Introduction to Matlab

By: Mohsen Farrokhi

2016/04/26

Outline:

- What is Matlab?
- Matlab Screen
- Variables, array, matrix, indexing
- Operators (Arithmetic, relational, logical)
- Display Facilities
- Flow Control
- Using of M-File
- Writing User Defined Functions
- Conclusion

MATLAB Introduction

MATLAB is both computer programming language and software environment for using that language effectively.

MATLAB is matrix-oriented, so what would take several statements in C or Fortran can usually be accomplished in just a few lines using MATLAB's built-in matrix and vector operations

MATLAB Introduction

FORTRAN:

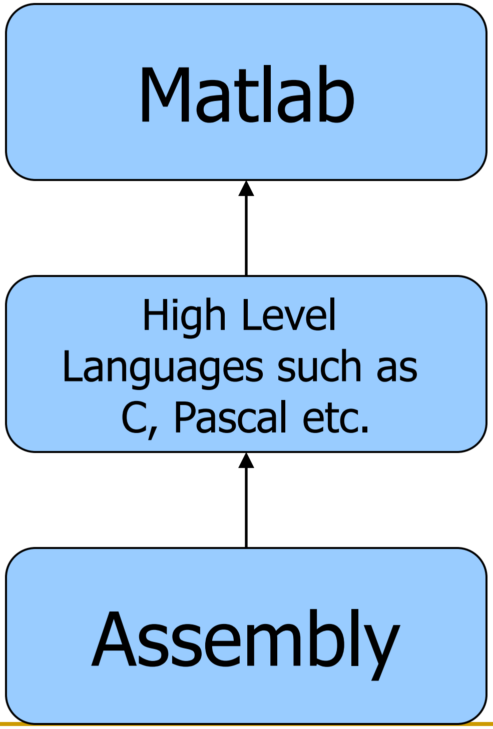
```
real*8 A(10,10), B(10,10), C(10,10)
do i=1,10
do j=1,10
   C(i,j) = A(i,j) + B(i,j)
10 continue
20 continue
```

MATLAB:

```
C = A + B
```

What is Matlab?

- Matlab is basically a high level language which has many specialized toolboxes for making things easier for us
- How high?



MATLAB Introduction

- MATLAB has a number of add-on software modules, called *toolbox* , that perform more specialized computations.

Signal & Image Processing

Signal Processing- Image Processing Communications - System Identification - Wavelet Filter Design

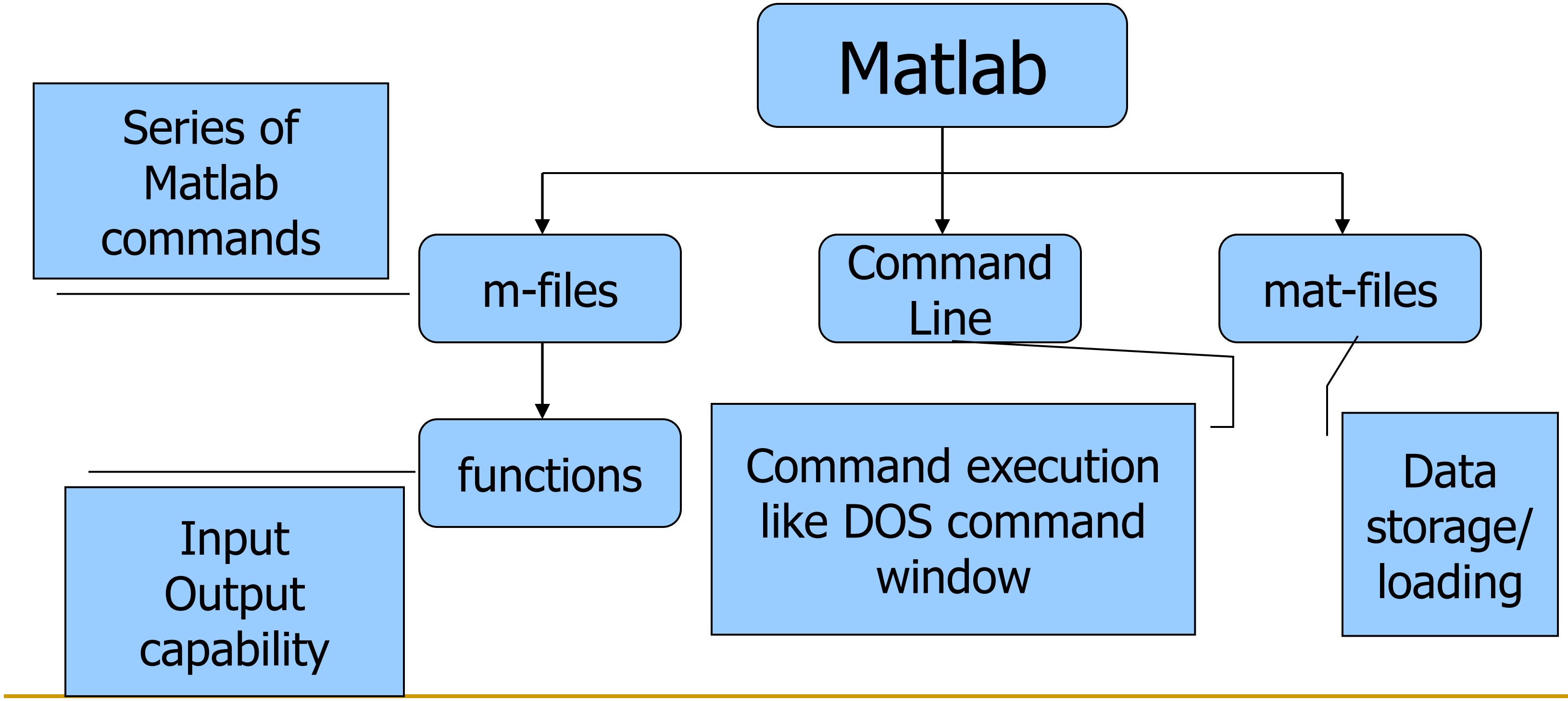
Control Design

Control System - Fuzzy Logic - Robust Control - µ-Analysis and Synthesis - LMI Control - Model Predictive Control Model-Based Calibration

More than 60 toolboxes!

