

Introduction to MATLAB

Instructor Özgür ZEYDAN

<http://cevre.beun.edu.tr/zeydan/>

About MATLAB

- MATLAB stands for **MATrix LABoratory**
- MATLAB is first developed by Cleve Barry Moler in 1970s
- Cleve Barry Moler and Jack Little are founded MathWorks Company in 1984
- Today MATLAB is developed and sold by MathWorks Company



Cleve Barry Moler



Jack Little

Özgür ZEYDAN

2

MATLAB

- MATLAB can be used for:
 - ❖ Numerical calculations
 - ❖ Algorithm development
 - ❖ Data analysis
 - ❖ Visualization


- Student version of MATLAB can be obtained from this link (\$99):
- http://www.mathworks.com/academia/student_version/index.html

Özgür ZEYDAN

3

Running MATLAB

- Start menu → All Programs → MATLAB → R2013b → MATLAB R2013b

 - Double click on MATLAB icon on desktop
- 
- The image shows the MATLAB R2013b desktop icon, which features a 3D surface plot in shades of blue and orange, with the text 'MATLAB R2013b' below it.
- Run... → “matlab.exe” or “matlab”

Özgür ZEYDAN

4

MATLAB Window

The screenshot displays the MATLAB R2013b environment. The **Command Window** (highlighted in red) shows the following code and output:

```

1 1 1 1 1
1 2 3 4 5
1 3 6 10 15
1 4 10 20 35
1 5 15 35 70

>> logo
>> a=2
a =
    2

>> b=5
b =
    5

>> c=a+b
c =
    7
  
```

The **Workspace** panel (highlighted in blue) lists variables and their values:

Name	Value
L	5x51 double
a	2
ans	5x5 double
b	5
c	7
l1	175.0011
l2	176.0011
logoFig	1
logoax	173.0011
s	174.0011

The **Command History** panel (highlighted in blue) shows the following commands:

```

-- 07.12.2013 18:05 -->
-- 07.12.2013 19:18 -->
    pasca1(5)
    logo
    a=2
    b=5
    c=a+b
  
```

The **Current folder** panel (highlighted in purple) shows the file explorer for the current directory, listing files like `final`, `konu3`, `konu4`, `konu5`, `konu6`, `konu12`, `konu13`, `konu15`, `vize3`, `asal.asv`, `asal.m`, `convert.asv`, `convert.m`, `imgel.bmp`, `logo.png`, `renk.bmp`, `renk.m`, `rgb2gray.m`, `topla.m`, and `ucgen.m`.

Özgür ZEYDAN

5

Simple Commands

- Simply type $9*5$
- Result is stored in “ans” variable
- After each calculation value of ans changes

Command Window

```

>> 9*5

ans =

    45

>> 28/4

ans =

    7
  
```

Workspace

Name	Value
ans	7

Özgür ZEYDAN

6

Defining Variables

➤ Type these commands:

➤ `a=5`

➤ `b=7`

➤ `c=a+b`

➤ All defined variables can be seen on Workspace

```
>> a=5
```

```
a =  
    5
```

```
>> b=7
```

```
b =  
    7
```

```
>> c=a+b
```

```
c =  
   12
```

Workspace	
Name ^	Value
a	5
ans	7
b	7
c	12

Özgür ZEYDAN

7

“clear” and “clc” commands

➤ “clear” command is used to delete all variables in Workspace

❖ Alternatively use  button

➤ “clc” command is used to clear Command Window

❖ Alternatively use  button

Özgür ZEYDAN

8

Mathematical Functions

Function	Definition	Example
<code>rem(x,y)</code>	Gives remainder of x/y	<code>rem(16,5)</code>
<code>sqrt(x)</code>	Gives square root of x	<code>sqrt(64)</code>
<code>power(x,y)</code>	Gives value of x^y (Also use <code>^</code>)	<code>power(2,5)</code>
<code>round(x)</code>	Rounds x to the nearest integer	<code>round(3.67)</code>
<code>exp(x)</code>	Gives value of e^x	<code>exp(2)</code>
<code>log(x)</code>	Gives logarithm of x in base e	<code>log(3)</code>
<code>log10(x)</code>	Gives logarithm of x in base 10	<code>log10(2)</code>
<code>abs(x)</code>	Gives absolute value of x	<code>abs(-5.9)</code>
<code>factorial(x)</code>	Gives factorial of x	<code>factorial(5)</code>

