# نتيجة بحث الصور عن ‪university of bahrain logo‬‏Department of Computer Engineering College of Information Technology University of Bahrain

ITCE417: Mobile and Wireless Networking Lab 1: Introduction to MATLAB

MATLAB is a high-level programming language that has been used extensively to solve complex engineering problems. The language itself bears some similarities with ANSI C and FORTRAN. MATLAB works with three types of windows on your computer screen. These are the Command window, the Figure window and the Editor window. The Figure window only pops up whenever you plot something. The Editor window is used for writing and editing MATLAB programs (called M-files) and can be invoked in Windows from the pull-down menu after selecting File | New | M-file. In UNIX, the Editor window pops up when you type in the command window: edit filename (‘filename’ is the name of the file you want to create).

The command window is the main window in which you communicate with the MATLAB interpreter. The MATLAB interpreter displays a command >> indicating that it is ready to accept commands from you.

* View the MATLAB introduction by typing intro

>> intro

At the MATLAB prompt. This brief introduction will demonstrate some basic MATLAB commands

* Explore MATLAB’s help capability by trying the following:

>> help

>> help plot

Say you want to evaluate the expression a3+√bd + 4c, where a=1.2, b=2.3, c=4.5 and d=4. Then in the command window type:

>> a=1.2

>> b=2.3

>> c=4.5

>> d=4

>> a^3+sqrt(b\*d)-4\*c Ans =

-13.2388

Note the semicolon after each variable assignment. If you omit the semicolon, then MATLAB echoes back on the screen the variable value

# Arithmetic Operations

There are four different arithmetic operators:

+ Addition

- Subtraction

/ Division

\* multiplication

There are also three other operators that operate on an element by element basis:

.\* multiplication of two vectors, element by element

./ division of two vectors, element-wise

.^ raise all the elements of a vector to a power

Suppose that we have the vectors x = [x1, x2, ..., xn] and y = [y1, y2, ..., yn]. Then x. ∗ y = [x1∗y1, x2∗y2, ..., xn∗yn]

x./y = [x1/y1, x2/y2, ..., xn/yn ]

x.ˆp = [xp^1, xp^2, ..., x^p]

The arithmetic operators + and - can be used to add or subtract matrices, scalars or vectors. By vectors we mean one-dimensional arrays and by matrices we mean multi-dimensional arrays. This terminology of vectors and matrices comes from Linear Algebra.

Example:

>> X=[1,3,4]

>> Y=[4,5,6]

>> X+Y

Ans

= 5 8 10

# Array Indexing

In MATLAB, all arrays (vectors) are indexed starting with 1, i.e., y(1) is the first element of the array y. Note that the arrays are indexed using parenthesis (.) and not square brackets [.] as in C/C++. To create an array having as elements the integers 1 through 6, just enter:

>> x=[1,2,3,4,5,6]

Alternatively, you can use the : notation,

>> x=1:6

The : notation above creates a vector starting from 1 to 6, in steps of 1. If you want to create a vector from 1 to 6 in steps of say 2, then type:

>> x=1:2:6

Ans

= 1 3 5

Try the following code:

>> ii=2:4:17

>> jj=20:-2:0

>> ii=2:(1/10):4

Extracting or inserting numbers in a vector can be done very easily. To concatenate an array, you can use the [ ] operator, as shown in the example below:

>> x=[1:3 4 6 100:110]

To access a subset of the array, try the following:

>> x(3:7)

>> length(x) % gives the size of the array or vector

# Allocating Memory

You can allocate memory for one-dimensional arrays (vectors) using the zeros command. The following command allocates memory for a 100-dimensional array:

>> Y=zeros(100,1);

>> Y(30)

Ans=

= 0

Similarly, you can allocate memory for two-dimensional arrays (matrices). The command

>> Y=zeros(4,5)

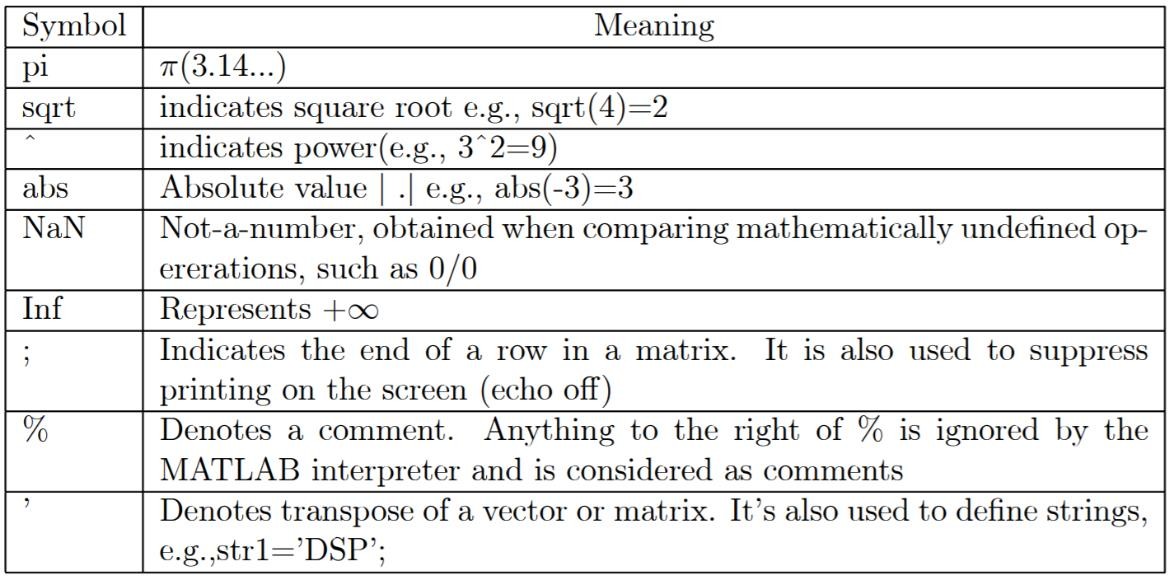
defines a 4 by 5 matrix. Similar to the zeros command, you can use the command ones to define a vector containing all ones,