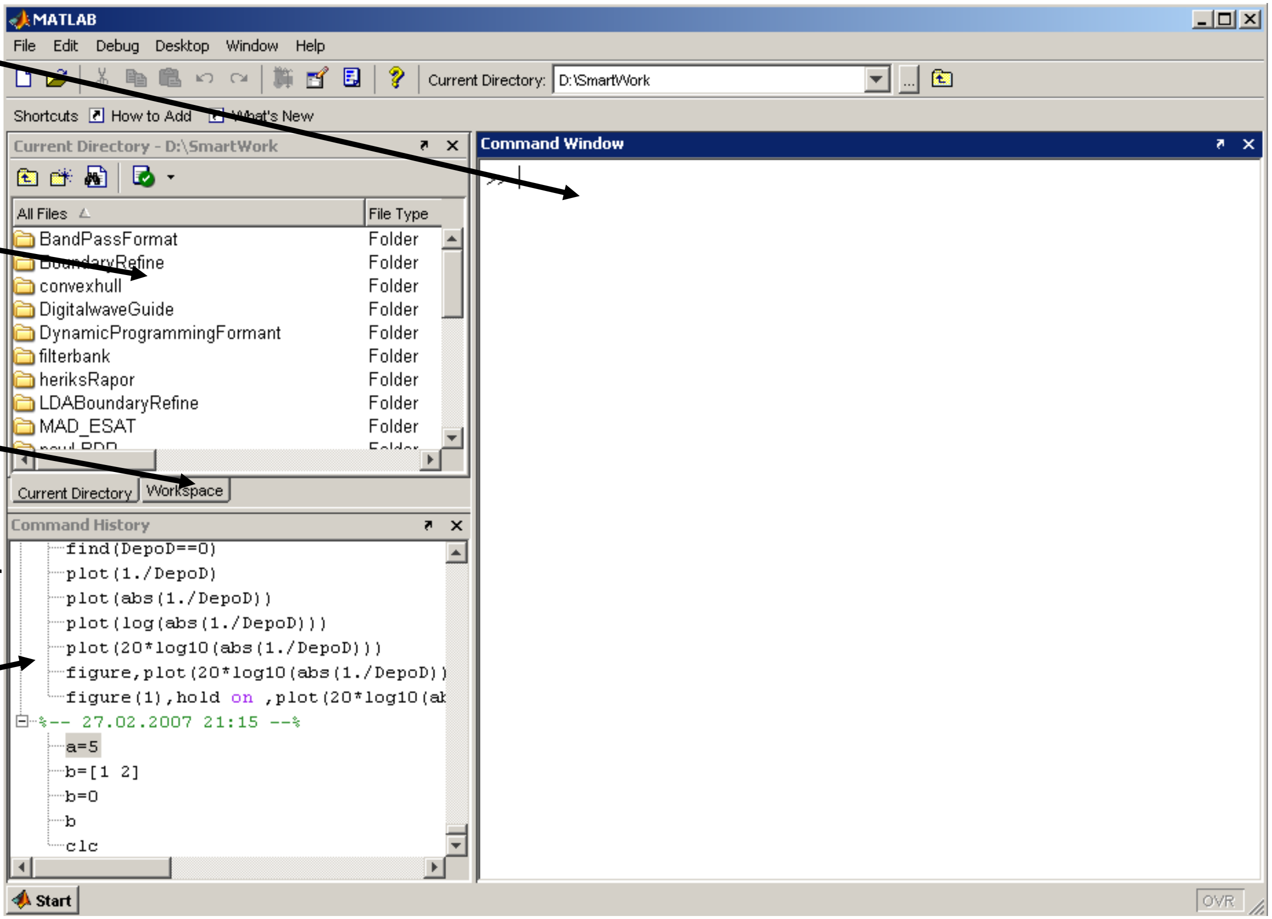
What are we interested in?

- Matlab is too broad for our purposes in this course.
- The features we are going to require is



Matlab Screen

- 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

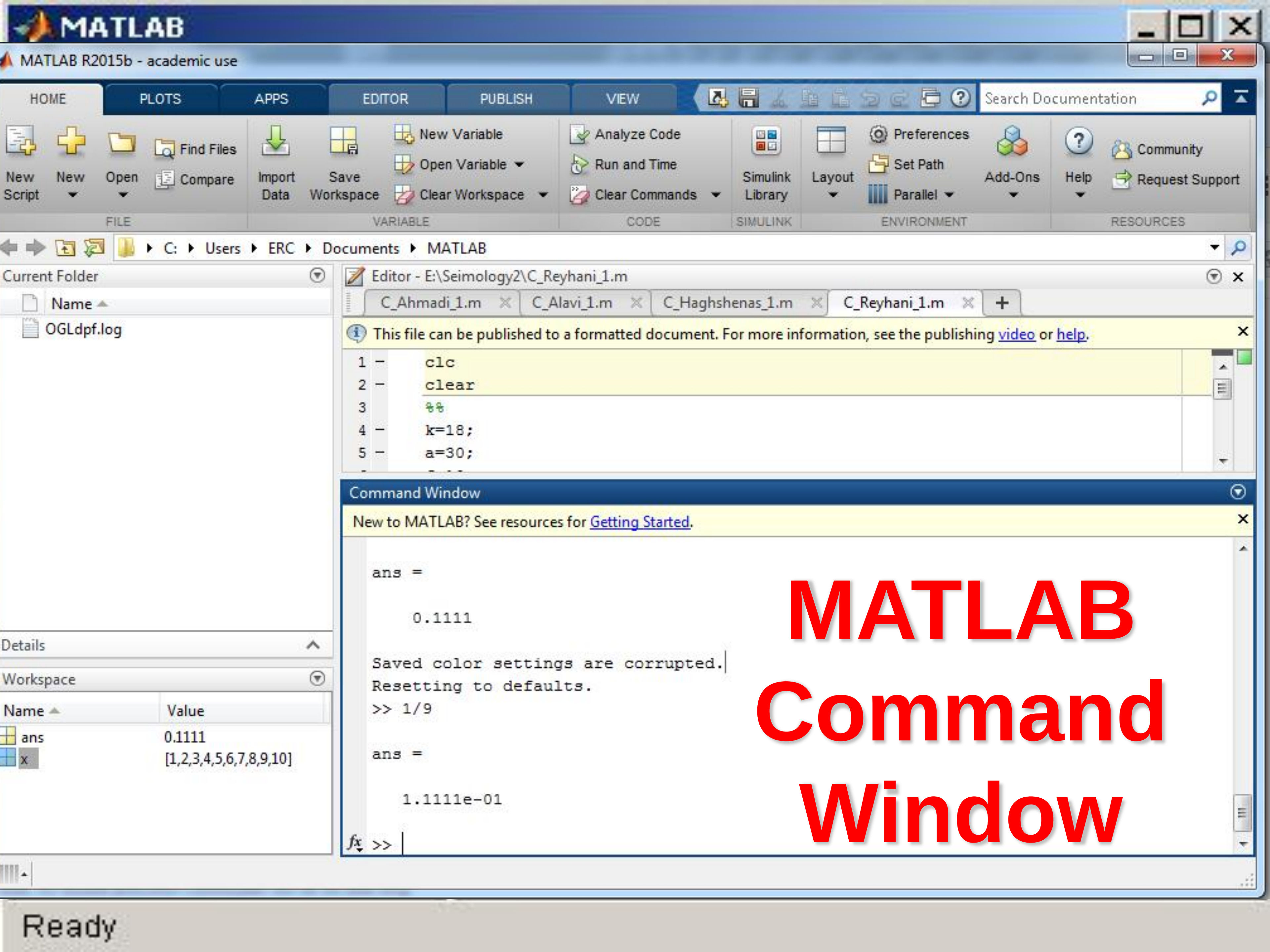

MATLAB
File Edit Debug Desktop Window Help
Current Directory: D:\SmartWork
Shortcuts How to Add What's New
Current Directory - D:\SmartWork
Command Window
Current Directory
Workspace
Command History
Start
OVR

