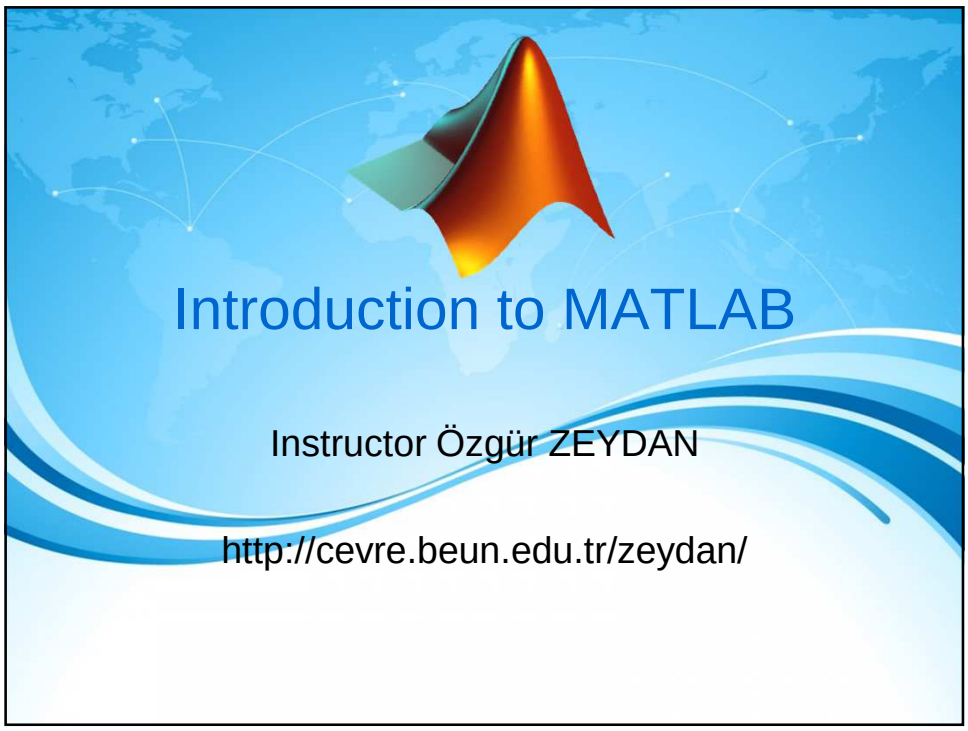


About MATLAB

- MATLAB stands for **MAT**rix **LAB**oratory
- 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

MATLAB

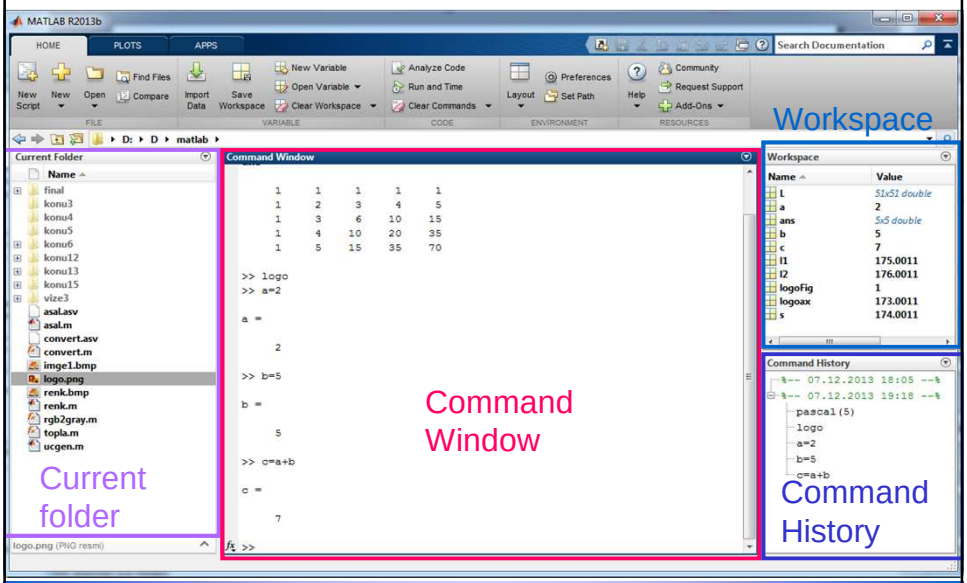
- MATLAB can be used for:
  - Numerical calculations
  - Algorithm development
  - Data analysis
  - Visualization
- Student version of MATLAB can be obtained from this link ($99):
- http://www.mathworks.com/academia/student_version/index.html

