# Structural Engg *(Lec-1)*

# What is MATLAB?

- MATLAB stands for matrix laboratory
- MATLAB provides a language and environment for numerical computation, data analysis, visualisation and algorithm development
- MATLAB provides functions that operate on
- Integer, real and complex numbers
- Vectors and matrices
- Structures

# Functionality

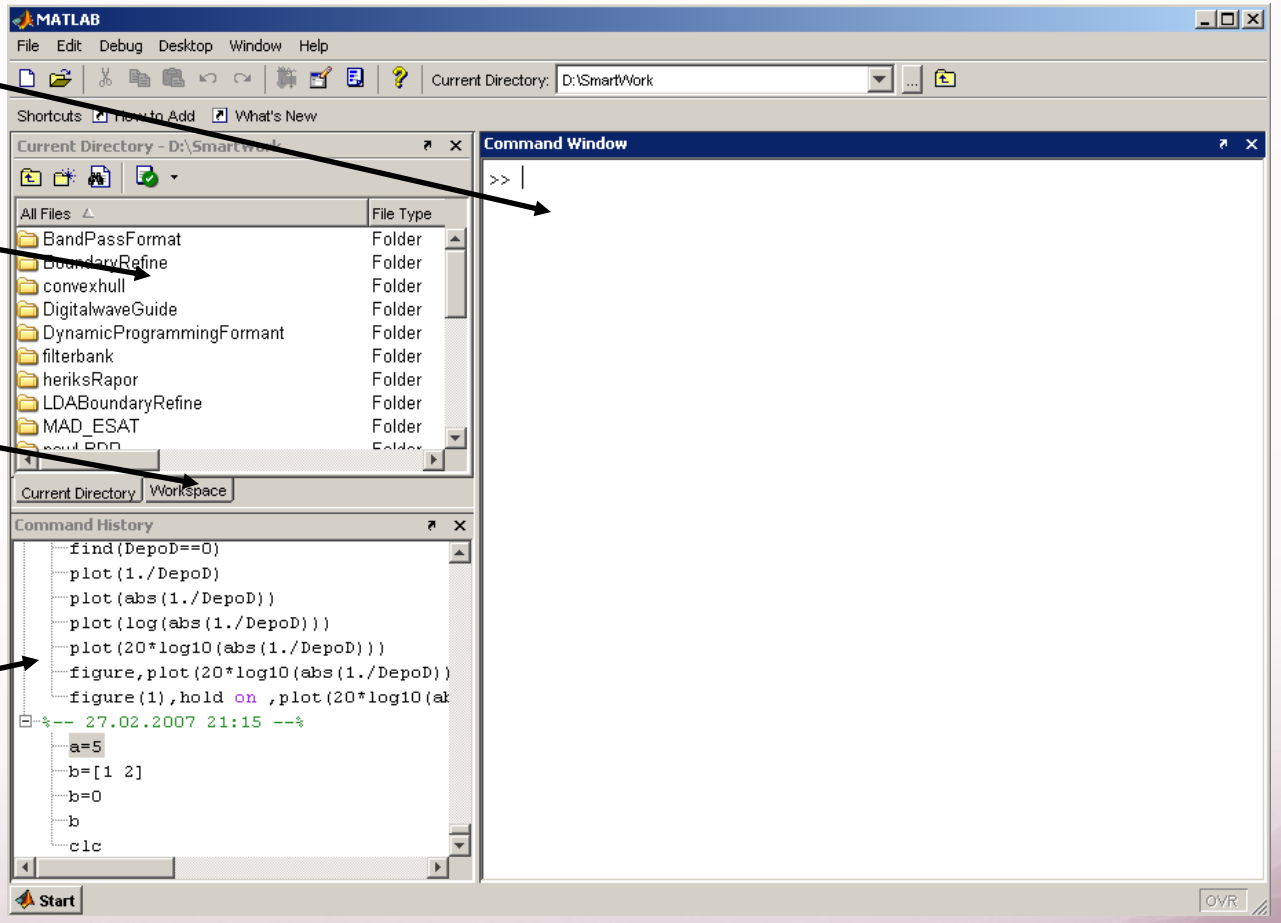
- Built-in Functionality includes
  - Matrix manipulation and linear algebra
  - Mathematical computation
  - Algorithm development
  - Data acquisition
  - Modeling, simulation, and prototyping
  - Data analysis, exploration, and visualization
  - Scientific and engineering graphics
- Add-on toolboxes provide
  - Image processing
  - Signal Processing
  - Optimization
  - Genetic Algorithms

