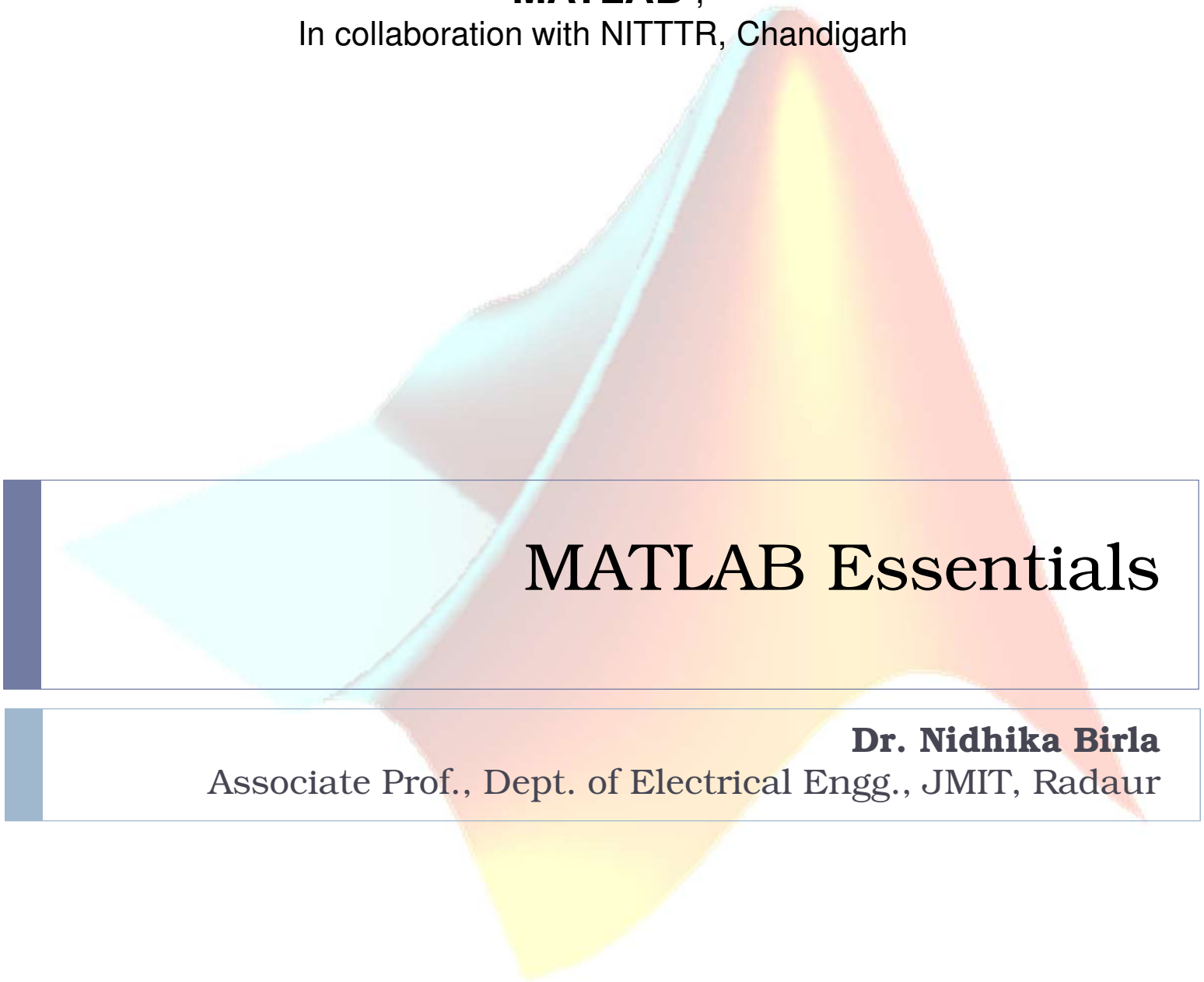
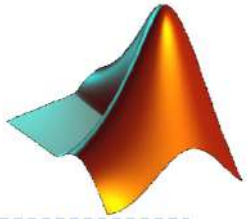
Short Term Training Program on
"**MATLAB**",
In collaboration with NITTTR, Chandigarh



MATLAB Essentials

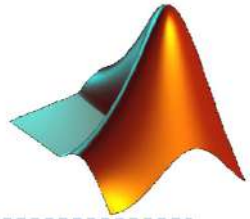
Dr. Nidhika Birla
Associate Prof., Dept. of Electrical Engg., JMIT, Radaur