Running MATLAB

- Start menu → All Programs → MATLAB → R2013b → MATLAB R2013b
- Double click on MATLAB icon on desktop

- Run… → “matlab.exe” or “matlab”

MATLAB Window



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
```

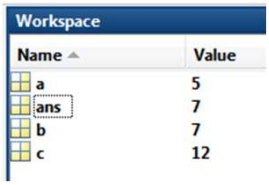
Workspace	
Name	Value
ans	7

Defining Variables

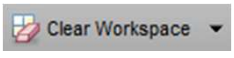
- Type these commands:
- `a=5`
- `b=7`
- `c=a+b`

- All defined variables can be seen on Workspace

```
>> a=5

a =

     5

>> b=7

b =

     7

>> c=a+b

c =

    12
```



"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

Mathematical Functions

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

Trigonometric Functions - 1

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

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

Example

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

Quadratic Formula

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

Solution

inputs

Command Window

```
>> a=1

a =

     1

>> b=-2

b =

    -2

>> c=-15

c =

   -15
```

outputs

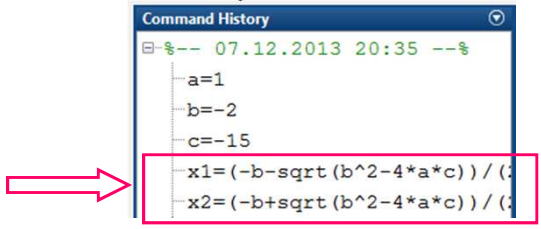
```
>> x1=(-b-sqrt(b^2-4*a*c))/(2*a)

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

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



Example

- Find the BOD_5 (y) for a wastewater with
- ultimate BOD (L) = 282 mg/L
- k_1 = 0.348 /day

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

Solution

```
L =

   282

>> t=5

t =

     5

>> k1=0.348

k1 =

    0.3480
```

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

y =

  232.5032
```

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×n):
  - m: number of rows
  - n: number of columns

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

Special Matrices

- Vector Matrix (one row or one column matrix)
- Row vector (size: 1×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×1)
- `H=[1;0;6;-7;8]`

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

H =

     1
     0
     6
    -7
     8
```


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
```

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

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)

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

A

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)

Getting Information about Matrices

- `numel()`
  - Gives number or elements in matrix

```
>> numel(A)

ans =

    16
```

- `size()`
  - Gives dimensions of matrix

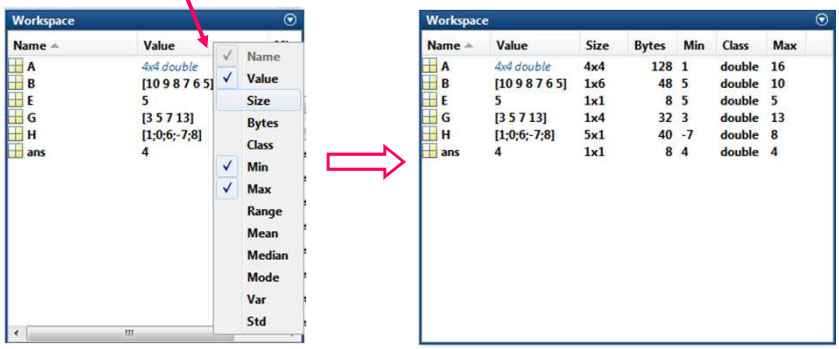
```
>> size(A)

ans =

     4     4
```

- `length()`
  - Gives length of matrix

```
>> length(A)

ans =

     4
```


Changing Workspace View

Right click here



Select properties, you want to see

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
```


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
```

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
```


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
```

Transpose of a Matrix

- ' : gives transpose of a matrix

```
A =

     6     0    -7
     2     5    -8

>> A'

ans =

     6     2
     0     5
    -7    -8
```


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
```

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    1.0000         0
    0.0000   -0.0000    1.0000
```


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

>> max(A)

ans =

    0.9502

>> min(A)

ans =

    0.0344
```

```
>> sum(A)

ans =

    2.8169

>> median(A)

ans =

    0.4102

>> std(A)

ans =

    0.3172
```

```
>> sort(A)

ans =

    0.0344    0.3171    0.3816    0.4387    0.6948    0.9502
```