Probability Methods in Engineering

Dr. Safdar Nawaz Khan Marwat
DCSE, UET Peshawar

Lecture 25

safdar@uetpeshawar.edu.pk

MATLAB

- High level language for technical computing
- Stands for **MAT**rix **LAB**oratory
- Everything is matrix

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

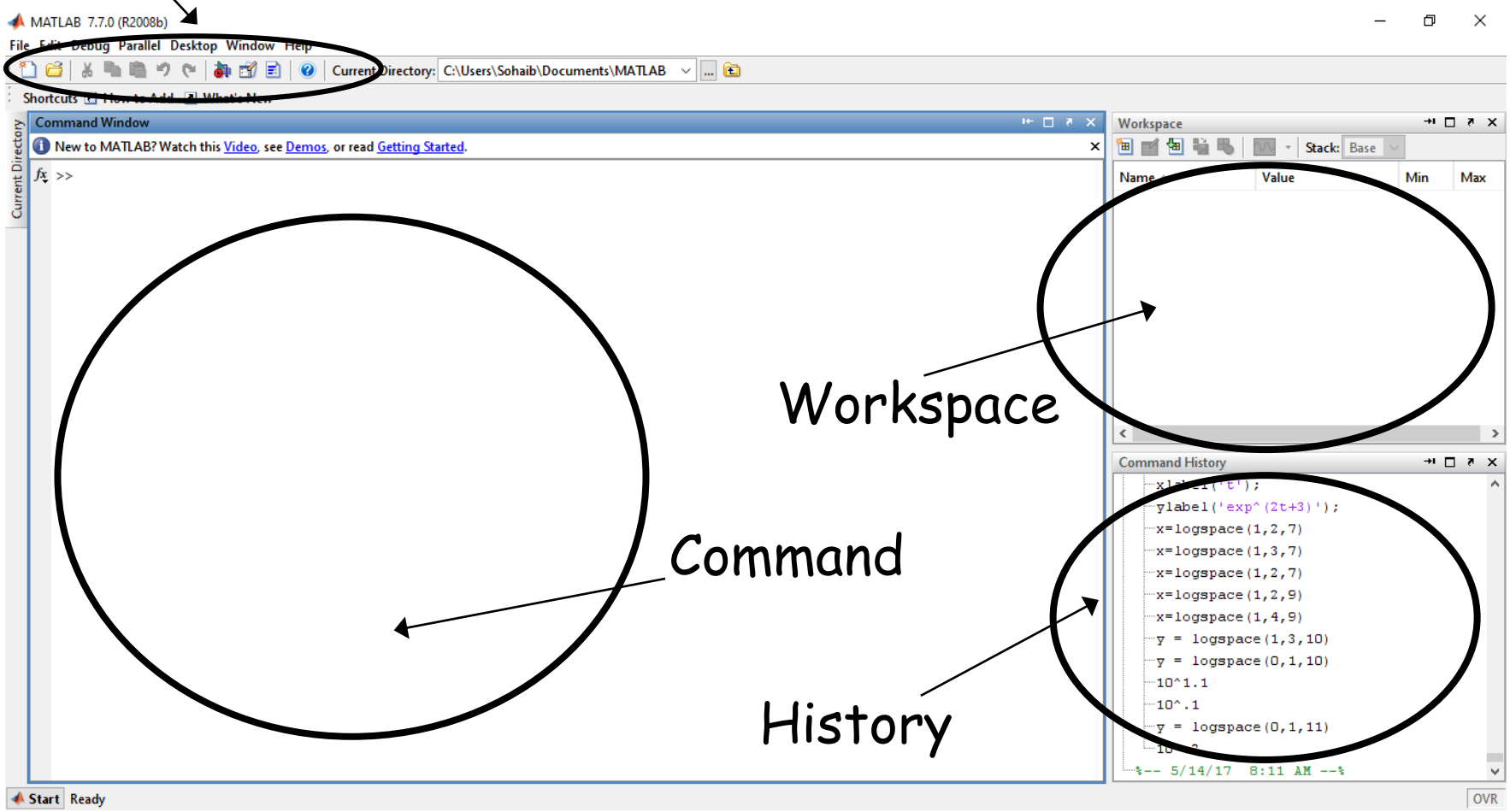
MATLAB System

- Development environment
- Mathematical function library
- MATLAB language

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

MATLAB Desktop

Menu and toolbar



Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Matrices and Vectors

- Almost all entities in MATLAB are matrices
- Easy to define
  - Use ',' or ' ' to separate row elements
  - Use ';' to separate rows
- Order of matrix – m x n
  - m = no. of rows, n = no. of columns
- Vectors - special cases
  - n = 1 column vector
  - m = 1 row vector

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

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
```

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Creating Vectors and Matrices

- `pi` used to represent π
- `i` or `j` used to represent imaginary unit

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
```

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
```

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Creating Vectors and Matrices (cont.)

- `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