MATrix LABoratory



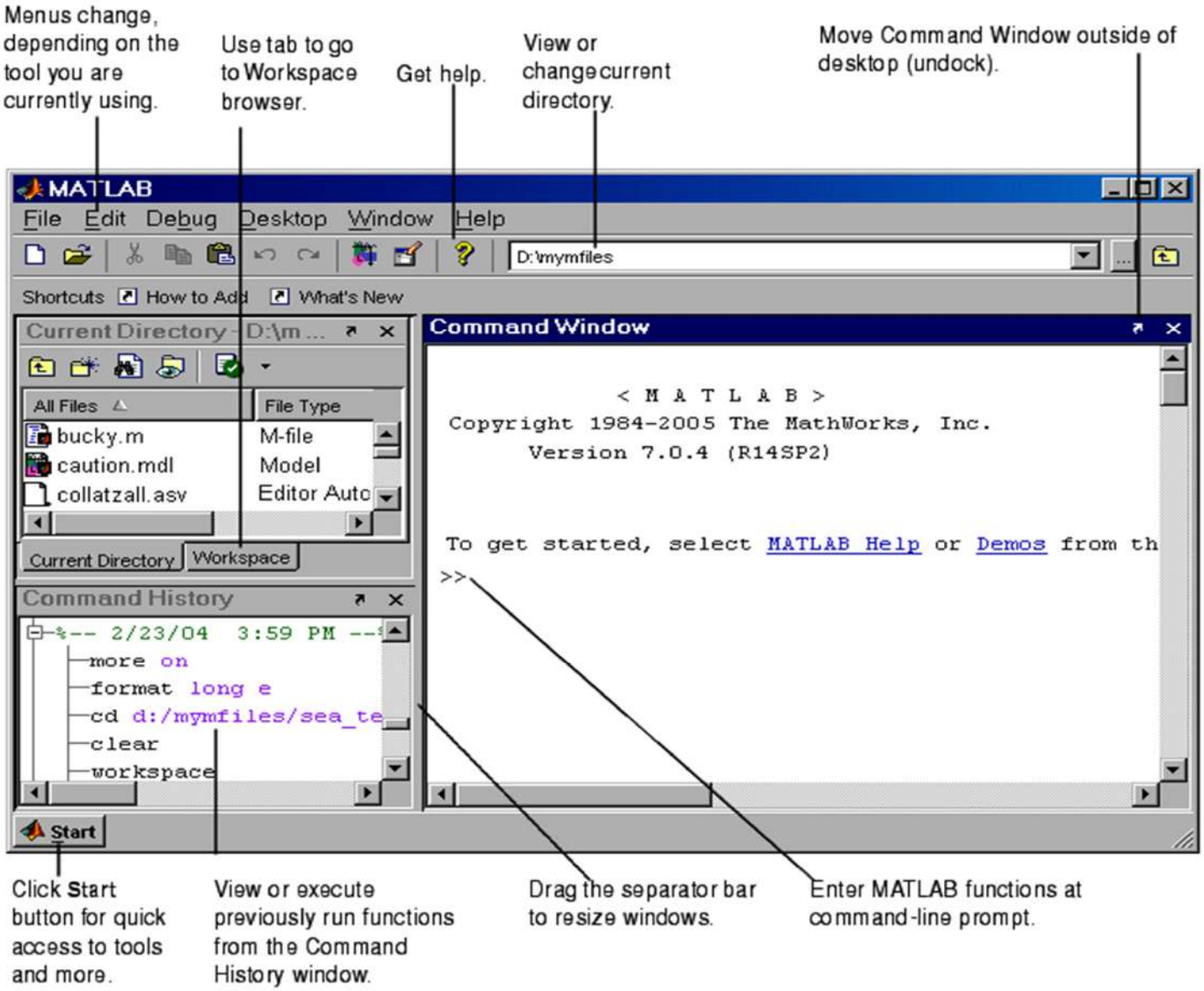
- Powerful, extensible, highly integrated computation, programming, visualization, and simulation package.
- Widely used in engineering, mathematics, and science.

MATLAB's Appeal

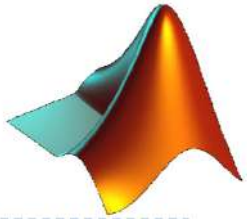


- Interactive code development proceeds incrementally; excellent development and rapid prototyping environment.
- Basic data element is the auto-indexed array.
- This allows quick solutions to problems that can be formulated in vector or matrix form.
- Powerful GUI tools.
- Large collection of *toolboxes*: collections of topic-related MATLAB functions that extend the core functionality significantly.

Basic Interface

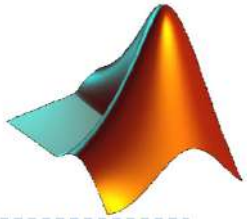


Few Useful Tips



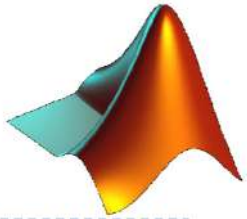
- To clear the Command Window, type clc.
- Commands can be retrieved with arrow up / arrow down keys ↓↑.
- Moving around the command line is possible with left / right arrow keys → ←. Go to the beginning of the line with Home and to the end with End. Esc deletes the whole line.
- A program can be stopped with Ctrl+c.

