MATLAB Workshop
Dr. M. T. Mustafa
Department of Mathematical Sciences

Introductory remarks

- MATLAB: a product of mathworks
  www.mathworks.com
  MATrix LABoratory

A matrix based system for scientific computation & visualization.

- What can we do (in or) with MATLAB
  - Use like a calculator
  - Graphics & visualization
  - Symbolic computation
  - Numerics
  - Use as a programming language
  - Solve applied & industrial problems using its sophisticated tool boxes

Selected list of toolboxes

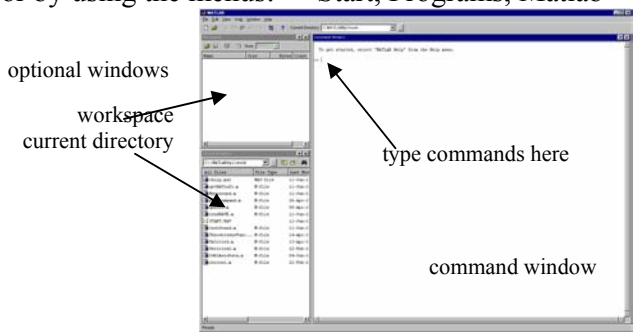
- Optimization toolbox
- Statistics toolbox
- Symbolic Math toolbox
- PDE toolbox
- Curve fitting toolbox
- Spline toolbox
- Image processing toolbox
- Wavelet toolbox
- Control system toolbox
- Fuzzy logic toolbox
- Financial derivative toolbox

Contents of workshop (PART I):

1. Getting started

1.1 Starting MATLAB

By double clicking on MATLAB icon
or by using the menus: Start, Programs, Matlab



screen shot of the Matlab window

1.2 Quitting & interrupting MATLAB

Type *quit* in command window
or use menus: File, Exit Matlab
To interrupt: Use ctrl-C

1.3 Asking for help

>> helpwin
>> helpdesk
>> demo
We can also obtain information about a particular function or command, for example
>> help sqrt
or
>> lookfor sqrt

1.4 Command line editing

Command lines are entered in the command window, in front of MATLAB prompt >>. A command line consists of one or more statements separated by semicolons.
For example
>> a=3; [This assigns the value 3 to the variable "a"]
>> b=4; c=5;
Try
>> a+b+c
or
>> d=a+b+c as well as *>> d=a+b+c ;* [do you see any difference in the output]

Few useful points to note:

- Every command line terminates with an <Enter>
- A single command line can have more than one statement
- The use of semi-colon
- How to recall previous commands

2. Simple computations in MATLAB (using built in scalar functions)

- Basic operations
  '+' addition, '-' subtraction, '*' multiplication,
  '/ 'right division, '\' left division, '^' exponentiation
- MATLAB has many built in functions which operate on scalars as well as element-wise on vectors & matrices (as we will see later). These can be used as commands or part of a command.

Elementary math functions	
abs	Absolute value or complex magnitude
angle	Phase angle
sqrt	Square root
real	Real part
imag	Imaginary part
conj	Complex conjugate
gcd	Greatest common divisor
lcm	Least common multiple
round	Round to nearest integer
fix	Round towards 0
floor	Round towards $-\infty$
ceil	Round towards ∞
sign	Signum functions
rem	Remainder
exp	Exponential base e
log	Natural logarithm
log10	Log base 10

Trigonometric & hyperbolic functions	
sin, asin, sinh, asinh	sine, arcsine, hyperbolic sine, hyperbolic arcsine
cos, acos, cosh, acosh	cosine, arccosine, hyperbolic cosine, hyperbolic arccosine
tan, atan, tanh, atanh	tangent, arctangent, hyperbolic tangent, hyperbolic arctangent
cot, acot, coth, acoth	cotangent, arccotangent, hyperbolic cotangent, hyperbolic arccotang
sec, asec, sech, asech	secant, arcsecant, hyperbolic secant, hyperbolic arcsecant
csc, acsc, csch, acsch	cosecant, arccosecant, hyperbolic cosecant, hyperbolic arccosecant

Examples:

>> sqrt(26.5)
>> x=4; y=sin(x) [for trig. functions, MATLAB works in radians]

See what the output of following is:
*>> cos(pi/3)*abs(-10)*

>> log(exp(2))+log10(exp(2))

>> rem(13,3)

- We can have more information by help command.