All Files	File Type
BandPassFormat	Folder
BoundaryRefine	Folder
convexhull	Folder
DigitalwaveGuide	Folder
DynamicProgrammingFormant	Folder
filterbank	Folder
heriksRapor	Folder
LDABoundaryRefine	Folder
MAD_ESAT	Folder

```
find(DepoD==0)
plot(1./DepoD)
plot(abs(1./DepoD))
plot(log(abs(1./DepoD)))
plot(20*log10(abs(1./DepoD)))
figure,plot(20*log10(abs(1./DepoD))
figure(1),hold on ,plot(20*log10(ak
%-- 27.02.2007 21:15 --%
a=5
b=[1 2]
b=0
b
clc
```

```
>>
```

MATLAB Command Window



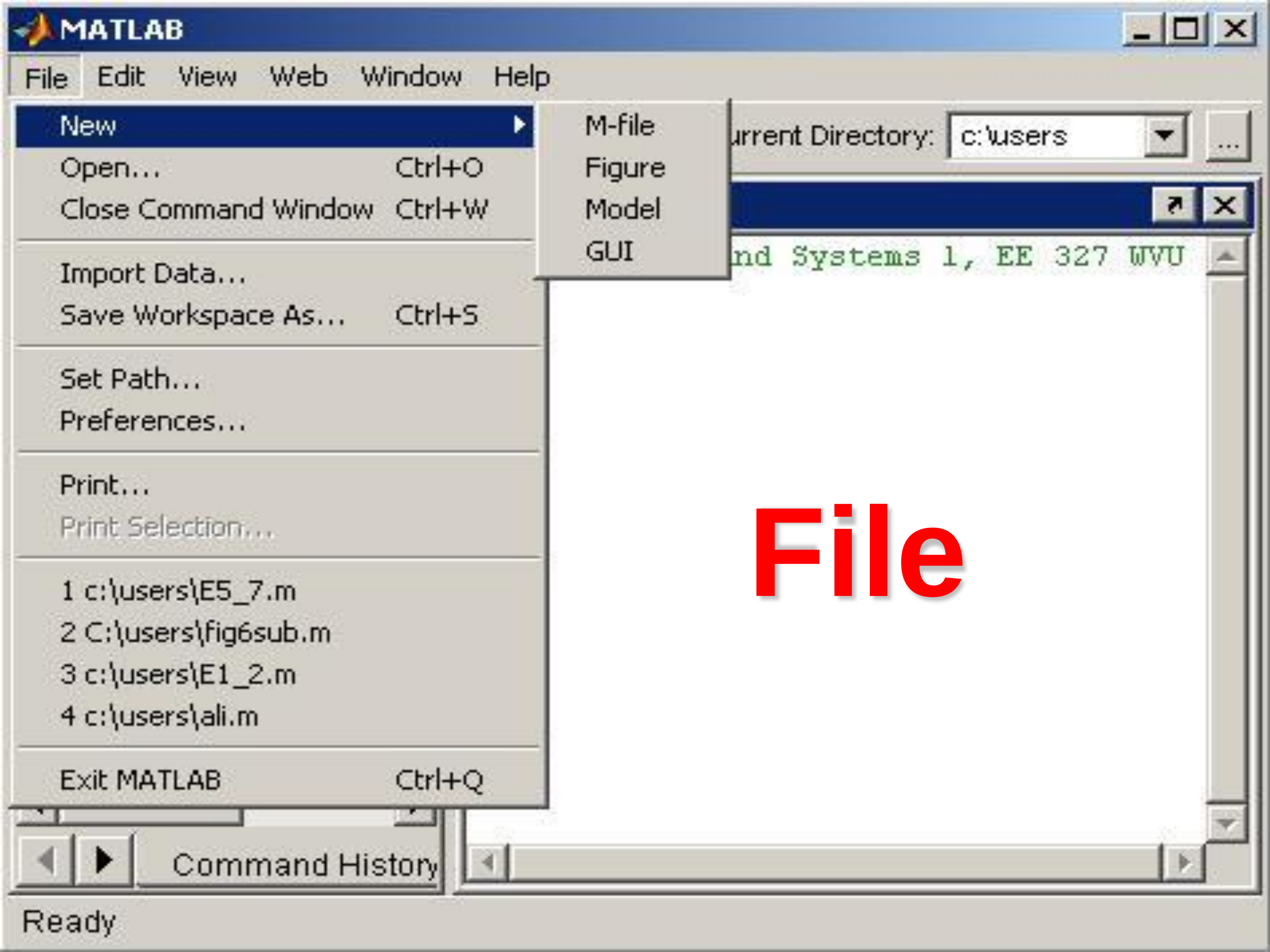
This file can be published to a formatted document. For more information, see the publishing video or help.

```
clc
clear
%%
k=18;
a=30;
```

New to MATLAB? See resources for Getting Started.