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Matrix Operations

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

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Array Operations

- Evaluated element by element
  - `.'` : array transpose (non-conjugated transpose)
  - `.^` : array power
  - `.*` : array multiplication
  - `./` : array division

- 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
```

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

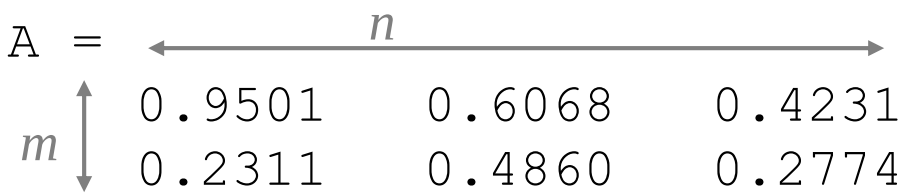
Built-in Functions

- `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
- `inv(A)`: inverse of a matrix A

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Indexing Matrices

- Given the matrix



- Then

```
A(1,2) = 0.6068

A(3) = 0.6068

A(:,1) = [0.9501
   1:m      0.2311 ]

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

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Adding Elements to a Vector or a Matrix

```
>> A=1:3
A=
    1   2   3
>> A(4:6)=5:2:9
A=
    1   2   3   5   7   9

>> B=1:2
B=
    1   2
>> B(5)=7;
B=
    1   2   0   0   7
```

```
>> C=[1  2;  3  4]
C=
    1   2
    3   4
>> C(3,:)=[5  6];
C=
    1   2
    3   4
    5   6
>> D=linspace(4,12,3);
>> E=[C  D']
E=
    1   2   4
    3   4   8
    5   6   12
```

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics

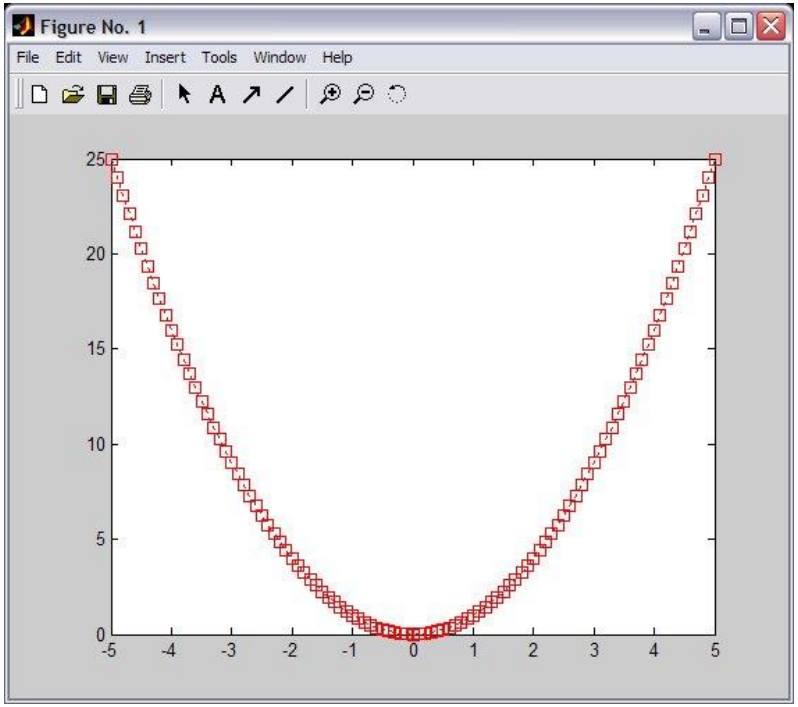
- 2-D plots

