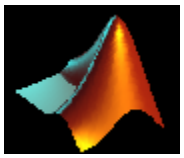
General MATLAB Information [1]

Introduction to MATLAB



Once you initiate the MATLAB software, you will see the MATLAB logo appear and then the MATLAB prompt **>>**. The prompt **>>** indicates that MATLAB is awaiting a command.

MATLAB is ***case sensitive***; all built-in MATLAB commands are LOWER CASE.

There are certain MATLAB features you should be aware of before you begin working with MATLAB.

- *Variables, Expressions and Statements.*

MATLAB statements typically take one of two forms:
variable = expression or **expression**

Examples val = cos(pi/7) exp(2.13)

All variable (and function) names consist of a letter followed by any number of numbers, letters and underscores. MATLAB is case sensitive and only the first 19 characters of any name are significant.

The equal sign **=** is called the **assignment operator**.

Expressions are composed from operators, function calls and variable names. Pressing ENTER normally signifies the end of a statement, causing MATLAB to interpret the command and print its result. If the last character of a statement is a; (semicolon), however, display of the result is suppressed. This feature may be especially useful when the result of a computation is a large matrix. Finally, several statements separated by commas may be placed on a single line.

When an expression is not explicitly assigned to a variable with the **assignment operator (=)**, MATLAB automatically stores the result in the special variable **ans**.

During a MATLAB session you may forget the names of variables stored in your *workspace*. The command **who** lists the names of all your variables. If you want to know their size as well, use the command **whos**. By default MATLAB stores all variables until the session is terminated. To remove a variable from the workspace use the command **clear var_name**. WARNING: **clear** with no arguments removes all variables from the workspace.

- *Numbers.*
  MATLAB uses conventional decimal notation with an optional decimal point and minus sign for negative values. Scientific notation uses the letter e to specify a power of ten scale factor. Some valid numbers are: 34, -7, 0.0017, 6.3457, 2.718281828459046, 3.060196847852814e+002.

  All numbers are stored internally using the “long” format specified by the IEEE floating point standard. Floating point numbers have about 16 significant decimal digits and a finite range of about 10^{-308} to 10^{308}.

- *Starting execution of a command.*

After you have typed a command name and any arguments or data required, you must press **ENTER** before it will begin to execute.

- *Search Path.*

To determine how to execute commands MATLAB uses a search path to find m-files and other data sets. Any file you want to execute must reside in the current directory or in a folder that is on the search path. By default, the files supplied with MATLAB and MathWorks toolboxes are included in the search path.

To see which directories/folders are on the search path type the command **path**. To change the search path select Set Path from the File menu in the desktop, and use the Set Path dialog box. The command **addpath** can be used to add directories to the path, and command **rmpath** can be used to remove directories from the path,

- *Getting help.*

If you know the name of command typing **help** followed by the name displays information about the command.

```
>> help sin
 SIN    Sine of argument in radians.
    SIN(X) is the sine of the elements of X.
```

```
>> help sqrt
 SQRT   Square root.
    SQRT(X) is the square root of the elements of X. Complex
    results are produced if X is not positive.
```

- *The command stack.*

As you enter commands, MATLAB saves a number of the most recent commands in a stack. Previous commands saved on the stack can be recalled using the **up arrow** key. The number of commands saved on the stack varies depending on the length of the commands and other factors.

- *Editing commands.*

If you make an error or mistype something in a command, you can use the **left arrow** and **right arrow** keys to position the cursor for corrections. The **home** key moves the cursor to the beginning of a command, and the **end** key moves the cursor to the end. The **backspace** and **delete** keys can be used to remove characters from a command line. The **insert** key is used to initiate the insertion of characters. Pressing the insert key a second time exits the insert mode. If MATLAB recognizes an **error** after you have pressed ENTER, then MATLAB responds with a beep and a message that helps define the error. You can **recall the command line** using the up arrow key in order to edit the line. To **delete an entire command line** press ESC, escape.

