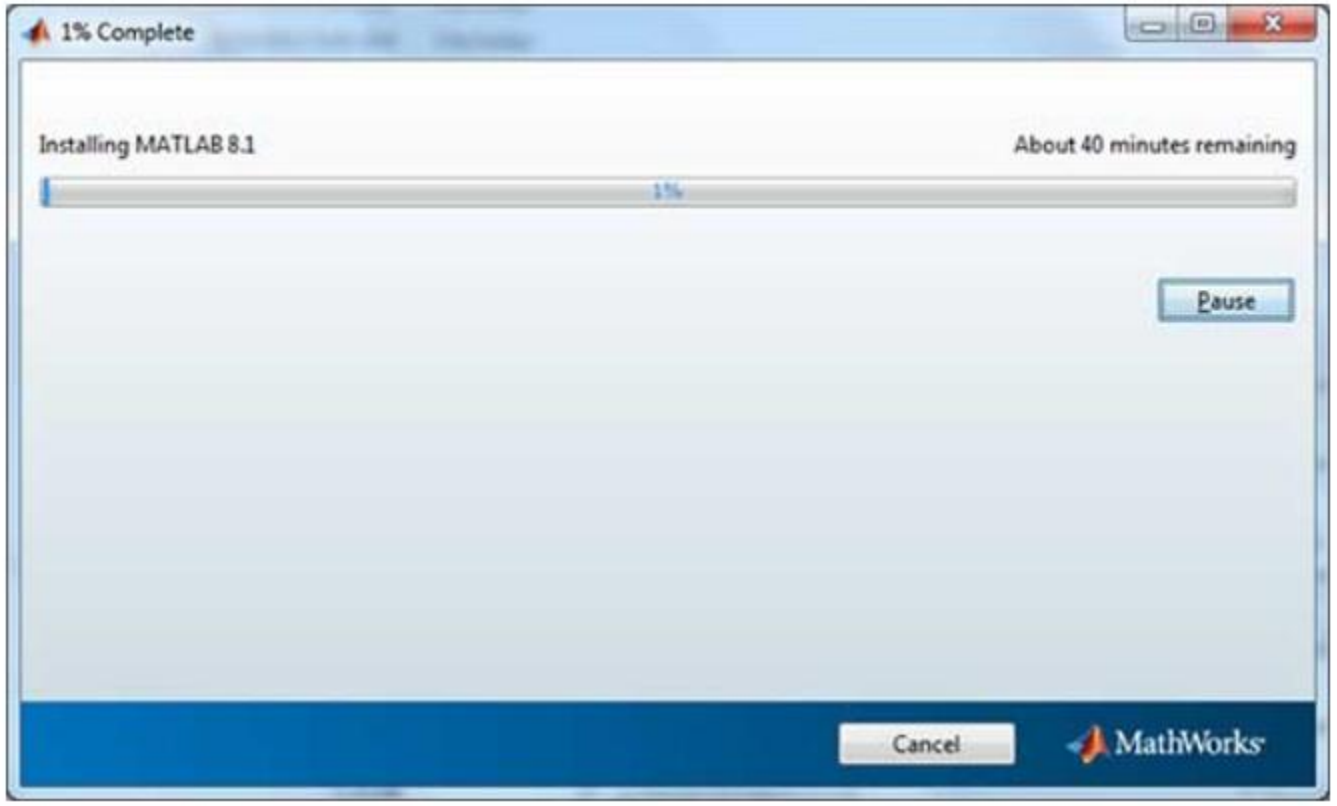
- Signal Processing and Communications
- Algorithm development
- Control Systems
- Computational Finance; Computational Biology

Local Environment Setup

- Setting up MATLAB environment is a matter of few clicks.
- MathWorks provides the licensed product, a trial version, and a student version as well.
- We need to log into the site and wait a little for their approval.
- After downloading the installer, the software can be installed through few clicks.