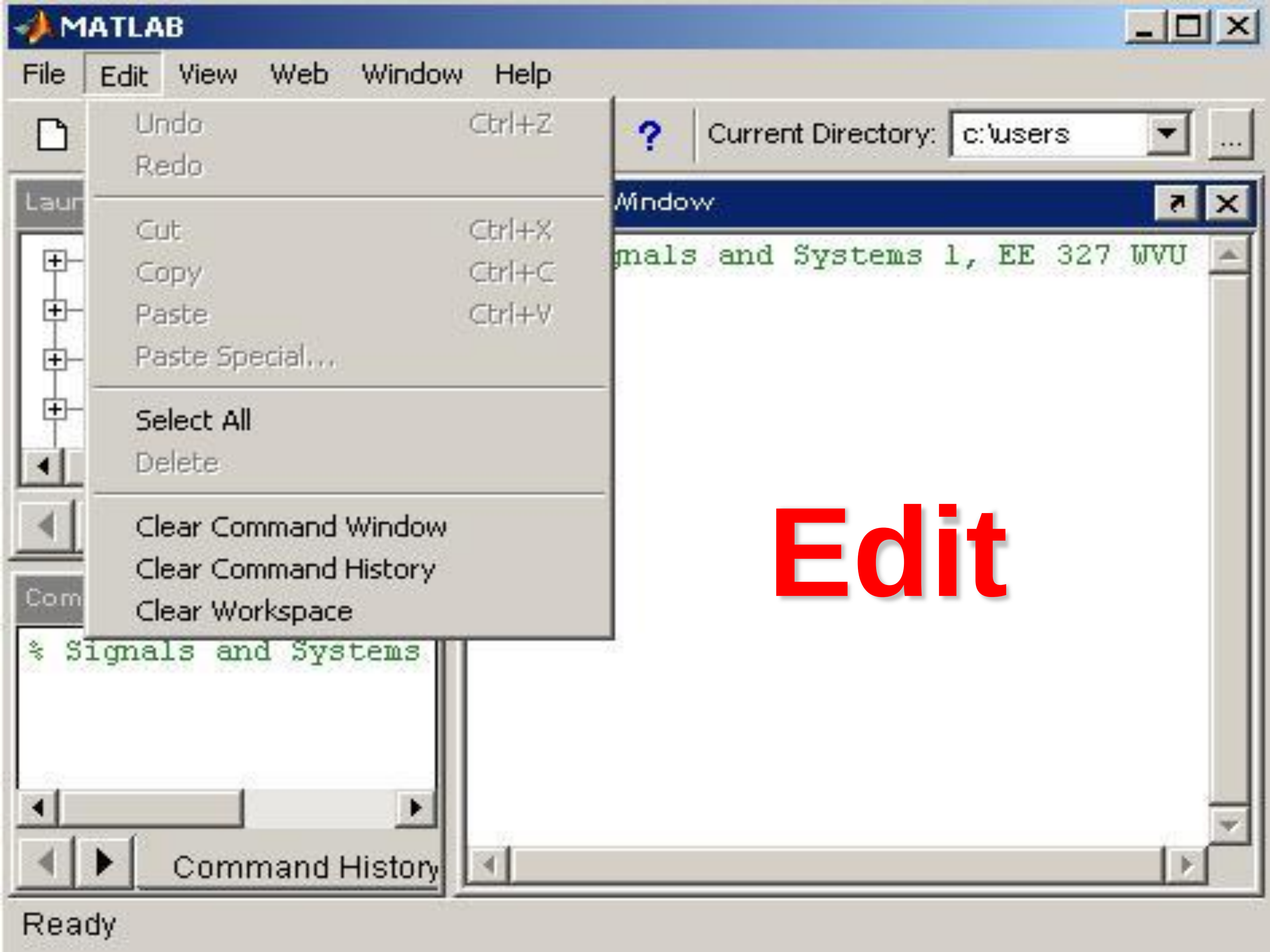
```
ans =

    0.1111

Saved color settings are corrupted.
Resetting to defaults.
>> 1/9

ans =

   1.1111e-01

>> 
```

Name	Value
ans	0.1111
x	[1,2,3,4,5,6,7,8,9,10]



```
nd Systems 1, EE 327 WVU
```

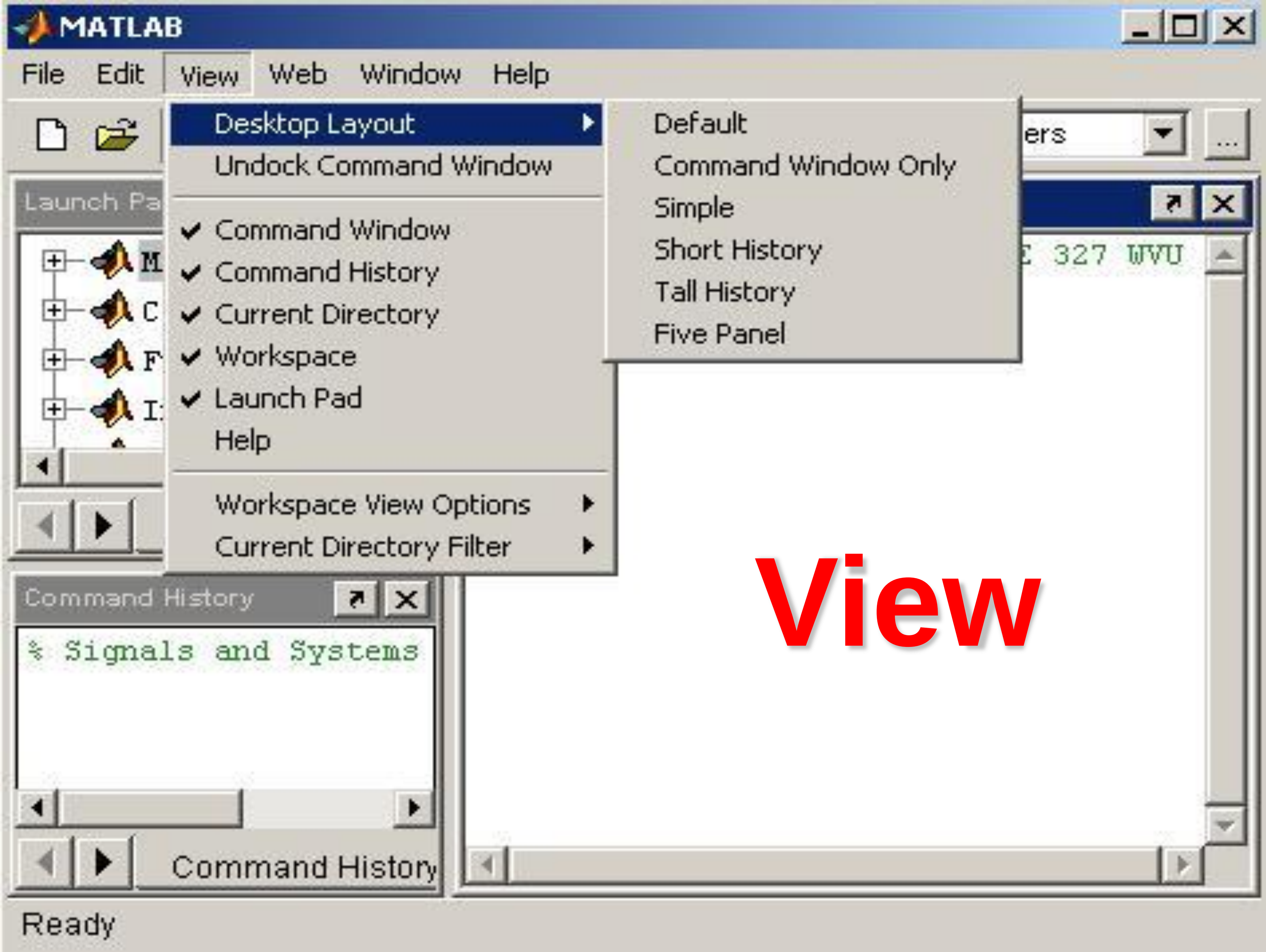
File



```
gnals and Systems 1, EE 327 WVU
```

```
% Signals and Systems
```

