Salahaddin University – Erbil
Education College
Physics Department

MATLAB

Lecture Two

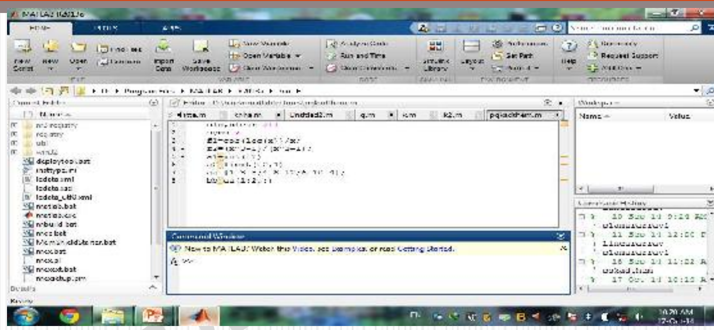
MATLAB Basics

Prepared By :
Assist. Prof. Dr. Haidar J. Ismail
Biophysics (Medical image processing)
Web address: https://academics.su.edu.krd/haidar.ismail

2. MATLAB Basics

- The differences between the MATLAB Professional Version and the MATLAB Student Version are rather minor, and virtually unnoticeable to a beginner, or even a mid-level user.
- MATLAB logo, MATLAB Desktop window will launch: title bar, a menu bar, a toolbar, and five embedded windows, one of which is hidden. The largest and most important window is the Command Window in the center.
- The Command History Window, the Current Directory Browser, and the Workspace Browser.
- Command prompt (>>). If the Command Window is "Active," its title bar will be dark, and the prompt will be followed by a cursor (a blinking vertical line).

2. MATLAB Basics



- **Help Browser:** While help in the Command Window is useful for getting quick information on a particular command, more extensive documentation is available via the MATLAB Help Browser. Different ways of invoke, one is following:
  >>doc sin

2. MATLAB Basics

- You can type **demo**(or select it in the help browser) to try some of MATLAB's online demonstrations.
- Methods to **exit** MATLAB: type quit at the prompt, click on (×), close icon, Alt+F4.

Arithmetic

Operation	Algebraic form	MATLAB
Addition	$a+b$	a + b
Subtraction	$a-b$	a - b
Multiplication	$a \times b$	a * b
Right division	a/b	a / b
Left division	b/a	a \ b
Power	a^b	a ^ b

Precedence	Operator
1	Parentheses (round brackets)
2	Power, left to right
3	Multiplication and division, left to right
4	Addition and subtraction, left to right

2. MATLAB Basics

Note: MATLAB prints the answer and assigns the value to a variable called ans. If you want to perform further calculations with the answer, you can use the variable ans rather than retype the answer.

Note: Trigonometric functions in MATLAB use radians, not degrees.

Note: MATLAB displays only 5 digits by default. To display more digits, type format long (15 digits). Type format short to return to 5-digit display.

- **Recovering from Problems:** If you make an error in an input line, MATLAB will normally print an error message.

```
>> 3u^2

??? 3u^2
     |
Error: Missing MATLAB operator.
```


2. MATLAB Basics

Note: MATLAB places a marker (a vertical line segment) at the place where it thinks the error might be; however, the actual error may have occurred earlier or later in the expression.

Note: The UP- and DOWN-ARROW keys allow you to scroll back and forth through all the commands you've typed in a MATLAB session, and are very useful when you want to correct, modify, or reenter a previous command.

Aborting Calculations: If MATLAB gets hung up in a calculation, or seems to be taking too long to perform an operation, you can usually abort it by typing CTRL+C.