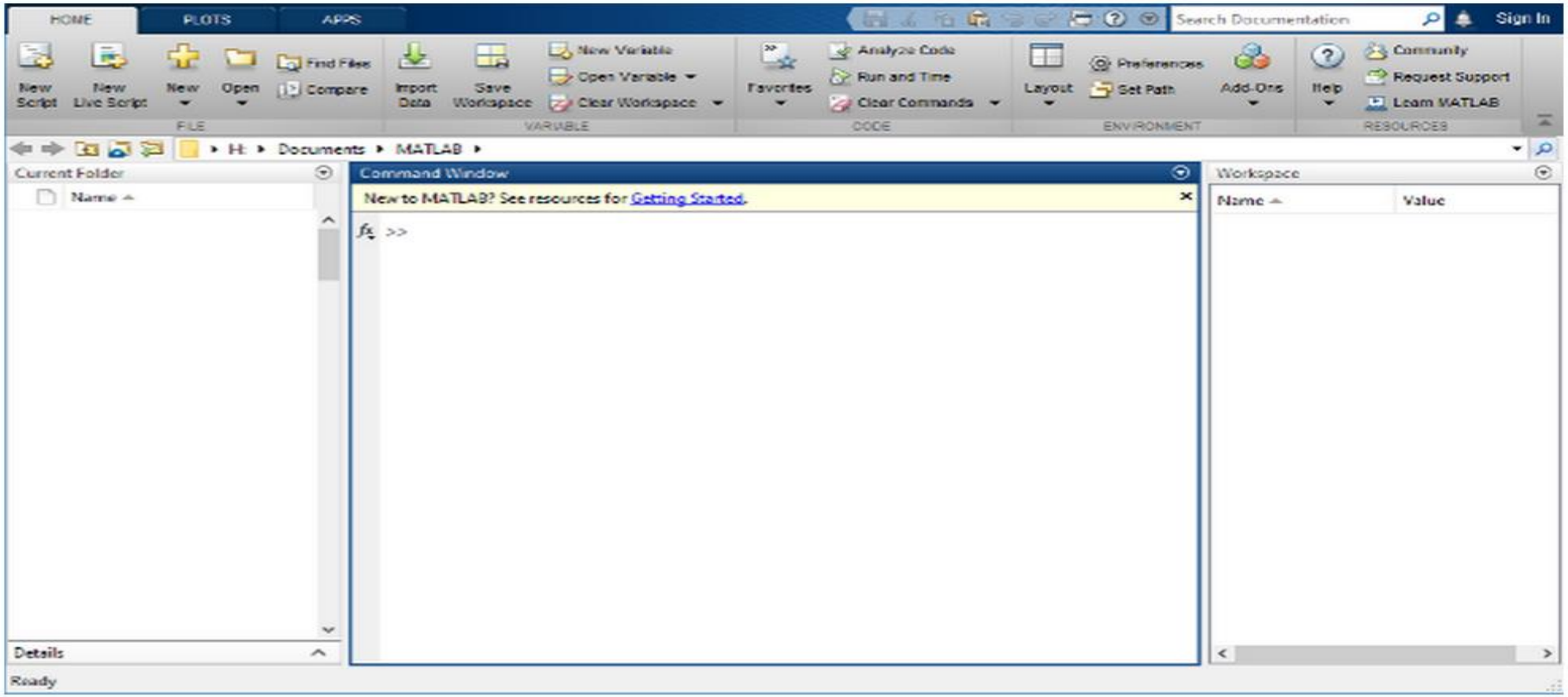
Local Environment Setup, Continued...

Local Environment Setup, Continued...



Understanding the MATLAB Environment

When you start MATLAB®, the desktop appears in its default layout.