Few Useful Tips



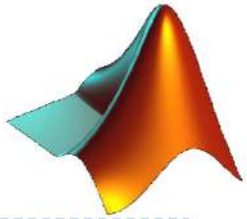
- There **can not be empty spaces** in the name of the Matlab files.
- Write "**;**" at the end of a line If you don't want that the intermediate calculus is written in the window while the program is running.
- Write "**%**" at the beginning of a line to write a comment in the program.
- Write "**…**" at the end of a line if you are writing a very long statement and you want to continue in the next line.

Getting Help



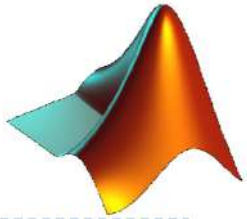
- Help Command
  - >> *help FunctionName*
- Lookfor Command
  - >> lookfor FunctionName
- The *help* command searches for an exact function name match, while the *lookfor* command searches the quick summary information in each function for a match.

Managing Workspace



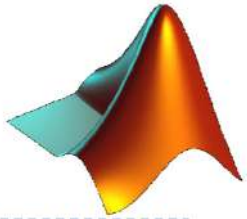
- The command *clear* or *clear all* removes all variables from the workspace. This frees up system memory.
  - >> clear all
- To display a list of the variables currently in the memory, type
  - >> who
- while, *whos* will give more details which include size, space allocation, and class of the variables.

MATLAB as Scratchpad



Arithmetic operation	Symbol	Example
Addition	+	$6+3=9$
Subtraction	–	$6-3=3$
Multiplication	*	$6*3=18$
Right division	/	$6/3=2$
Left division	\	$6\backslash 3=3/6=1/2$
Exponentiation	^	$6\,\hat{}\,3=6^3=216$

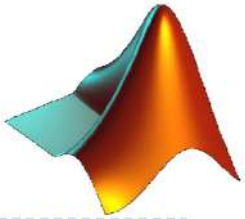
MATLAB as Scratchpad



Calculate the area of a circle with a radius of 2.5 m.

```
» area = pi * 2.5^2
area =
        19.6350
```

Variables



- No need for types. i.e.,

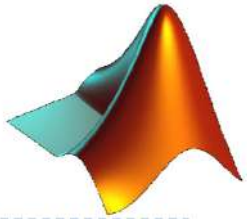
```
int x;
double y;
float z;
```

- All variables are created with double precision unless specified and they are matrices.

```
Example:
>>a=15;
>>b=24;
```

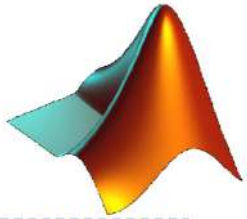
- After these statements, the variables are 1x1 matrices with double precision.

Variables



- Resize arrays dynamically.
- To reuse a variable name, simply use it in the left hand side of an assignment statement.
- Variable names are case sensitive and over-written when re-used.
- Terminology:
  - "scalar" (1 x 1 array),
  - "vector" (1 x N array),
  - "matrix" (M x N array)
- Representation of Real Numbers: $4.3e\text{-}3 = 4.3 * 10^{-3}$

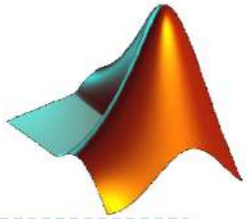
Special Variables



- **Characteristic constants**: pi=π, NaN (not a number, 0/0), Inf=∞.

- **Complex numbers**: i=sqrt(-1) (only i or j can be used),
  - z=2+i*4,
  - z=2+4i

Play with Variables



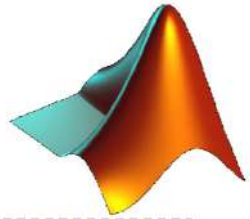
Suppose that $x = 3$ and $y = 4$. Use MATLAB to evaluate the following expression:

$$\frac{x^2y^3}{(x-y)^2}$$

```
» x = 3;
» y = 4;
» res = x^2 * y^3 / (x - y)^2
res =
    576
```

From Online Resources

Play with Variables



The distance traveled by a ball falling in the air is given by the equation

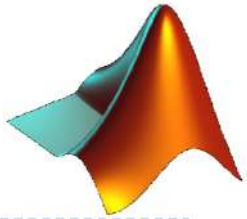
$$x = x_0 + v_0 t + \frac{1}{2} a t^2$$

Use MATLAB to calculate the position of the ball at time $t = 5$ s if $x_0 = 10$ m, $v_0 = 15$ m/s, and $a = -9.81$ m/sec^2.

```
» t = 5;
» x0 = 10;
» v0 = 15;
» a = -9.81;
» x = x0 + v0 * t + 1/2 * a * t^2
x =
   -37.6250
```