Özgür ZEYDAN

9

Trigonometric Functions - 1

Function	Definition	Example
<code>pi</code>	Gives value of π	<code>pi</code>
<code>sin(x)</code>	Gives sinus of x in radians	<code>sin(30)</code>
<code>sind(x)</code>	Gives sinus of x in degrees	<code>sind(30)</code>
<code>asin(x)</code>	Gives inverse sinus of x in radians	<code>asind(45)</code>
<code>asind(x)</code>	Gives inverse sinus of x in degrees	<code>asind(45)</code>
<code>sinh(x)</code>	Gives hyperbolic sinus of x in radians	<code>sinh(45)</code>
<code>asinh(x)</code>	Gives inverse hyperbolic sinus of x in radians	<code>asinh(45)</code>

Özgür ZEYDAN

10

Trigonometric Functions - 2

Function	Definition	Example
$\cos(x)$	Gives cosinus of x in radians	$\cos(45)$
$\tan(x)$	Gives tangent of x in radians	$\tan(45)$
$\cot(x)$	Gives cotangent of x in radians	$\cot(45)$
$\sec(x)$	Gives secant of x in radians	$\sec(45)$
$\csc(x)$	Gives cosecant of x in radians	$\csc(45)$

Other functions are similar to sinus functions

Özgür ZEYDAN

11

Example

- Solve quadratic equation ($ax^2+bx+c=0$) using MATLAB
- For example: $x^2 - 2x - 15 = 0$

Quadratic Formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Özgür ZEYDAN

12

Solution

inputs

```

Command Window
>> a=1
a =
    1
>> b=-2
b =
   -2
>> c=-15
c =
  -15
  
```

outputs

```

>> x1=(-b-sqrt(b^2-4*a*c))/(2*a)
x1 =
   -3
>> x2=(-b+sqrt(b^2-4*a*c))/(2*a)
x2 =
    5
  
```

Workspace	
Name ^	Value
a	1
b	-2
c	-15
x1	-3
x2	5

Özgür ZEYDAN 13

Command History

- In order to solve new quadratic equations, just write new values for a, b and c
 - ❖ a=1
 - ❖ b=0
 - ❖ c=-4
- Then **double click** on or **drag and drop** “x1” and “x2” formulas on command history

```

Command History
07.12.2013 20:35
a=1
b=-2
c=-15
x1=(-b-sqrt(b^2-4*a*c))/(2*a)
x2=(-b+sqrt(b^2-4*a*c))/(2*a)
  
```



Özgür ZEYDAN

14

Example

- Find the BOD₅ (y) for a wastewater with
- ultimate BOD (L) = 282 mg/L
- k₁ = 0.348 /day

$$y = L(1 - e^{-k_1 t})$$

Özgür ZEYDAN

15

Solution

```

L =
    282

>> t=5
t =
    5

>> k1=0.348
k1 =
    0.3480

>> y=L*(1-exp(-k1*t))
y =
    232.5032
  
```

Özgür ZEYDAN

16

Matrices in MATLAB

- All variables are stored in MATLAB in matrix form

- $A = [5, 10, -2; -6, 0, 7]$

```
>> A=[5,10,-2;6,0,-7]
A =
     5     10     -2
     6      0     -7
```

- Use ',' (comma) or ' ' (space) to separate row elements
- Use ';' (semicolon) to separate rows
- Size of Matrix $A(m \times n)$:
 - ❖ m: number of rows
 - ❖ n: number of columns

