Introduction to MATLAB

Some of the basic functions available in MATLAB is listed in the table below. Use this table as a starting point and mess around with codes to learn more about MATLAB.

Basic Functions

Function	**Sample**	**Description**
clc	clc	clears all input and output from the Command Window display, giving "clean screen." After using clc, the scroll bar cannot be used to see the history of functions, but still the up arrow can be used to recall statements from the command history.
close	close all	close deletes the current figure or the specified figure(s). It optionally returns the status of the close operation.
clear	clear	Clears the workspace

zeros	B = zeros(n)
	B = zeros(m,n)
		Creates an n-by-n matrix of zeros.
		Creates an m-by-n matrix of zeros.

ones	B = ones(n)
	B = ones(m,n)
		Creates an n-by-n matrix of ones.
		Creates an m-by-n matrix of ones.

disp	disp(X)	displays an array, without printing the array name. If X contains a text string, the string is displayed.
length	length(n)	Returns the length of vector n
rand	rand(m,n)	Returns a randomly generated m * n matrix
figure	figure	figure creates figure graphics objects. Figure objects are the individual windows on the screen in which MATLAB displays graphical output.
subplot	subplot(m,n,p) or subplot(mnp) or subplot mnp	breaks the figure window into an m-by-n matrix of small axes, selects the pth axes object for the current plot, and returns the axes handle. The axes are counted along the top row of the figure window, then the second row, etc. subplot(2,1,1), plot(income) subplot(2,1,2), plot(outgo) plots income on the top half of the window and outgo on the bottom half.
title	title('string')	Each axes graphics object can have one title. The title is located at the top and in the center of the axes.
xlabel, ylabel, zlabel	xlabel('time')	Each axes graphics object can have one label for the x-, y-, and z-axis. The label appears beneath its respective axis in a two-dimensional plot and to the side or beneath the axis in a three-dimensional plot.
Input	Input('what is the input?', 's')	Displays "what is the input?" as a prompt on the screen, waits for input from the keyboard, and returns the value entered in user_entry. returns the entered string as a text variable rather than as a variable name or numerical value.

Mathematical Conversions

Function	Sample	Description
abs	abs(X)	returns an array Y such that each element of Y is the absolute value of the corresponding element of X
mod	mod(X,Y)	returns X - n.*Y where n = floor(X./Y). If Y is not an integer and the quotient X./Y is within round off error of an integer, then n is that integer. The inputs X and Y must be real arrays of the same size, or real scalars.
sqrt	sqrt(X)	returns the square root of each element of the array X. For the elements of X that are negative or complex, sqrt(X) produces complex results.
ceil	ceil(A)	rounds the elements of A to the nearest integers greater than or equal to A. For complex A, the imaginary and real parts are rounded independently.
min	min(A)	returns the smallest elements along different dimensions of an array. If A is a vector, min(A) returns the smallest element in A.If A is a matrix, min(A) treats the columns of A as vectors, returning a row vector containing the minimum element from each column. If A is a multidimensional array, min operates along the first nonsingleton dimension.
max	max(A)	Returns the largest elements along different dimensions of an array. If A is a vector, max(A) returns the largest element in A.If A is a matrix, max(A) treats the columns of A as vectors, returning a row vector containing the maximum element from each column. If A is a multidimensional array, max(A) treats the values along the first non-singleton dimension as vectors, returning the maximum value of each vector.
angle	angle(Z)	Returns the phase angles, in radians, for each element of complex array Z. The angles lie between + π and – π.
^	F= a^b	Returns a to the power b. if syms enabled
dec2bin	dec2bin(n)	Converts decimal n to binary

Mathematical functions and functions related to signals and systems

Function	**Sample**	**Description**
sin	sin(45)	Sine of argument in radians
cos	cos(60)	Cosine of argument in radians
tan	tan(100)	Tangent of argument in radians
exp	exp(X)	The exp function is an elementary function that operates element-wise on arrays. Its domain includes complex numbers. Y = exp(X) returns the exponential for each element of X.
log	log(X)	returns the natural logarithm of the elements of X. For complex or negative z, where z = x +y*i , the complex logarithm is returned.
conv	w = conv(u,v)	convolves vectors u and v. Algebraically, convolution is the same operation as multiplying the polynomials whose coefficients are the elements of u and v.
fourier	F = fourier(f)	Returns the Fourier transform of the symbolic scalar f with default independent variable x. The default return is a function of w. The Fourier transform is applied to a function of x and returns a function of w.
fft	Y = fft(X)	Returns the discrete Fourier transform (DFT) of vector X, computed with a fast Fourier transform (FFT) algorithm. If X is a matrix, fft returns the Fourier transform of each column of the matrix. If X is a multidimensional array, fft operates on the first nonsingleton dimension.
	Y = fft(X,n)	Returns the n-point DFT. If the length of X is less than n, X is padded with trailing zeros to length n. If the length of X is greater than n, the sequence X is truncated. When X is a matrix, the length of the columns are adjusted in the same manner.