2. MATLAB Basics

Symbolic Computation

- Type help symbolic to make sure that the Symbolic Math Toolbox is installed on your system.
- To perform symbolic computations, you must use **syms** to declare the variables you plan to use to be symbolic variables.

```
>> syms x y
>> (x - y)*(x - y)^2

ans =
(x-y)^3
```

```
>> expand(ans)

ans =
x^3-3*x^2*y+3*x*y^2-y^3
```

```
>> factor(ans)

ans =
(x-y)^3
```

- The command **expand** told MATLAB to multiply out the expression, and **factor** forced MATLAB to restore it to factored form.
- MATLAB has a command called **simplify**, which you can sometimes use to express a formula as simply as possible.

2. MATLAB Basics

```
>> simplify((x^3 - y^3)/(x - y))

ans =
x^2+x*y+y^2
```

- When you work with symbolic expressions you often need to substitute (using subs) a numerical value, or even another symbolic expression, for one (or more) of the original variables in the expression.

```
>> d = 1, syms u v    >> w = u^2 - v^2    >> subs(w, u, 2)
   >> subs(w, v, d)    >> subs(w, v, u + v)    >> subs(w,[u v],[4 3])
```

Note: when you enter multiple commands on a single line separated by commas, MATLAB evaluates each command and displays the output on separate lines.

- **Exact Arithmetic:** MATLAB uses floating-point arithmetic for its calculations. You can do exact arithmetic with symbolic expressions.

2. MATLAB Basics

>>cos(pi/2) % really cos(π/2)=0   ans = 6.1232e-17

- The inaccuracy is due to MATLAB gives an approximation to π accurate to about 15 digits, not its exact value.
- To compute an exact answer, instead of an approximate answer, we must create an exact symbolic representation of π/2 by: >> cos(sym('pi/2')) ans = 0
- **syms** has a lasting effect on its argument (even if x was previously defined, syms x clears that definition and renders x a symbolic variable – which it remains until it is redefined) but sym has only a temporary effect unless you assign the output to a variable, as in x = sym('x').
- Here is how to add 1/2 and 1/3 symbolically:
  >> sym('1/2') + sym('1/3') ans = 5/6

2. MATLAB Basics

- **Variable-precision arithmetic:** with vpa. For example, to print 44 digits of √2, type:
- >> vpa('sqrt(2)', 44)
  ans = 1.4142135623730950488016887242096980785696718
- If you don't specify the number of digits, the default setting is 32.

Note: one should be wary of using vpa on an expression that MATLAB must evaluate before applying variable-precision arithmetic.

3ˆ45	gives a floating-point approximation
vpa(3ˆ45)	gives an answer that is correct only in its first 16 digits
vpa('3ˆ45')	gives the exact answer

2. MATLAB Basics

Vectors and Matrices

- **Vectors:** A vector is an ordered list of numbers. You can enter a vector of any length in MATLAB by typing a list of numbers, separated by commas and/or spaces, inside square brackets. For example:
  >> z = [1,4,7,18]   >> y = [4 -3 5 -2 8 1]
- vector of values running from 1 to 9:
  >> x = 1:9   X = 1 2 3 4 5 6 7 8 9
- The increment can be specified as the **middle** of three argument: **>> x = 0:2:10**   x = 0 2 4 6 8 10
- Increments can be **fractional or negative**, for example, **0:0.1:1** or **100:-1:0**.
  linspace(0,10,6) ans = 0 2 4 6 8 10

2. MATLAB Basics

- The elements of the vector **x** can be **extracted** as **x(1)**, **x(2)**, etc. For example:
  >> **x(3)**   ans = 4   >> **x(4:7), x([4,7])**
- To **change** the vector **x** from a row vector to a column vector, put a **prime** (') after **x**: >> **x'**
  >>x1=[5,3,1,23,11],min(x1),max(x1),mean(x1),sort(x1),sum(x1)
- You can perform **mathematical operations** on vectors.
  for example, to **square** the elements of the vector **x**,
  >> **x.^2**   ans = 0 4 16 36 64 100
- The **period(.)** in this expression says that the numbers in **x** should be squared individually, or *element-by-element*.
- Typing **x^2** would tell MATLAB to use matrix multiplication to multiply **x** by itself and would produce an **error** message in this case.

