MATLAB is a programming language/Tool developed by MathWorks. It started out as a matrix programming language where linear algebra programming was simple. It can be run both under interactive sessions and as a batch job.

MATLAB's Power of Computational Mathematics

MATLAB is used in every facet 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 (dependent)
* Algebraic Equations
* Non-linear Functions (independent)
* Statistics (calculation)
* Data Analysis (comparsion)
* 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 built-in graphics for visualizing data and tools for creating custom plots.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
* MATLAB's programming interface gives development tools for improving code quality maintainability and maximizing performance.
* 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

# MATLAB - Environment Setup

Local Environment Setup

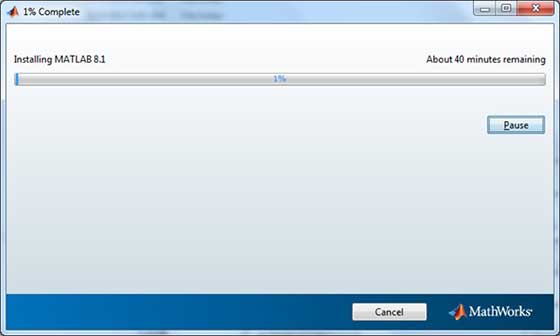
Setting up MATLAB environment is a matter of few clicks. The installer can be downloaded from [here](https://www.mathworks.com/downloads/web_downloads/).

https://www.mathworks.com/downloads/

MathWorks provides the licensed product, a trial version and a student version as well. You need to log into the site and wait a little for their approval.

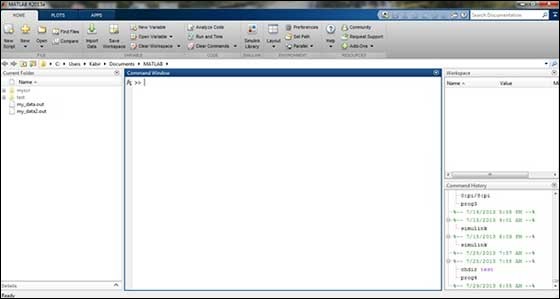
After downloading the installer the software can be installed through few clicks.





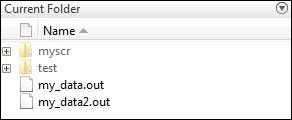
Understanding the MATLAB Environment

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 −

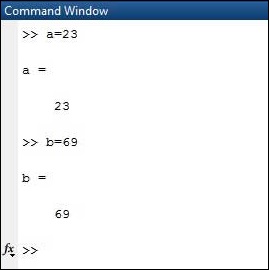


The desktop has the following panels −

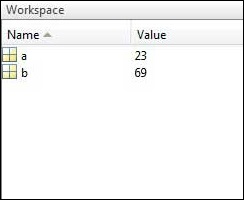
* **Current Folder** − This panel allows you to access the project folders and files.



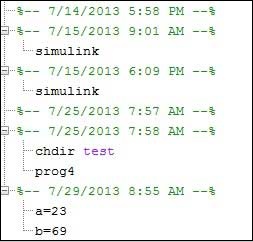
* **Command Window** − This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



* **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.



# Basic Syntax

5 + 5

And press ENTER

When you click the Execute button, or type Ctrl+E, MATLAB executes it immediately and the result returned is −

ans = 10

3 ^ 2 % 3 raised to the power of 2

sin(pi /2) % sine of angle 90o

7/0 % Divide by zero

## Use of Semicolon (;) in MATLAB

Semicolon (;) indicates end of statement. However, if you want to suppress and hide the MATLAB output for an expression, add a semicolon after the expression.

For example,

x = 3;

y = x + 5

## Adding Comments

The percent symbol (%) is used for indicating a comment line. For example,

x = 9 % assign the value 9 to x

You can also write a block of comments using the block comment operators % { and % }.

The MATLAB editor includes tools and context menu items to help you add, remove, or change the format of comments.