```
plot(xdata, ydata, 'marker_style');
```

For example:

```
x=-5:0.1:5;
sqr=x.^2;
pl1=plot(x, sqr, 'r:s');
```

Gives:



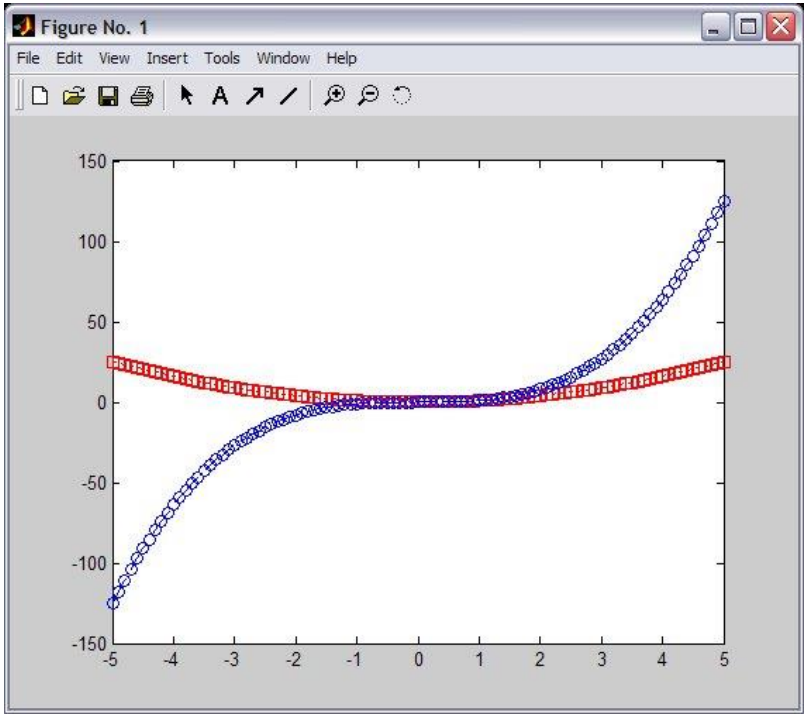
Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics (cont.)

- Overlay plots

Use `hold on` for overlaying graphs