Check
>> help rem
To get information about elementary functions, start for example from
>> help elfun

3. Vectors

MATLAB is a vector or matrix oriented software. It is important to get a good grip on *generating, manipulating and applying different operations* on vectors.

3.1 Building vectors

- By directly typing as

  >> a = [1, 3, 5, 7, 9]
  or
  >> a= [1 3 5 7 9] *See the output on your screen*

- By specifying the stepsize from the starting point to ending point

  >> a=1:2:9
  or *check the output on your screen*
  >> a=[1:2:9]
  means: start with 1, increase in steps of size 2 and end with 9.

  Exercise 1
  Check what happens with the commands
  >> b = [6 4 2 0]
  >> b = 6:-2:0
  >> b = 6:-2:-1

- By specifying the number of elements in the vector
  - *Specify the number of (equally spaced) points between the starting point and end points*
  - *Most practical method*

  Examples
  >> a = linspace(1,5,7) *check the output on your screen*
  Creates a vector of 7 equally spaced points from 1 to 5

  >> v = linspace(-10,10,500);

  What do you get if you try
  >> length(v)
  or
  >> size(v)

- By using built in functions

  MATLAB provides commands to directly generate some specific vectors and matrices

  For example,
  >> b = zeros(1,N)
  creates a vector of length N (or a 1-by-N matrix) with zero entries

Now type
>> size(a)

What do you think will be output of
>> zeros(size(a))

What is your guess about the output of following?
>> ones(size(a))
or
>> ones(1,7)

>> rand(size(a)) [uniformly distributed random entries]

>> randn(1,7) [normally distributed random entries]

- By piecing together two vectors

  For example
  >> a = [1 3 5];
  >> b = [2 4 6];
  Then
  >> c = [a b]
  creates the vector c=[1 3 5 2 4 6]

  Exercise 2 **{How to access entries of vector}**
  Create a vector "a" of 5 values from 0 to 2. What is the 3rd value. Set the 3rd value as 7.
  [Hint: the ith entry of vector a can be accessed as a(i)]

3.2 Vector operations and functions

- Addition, subtraction of vectors (as usual)

  Here are a few examples
  >> A=[1 2 3 4]; B= [2 1 0 4]
  >> C = A+B;
  >> A = A-B;

- Transpose

  See what happens with command
  >> D = A'

- Product

  Done as a matrix product
  Try the following commands and check the output
  >> A*A'
  >> A'*A
  >> V = [1 2 3]
  >> A*V

- Multiplication, division by a scalar

  As usual, for example
  >> a=2;
  >> a*A produces [2 4 6 8]
  >> (a*A)/2 produces [1 2 3 4]

- Addition/subtraction of a scalar

  To understand this point, see what happens if you type
  >> c = A+100;
  >> c-50

- Scalar functions on vectors

  The operation explained here is very useful for many purposes. All the built in scalar functions (introduced in section 2) operate on vectors (as well as matrices) element-wise.

  Here are a few examples. Try these or use scalar functions of your liking.
  >> A=1:4;

  >> log(A)

  >> sqrt(A)

  Exercise 3

  Construct a vector x containing 8 values, starting with 0 and ending with π . Next construct a vector y which contains *sine* of the entries of the vector x. What is the sum of the 4th elements of x and y.

- Vector functions

  The following examples illustrate some of the vector functions.

  >> v = [-7 3 5];

sum(v) = 1	[adds elements of v]
max(v)=3	[gives the maximum element]
abs(v)=[7 3 5]	
sort(-v)= [-5 -3 7]	[sorts in increasing order]
median(v)=3	[gives median of the elements]
mean(v)=0.3333	[gives mean of the elements]
std(v)=6.4291	[gives the standard deviation]

Exercise 4 **{help is indeed help}**

Type

>> help rand [to know details about & behind this command]

Now type

>> v = rand(1,1000);

What to you expect the mean of the elements of v to be? Compare your answer with MATLAB output.

3.3 Dot operation

This is another frequently used operation of MATLAB. This can be used to enforce MATLAB to perform an entry-wise operation.

Try following examples to learn more about it.