## Commonly used Operators and Special Characters

MATLAB supports the following commonly used operators and special characters −

|  |  |
| --- | --- |
| **Operator** | **Purpose** |
| **+** | Plus; addition operator. |
| **-** | Minus; subtraction operator. |
| **\*** | Scalar and matrix multiplication operator. |
| **.\*** | Array multiplication operator. |
| **^** | Scalar and matrix exponentiation operator. |
| **.^** | Array exponentiation operator. |
| **\** | Left-division operator. |
| **/** | Right-division operator. |
| **.\** | Array left-division operator. |
| **./** | Array right-division operator. |
| **:** | Colon; generates regularly spaced elements and represents an entire row or column. |
| **( )** | Parentheses; encloses function arguments and array indices; overrides precedence. |
| **[ ]** | Brackets; enclosures array elements. |
| **.** | Decimal point. |
| **…** | Ellipsis; line-continuation operator |
| **,** | Comma; separates statements and elements in a row |
| **;** | Semicolon; separates columns and suppresses display. |
| **%** | Percent sign; designates a comment and specifies formatting. |
| **\_** | Quote sign and transpose operator. |
| **.\_** | Nonconjugated transpose operator. |
| **=** | Assignment operator. |

## 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 √-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**.

Variable names can be of any length, however, MATLAB uses only first N characters, where N is given by the function **namelengthmax**.

## Saving Your Work

The **save** command is used for saving all the variables in the workspace, as a file with .mat extension, in the current directory.

For example,

save myfile %here myfile is name of the file

You can reload the file anytime later using the **load** command.

load myfile %open the existing code file

# MATLAB - Variables

## Multiple Assignments

You 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 following result −

c = 14

## I have forgotten the Variables!

The **who** command displays all the variable names you have used.

who

MATLAB will execute the above statement and return the following result −

Your variables are:

a ans b c

The **whos** command displays little more about the variables −

* Variables currently in memory
* Type of each variables
* Memory allocated to each variable
* Whether they are complex variables or not

whos

MATLAB will execute the above statement and return the following result −

Attr Name Size Bytes Class

==== ==== ==== ==== =====

a 1x1 8 double

ans 1x70 757 cell

b 1x1 8 double

c 1x1 8 double

Total is 73 elements using 781 bytes

The **clear** command deletes all (or the specified) variable(s) from the memory.

clear x % it will delete x, won't display anything

clear % it will delete all variables in the workspace

% peacefully and unobtrusively

format long

x = 7 + 10/3 + 5 ^ 1.2

format short

x = 7 + 10/3 + 5 ^ 1.2

format bank

format short e

format long e

format rat

Creating Vectors

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

* Row vectors ; or blank
* Column vectors ,

**Row vectors** are created by enclosing the set of elements in square brackets, using space or comma to delimit the elements.

r = [7 8 9 10 11]

r = [7 8 9 10 11];

t = [2, 3, 4, 5, 6];

res = r + t

**Column vectors** are created by enclosing the set of elements in square brackets, using semicolon(;) to delimit the elements.

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

## 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 demarcated by a semicolon. For example, let us create a 3-by-3 matrix as −

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

Commands for Managing a Session

MATLAB provides various commands for managing a session. The following table provides all such commands −

|  |  |
| --- | --- |
| **Command** | **Purpose** |
| Clc | Clears command window. |
| Clear | Removes variables from memory. |
| Exist | Checks for existence of file or variable. |
| global | Declares variables to be global. |
| help | Searches for a help topic. |
| lookfor | Searches help entries for a keyword. |
| quit | Stops MATLAB. |
| who | Lists current variables. |
| whos | Lists current variables (long display). |

Commands for Working with the System

MATLAB provides various useful commands for working with the system, like saving the current work in the workspace as a file and loading the file later.

It also provides various commands for other system-related activities like, displaying date, listing files in the directory, displaying current directory, etc.