# MATLAB paradigm

- MATLAB is an interactive environment
  - Commands are interpreted one line at a time
  - Commands may be scripted to create your own functions or procedures
- Variables are created when they are used
- Variables are typed, but variable *names* may be reused for different types
- Basic data structure is the matrix
  - Matrix dimensions are set dynamically

# MATLAB Desktop

- Command Window
  - type commands
- Current Directory
  - View folders and m-files
- Workspace
  - View program variables
  - Double click on a variable to see it in the Array Editor
- Command History
  - view past commands
  - save a whole session using diary

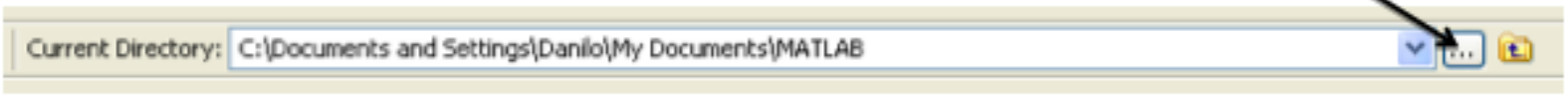


- Command Window: Where you enter commands
- Command History: running history of commands

  which is *preserved  across MATLAB sessions*
- Current directory: Default is *$matlabroot/work*
- Workspace: GUI for viewing, loading and saving MATLAB variables
- Editor/Debugger: text editor, debugger; editor works with file types in addition to .m (MATLAB “m files”)

# Making Folders

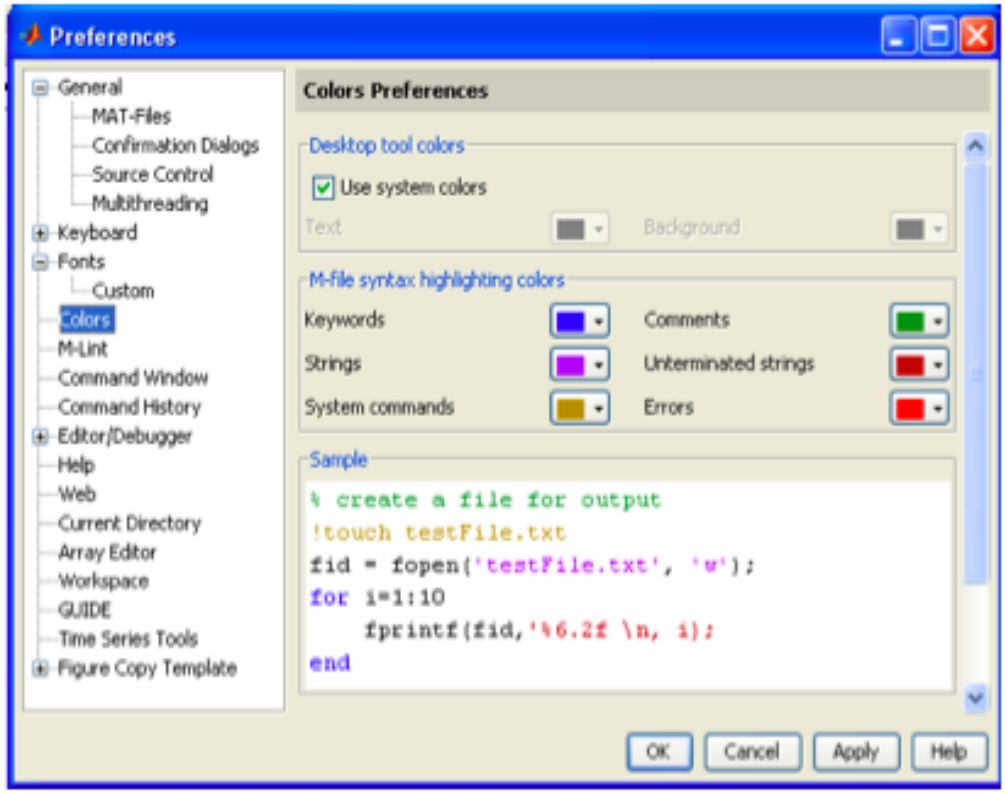
- Use folders to keep your programs organized
- To make a new folder, click the 'Browse' button next to 'Current Directory'