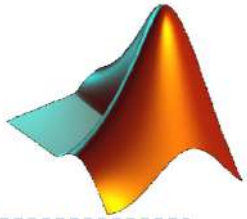
From Online Resources

M Files



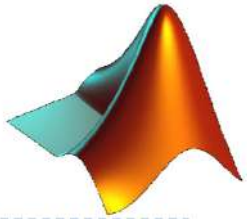
- An M-file is a MATLAB document the user creates to store the code they write for their specific application.
- Creating an M-file – To create an M-file, select File\New ►M-file.
- Saving – The next step is to save the newly created M-file. In the M-file window, select File\Save As…
- Opening an M-file – To open up a previously designed M-file, simply open MATLAB in the same manner as described before. Then, open the M-file by going to File\Open…, and selecting your file.

M Files



- Writing Code
- Resaving
- Running Code – To run code, simply go to the main MATLAB window and type the name of your M-file after the >> prompt. Other ways to run the M-file are to press F5 while the M-file window is open, select Debug\Run, or press the Run button in the M-file window toolbar.

Vector and Matrix



- **Vector**

```
x = [1 2 5 1]

x =
     1     2     5     1
```

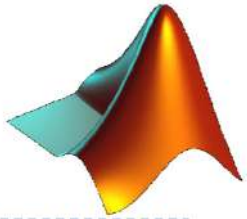
- **Matrix**

```
x = [1 2 3; 5 1 4; 3 2 -1]

x =
     1     2     3
     5     1     4
     3     2    -1
```

- **Transpose**

```
y = x'

y =
     1
     2
     5
     1
```

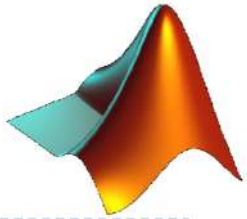
Vectors



- Generating row vectors:
  - Specifying the increment **h** between the elements **v=a:h:b**

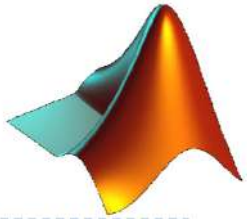
```
t =1:10
 t =
       1    2    3    4   5   6    7   8    9    10
k =2:-0.5:-1
 k =
       2   1.5   1   0.5   0   -0.5   -1
```

Matrices



- Information about an element:
  - *M(1,3)*,
  - A Row *M(2,:)*,
  - A Column *M(:,3)*.
- Changing the value of an element: *M(2,3)=1*;
- Deleting
  - A Column: *M(:,1)=[ ]*,
  - A Row: *M(2,:)=[ ]*;

Generating Vectors and Matrices from Functions



- zeros(M,N) MxN matrix of zeros

```
x = zeros(1,3)
x =
     0     0     0
```

- ones(M,N) MxN matrix of ones

```
x = ones(1,3)
x =
     1     1     1
```

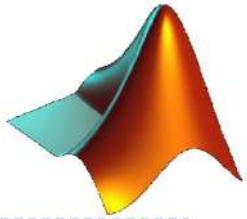
- rand(M,N) MxN matrix of uniformly distributed random numbers on (0,1)

```
x = rand(1,3)
x =
 0.9501  0.2311 0.6068
```

- eye(M,N) MxN identity matrix

```
x = eye(2,2)
x =
     1     0
     0     1
```

Concatenation of Matrices



x = [1 2], y = [4 5], z=[0 0]

A = [x y]

```
1   2   4   5
```

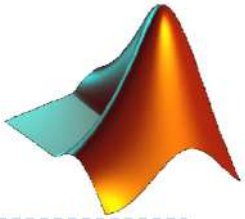
B = [x ; y]

```
1 2
4 5
```

C = [x y ;z]

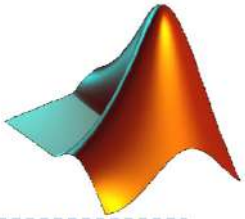
Error:??? Error using ==> vertcat CAT arguments dimensions are not consistent.