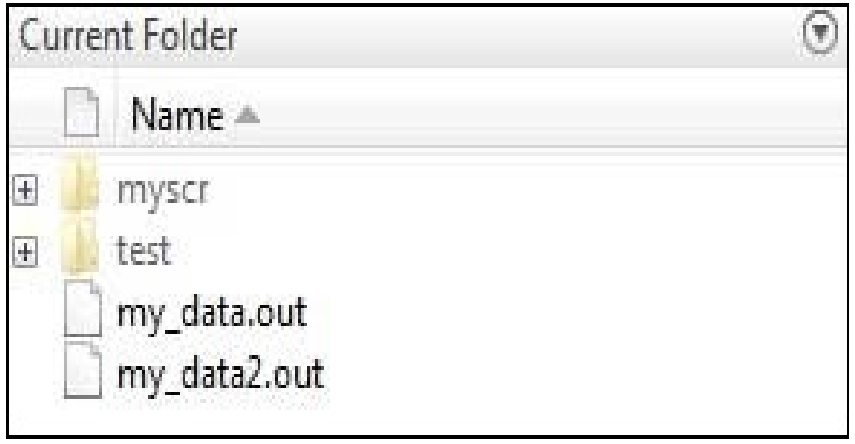


The desktop includes these panels:

- **Current Folder** — Access your files.
- **Command Window** — Enter commands at the command line, indicated by the prompt (>>).
- **Workspace** — Explore data that you create or import from files.

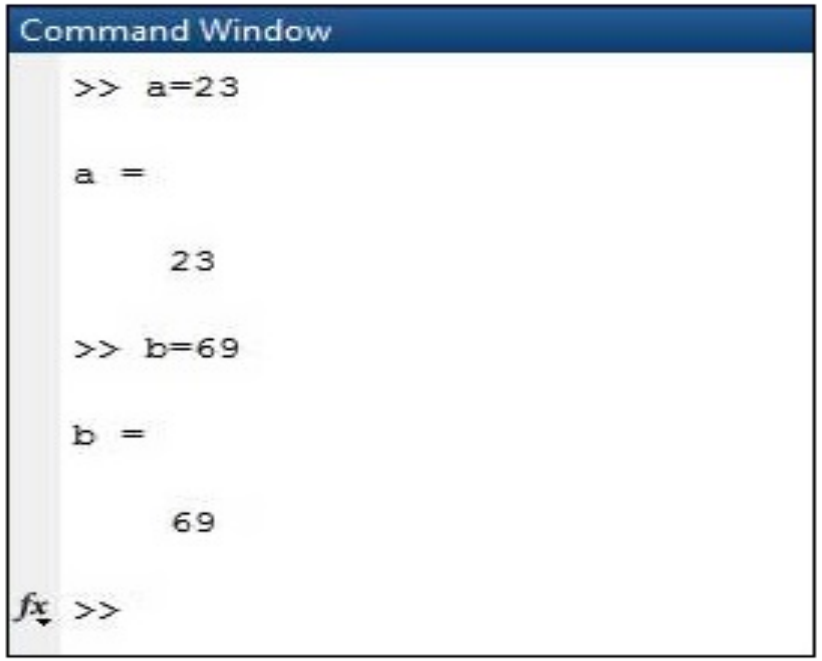
Understanding the MATLAB Environment, Continued…

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



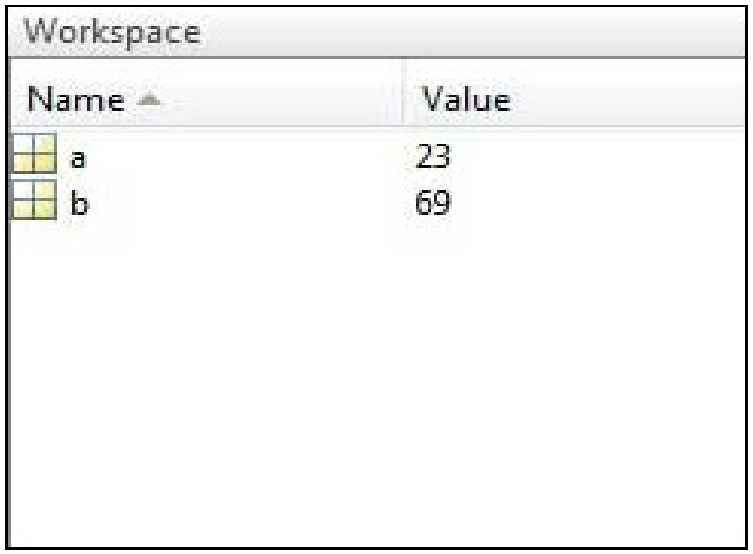
Understanding the MATLAB Environment, Continued…