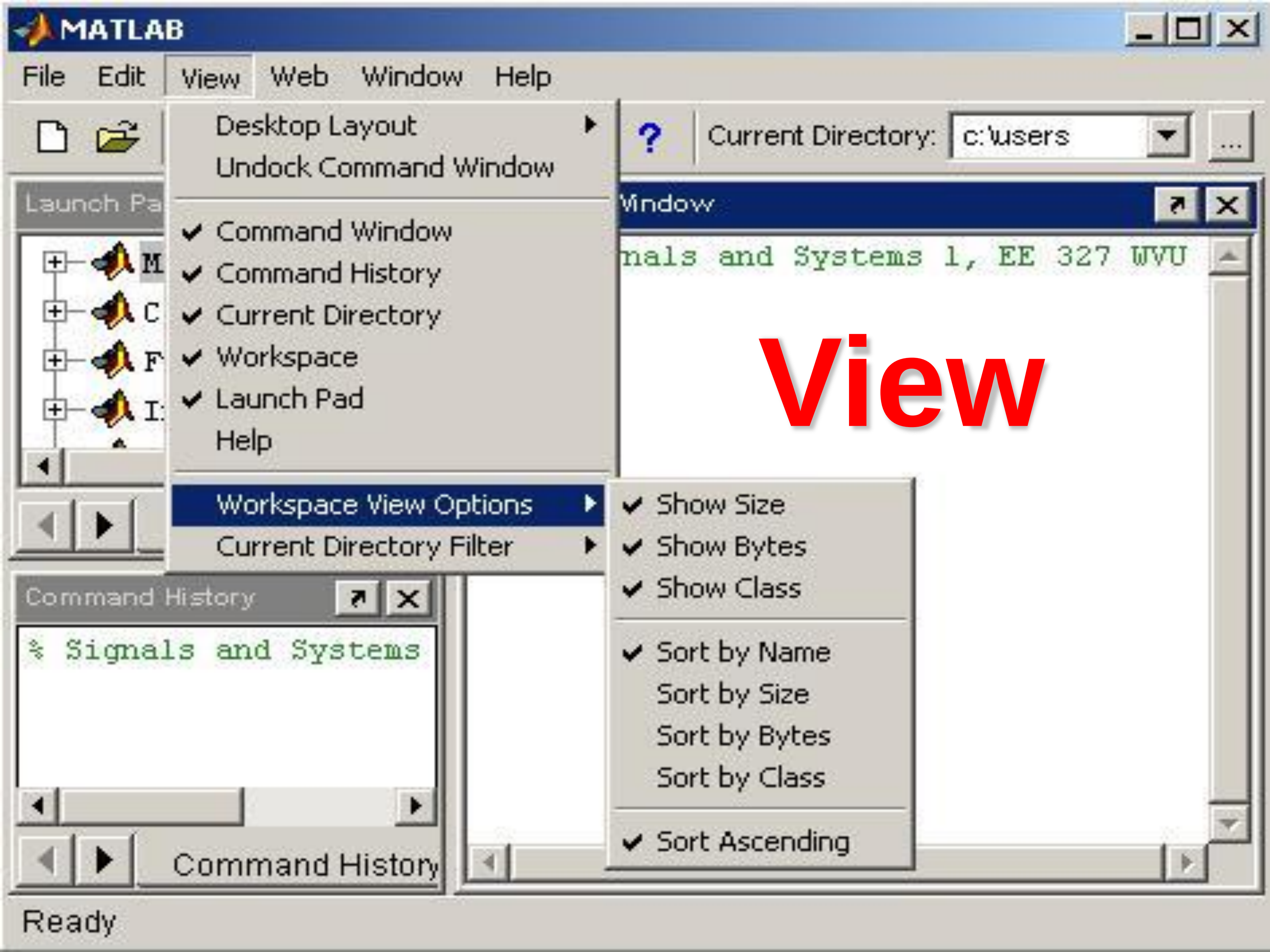
Edit



```
% Signals and Systems
```

```
E 327 WVU
```