Operators



+ Addition
- Subtraction
* Multiplication
/ Division
^ Power
’ Complex Conjugate Transpose

Matrix Operations



Given A and B:

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

A =

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

```
>> B = [3 5 2; 5 2 8; 3 6 9]

B =

     3     5     2
     5     2     8
     3     6     9
```

Addition

```
>> X = A + B

X =

     4     7     5
     9     7    14
    10    14    18
```

Subtraction

```
>> Y = A - B

Y =

    -2    -3     1
    -1     3    -2
     4     2     0
```

Product

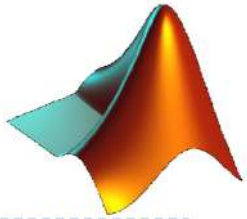
```
>> Z = A * B

Z =

    22    27    45
    55    66   102
    88   105   159
```

Transpose

```
>> T = A'

T =

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

From Online Resources

Operators (Element by Element)



.* element-by-element multiplication

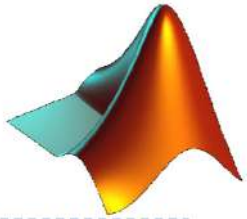
./ element-by-element division

.^ element-by-element power

```
>> A = [1 5 6; 11 9 8; 2 34 78]
A =
     1     5     6
    11     9     8
     2    34    78
>> B = [16 4 23; 8 123 86; 67 259 5]
B =
    16     4    23
     8   123    86
    67   259     5
```

```
>> C = A * B   % "normal" matrix multiply
C =
         458        2173         483
         784        3223        1067
        5530       24392        3360

>> CDOT = A .* B   % element-by-element
CDOT =
          16          20         138
          88        1107         688
         134        8806         390
```

2D Graphics



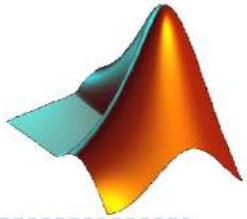
- **plot()** creates a graphic from vectors, with linear scales on both axes,

  >> plot(X,Y,'option') (option: allows chosing color and stroke of the curve, 'Colour Marker Linestyle')

Color style-option		Line style-option		Marker style-option	
y	yellow	–	solid	+	plus sign
m	magenta	– –	dashed	o	circle
c	cyan	:	dotted	*	asterisk
r	red	– .	dash-dot	x	x-mark
g	green			.	point
b	blue			^	up triangle
w	white			s	square
k	black			d	diamond, etc.

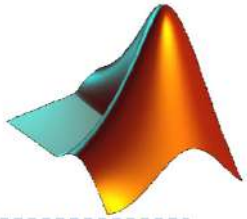
- **hold on:** allows to draw more graphics on the same figure (deactivate with **hold off)**
- **grid** activates a grid on the drawing. Writing grid again deactivates it.

2D Graphics



- **loglog()** logarithmic scale on both axes,
- **semilogx():** logarithmic scale on the abcises axis, and linear on the ordinates axis,
- **semilogy():** linear scale on abscises and logarithmic on ordinates.

2D Graphics



- To open a new figure window:

>> figure

- Graph Annotation

>>legend ('Text')

>> title ('Text')

>> xlabel ('Text')

>> ylabel ('Text')

>>axis([x0 x1 y0 y1])

2D Graphics

- Create an x-array of 100 samples between 0 and 4π.

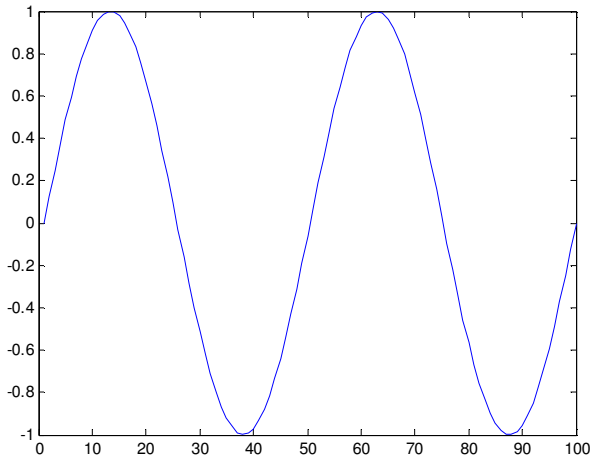
```
>>x=linspace(0,4*pi,100);
```

- Calculate sin(.) of the x-array