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



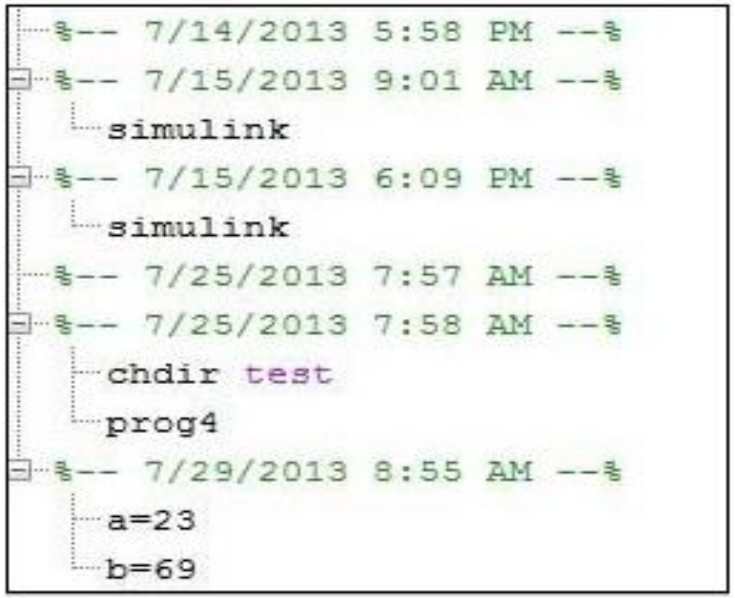
Understanding the MATLAB Environment, Continued…

- **Workspace:** The workspace shows all the variables created and/or imported from files.



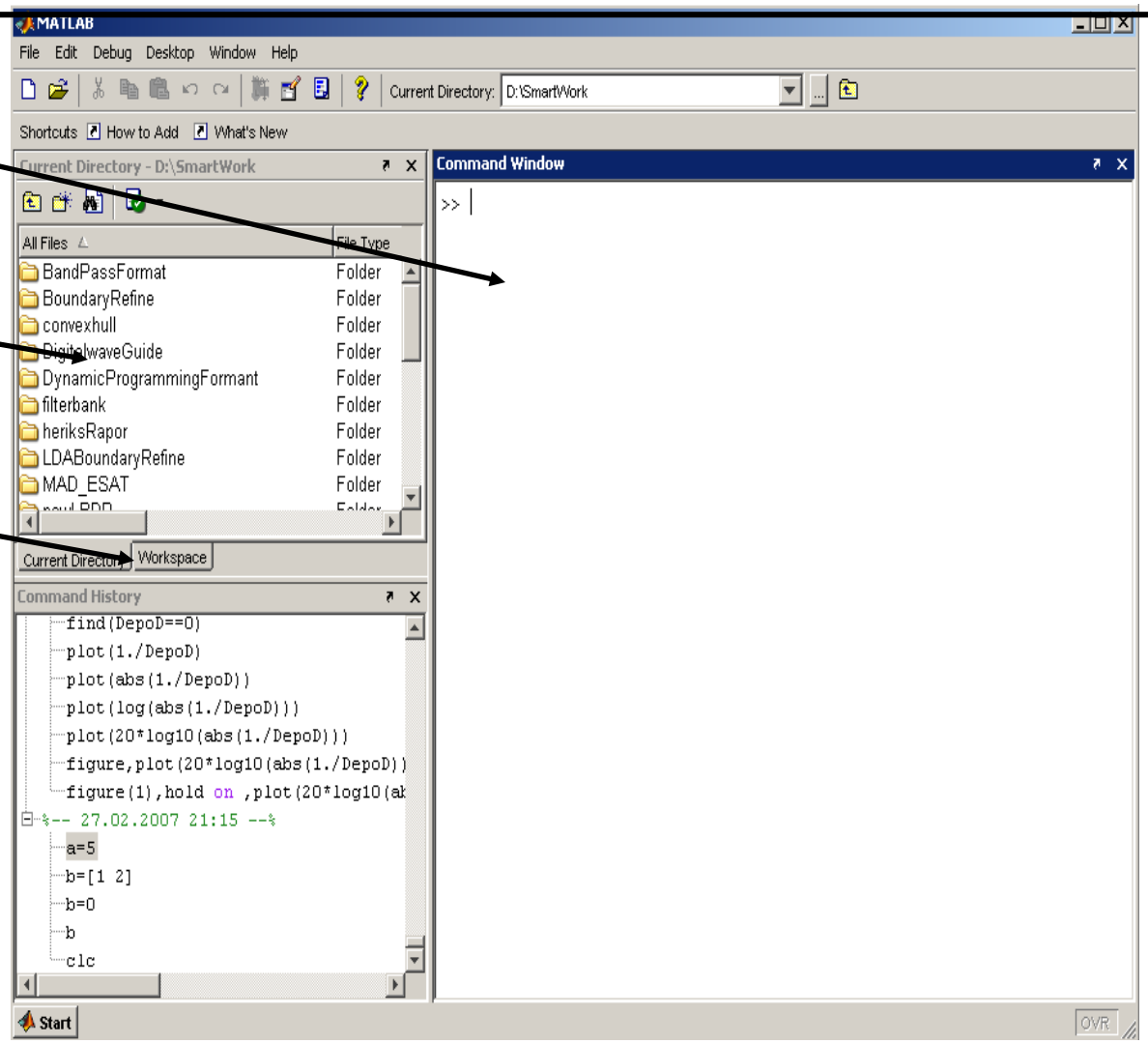
Understanding the MATLAB Environment, Continued…