- *Continuing commands.*

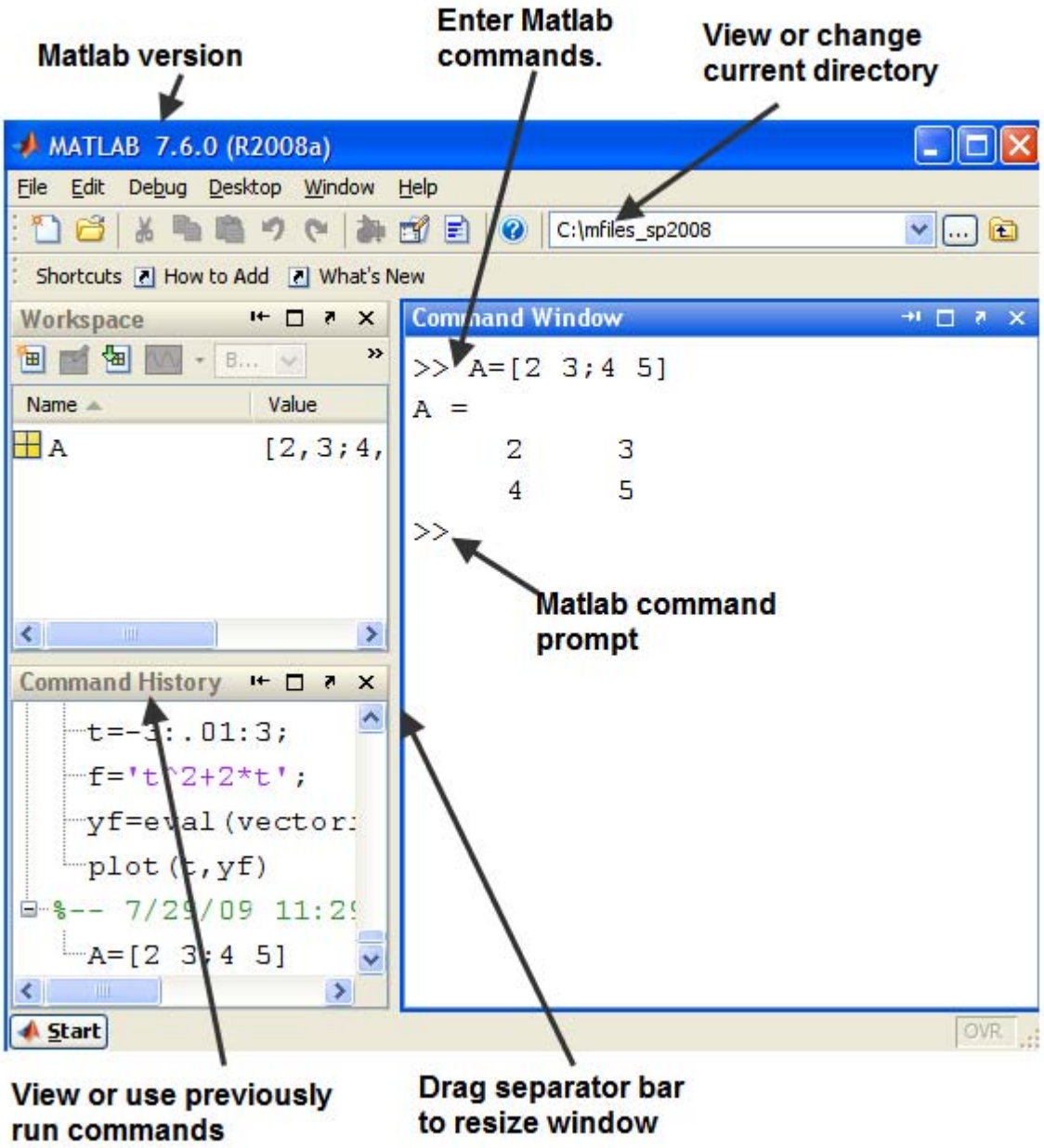
MATLAB commands that do not fit on a single line can be continued to the next line using an ellipsis, which is three consecutive periods, followed by ENTER.