```
>>y=sin(x);
```

- Plot the y-array

```
>>plot(y)
```



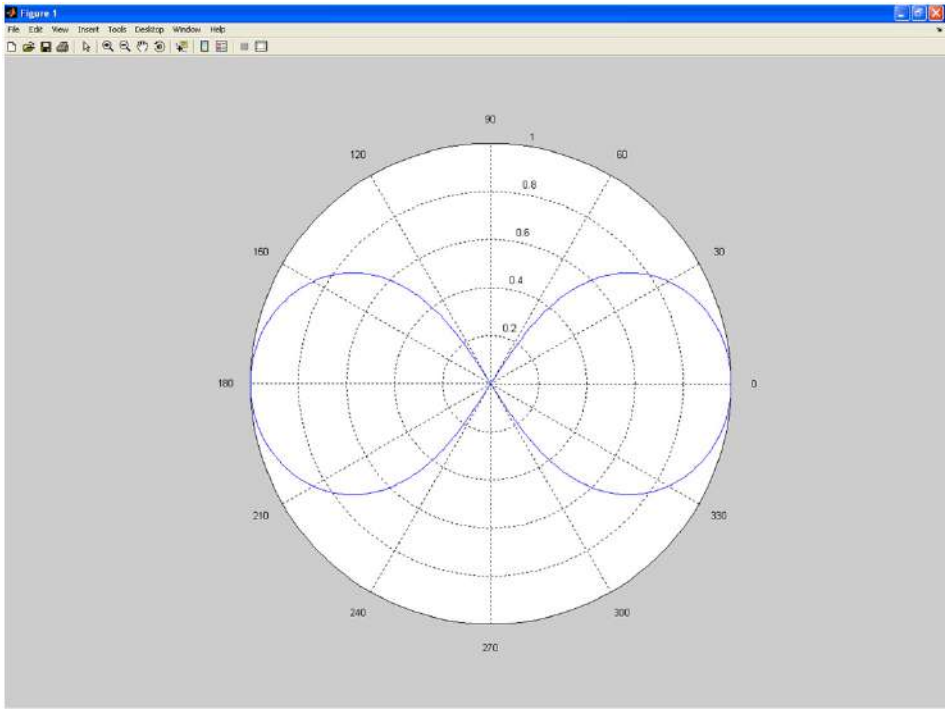
2D Graphics

- Polar coordinates

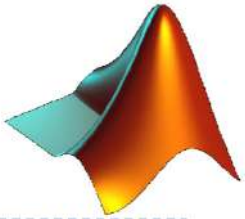
>> polar(theta, rho)

e.g. **>> t = [0 : 0.01 : 2*pi]**

>> polar (sin(t), cos(t))



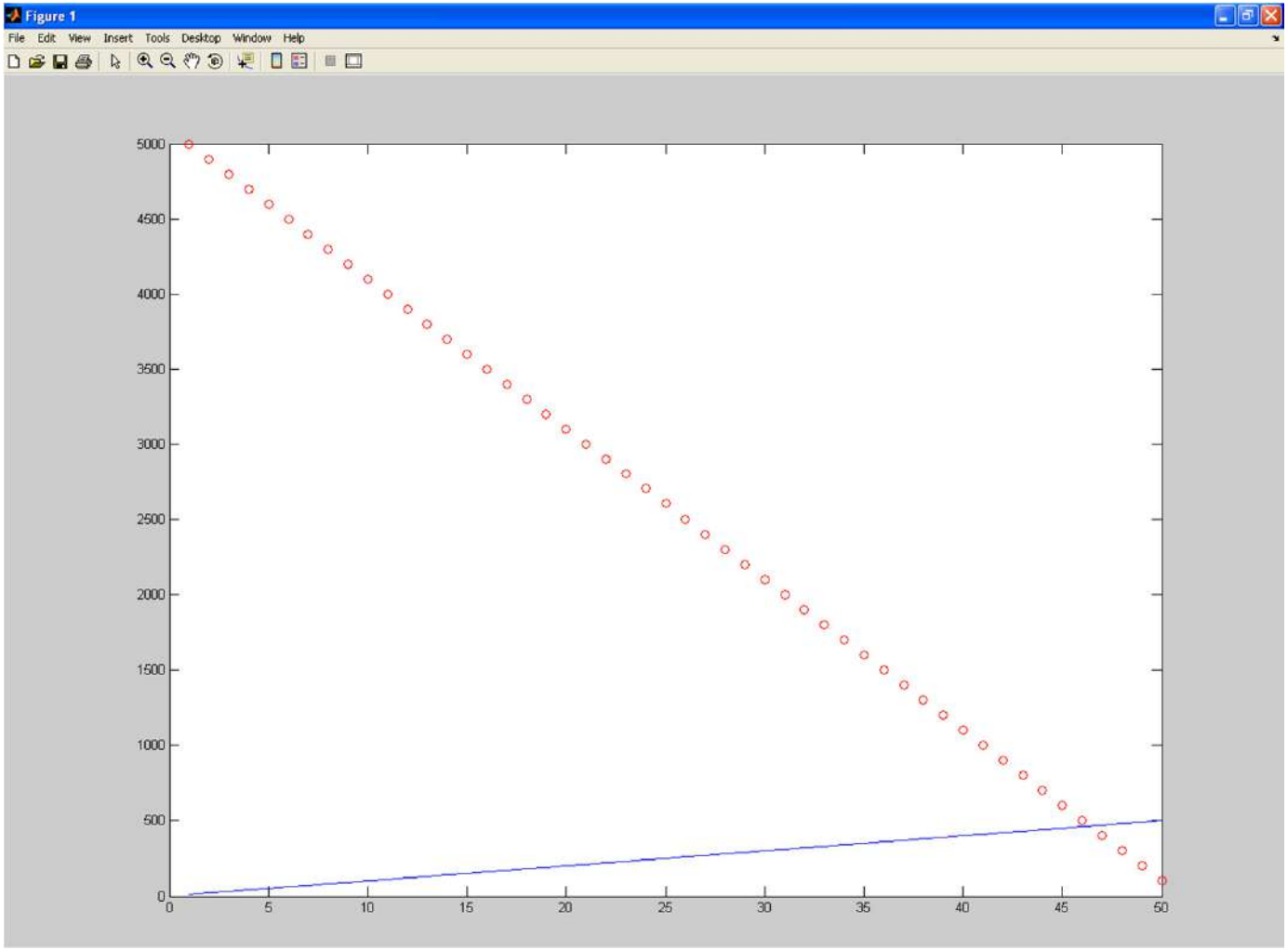
Multiple Plots



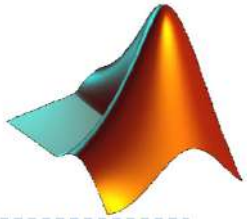
- Multiple datasets on a plot