>> Y=ones(1,5) Ans=

11111

# Special characters and functions

Some common special characters used in MATLAB are given below:



Some special functions are given below:

length(x) - gives the dimension of the array x find - Finds indices of nonzero elements.

Examples :

>> x=1:10;

>> length(x) ans =

10

The function find returns the indices of the vector X that are non-zero. For example, I = find(X>100), finds all the indices of X when X is greater than 100. So for the above example:

>> find(x> 4) ans =

5 6 7 8 9 10

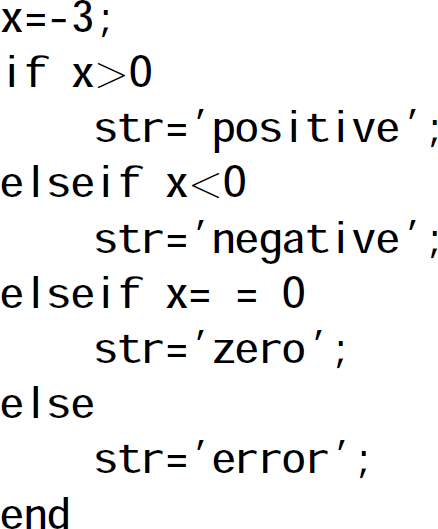
# Control Flow

MATLAB has the following flow control constructs:

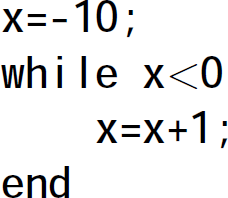
* if statements
* switch statements
* for loops
* while loops

The if, for, switch and while statements need to terminate with an end statement. Examples:

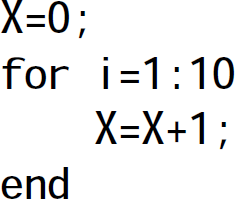
IF:



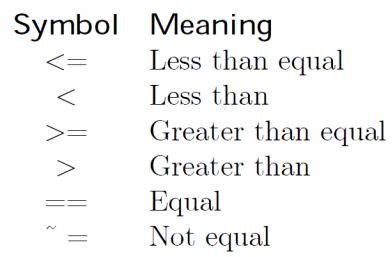
WHILE:



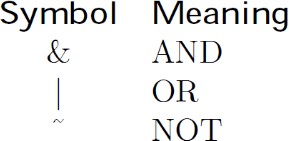
FOR loop:



MATLAB supports the following relational and logical operators: Relational Operators



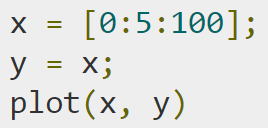
Logical Operators

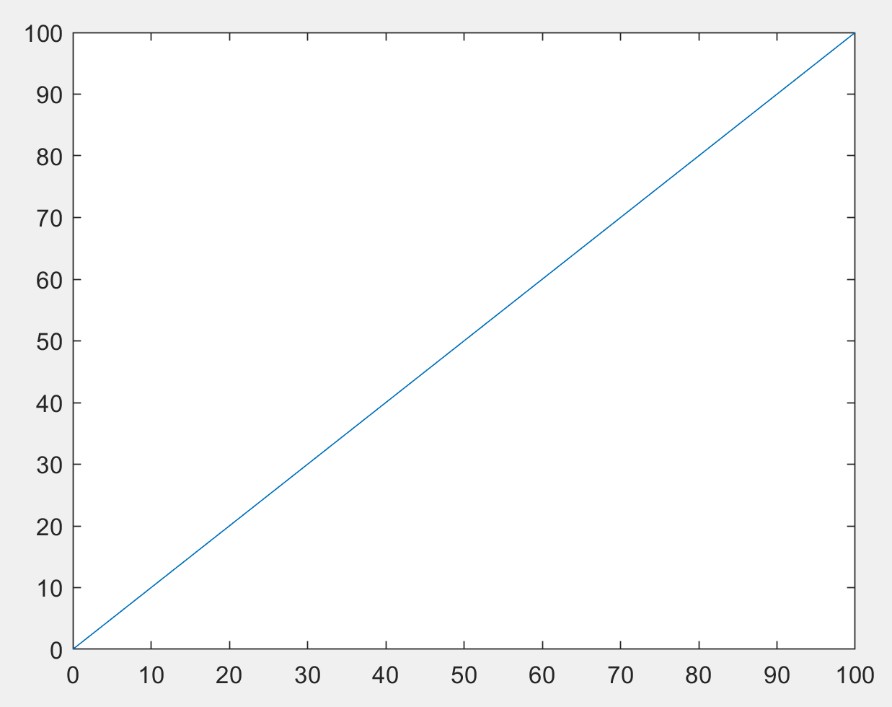


# Plotting

The plot function has different forms, depending on the input arguments.