- Click the 'Make New Folder' button, and change the name of the folder. **Do NOT use spaces** in folder names. In the MATLAB folder, make two new folders: IAPMatlab\day1
- Highlight the folder you just made and click 'OK'
- The current directory is now the folder you just created
- To see programs outside the current directory, they should be in the Path. Use File-> Set Path to add folders to the path

# Customization

- File → Preferences
  - Allows you personalize your MATLAB experience



# Help/ Doc

- **help**
  - **The most** important function for learning MATLAB on your own
- To get info on how to use a function:

  **» help sin**
  - Help lists related functions at the bottom and links to the doc
- To get a nicer version of help with examples and easy-to-read descriptions:

  **» doc sin**

- **>> lookfor** …"keyword search"… to search for a particular string in Help text of functions
- **>> demo …** for a demonstartion program

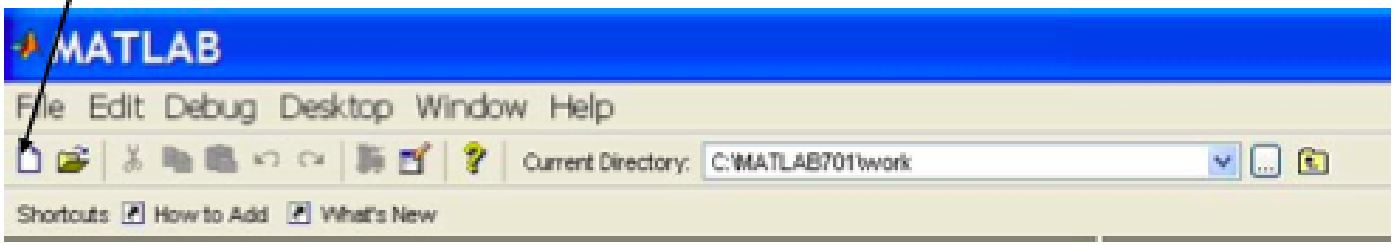
# Scripts…Overview

- Scripts are
  - collection of commands executed in sequence
  - written in the MATLAB editor
  - saved as m-files (.m extension)