The following table displays some commonly used system-related commands −

|  |  |
| --- | --- |
| **Command** | **Purpose** |
| cd | Changes current directory. |
| date | Displays current date. |
| delete | Deletes a file. |
| diary | Switches on/off diary file recording. |
| dir | Lists all files in current directory. |
| load | Loads workspace variables from a file. |
| path | Displays search path. |
| pwd | Displays current directory. |
| save | Saves workspace variables in a file. |
| type | Displays contents of a file. |
| what | Lists all MATLAB files in the current directory. |
| wklread | Reads .wk1 spreadsheet file. |

Input and Output Commands

MATLAB provides the following input and output related commands −

|  |  |
| --- | --- |
| **Command** | **Purpose** |
| disp | Displays contents of an array or string. |
| fscanf | Read formatted data from a file. |
| format | Controls screen-display format. |
| fprintf | Performs formatted writes to screen or file. |
| input | Displays prompts and waits for input. |
| ; | Suppresses screen printing. |

The **fscanf** and **fprintf** commands behave like C scanf and printf functions. They support the following format codes −

|  |  |
| --- | --- |
| **Format Code** | **Purpose** |
| **%s** | Format as a string. |
| **%d** | Format as an integer. |
| **%f** | Format as a floating point value. |
| **%e** | Format as a floating point value in scientific notation. |
| **%g** | Format in the most compact form: %f or %e. |
| **\n** | Insert a new line in the output string. |
| **\t** | Insert a tab in the output string. |

The format function has the following forms used for numeric display −

|  |  |
| --- | --- |
| **Format Function** | **Display up to** |
| format short | Four decimal digits (default). |
| format long | 16 decimal digits. |
| format short e | Five digits plus exponent. |
| format long e | 16 digits plus exponents. |
| format bank | Two decimal digits. |
| format + | Positive, negative, or zero. |
| format rat | Rational approximation. |
| format compact | Suppresses some line feeds. |
| format loose | Resets to less compact display mode. |

Vector, Matrix and Array Commands

The following table shows various commands used for working with arrays, matrices and vectors −

|  |  |
| --- | --- |
| **Command** | **Purpose** |
| cat | Concatenates arrays. |
| find | Finds indices of nonzero elements. |
| length | Computes number of elements. |
| linspace | Creates regularly spaced vector. |
| logspace | Creates logarithmically spaced vector. |
| max | Returns largest element. |
| min | Returns smallest element. |
| prod | Product of each column. |
| reshape | Changes size. |
| size | Computes array size. |
| sort | Sorts each column. |
| sum | Sums each column. |
| eye | Creates an identity matrix. |
| ones | Creates an array of ones. |
| zeros | Creates an array of zeros. |
| cross | Computes matrix cross products. |
| dot | Computes matrix dot products. |
| det | Computes determinant of an array. |
| inv | Computes inverse of a matrix. |
| pinv | Computes pseudoinverse of a matrix. |
| rank | Computes rank of a matrix. |
| rref | Computes reduced row echelon form. |
| cell | Creates cell array. |
| celldisp | Displays cell array. |
| cellplot | Displays graphical representation of cell array. |
| num2cell | Converts numeric array to cell array. |
| deal | Matches input and output lists. |
| iscell | Identifies cell array. |

Plotting Commands

MATLAB provides numerous commands for plotting graphs. The following table shows some of the commonly used commands for plotting −

|  |  |
| --- | --- |
| **Command** | **Purpose** |
| axis | Sets axis limits. |
| fplot | Intelligent plotting of functions. |
| grid | Displays gridlines. |
| plot | Generates xy plot. |
| print | Prints plot or saves plot to a file. |
| title | Puts text at top of plot. |
| xlabel | Adds text label to x-axis. |
| ylabel | Adds text label to y-axis. |
| axes | Creates axes objects. |
| close | Closes the current plot. |
| close all | Closes all plots. |
| figure | Opens a new figure window. |
| gtext | Enables label placement by mouse. |
| hold | Freezes current plot. |
| legend | Legend placement by mouse. |
| refresh | Redraws current figure window. |
| set | Specifies properties of objects such as axes. |
| subplot | Creates plots in subwindows. |
| text | Places string in figure. |
| bar | Creates bar chart. |
| loglog | Creates log-log plot. |
| polar | Creates polar plot. |
| semilogx | Creates semilog plot. (logarithmic abscissa). |
| semilogy | Creates semilog plot. (logarithmic ordinate). |
| stairs | Creates stairs plot. |
| stem | Creates stem plot. |