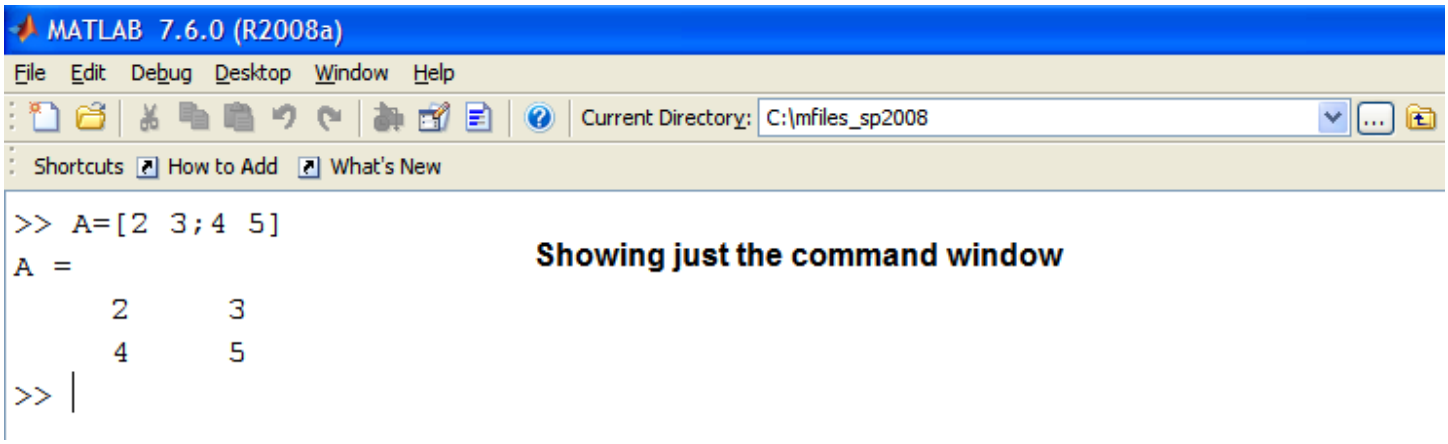
- *Stopping a command.*

To stop execution of a MATLAB command, press **Ctrl** and **C** simultaneously, then press ENTER. Sometimes this sequence must be repeated.

- *Quitting.*

To quit MATLAB, type **exit** or **quit** followed by pressing ENTER.





You can change the way the desktop looks. Click on the Desktop drop down menu.

General:

To enter a matrix: > type the entries between square brackets, [.....]
> enter it by rows with elements separated by a space or comma
> rows are terminated by a semicolon

command **A= [1 2 3;4 5 6;7 8 9]** displays A =

1 2 3
4 5 6
7 8 9

Suppressing Displays: When you place a semicolon at the end of a command, the command will be executed but the result will not be displayed on the screen. This is very useful when you are creating big matrices or to not reveal intermediate steps of a calculation. Try typing the command

w = linspace(1,20,100); ← Note the semicolon

to create a row vector with equally spaced entries. The contents of **w** will not be displayed. To see the contents just type **w**.

To enter a string: type the contents of the string between **single quotes** ' ... '

command **st = 'Hi there.'** displays st =

Hi there.

Strings are not commands; they information like messages and formulas.

What you see on the screen: MATLAB has various **display formats**. That is, ways to show things on the screen. All numerical values stored by MATLAB have about 16 decimal places. However, we do not always want to see all the decimals places. That is why MATLAB has display formats.

The default format is format short.
Format short shows 4 places behind the decimal point for reasonable sized numbers.

Examples: pi is shown as 3.1416 in format short
2/3 is shown as 0.6667 in format short
1/8 is shown as 0.1250 in format short

(Format short rounds to obtain the last digit which is displayed. **Beware,** there may be many more decimal places than those shown.)

Examples of other display formats: (**e formats** display a number and an exponent which is a power of ten ; 1.2345e+002 means $1.2345*10^2$)

FORMAT SHORT Scaled fixed point format with 5 digits.
FORMAT LONG Scaled fixed point format with 15 digits.
FORMAT SHORT E Floating point format with 5 digits.
FORMAT LONG E Floating point format with 15 digits.
FORMAT SHORT G Best of fixed or floating point format with 5 digits.
FORMAT LONG G Best of fixed or floating point format with 15 digits.
FORMAT RAT Approximation by ratio of small integers.

format short e	shows pi as	3.1416e+000
format long	shows pi as	3.14159265358979
format long e	shows pi as	3.141592653589793e+000
format rat	shows pi as	355/113

Note: 355/113 is only an approximation to pi; computing 355/113 in format long gives 3.14159292035398

Use command help format for the description of other display formats.

Special case: A value which is exactly zero will be displayed as a single zero. If you see 0.0000, the value is not exactly zero. Change to a long format and re-display it.

MATLAB also works with complex numbers. Use i as a suffix on the imaginary part. For example the complex number 5 – 8i is entered at a MATLAB prompt as >> z = 5-8i and will be displayed as z = 5.0000 - 8.0000i in format short. Changing the format can alter the display.

Illustrations of ALGEBRA Expressions and their MATLAB form.