2. MATLAB Basics

- Similarly, you must type **.*** or **./** if you want to multiply or divide vectors element-by-element.
  >> x.*y ans = 0 -6 20 -12 64 10
- Most MATLAB operations are, by **default**, performed element-by-element. For example, you do not type a period**(.)** before: **addition, subtraction** and **exp(x)** (the **matrix exponential** function is **expm**).
- **Matrices:** It is a rectangular array of numbers. Row and column vectors are examples of matrices. Consider the 3 × 4 matrix:

$$a = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 7 & 15 & 3 \\ 12 & 9 & 6 & 3 \end{pmatrix}$$

>> a = [1:4;5,7,15,3;12:-3:3]

2. MATLAB Basics

Note: the matrix elements in any row are separated by commas, and the rows are separated by semicolons. The elements in a row can also be separated by spaces.

- Extract: >> a(7),a(3,2),a(2,:),a(1:3,2:3),a([2 3],[1 3])
- If two matrices A and B are the same size:
  sum: A+B add a scalar (a single number): A + c
  difference: A-B subtracts: A - c
- If A and B are multiplicatively compatible, i.e., if A is n×m and B is m×ℓ, then their product A*B is n×ℓ.

$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} \cdot \begin{bmatrix} 3 & 6 \\ 1 & 4 \\ 5 & 2 \end{bmatrix} = \begin{bmatrix} 31 & 28 \\ 42 & 2\cdot 6+4\cdot 4+6\cdot 2 \end{bmatrix} = \begin{bmatrix} 31 & 28 \\ 42 & 40 \end{bmatrix}$$

$$A\cdot B = \begin{bmatrix} 2 & 1 \\ 3 & 5 \end{bmatrix} \cdot \begin{bmatrix} -2 & 3 \\ 4 & -1 \end{bmatrix} = \begin{bmatrix} 2\cdot(-2)+1\cdot 4 & 2\cdot 3+1\cdot(-1) \\ 3\cdot(-2)+5\cdot 4 & 3\cdot 3+5\cdot(-1) \end{bmatrix} = \begin{bmatrix} 0 & 5 \\ 14 & 4 \end{bmatrix}$$

- zeros(1,3), ones(2,4), rand(3,5), randn(2,5), eye(n,m), det(A)

2. MATLAB Basics

- The product of a number **c** and the matrix **A** is given by **c*A**, and **A'** represents the **conjugate transpose** of A.

$$A := \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 6 & 7 & 8 \\ w & x & y & z \end{bmatrix} \qquad A = \begin{bmatrix} 1+2i & 2-i & 3i \\ 4 & -2+7i & 6+6i \end{bmatrix}, \text{ then } A^* = \begin{bmatrix} 1-2i & 4 \\ 2+i & -2-7i \\ -3i & 6-6i \end{bmatrix}.$$

Note: Typing a semicolon at the end of an input line suppresses printing of the output of the MATLAB command.

Functions

- In MATLAB you will use both **built-in functions** and **functions** that you create yourself.

Built-in Functions

- MATLAB has many built-in functions.

1. These include: **sqrt**, **cos**, **sin**, **tan**, **log**, **exp**, and **atan** (for arctan).

2. MATLAB Basics

2. Specialized mathematical functions like: **gamma, erf,** and **besselj**.
3. MATLAB also has several **built-in constants**, including: **pi** (the number π), **i** (the complex number i $=\sqrt{-1}$), and **Inf** (∞).

User-Defined Functions

- Two methods to define your own functions are:

1. The first uses the command **inline**, and the second uses the operator **@** to create what is called an "**anonymous function**".

```
>> f = @(x) x^2        f = @(x) x^2
>> f1 = inline('x^2', 'x')        f1 = Inline function: f1(x) = x^2
>> f(4)    ans = 16        >> f1(4)    ans = 16
```

2. MATLAB Basics

Note: To insure that your user-defined function can act on vectors, insert dots before the mathematical operators *, /, and ^.