```
>> p1 = plot(xcurve, ycurve)
>> hold on
>> p2 = plot(Xpoints, Ypoints, 'ro')
>> hold off
```

Multiple Datasets



Multiple Plots



- Subplots on a figure: **subplot(n,m,k)** divides a drawing window in **n** horizontal parts and **m** vertical parts, where **k** is the activated partition.

```
>> s1 = subplot(1, 2, 1)
>> p1 = plot(time, velocity)
>> s2 = subplot(1, 2, 2)
>> p2 = plot(time, acceleration)
```

SubPlot

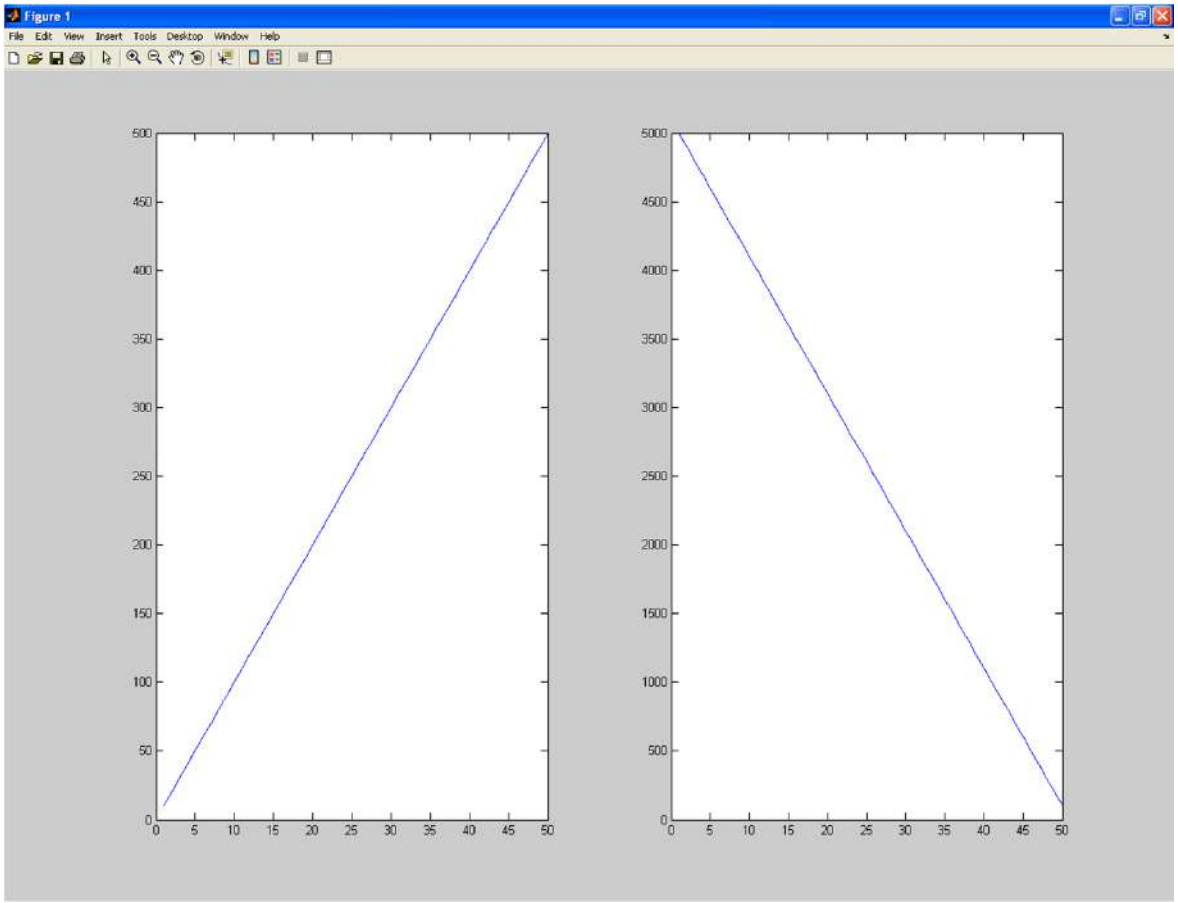