>> a = [2 4 6]; b = [1 -3 2];

>> a*b

>> a.*b

>> a./b

>> a.^2

>> a.^b

Exercise 5

Create a vector x from 1 to 4 with four elements and a vector y from 4 to -3 with step size -2.
Find [max(b.^a) – min(b.^a)]. Find sin(a.*b).

Exercise 6

Create a vector v containing integers from 10 to 100. How can you create a vector w containing elements which are reciprocals of the corresponding elements of v.

Exercise 7

What do you expect the outcome of following command line to be?

>> v=[1 4 16 25]; z = (sqrt(v)).^2

4. Two dimensional graphics

4.1 XY plots

If we want to plot the graph of y = f(x) by hand, one way is

Make a list of X-values
Compute corresponding Y-values
Plot the point (x,y) and connect.

MATLAB does exactly the same thing. The command

>> plot(X,Y)

produces a graph of the elements of vector X versus the corresponding elements of the vector Y.

We learn more via examples.

Example To plot $y = \sin x$ on $[0, 2\pi]$

The following set of commands produces the required graph.

```
>> X = linspace(0,2*pi,50);
>> Y = sin(X);
>> plot(X,Y)
```

compare your output with others

Example **{more than one graphs in one window}**

To plot $f(x) = \frac{x}{1+x^2}$ and $g(x) = \frac{x^2}{1+x^2}$ on $[-2, 2]$.

We need the following set of commands

```
>> x = linspace(-2,2,100);
>> y1 = x./(1+x.^2);
>> plot(x,y1)
>> hold on          [freezes the current graph window]
>> y2 = (x.^2)./(1+x.^2);
>> plot(x,y2)
>> hold off         [releases the hold]
```

compare your output with others

Optional arguments

- The graphs can be given **titles, axes labels** etc using the **menu** of graph window or by different **commands.**

```
>> title('My useless graph')
>> xlabel('X')
>> ylabel('do not know')
>> grid on
```

- Line types: Using graph window or
  Solid(-), dashed(--), dotted(:)
- Mark types Using graph window or
  Point(.), plus(+), circle(o)
- Colors Using graph window or
  Red(r), green(g), blue(b)

4.2 Saving, re-using and printing graphs

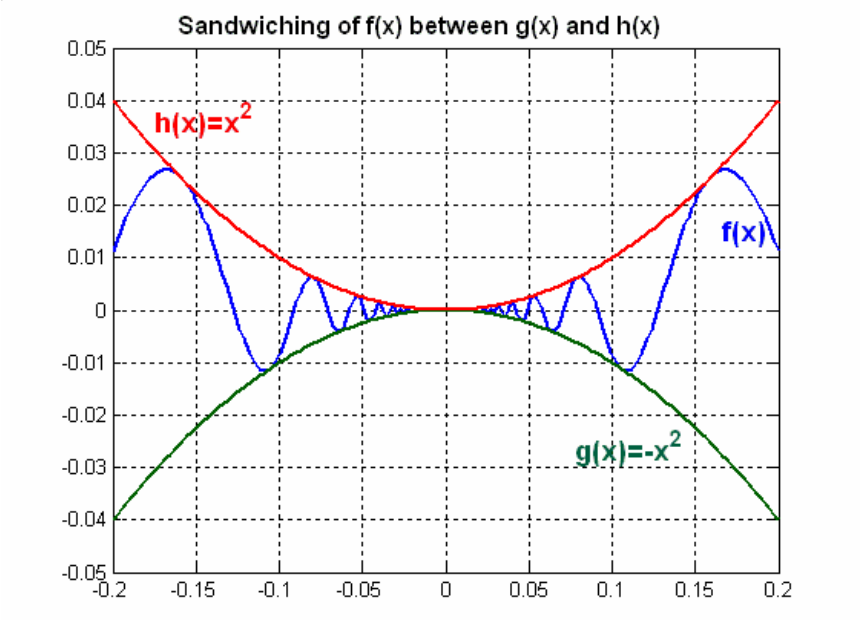
- One option is to **save** (using graph window menu) as *fig* file. The advantage of this option is that you can open these again in MATLAB and edit if needed.
- Another option is to **export** (using graph window menu) as *bmp or jpg or eps* file as required. The advantage of this option is that you can use these elsewhere, e.g. in MSWORD, LaTeX, Powerpoint etc.