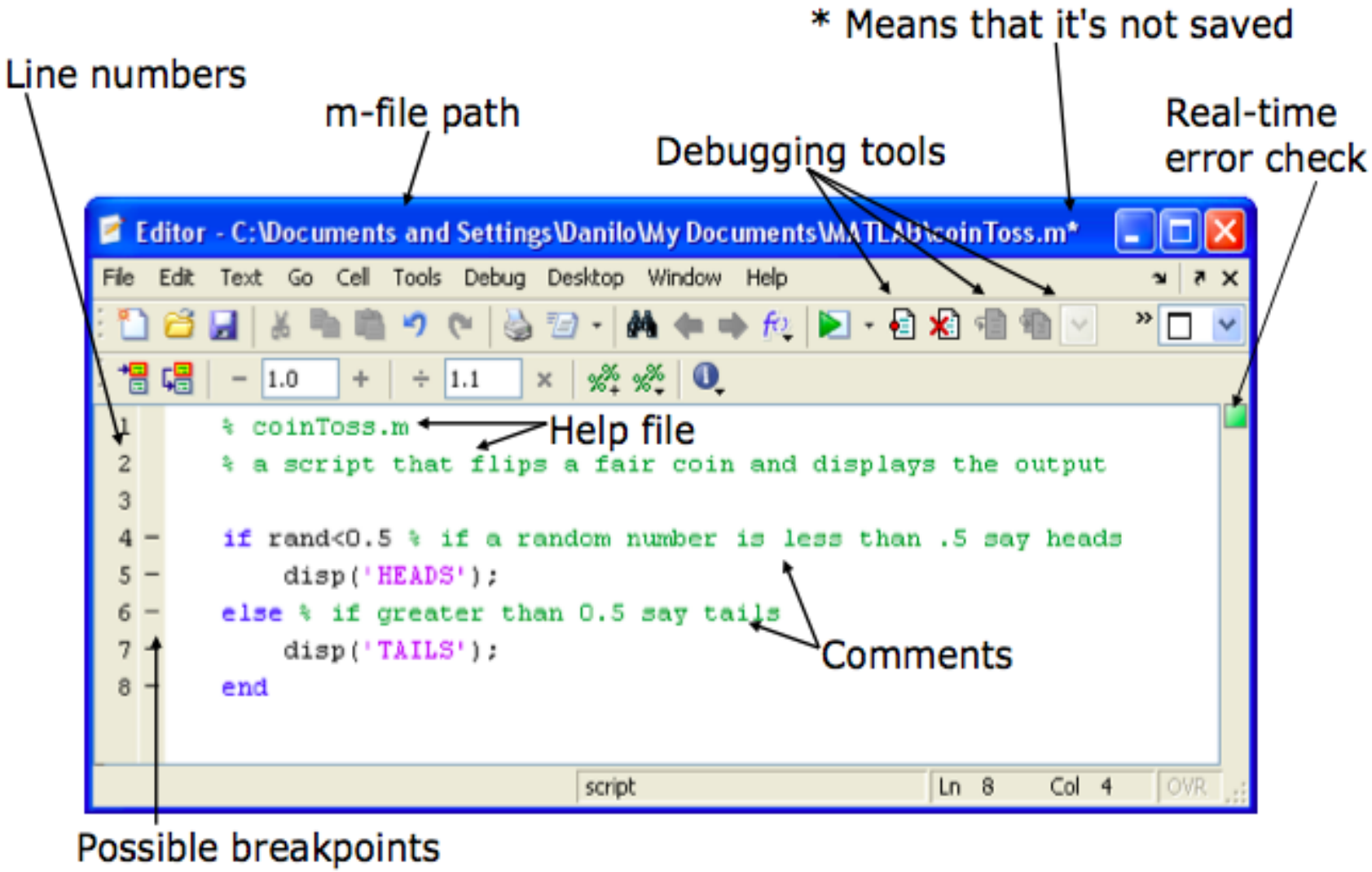
- To create an m-file from command-line

```
» edit helloWorld.m
```

- or click



# Scripts…the Editor



# Scripts…Some Notes

**Comments**

- Anything following a % is seen as a comment.
- Comment thoroughly to avoid wasting time later.

# Quitting MATLAB

- To end your MATLAB session

  **File -> Exit MATLAB**… in the desktop

  OR

- Type

  **quit / exit**… in the command window

# Arithmatic

- To sum up — x **+** y <Enter>
- Subtraction — x **-** y <Enter>
- Multiplication — x ***** y <Enter>
- Division — x **/** y <Enter>

  x **** y <Enter>
- Exponent — x **^** y <Enter>

Try with: x=2, y=3

# Operators Precedence

**Table 2.1** Arithmetic Operations between Two Scalars

| 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 |

# Variables

- Assign variables to do the arithmetic operations
- To use the result in further calculations

>> a=2 <Enter>

>> a= a+7 <Enter>

>> a=a*10 <Enter>

>> b=3 <Enter>

>> z=a+b

**NOTE:** In the last case the latest result is saved in a variable **Ans** replacing the result of the previous calculation.