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



MATLAB Screen

- Command Window
  - type commands
- Current Directory
  - View folders and m-files
- Workspace
  - View program variables
  - Double click on a variable to see it in the Array Editor
- Command History
  - view past commands
  - save a whole session using diary



Hands on Practice

- MATLAB environment behaves like a super-complex calculator. We can enter commands at the >> command prompt.
- In MATLAB, we give a command and it executes the command right away.
- Type a valid expression, for example, >>20 + 21; and press ENTER.
- When we click the Execute button, MATLAB executes it immediately and the result returned is ans=41.

Hands on Practice, Continued…

- Let us take up few more examples:
- >>3 ^ 2 (%3 raised to the power of 2). When we click the Execute button, MATLAB executes it immediately and the result returned is ans=9.
- >>sin(pi/2). (% sine of angle 90). When we click the Execute button, MATLAB executes it immediately and the result returned is ans=1.
- >>7/0 (% Divide by zero). When we click the Execute button, MATLAB executes it immediately and the result returned is ans=Inf.

Hands on Practice, Continued…

- >>732 * 20.3. When we click the Execute button, MATLAB executes it immediately and the result returned is ans=1.4860e+04.
- MATLAB provides some special expressions for some mathematical symbols, like pi for π, Inf for ∞, i (and j) for $\sqrt{-1}$.
- In MATLAB, **NaN** stands for 'not a number'.

  For example, >> 0/0; -inf/inf

Use of Semicolon (;) in MATLAB

- Semicolon (;) indicates end of statement. However, if we want to suppress and hide the MATLAB output for an expression, add a semicolon after the expression.
- For example, >>x = 5; y = x + 5. When we click the Execute button, MATLAB executes it immediately and the result returned is y=8.

Adding Comments in MATLAB

- To add comments to MATLAB code, we use the percent (%) symbol. Comment lines can appear anywhere in a program file, and we can add comments to the end of a line of code.

  For example, y = sum(x) % Use the sum function

Commonly used Operators and Special Characters

Operation	Symbol	Example
Addition	+	5+3=8
Subtraction	-	5-3=2
Multiplication	*	5*3=15
Right Division	/	5/3=1.6667
Left Division	\	5/3=3\5=1.667
Exponentiation	^	5^3=125

Special Variables and Constants

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 $\sqrt{-1}$.
Inf	Infinity.
NaN	Undefined numerical result (not a number).
pi	The number π

Naming Variables

- Variable names consist of a letter followed by any number of letters, digits or underscore.
- MATLAB is **case-sensitive**.
- For example, >>x = 10 (% defining x and initializing it with a value). MATLAB will execute the above statement and return the following result x = 10.
- >>x = sqrt(25) (% defining x and initializing it with an expression) MATLAB will execute the above statement and return the result as x = 5.