View

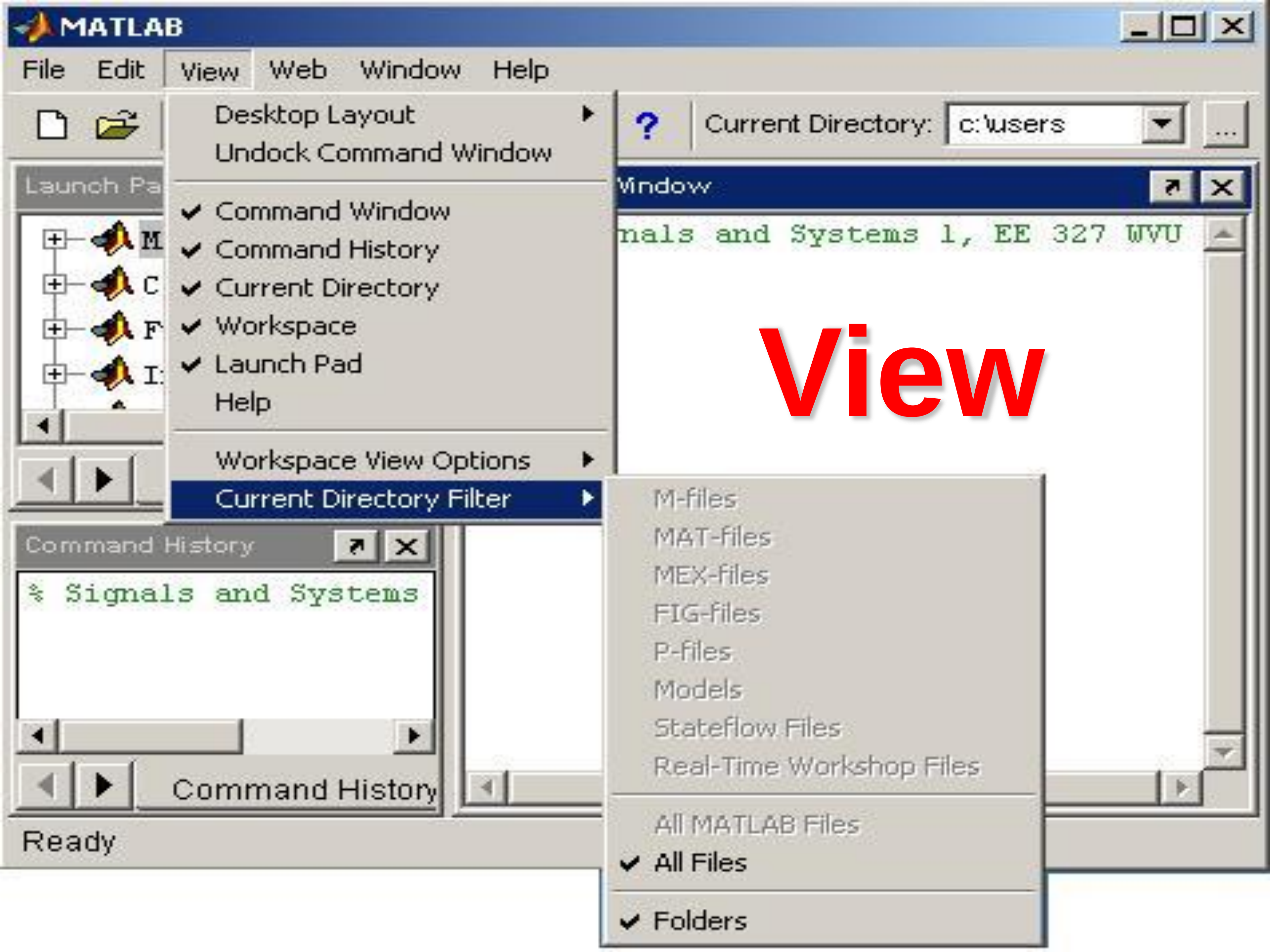


View

```
nals and Systems 1, EE 327 WVU
```

```
% Signals and Systems
```

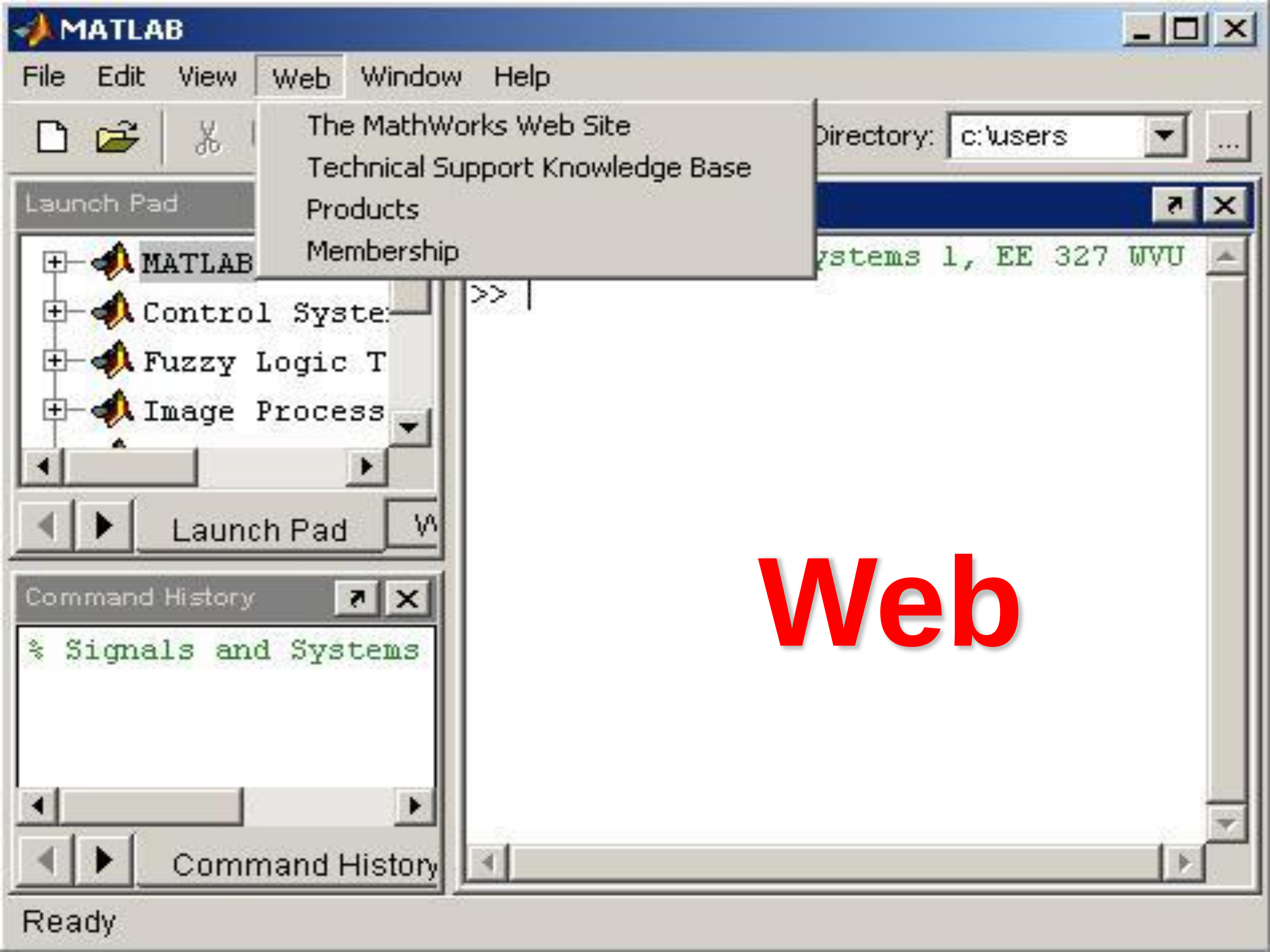
View



```
nals and Systems 1, EE 327 WVU
```

```
% Signals and Systems
```