Özgür ZEYDAN

17

Special Matrices

- **Empty matrix**
 - ❖ $B = []$
 - ❖ $C = ' '$
- **Scalar matrix** (size: 1×1)
 - ❖ $D = 23$
 - ❖ $E = -117$
 - ❖ $F = 11.68$
- To check whether matrix is scalar or not
 - ❖ `isscalar(E)`
 - ❖ Gives boolean "1" if true, "0" if false

Özgür ZEYDAN

18

Special Matrices

- **Vector Matrix** (one row or one column matrix)
- Row vector (size: $1 \times n$)
- $G = [3 \ 5 \ 7 \ 13]$ or $G = [3,5,7,13]$

```
>> G = [3 5 7 13]
G =
     3     5     7    13
```

- Column vector (size: $m \times 1$)
- $H = [1; 0; 6; -7; 8]$

```
>> H = [1; 0; 6; -7; 8]
H =
     1
     0
     6
    -7
     8
```

Özgür ZEYDAN

19

Creating Row Matrix

- Matrix name = start : increment : end
- $A = 2 : 0.4 : 4$

```
>> A = 2 : 0.4 : 4
A =
     2.0000     2.4000     2.8000     3.2000     3.6000     4.0000
```

- Matrix name = start : decrement : end
- $B = 10 : -1 : 5$

```
>> B = 10 : -1 : 5
B =
    10     9     8     7     6     5
```

Özgür ZEYDAN

20

Checking Row or Column Vectors

- To check whether matrix is row vector or not
 - ❖ `isrow(A)`
 - ❖ Gives boolean "1" if true, "0" if false

- To check whether matrix is column vector or not
 - ❖ `iscolumn(B)`
 - ❖ Gives boolean "1" if true, "0" if false

Özgür ZEYDAN

21

Indexing Values in Matrix

- `A(3,4)` % Extract the element in row 3, column 4
- `A(1,:)` % Extract first row
- `A(:,2)` % Extract second column

A =

	A(:,2)			
A(1,:)	16	2	3	13
	5	11	10	8
	9	7	6	12
	4	14	15	1

A(3,4)

Özgür ZEYDAN

22

Linear Indexing in Matrices

➤ $A(11) = A(3,3) = 6$

```
>> A(11)
ans =
     6
```

➤ $A(5) = A(1,2) = 2$

```
>> A(5)
ans =
     2
```

1	5	9	13
16	2	3	13
2	6	10	14
5	11	10	8
3	7	11	15
9	7	6	12
4	8	12	16
4	14	15	1

A

Özgür ZEYDAN

23

Linear Indexing in Matrices

A(1)	A(4)	A(7)
A(2)	A(5)	A(8)
A(3)	A(6)	A(9)

B(1)	B(6)	B(11)
B(2)	B(7)	B(12)
B(3)	B(8)	B(13)
B(4)	B(9)	B(14)
B(5)	B(10)	B(15)

Özgür ZEYDAN

24

Getting Information about Matrices

➤ numel()

- ❖ Gives number of elements in matrix

```
>> numel(A)
ans =
    16
```

➤ size()

- ❖ Gives dimensions of matrix

```
>> size(A)
ans =
     4     4
```

➤ length()

- ❖ Gives length of matrix

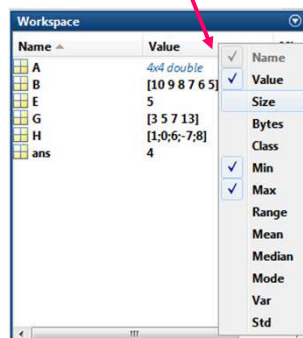
```
>> length(A)
ans =
     4
```

Özgür ZEYDAN

25

Changing Workspace View

Right click here



Name	Value	Size	Bytes	Min	Class	Max
A	4x4 double	4x4	128	1	double	16
B	[10 9 8 7 6 5]	1x6	48	5	double	10
E	5	1x1	8	5	double	5
G	[3 5 7 13]	1x4	32	3	double	13
H	[1;0;6;-7;8]	5x1	40	-7	double	8
ans	4	1x1	8	4	double	4

Select properties, you want to see

Özgür ZEYDAN

26

Creating Matrices Using Functions

- `zeros(m,n)` : matrix with all zeros

```
>> zeros(2,4)
ans =
     0     0     0     0
     0     0     0     0
```