Naming Variables, Continued…

- Note that once a variable is entered into the system, we can refer to it later. Variables must have values before they are used. When an expression returns a result that is not assigned to any variable, the system assigns it to a variable named answer, which can be used later.
- For example, >>x = 7 * 8; y = x * 7.89. MATLAB will execute the above statement and return the following result y = 441.8400.

Multiple assignments

- We can have multiple assignments on the same line.
- For example, >> a = 2; b = 7; c = a * b
- MATLAB will execute the above statement and return the result c = 14.

I have forgotten the Variables

- The who command displays all the variable names we have used.
- MATLAB will execute the above statement and return the result: Your variables are: ans - - - …
- The whos command. MATLAB will execute the above statement and return the following result.

Name	Size	Bytes	Class	Attributes
ans	1x1	8	double	
x	1x1	8	double	

I have forgotten the Variables

>> clear x. It will delete x, won't display anything.

>> clear. It will delete all variables in the workspace.

>> clc. It clears all the text from the Command Window, resulting in a clear screen.

Long Assignments

- Long assignments can be extended to another line as follows:

```
>> initial_velocity = 0;
acceleration = 9.8;
time = 20;
>> Final_velocity = initial_velocity + acceleration * time
```

- MATLAB will execute the above statement and return the following result: final velocity = 196.

The format Command

- By default, MATLAB displays numbers with four decimal place values. This is known as **short format**.
- However, if we want more precision, then we need to use the **format** command.
- The **format long** command displays 15 digits after the decimal point.
- For example: >> format long

  >> x = 7 + 10/3 + 5 ^ 1.2
- MATLAB will execute the above statement and return the following result: x = 17.231981640639408.

The format Command, Continued…

Another example,

```
>> format short
>> x = 7 + 10/3 + 5 ^ 1.2
```

- MATLAB will execute the above statement and return the following result: x = 17.232
- The **format bank** command rounds numbers to two decimal places.

The format Command, Continued…

For example,

```
>> format bank
>> daily_wage = 177.45; weekly_wage = daily_wage * 6
```

- MATLAB will execute the above statement and return the following result: weekly_wage = 1064.70
- MATLAB displays large numbers using exponential notation.
- The **format short e** command allows displaying in exponential form with four decimal places plus the exponent.

The format Command, Continued…

- For example,

  >> format short e

  4.678 * 4.9

- MATLAB will execute the above statement and return the following result: ans = 2.2922e+01
- The **format long e** command allows displaying in exponential form with four decimal places plus the exponent.

The format Command, Continued…

- For example, >> format long e
  >> x = pi
- MATLAB will execute the above statement and return the following result: x = 3.141592653589793e+00
- The **format rat** command gives the closest rational expression resulting from a calculation.
- For example, >> format rat
  4.678 * 4.9
- MATLAB will execute the above statement and return the following result: ans = 34177/1491

Creating Vectors

- A vector is a one-dimensional array of numbers.
- MATLAB allows creating two types of vectors:
- Row vectors and Column vectors.
- **Row vectors** are created by enclosing the set of elements in square brackets, using space or comma.
- For example, >> X = [7 8 9 10 11]. MATLAB will execute the above statement and return the following result:

  X =

  7 8 9 10 11

Creating Vectors, Continued…

- Another example,

  >> X = [7 8 9 10 11]; >> Y = [2, 3, 4, 5, 6]; >> Z = X +Y

- MATLAB will execute the above statement and return the following result:

  Z =

  9 11 13 15 17

Creating Vectors, Continued…

- **Column vectors** are created by enclosing the set of elements in square brackets, using semicolon(;).
- For example, >> X = [7; 8; 9; 10]
- MATLAB will execute the above statement and return the following result:

```
X =

     7

     8

     9

    10
```

Creating matrices

- A matrix is a two-dimensional array of numbers.
- In MATLAB, a matrix is created by entering each row as a sequence of space or comma separated elements, and end of a row is terminated by a semicolon.
- For example, let us create a 3-by-3 matrix as

```
>>A = [1 2 3 4; 4 5 6 7; 7 8 9 10]
```