Standard Algebra Form	MATLAB Representation
$3x^2 - 5x + 1$	3*x^2-5*x+1 **multiplication must be indicated using * and exponents require an ^**
$\frac{2x-3}{4-7x}$	(2*x-3)/(4-7*x) **the numerator and denominator must be enclosed in parentheses if they are more than 1 term**
$\sqrt{x^2+1}$	sqrt(x^2+1) or (x^2+1)^.5 or (x^2+1)^(1/2) **using sqrt is preferred; fractional exponents must be enclosed in parentheses**
e^{-x}	exp(-x)
ln(x)	log(x) **the natural log function is denoted log; log base 10 is denoted log10**
$\cos^2(x - \pi)$	cos(x-pi)^2 **constant π is denoted pi; all arguments of trig. functions are considered radians**
$\sin(x)\, e^{-2.3x}$	sin(x)*exp(-2.3*x) **note that the exponent is in parentheses and that we must use * to indicate multiplication**
|tan(x)|	abs(tan(x)) **the absolute value is denoted by abs; parentheses must be used**

Operators, Elementary Functions, & General Purpose Commands

Arithmetic operators.

plus	+
minus	-
times	*
power	^
divide	\

Relational operators.

Equal	==
Not equal	~=
Less than	<
Greater than	>
Less than or equal	<=
Greater than or equal	>=

Logical operators.

and	&
or	|
not	~

++

Trigonometric & Hyperbolic

sin	- Sine.
sinh	- Hyperbolic sine.
asin	- Inverse sine.
asinh	- Inverse hyperbolic sine.
cos	- Cosine.
cosh	- Hyperbolic cosine.
acos	- Inverse cosine.
acosh	- Inverse hyperbolic cosine.
tan	- Tangent.
tanh	- Hyperbolic tangent.
atan	- Inverse tangent.
atan2	- Four quadrant inverse tangent.
atanh	- Inverse hyperbolic tangent.
sec	- Secant.
sech	- Hyperbolic secant.
asec	- Inverse secant.
asech	- Inverse hyperbolic secant.
csc	- Cosecant.
csch	- Hyperbolic cosecant.
acsc	- Inverse cosecant.
acsch	- Inverse hyperbolic cosecant.
cot	- Cotangent.
coth	- Hyperbolic cotangent.
acot	- Inverse cotangent.
acoth	- Inverse hyperbolic cotangent.

Exponential

exp	- Exponential.
log	- Natural logarithm.
log10	- Common (base 10) logarithm.
log2	- Base 2 logarithm and dissect floating point number.
pow2	- Base 2 power and scale floating point number.
sqrt	- Square root.
nextpow2	- Next higher power of 2.

Complex

abs	- Absolute value.
angle	- Phase angle.
complex	- Construct complex data from real and imaginary parts.
conj	- Complex conjugate.
imag	- Complex imaginary part.
real	- Complex real part.

Calculus Operators

diff -Difference and approximate derivative when working on a vector.
diff -Differentiate when working on a string or symbolic expression; will do higher derivatives & partial derivatives.
int -Integrate; indefinite or definite depending upon the arguments.
taylor -Taylor series expansion.

Basic Plotting Tools

plot -Plots a set of ordered pairs.
plot3 -Plots a set of ordered triples.
ezplot -Easy to use function plotter.

++
+

General Purpose Commands

home - Moves the cursor to the upper left corner of the Command Window and clears the visible portion of the window. Use the scroll bar to see what was on the screen previously.
clc - Clear command window; scrollbar not available.
demo - Run demonstrations.
who - List current variables.
whos - List current variables, long form.
clear - Clear variables and functions from memory.

load - Load workspace variables from disk.
save - Save workspace variables to disk. (Use help save for more information.)
saveas - Save Figure or model to desired output format.
quit - Quit MATLAB session.
exit - Exit from MATLAB.
what - List MATLAB-specific files in directory.
type - List M-file.
which - Locate functions and files.
path - Get/set search path.
addpath - Add directory to search path.
rmpath - Remove directory from search path.
diary - Save text of MATLAB session.

Diary File: At times you will need to record the results of your MATLAB session as part of an assignment. Have a flash drive or floppy disk ready to use so you can save your MATLAB work. Change the **Current Directory field** in the desktop toolbar to **the appropriate drive letter and possibly** a particular directory.