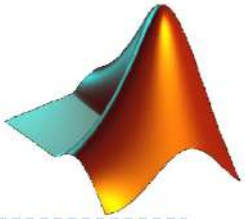


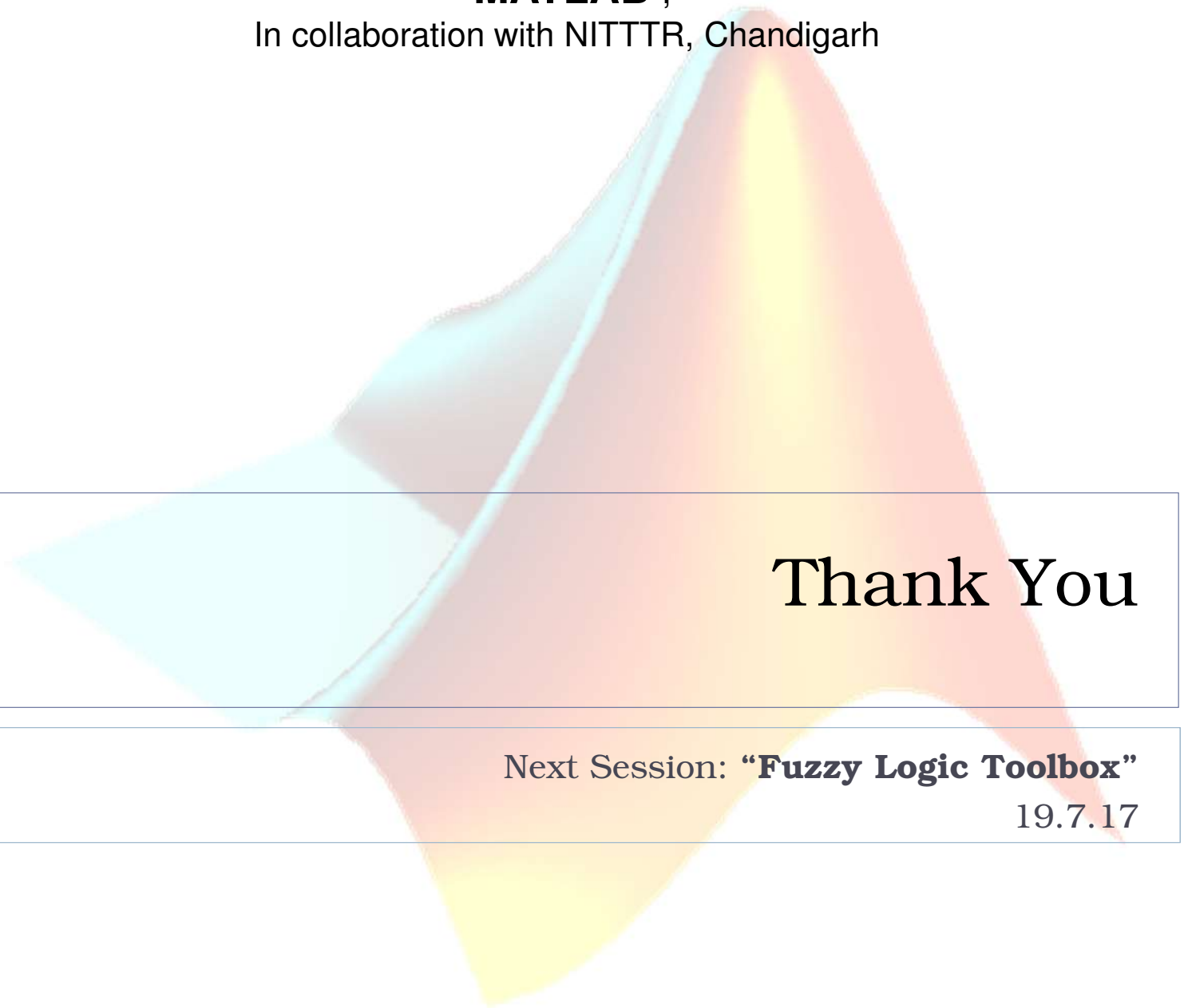
Figure 1
File Edit View Insert Tools Desktop Window Help
500
450
400
350
300
250
200
150
100
50
0
0 5 10 15 20 25 30 35 40 45 50
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0 5 10 15 20 25 30 35 40 45 50

References



- Various online Resources / Books / Tutorials

Short Term Training Program on
"**MATLAB**",
In collaboration with NITTTR, Chandigarh



Thank You

Next Session: **"Fuzzy Logic Toolbox"**
19.7.17