Creating matrices, Continued…

- MATLAB will execute the above statement and return the following result:

A =

1	2	3	4
4	5	6	7
7	8	9	10

Elementary Math Built – In Functions

Function	Description	Examples
sqrt(x)	Square root	>>sqrt(81) ans=9
nthroot(x,n)	Real nth root of a real number x. (If x is negative n must be an odd integer).	>>nthroot(8,3) ans=2
exp(x)	Exponential(e^x)	>>exp(5) ans=1.484131591025766e+02
abs(x)	Absolute value	>>abs(-24) ans=24
log(x)	Natural logarithm. Base e logarithm (ln)	>>log(1000) ans=6.907755278982137e+00
log10(x)	Base 10 logarithm	>>log10(1000) ans=3
factorial(x)	The factorial function x! (x must be a psitive integer)	>>factorial(5) ans=120

Trigonometric Math Functions

Function	Description	Examples
sin(x) sind(x)	Sine of angle x (x in radians) Sine of angle x (x in degrees)	>>sin(pi/6) ans=0.5000 >>sind(30) ans=0.5000
cos(x) cosd(x)	Cosine of angle x (x in radians) Cosine of angle x (x in degrees)	>>cos(0.5) ans=0.8776 >>cosd(30) ans=0.8660
tan(x) tand(x)	Tangent of angle x (x in radians) Tangent of angle x (x in degrees)	>>tan(pi/6) ans=0.5774 >>tand(45) ans=1
cotx) cotd(x)	Cotangent of angle x (x in radians) Cotangent of angle x (x in degrees)	>>cot(0.5) ans=1.8305 >>cotd(90) ans=1
asin(x) asind(x)	Inverse of sine of angle x (x in radians) Inverse of sine angle x (x in degrees)	>>asin(0.5) ans=0.5236 >>asin(0.8660) ans=59.9971
acos(x) acosd(x)	Inverse of cosine of angle x (x in radians) Inverse of cosine angle x (x in degrees)	>>acos(0.3) ans=1.2661 >>acosd(0.5) ans=60

Rounding Functions

Function	Description	Examples
round(x)	Round to the nearest integer	>>round(25/3) ans=8 >>round(4.49) ans=4 >>round(4.5) ans=5 >>round(4.9999) ans=5
fix(x)	Round towards zero. In other words, chops off the fraction part.	>>fix(17/5) ans=3 >>fix(4.49) ans=4 >>fix(4.9999) ans=4 >>fix(-4.6674) ans=-4
ceil(x)	Round towards positive infinity.	>>ceil(27/2) ans=14 >>ceil(2.1) ans=3 >>ceil(2.9) ans=3 >>ceil(-4.99) ans=-4
floor(x)	Round towards minus infinity	>>floor(-9/4) ans=-3 >>floor(2.1) ans=2 >>floor(2.99) ans=2
rem(x,y)	Returns the remainder after x is divided by y	>>rem(13,5) ans=3

Examples for Arithmetic operators

1) Calculate the following using MATLAB.

$$\frac{1}{2+3^2}+\frac{4}{5}*\frac{6}{7}$$

```
>>a=1/(2+3^2)
a=
    0.0909
>>b=4/5
b=
     0.8000
>>c=6/7
c=
    0.8571
>>a+b*c
ans=
       0.7766
```

2. Find the value of $y = e^{-a}\sin(x) + 10\sqrt{y}\ if\ a = 5, x = 2, y = 8.$

>>a=5; x=2;   y=8;

y=exp(-a)*sin(x)+10*sqrt(y)

y=

2.829039804533026e+013.

3. Compute $\sin\left(\frac{\pi}{4}\right), e^{10}$.

>>sin(pi/4)

ans=0.7071

>>exp(10)

ans=2.2026e+004

Reference:

1. Getting started with MATLAB, Rudra Pratap, Oxford University Press, 7th Edition, 2016.
2. https://www.tutorialspoint.com/matlab/matlab_data_import.htm

THANK YOU