Now type command **diary filename.txt** followed by the Enter key. Each computation (but not graphics) you make in MATLAB will be saved in your directory in a text file named filename.txt. You can then edit this file using your favorite text editor. When you have finished your MATLAB session you can turn off the recording by typing **diary off** at the MATLAB prompt . If you want to stop your MATLAB session for any reason, you can reopen the diary file the next time you start MATLAB. If you use the same file name, the results of your new MATLAB session will be written at the end of the old diary file. You may want to use different names for each session on an assignment, and then merge the files.

Function & Plotting Practice:

1. Type the following commands in MATLAB. Move the mouse pointer onto the plot. Notice the symbol is a cross-hair. Position the cross-hair to estimate the 6 local max-min points shown. Click the left mouse button to record a point. Compare the values you obtain with those of other students in the class.

f='x*(x-1)*sin(x)'
ezplot(f)
[x,y]=ginput(6);[x y]

Record your estimates below:

x	y

2. Enter the string **g =sym('exp(sin(x))')**. Compute its second derivative from the command **g2 = diff(g,2)** . Graph the second derivative over the interval [0.7,1.5]. Use command **ezplot(g2,[0.7,1.5])**. Type **[x y]=ginput(2);** Then using the mouse click on the curve to estimate the largest and smallest value on the picture displayed. To show these points type **[x y]**

What is an upper bound on the absolute value of **g2** ? _____

3. Find the max of the absolute value of **f(x)** over [2, 4] where **f(x) = 2xcos(2x) - (x-2)^2**. Explain your procedure. State pertinent MATLAB commands.

4. Labeling graphs; changing axes; and other interesting things. Enter the following commands.

close all ← closes all existing graphs
clear ← erases all data currently defined

t=0:.01:2*pi+.2; ← explain what these two commands do
x=cos(t);y=3*sin(t);

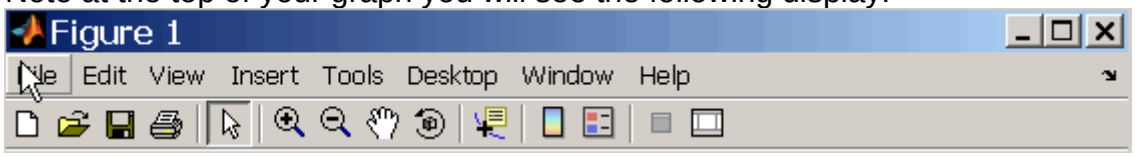
plot(x,y,'-r','linewidth',2) ← conjecture what this command does

axis([-3 3 -3 3]) ← explain what these commands do
grid on
addaxes

title('MY ELLIPSE','color','b','fontsize',14) ← putting some labels on
xlabel('X-axis')
ylabel('Y-axis')

If your graph is not visible, type command **figure(gcf)** .

Note at the top of your graph you will see the following display:



Click on **Insert** then Text Box in the drop down menu; move your mouse to white space on the graph then click and drag to draw a box. Now type your name in the resulting box. Click the mouse outside the box with your name. Now use your mouse to drag your name inside the ellipse. Click outside the box containing your name to affix it at that position.

Experiment with other icons on the tool bar.

Now let's superimpose another graph on this figure. Type the following commands.

hold on ← "freezes" the current graph so it is not erased when the next plot command is executed

```
t=-3:.01:3;
f='t^2+2*t';
yf=eval(vectorize(f));
plot(t,yf)
```

5. In approximating the solution of a system of differential equations there will be an independent variable t and two dependent variables $x_1(t)$ and $x_2(t)$. A table will be returned from the computing algorithm which will be a $3 \times n$ matrix. The first row will contain values of the independent variable t, the second row values of x_1 and the third row values of x_2. We will want to plot t vs x_1 and t vs x_2, and maybe even x_1 vs x_2. This exercise illustrates techniques for making such plots.

load lab1ex8data ← loading the data set; its name is data
data ← viewing the data set

plot(data(1,:),data(2,:),'-k') ← plot t vs x_1 with black line segments

plot(data(1,:),data(3,:),'-r'), figure(gcf) ← plot t vs x_2 with red line segments; previous plot is overlaid & new plot is shown
hold on
plot(data(1,:),data(2,:),'-k'), figure(gcf) ← plot t vs x_1 on the existing t vs x_2 graph & figure displayed

figure ← create a new figure for the next plot

plot(data(2,:),data(3,:),'-b') ← plot x_1 vs x_2 in blue

close all ← closes all existing figures