```
hold on
cub=x.^3;
pl2=plot(x, cub, 'b-o');
```



Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics (cont.)

- Annotations

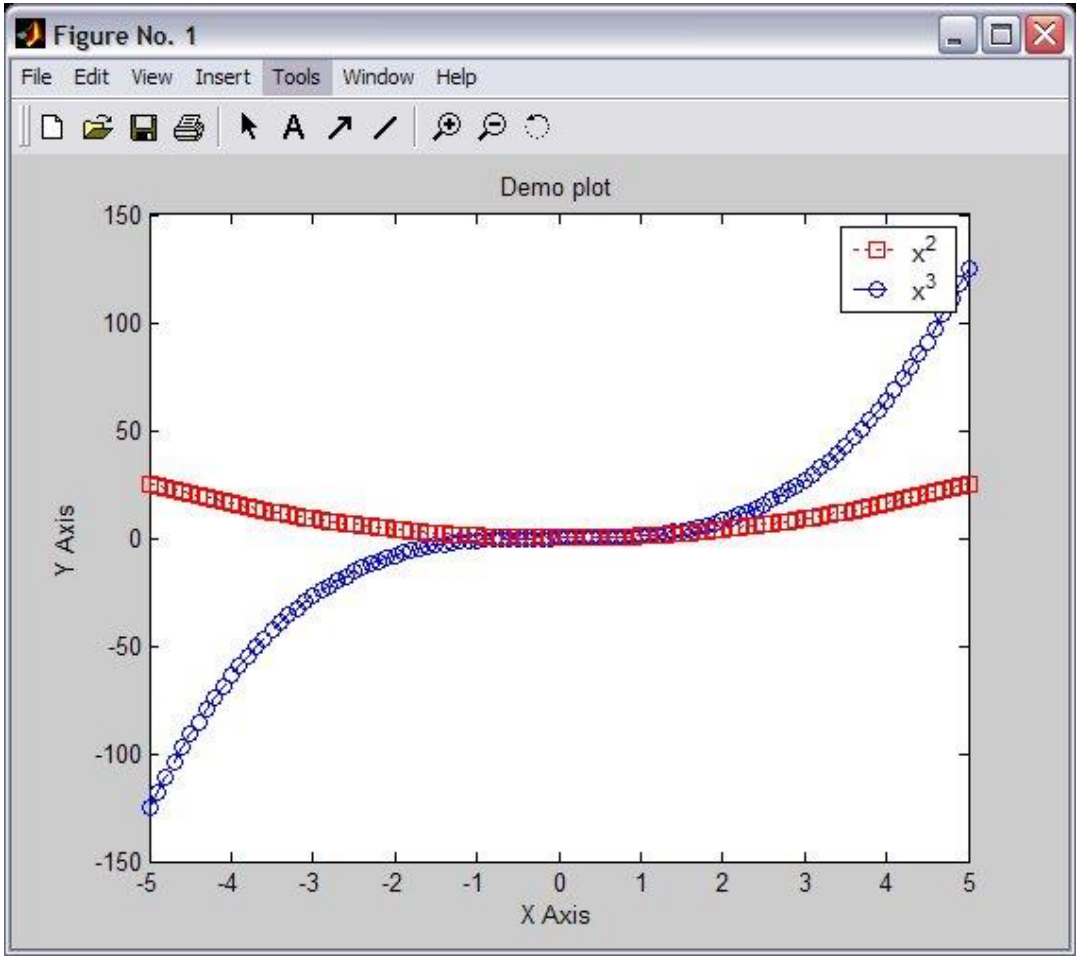
Use `title, xlabel, ylabel` and `legend` for annotation

```
title('Demo plot');
xlabel('X Axis');
ylabel('Y Axis');
legend([pl1, pl2], 'x^2', 'x^3');
```

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics (cont.)

- Annotations



Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

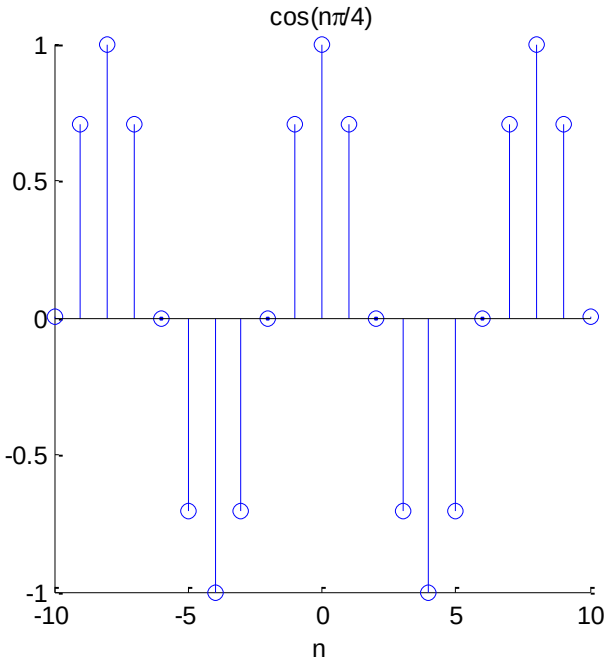
Graphics (cont.)

- Stem

Plot discrete sequence data

Usage of `stem()` similar to `plot()`