`>> f = @(x) x.^2`   or else   `>> f1 = inline('x.^2', 'x')`

`>> f(1:5)`   ans = 1 4 9 16 25

- One can also define functions of two or more variables.

  `>> g = @(x, y) x^2 + y^2; g(1, 2)`   ans = 5

  `>> g1 = inline('x^2 + y^2', 'x', 'y'); g1(1, 2)`   ans = 5

- If instead you define: `>> g = @(x, y) x.^2 + y.^2;` then you can evaluate on vectors; thus:

  `>> g([1 2], [3 4])`   ans = 10 20

gives the values of the function at the points (1,3) and (2,4).

g = @(x, y) x.^2 + y^2; g(1:5, 2)   ?

g1 = inline('x.^2 + y.^2', 'x', 'y'); g1(1:5, 2:6)   ?

g2 = inline('x.^2 + y.^2', 'x', 'y'); g2([1:3;2:4],2)   ?

1. Equations to be solved in Command window:

433.12*15.7

$5\left(\frac{3}{4}\right)+\frac{9}{5}$ and $4^3\left[\frac{3}{4}+\frac{9}{(2)3}\right]$

$C = \sqrt{A^2 + B^2}$

$2^5/(2^6 - 1)$

e^4

$\ln(e^4)$

$\log10(e^4)$

$e^{\pi\sqrt{121}}$

$\cos(\pi/4) + \sin^2(\pi/3)$

$\log_e(e^3) + \log_{10}(e)$

$area = \pi * (\pi/3)^2$

$a = e$

$b = e^3$

Function	Description	Mathematical Expression
sin(u)	Sinus	sin(u)
cos(u)	Cosinus	cos(u)
exp(u)	Exponential	e^u
log(u)	Natural logarithm	ln(u)
10^u	Power of base 10	10^u
log10(u)	Common (base 10) logarithm	log(u)
u^2	Power 2	u^2
sqrt(u)	Square root	$u^{0.5}$
1/u	Reciprocal	$1/u$

$g = e^{(3\sqrt{131})}$ $c = \ln(e) + \ln(e^3)$ $h = \log(5) + \log_e(5) + \log_2(5)$

$d = \log(e) + \log(e^3$

$e = \pi$

$f = \cos(\pi/4)$

>>Factor(12)   **ans: 2 2 3**

>>Factor(24)   **ans: 2 2 2 3**

$(x+2)(x-3) = x^2 - x - 6$

```
>> expand((x-1)*(x+4))
ans = x^2 +3*x - 4
expand(cos(x+y))
ans =

cos(x)*cos(y)-sin(x)*sin(y)
>> expand(sin(x-y))
>> syms y
>> expand((y-2)*(y+8))
ans =

y^2+6*y-16
```

$x(x^2-2) = x^3 - 2x$

```
>> collect(x*(x^2-2))
ans =
x^3-2*x
```

```
>> syms t
>> collect((t+3)*sin(t))
```

$x^2 - y^2 = (x+y)(x-y)$

```
>> factor(x^2-y^2)
ans =
(x-y)*(x+y)
>> factor([x^2-y^2, x^3+y^3])
```

$(x^2+9)(x^2-9) = x^4 - 81$

```
>> simplify((x^4-81)/(x^2-9)) ans =
x^2+9
```

$e^{2\log 3x} = e^{\log 9x^2} = 9x^2$

```
>> simplify(exp(2*log(3*x))) ans =
9*x^2
>> simplify(cos(x)^2-sin(x)^2)
ans =    2*cos(x)^2-1
>> simplify(cos(x)^2+sin(x)^2)
```

19


```
>> format long
>> x = 3 + 11/16 + 2^1.2
x =
    5.98489670999407
>> format short
>> x = 3 + 11/16 + 2^1.2
x =
    5.9849
>> format bank
>> hourly = 35.55
>> weekly = hourly*40
weekly =
            1422.00
```