# Variable Names

- It may consist only of:
  - Letters from a-z
  - Numbers from 0-9
  - underscore ( _ )
- It must start with a letter

**NOTE:** MATLAB is case sensitive

# Matrices & Vectors

- All (almost) entities in MATLAB are matrices
- Easy to define:

```
>> A = [16 3; 5 10]
  A =    16     3
          5    10
```

- Use ‘,’ or ‘ ’ to separate row elements -- use ‘;’ to separate rows

# Matrices & Vectors - II

- Order of Matrix -
  - m=no. of rows, n=no. of columns

$$m \times n$$

- Vectors - special case
  - n = 1 column vector
  - m = 1 row vector

# Creating Vectors and Matrices

- Define

```
>> A = [16 3; 5 10]
   A =    16     3
          5    10
>> B = [3 4 5
   6 7 8]
   B = 3  4  5
       6  7  8
```

- Transpose

Vector :

```
>> a=[1 2 3];
>> a'
     1
     2
     3
```

Matrix:

```
>> A=[1 2; 3 4];
>> A'
ans =
     1     3
     2     4
```

# Creating Vectors

Create vector with equally spaced intervals

```
>> x=0:0.5:pi
x =
 0 0.5000 1.0000 1.5000 2.0000 2.5000 3.0000
To obtain non-unit spacing, specify an increment.
For example: 100:-7:50       will give you
         100    93    86    79    72    65    58    51
```

Create vector with *n* equally spaced intervals

```
>> x=linspace(0, pi, 7)
x =
   0 0.5236 1.0472 1.5708 2.0944 2.6180 3.1416
```

Equal spaced intervals in logarithm space

```
>> x=logspace(1,2,7)
x =
   10.0000 14.6780 21.5443 … 68.1292  100.0000
```

Note: MATLAB uses pi to represent $\pi$ , uses i or j to represent imaginary unit

# Creating Matrices

- `zeros(m, n)`: matrix with all zeros
- `ones(m, n)`: matrix with all ones.
- `eye(m, n)`: the identity matrix
- `magic(m)`: square matrix whose elements have the same sum, along the row, column and diagonal.
- `pascal(m)` : Pascal matrix( product of Upper triangulat matrix and lower triangular matrix)

# Matrix operations

- `^`: exponentiation
- `*`: multiplication
- `/`: division
- `\`: left division. The operation `A\B` is effectively the same as `INV(A)*B`, although left division is calculated differently and is much quicker.
- `+`: addition
- `-`: subtraction

# Array Operations

- Evaluated element by element
  - `.'` : array transpose
  - `.^` : array power
  - `.*` : array multiplication
  - `./` : array division
- Very different from Matrix operations

```
>> A=[1 2;3 4];
>> B=[5 6;7 8];
>> A*B
    19    22
    43    50
```

But:

```
>> A.*B
     5    12
    21    32
```

# Some Built-in functions

- `mean(A)`: mean value of a vector
- `max(A), min (A)`: maximum and minimum.
- `sum(A)`: summation.
- `sort(A)`: sorted vector
- `median(A)`: median value
- `std(A)`: standard deviation.
- `det(A)` : determinant of a square matrix
- `dot(a,b)`: dot product of two vectors
- `Cross(a,b)`: cross product of two vectors
- `Inv(A)`: Inverse of a matrix A

# Mathematical Functions

- MATLAB has all of the useful mathematical functions.

>> x=4; <Enter>

>> sqrt(x)

>> sin (x)  excepts ‘x’ in radians[Syntax 4 sin(90): **sin(90*pi/180**)

>> cos(x)

>> log (x)

>> exp(x)  exponential function $\mathbf{e^x}$

>> pi*x

**NOTE**: clear, clc

# Indexing Matrices

Given the matrix:

```
A =          n
        0.9501    0.6068    0.4231
m       0.2311    0.4860    0.2774
```

Then:

```
A(1,2) = 0.6068
```

$\longrightarrow A_{ij}, i = 1...m, j = 1...n$

```
A(3) = 0.6068
```

$\longrightarrow index = (i-1)m + j$

```
A(:,1) = [0.9501
```

1:*m*

```
          0.2311 ]

A(1,2:3)=[0.6068    0.4231]
```