Lab #1 - Part I
The purpose of this document is to provide you with some basic commands on how to build arrays, do simple manipulations on them and create 2D plots in MATLAB.

1. Construct the row vector a={3 1 5 7 9 2 6}

```
a = [3 1 5 7 9 2 6] %this sign tells MATLAB to ignore rest of this line
```

2. Extract the 4th element of a:

```
a(4)      % takes the 4th element of a and assigns the result to the
          % array ans.
c = a(4)  % takes the 4th element of a and assigns the result to the
          % array named c
```

3. Change the 4th element of a to 8:

```
a(4)=8
```

4. Delete the 5th element of a:

```
a(5)=[]  %[] is the empty matrix whose size is 0x0
```

5. Construct the column vector :

```
b = [2; 5; 7]      % or

b = [2 5 7]'       %(Note: ' is the transpose operator!)
```

6. Construct the matrices :

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

7. Extract the element from matrix :

```
A(3,2)
```

8. Extract the following submatrix from A:

```
A(1:2,2:3)
```

9. Extract the second row of A:

```
A(2,:)             % ":" means all elements of the array
```

10. Append b to the right of A:

```
[A b]              %the sizes of the two arrays must be compatible
```

11. Delete the last column of A:

```
A(:,3) = []
```

12. Construct the symmetric matrix

```
B = [1 5 6; 5 2 4; 6 4 1]
```

13. Compute A+B, A-B, A*B:

```
A+B             %(A and B should be of the same size)

A-B             %(A and B should be of the same size)

A*B             %(the no. of columns of A should be equal to the no. of rows of B)
```

14. Obtain the element-wise multiplication of A and B matrices:

```
A.*B % A and B should have the same size
```

15. Compute the solution of the system of linear equations Ax=b (i.e., solve for x):

```
x = inv(A)*b    %or
x = A\b         %backslash operator \ is used to solve linear systems of equations
```

16. Construct an array of integers from 0 to 100 (inclusive):

```
r = 0:100;      % a ";" at the end of the line prevents the output
                % to be printed on the command window.
```

17. Construct an array of even numbers from 0 to 100 (inclusive):

```
s= 0:2:100;     %format is first:step:last. I step is omitted, it
                %is equal to 1 by default.
```

18. Construct an array that is bounded by [0 2π] and has elements that are spaced with increments $\pi/100$

```
t = 0:pi/100:2*pi;
```

19. Plot t versus f(t) = t*cos(t)-t:

```
f = t.*cos(t)-t;
plot(t,f)
xlabel('t')
ylabel('f(t) = t*cos(t)-t')
```

20. The MATLAB command to get help about a command:

```
help plot       %"help" followed by the command
```

Lab #1 - Part 2

Relational Operators:

== equal to	~= not equal to
< less than	> greater than
<= less than or equal to	>= greater than or equal to

Logical Operators

&	AND
|	OR
~	NOT

Conditionals: "if"

If statements allow us to execute different commands depending on the truth or falsity of some logical tests. The general form of the statement is

Example: This code is used to assign letter grades to students:

```
if points >= 90
        gradepoints = 'A';
elseif points >= 80 & points < 90
        gradepoints = 'B';
elseif points >= 70 & points < 80
        gradepoints = 'C';
elseif points >= 60 & points < 70
        gradepoints = 'D';
else
        gradepoints = 'E';
end
```

Loops: "for"

For loops are used when we want to repeat a segment of code for a predetermined number of times.

Example: This code computes the result of the summation $\sum_{1}^{20} x$

```
sum=0;
for x=1:1:20
        sum=sum+x;
end
```

Loops: "while"

'While loops' are used when we want to repeat a segment of a code until a condition is satisfied.

Example: This code calculates the maximum value of n satisfying 1 + 2 + 3 + … + n <= 2826.

```
sum=0;
n=1;
while sum<=2826
        sum=sum+n
        n=n+1;
end
```