Exercise 8

Make the graphs of $f(x)=x^2\cos\left(\frac{1}{x}\right)$, $g(x)=-x^2$ and $h(x)=x^2$ in one window and complete a transparency like the following (using MS word). A file sandwich.doc is on your desktop, you are required to make the graphs and include those in the word file.

Geometric demonstration of the squeezing theorem

Squeezing of $f(x)=x^2\cos\left(\frac{1}{x}\right)$ between $g(x)=-x^2$ and $h(x)=x^2$



4.3 Plotting parametric equations

Plots of parametrically defined curves can also be made in a manner similar to XY-plots.

For example, the following commands plot $x = \sin 2t$, $y = \cos 3t$ over $[0, 2\pi]$.

```
>> t = 0: 0.001:2*pi;
>> x = sin(2*t);     y= cos(3*t);
>> plot(x,y)
>> title('Lissajous figure')
```

compare your output with others

Exercise 9

Plot the circle $x^2 + y^2 = 1$ using parametric representation.

Do you see any problem? Did you get a figure looking like an ellipse.

Now open another figure window by the command

```
>> figure
```

and plot the same graph here using

```
>> plot(x,y)
```

and then type

```
>> axis equal
```

Can you see what was the problem with previous figure.

Related useful commands are

```
>> axis equal          [creates equal scales on the axes]
>> axis normal         [returns to MATLAB's original scaling]
```

Check

```
>> help axis
```

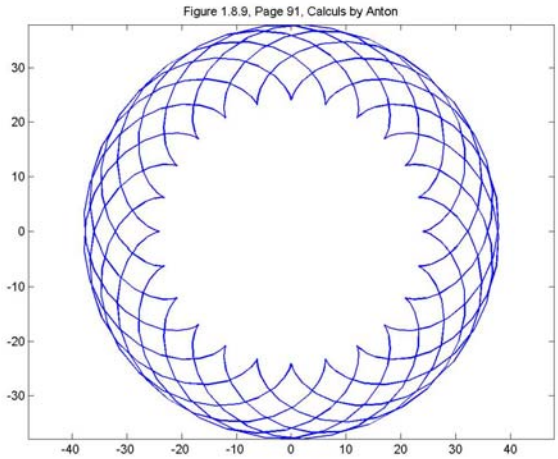
for more information.

Exercise 10

Plot $x = 31\cos t - 7\cos(\frac{31}{7}t),\ y = 31\sin t - 7\sin(\frac{31}{7}t)$

$0 \le t \le 14\pi$

to get a figure like the following. Also compare your output with Figure1.8.9 page 91 of Calculus (7th edition) by Anton.



4.4 Plotting polar equations

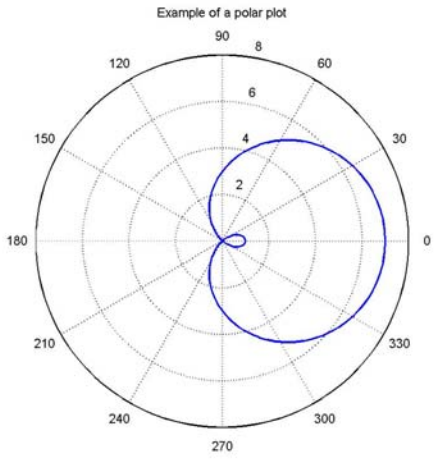
Plot in polar coordinates can be done using the command ***polar(theta,r)*** where ***theta*** is the angle in radians and ***r*** is the radius vector.
This is illustrated below.

Example Graph of $r = 3 + 4\cos\theta$

The following set of commands

```
>> theta = linspace(0,2*pi,100);
>> r = 3 + 4*cos(theta);
>> polar(theta,r)
```

produce

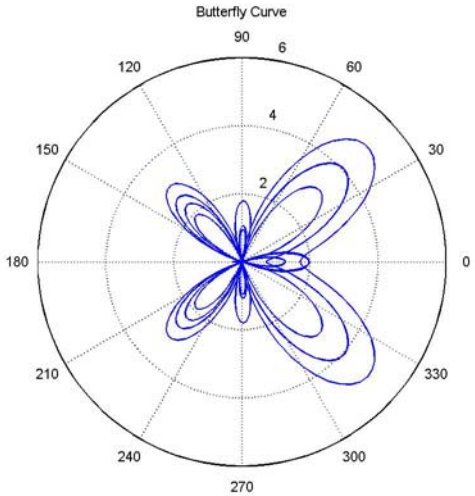


Exercise 11 **[Q.56, Ex. 11.1, Calculus (7th edition) by Anton]**

The accompanying figure shows the graph of the "butterfly curve"

$$r = e^{\cos\theta} - 2\cos 4\theta + \sin^3\frac{\theta}{4}.$$

Generate the complete butterfly with MATLAB and state the parameter interval you used.