* If y is a vector, plot(y) produces a piecewise linear graph of the elements of y versus the index of the elements of y.
* If you specify two vectors as arguments, plot(x,y) produces a graph of y versus x.

Example:

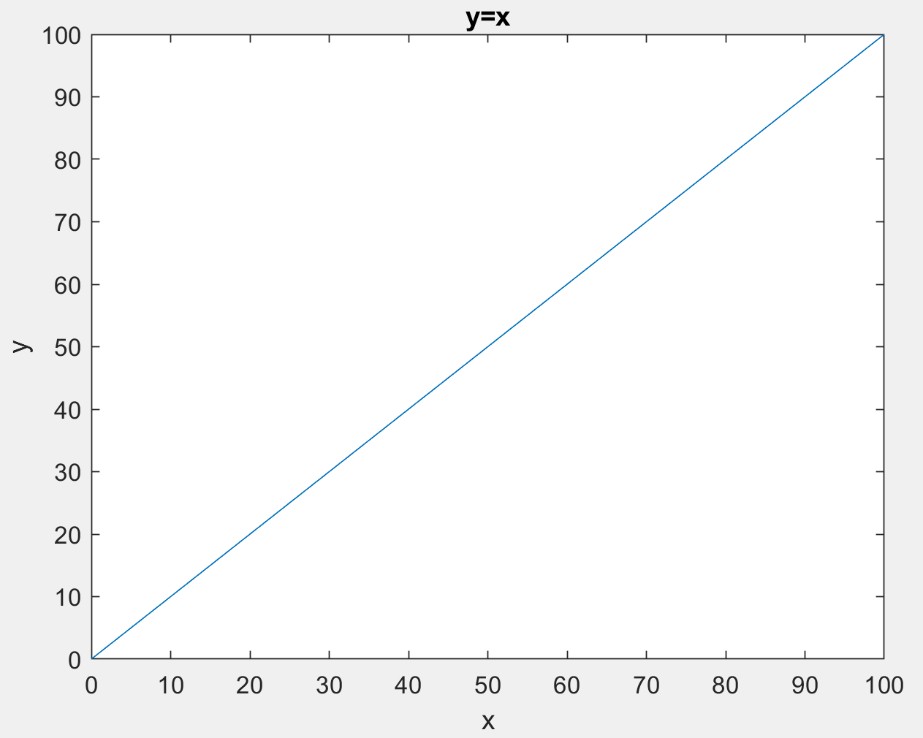


You can also add axis labels and a title as follows:

xlabel(‘x’)

ylabel(‘y’)

title(‘y=x’)



Saving figures

Save a figure by selecting **Save** from the **File** menu. This writes the figure to a file, including property data, its menus, uicontrols, and all annotations (i.e., the entire window). If you have not saved the figure before, the **Save As** dialog displays. This dialog box provides options to save the figure as a.fig file or export it to a graphics format.

If you have previously saved the figure, using **Save** again saves the figure “silently,” without displaying the **Save As** dialog.

To save a figure using a standard graphics format for use with other applications, such as TIFF or JPG, select **Save As** (or **Export Setup**, if you want additional control) from the **File** menu.

# Programming in MATLAB (M-File)

MATLAB programming is done using M-files, i.e., files that have the extension .m. These files are created using a text editor. To open the text editor, go to the File pull-down menu, choose New, then M-file. After you type in the program, save it, and then call it from the command window to execute it.

# Report Tasks

1. write a MATLAB function to compute the average (mean) of a vector x, then call and execute the function from the command window

* Take a screenshot of your code.

Text

Description automatically generated

* Take three screenshots for the output after calling the function with different values of x

We wrote a simple function to generate a rand\_vector with 10 element between values 1 and 50

Text

Description automatically generated

For loop to call 2 function and then prinout the result

A screenshot of a computer

Description automatically generated with medium confidence

.

1. https://www.mathworks.com/help/examples/matlab/win64/CreatingAPlotExample_01.pngPlot the graph of the cosine function using values of x ranging from 0 to and label the x- and y- accesses.

* Add labels for the x- and y- accesses.
* Add a title for your graph and call it ‘Cosine Wave Plot’.
* Add a grid.
* Add a legend and call your result ‘Cosine Wave’.
* Take a screenshot of your code and the cosine graph.

Chart, line chart

Description automatically generated

Text

Description automatically generated