- `ones(m,n)` : matrix with all ones

```
>> ones(1,5)
ans =
     1     1     1     1     1
```

- `eye(m,n)` : the identity matrix

```
>> eye(3,4)
ans =
     1     0     0     0
     0     1     0     0
     0     0     1     0
```

Özgür ZEYDAN

27

Creating Matrices Using Functions

- `rand(m,n)` : uniformly distributed random matrix

```
>> rand(1,6)
ans =
     0.6557     0.0357     0.8491     0.9340     0.6787     0.7577
```

- `randn(m,n)` : normally distributed random matrix

```
>> randn(2,5)
ans =
     0.7269     0.2939     0.8884    -1.0689    -2.9443
    -0.3034    -0.7873    -1.1471    -0.8095     1.4384
```

Özgür ZEYDAN

28

Matrix Operations

- + : addition
- - : subtraction
- * : multiplication

```
>> C=[1 2 3; 4 5 6; 7 8 9]
```

```
C =
```

```
1 2 3
4 5 6
7 8 9
```

```
>> D= [10 11 12;13 14 15; 16 17 18]
```

```
D =
```

```
10 11 12
13 14 15
16 17 18
```

```
>> E=C+D
```

```
E =
```

```
11 13 15
17 19 21
23 25 27
```

```
>> E=D-C
```

```
E =
```

```
9 9 9
9 9 9
9 9 9
```

```
>> E=C*D
```

```
E =
```

```
84 90 96
201 216 231
318 342 366
```

Özgür ZEYDAN

29

Array Operations

- Array (Element by element) operations
- .* : element-by-element multiplication
- ./ : element-by-element division
- .^ : element-by-element power

```
>> F=[1 2 ; 3 4]
```

```
F =
```

```
1 2
3 4
```

```
>> G=[2 4 ; 6 8]
```

```
G =
```

```
2 4
6 8
```

```
>> H=F.*G
```

```
H =
```

```
2 8
18 32
```

```
>> H=G./F
```

```
H =
```

```
2 2
2 2
```

```
>> H=F.^2
```

```
H =
```

```
1 4
9 16
```

Özgür ZEYDAN

30

Transpose of a Matrix

- `'` : gives transpose of a matrix

```
A =
     6     0    -7
     2     5    -8

>> A'

ans =
     6     2
     0     5
    -7    -8
```

Özgür ZEYDAN

31

Determinant of a Square Matrix

- `det()` : gives determinant of a square matrix

```
C =
     1     2     3
     4     5     6
     7     8     0

>> det(C)

ans =

    27.0000
```

Özgür ZEYDAN

32

Inverse of a Matrix

- `inv()` : Gives inverse of a matrix
- Multiplication of a matrix with its inverse matrix gives identity matrix

```
C =
     1     2     3
     4     5     6
     7     8     0

>> D=inv(C)

D =
    -1.7778    0.8889   -0.1111
     1.5556   -0.7778    0.2222
    -0.1111    0.2222   -0.1111

>> E=C*D

E =
     1.0000         0   -0.0000
    -0.0000         1   0.0000
     0.0000   -0.0000     1.0000
```

Özgür ZEYDAN

33

Basic Statistical Functions

- `mean()` : mean value
- `max()` : maximum
- `min()` : minimum
- `sum()` : summation
- `sort()` : sorted values
- `median()` : median value
- `std()` : standard deviation

```
>> A=rand(1,6)
A =
    0.6948    0.3171    0.9502    0.0344    0.4387    0.3816
```

```
>> mean(A)
ans =
    0.4695

>> sum(A)
ans =
    2.8169

>> max(A)
ans =
    0.9502

>> median(A)
ans =
    0.4102

>> min(A)
ans =
    0.0344

>> std(A)
ans =
    0.3172
```

```
>> sort(A)
ans =
    0.0344    0.3171    0.3816    0.4387    0.6948    0.9502
```

Özgür ZEYDAN

34