Example {more than one polar graphs in one window}

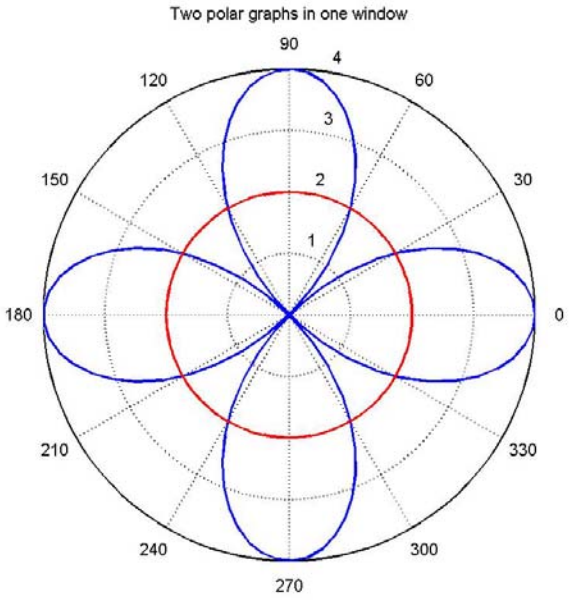
To plot $r = 4\cos 2\theta$ and $r = 2$ in one window.

A script file consisting of following commands produces the desired graphs.

Commands to be written in script file

```
clear all
close all
theta=linspace(0,2*pi,100);
r1=4*cos(2*theta);
polar(theta,r1)
hold on
r2=2+0*theta;
pause
polar(theta,r2,'r')
title('Two polar graphs in one window')
hold off
```

The graphs produced by above script file are as follows.



Exercise 12

Plot the following polar curves in one window

$r = 2(1+\cos\theta)$.

$r = 2(1-\cos\theta)$

Viewing a polar graph as it is traced

In some of the polar curves (for example those with many loops or petals) it may be useful to show the graph as it is traced. We will do it with the help of very simple MATLAB program which explained in the example below.

Example {To show the graph as it is traced}

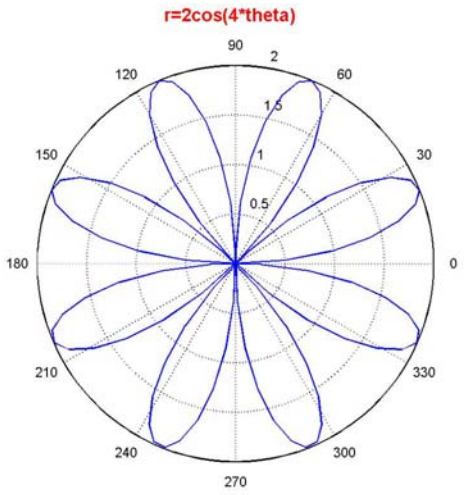
To plot the rose with 8 leaves given by equation $r = 2\sin 4\theta$. Further write a program such that the graph is shown as it is traced.

A script file consisting of following commands

Commands to be written in script file

```
clear all
close all
theta=linspace(0,2*pi,100);
r=2*sin(4*theta);
polar(theta,r)
```

produce



In order to view the tracing of graph we shall use the following script file whose commands will be explained in the workshop session 2.

```
clear all
close all
x=linspace(0,2*pi,100);
figure
theta=x(1);
for i=2:100
        theta=[theta x(i)];
        r=2*sin(4*theta);
        polar(theta,r);
        drawnow
end
```

Exercise 13 {Rose within a rose

Plot the following polar curve

$$r = 1 - 2\sin 3\theta.$$

Further write a program such that the graph is shown as it is traced.

5. Few words about script files

- Covered in stages in workshop sessions.

End of Part 1