```
n=-10:10;
f=stem(n,cos(n*pi/4))
title('cos(n\pi/4)')
xlabel('n')
```



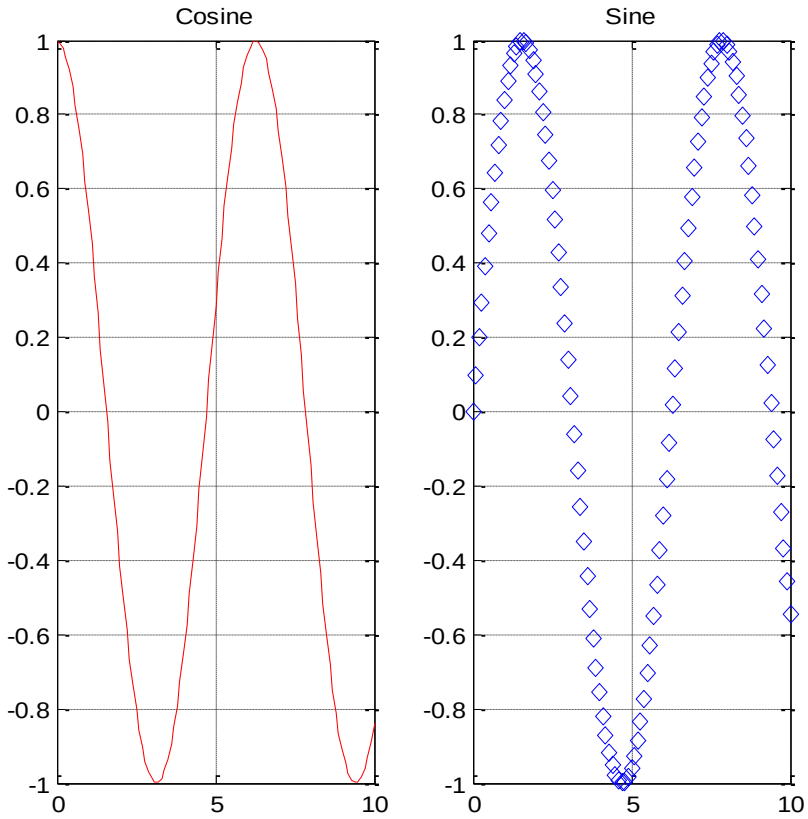
Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics (cont.)

- Subplots

Divide a plotting window into several panes

```
x=0:0.1:10;
f=figure;
f1=subplot(1,2,1);
plot(x,cos(x),'r');
grid on;
title('Cosine')
f2=subplot(1,2,2);
plot(x,sin(x),'d');
grid on;
title('Sine');
```



Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Graphics (cont.)

- Save

Use `saveas(h,'filename.ext')` to save figure to file

```
f=figure;
x=-5:0.1:5;
h=plot(x,cos(2*x+pi/3));
title('Figure 1');
xlabel('x');
saveas(h,'figure1.fig')
saveas(h,'figure1.eps')
```

Useful extension types
bmp: Windows bitmap
emf: Enhanced metafile
eps: Encapsulated Postscript
fig: MATLAB figure
jpg: JPEG image
m: MATLAB M-file
tif: Tagged Image File Format, compressed

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Workspace

- Matlab remembers old commands
  - Variables as well
- Each function maintains its own scope
- The keyword `clear` removes all variables from workspace
- The keyword `who` lists the variables

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

File I/O

- Matlab has a native file format to save and load workspaces
  - Use keywords `load` and `save`
- In addition MATLAB knows a large number of popular formats
  - Type "`help fileformats`" for a listing
- In addition MATLAB supports 'C' style low level file I/O
  - Type "`help fprintf`" for more information

Source: *isites.harvard.edu/fs/docs/icb.topic137910.files/MATLAB_session_1.ppt*

Problems

- Plot the following signals in linear scale

$$x(t) = \sin(3t) \quad -5 < t < 5$$

$$y(t) = e^{2t+3} \quad 0 < t < 5$$