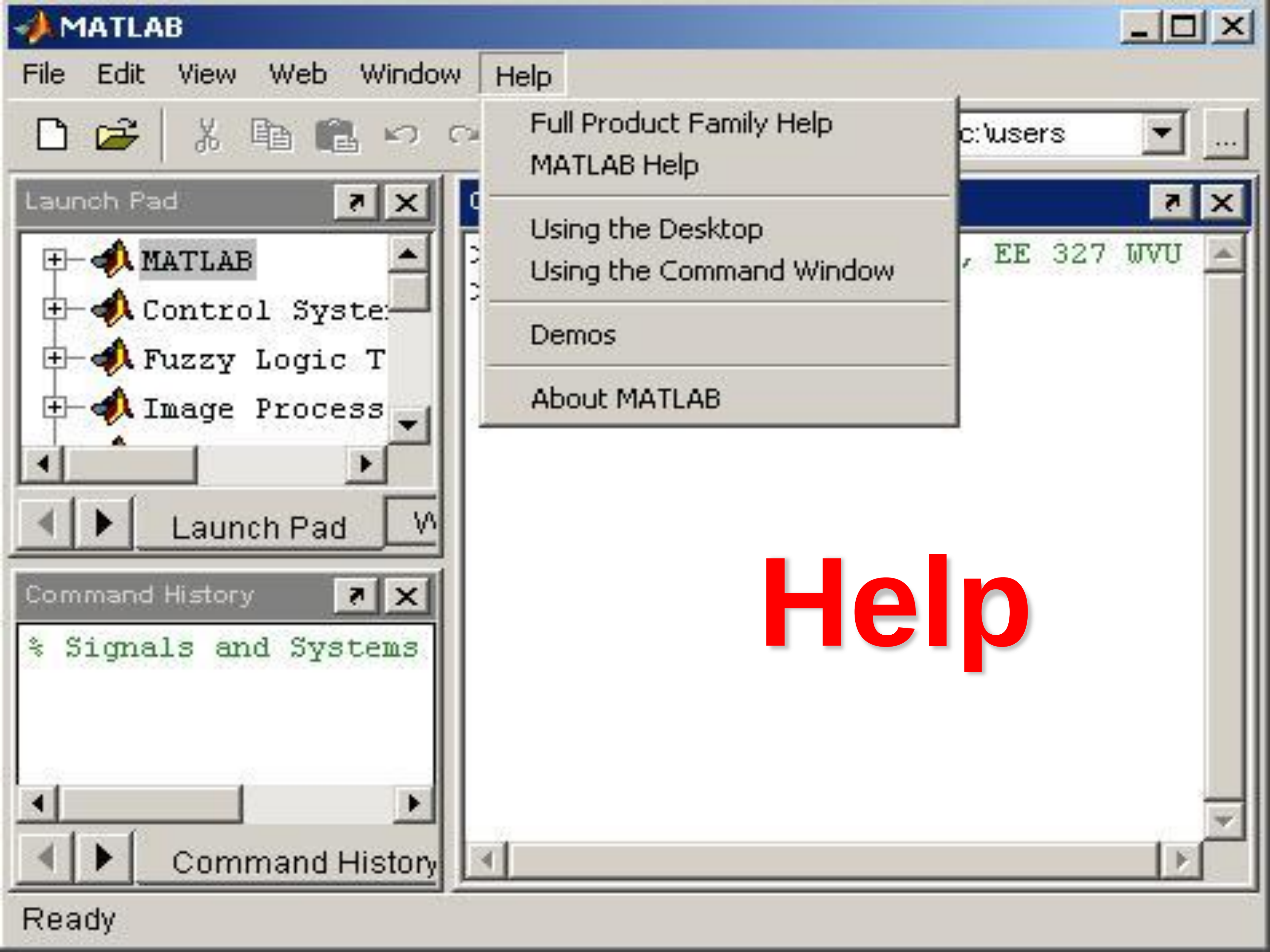
Web



```
ystems 1, EE 327 WVU
>>
```

```
% Signals and Systems
```