5.4387×10^3 as $5.4387e+003$.

```
>> format short e
>> 7.2*3.1
ans =
   2.2320e+001
```

zeros(n) ones(n) size (A) length(A)

A = zeros(1,3) A = ones(2,4)
A = rand(3,5) A = randn(2,5)

20

```
>>v=[1, 2, 4, 5] >>w=[1;2;4;5]
>> A = [1,2,3;4,-5,6;5,-6,7]
>>v+2  >>B=A'  >>A*B  >>A+B  >> B=A.'
```

Transpose	B = A'
Identity Matrix	eye(n) ➔ returns an n x n identity matrix eye(m,n) ➔ returns an m x n matrix with ones on the main diagonal and zeros elsewhere.
Addition and subtraction	C = A + B C = A – B
Scalar Multiplication	B = αA, where α is a scalar.
Matrix Multiplication	C = A*B
Matrix Inverse	B = inv(A), A must be a square matrix in this case. rank (A) ➔ returns the rank of the matrix A.
Matrix Powers	B = A.^2 ➔ squares each element in the matrix C = A * A ➔ computes A*A, and A must be a square matrix.
Determinant	det (A), and A must be a square matrix.

A, B, C are matrices, and m, n, α are scalars.

```
>> A(2,3)      >> A([2 3],[1 2])  B(:,2) = [ ]
>> B=A([3 2],[2 1])
>> B=[A(3,2),A(3,1);A(2,2);A(2,1)]
>> A(1,:)          >> A(1:2,:)     >> A([1 2],:)
>> v=1:5     >> w=1:2:5        >> z=[1;2;3];
>> x=A\z          >> det(A)        >> inv(A)
```

$$\det\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{21}a_{12}$$

Inverse:

A must be square

$$\det\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

$$A_{n\times n}A^{-1}{}_{n\times n} = A^{-1}{}_{n\times n}A_{n\times n} = I$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}^{-1} = \frac{1}{a_{11}a_{22} - a_{21}a_{12}}\begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

```
>> f = @(x) exp(x)-1
>> f = inline('exp(x)-1')
>> alpha = 1;
>> g = @(x,y,z) x^2+y^2-alpha*z^2;
>> g(1,2,3)
```

21


```
>> f = inline('log(a*x)/(1+y^2)')
>> f = inline('log(a*x)/(1+y^2)','x','y','a')
>> f = inline('exp(-x)/(1+x^2)');
```

```
% this lines writted in command
%window seperately and
% bwith enter key after each
syms x y
(x-y)*(x-y)^2
expand(ans)
factor(ans)
simplify((x^3-y^3)/(x-y))
simple(ans)
```

Table 2.1: Elementary functions

cos(x)	Cosine	abs(x)	Absolute value
sin(x)	Sine	sign(x)	Signum function
tan(x)	Tangent	max(x)	Maximum value
acos(x)	Arc cosine	min(x)	Minimum value
asin(x)	Arc sine	ceil(x)	Round towards $+\infty$
atan(x)	Arc tangent	floor(x)	Round towards $-\infty$
exp(x)	Exponential	round(x)	Round to nearest integer
sqrt(x)	Square root	rem(x)	Remainder after division
log(x)	Natural logarithm	angle(x)	Phase angle
log10(x)	Common logarithm	conj(x)	Complex conjugate

Punctuation marks

Punctuation	
apostrophe	(' ')
brackets	([ ], ( ), { }, ⟨ ⟩)
colon	(:)
comma	(, ،)
dash	(‒, –, —, ―)
ellipsis	(…, ..., . . .)
exclamation mark	(!)
full stop / period	(.)
hyphen	(-)
hyphen-minus	(-)
question mark	(?)
quotation marks	(' ', " ", ' ', " ")
semicolon	(;)
slash / stroke / solidus	(/, ⁄)