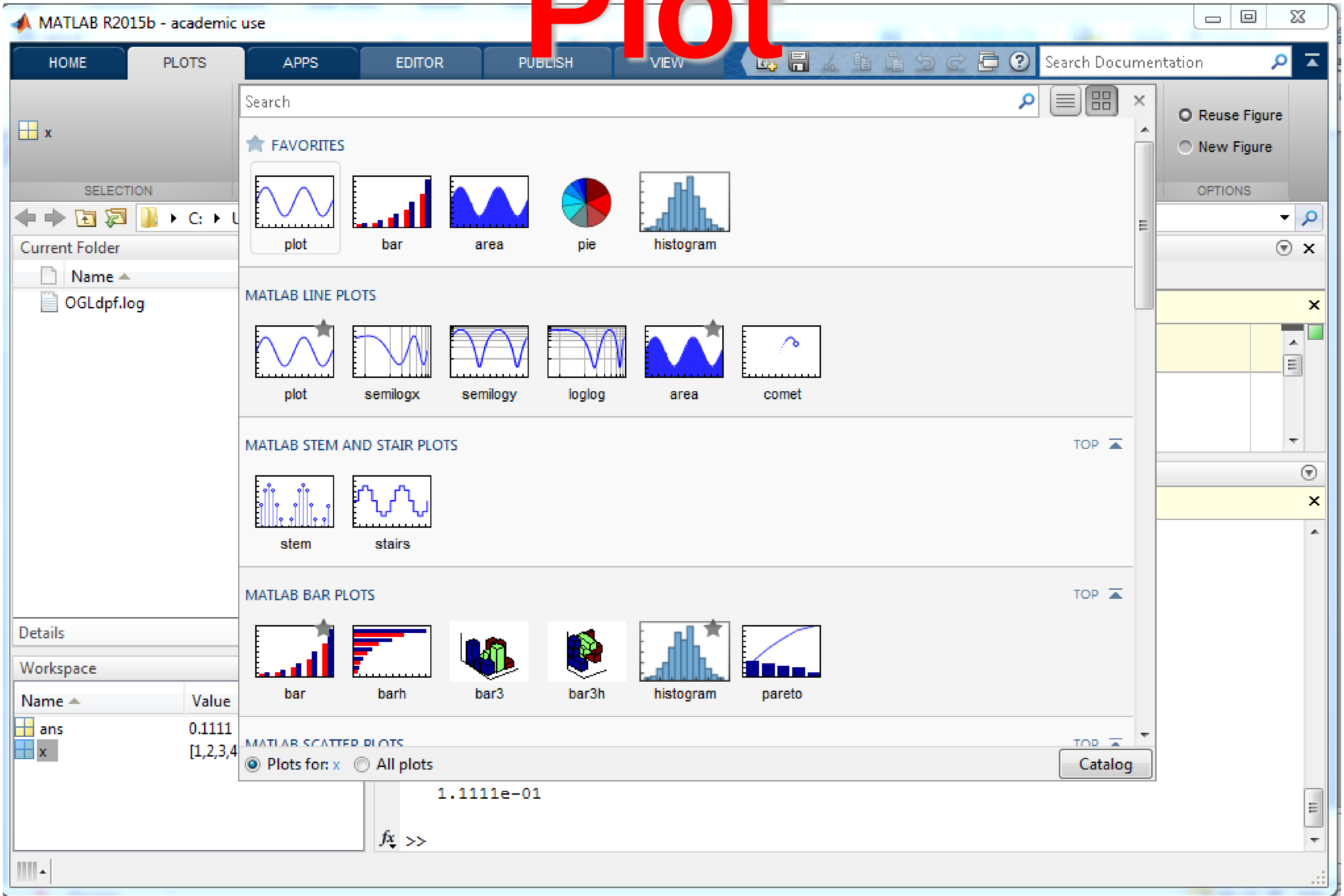
Help



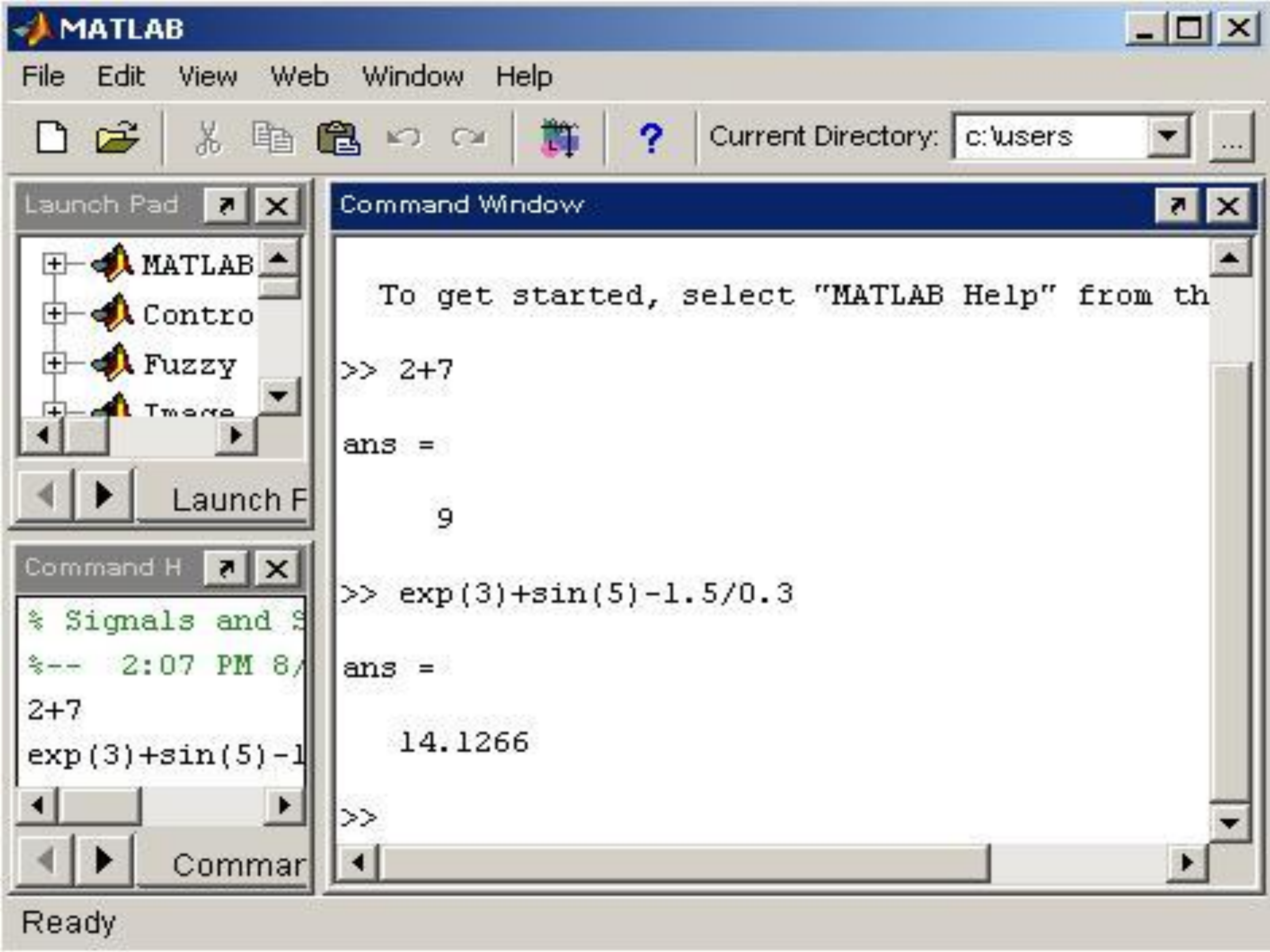
```
% Signals and Systems
```

```
, EE 327 WVU
```

Plot



```
  1.1111e-01

fx >>
```



```
 To get started, select "MATLAB Help" from th

>> 2+7

ans =

     9

>> exp(3)+sin(5)-1.5/0.3

ans =

   14.1266

>>
```

```
% Signals and S
%-- 2:07 PM 8/
2+7
exp(3)+sin(5)-1
```

Variables

- No need for types. i.e.,

~~int a;~~
~~double b;~~
~~float c;~~

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

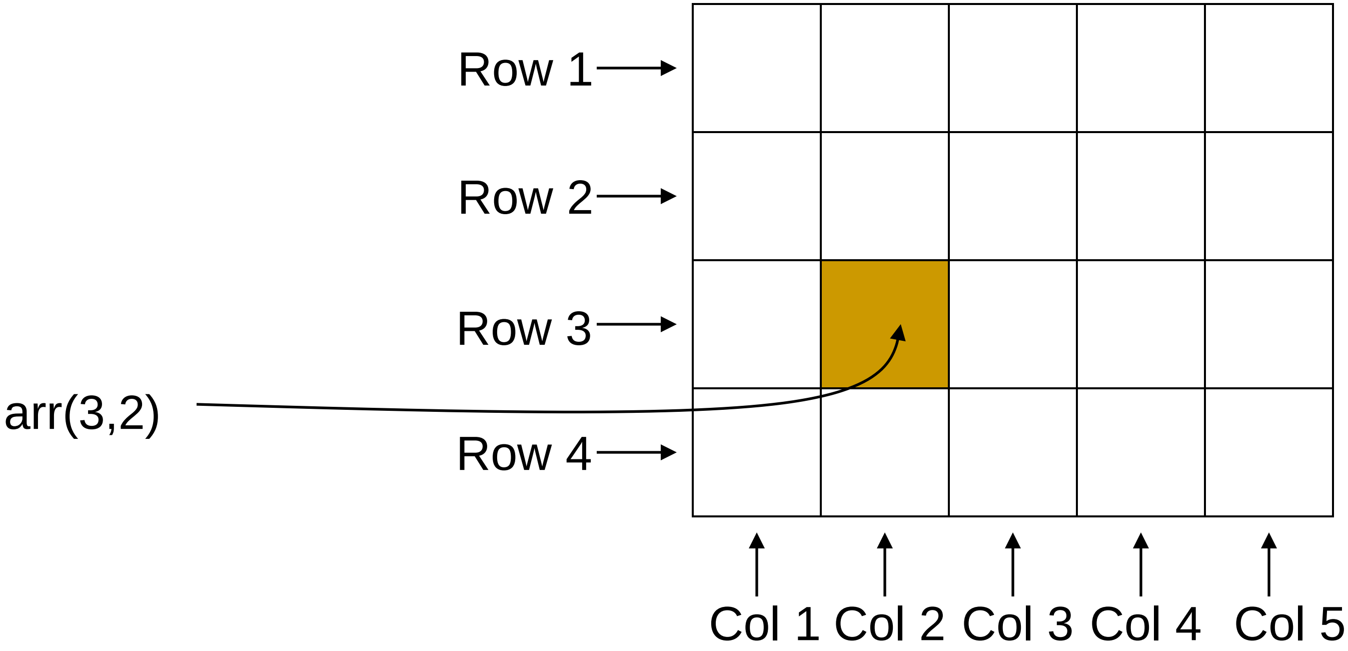
```
Example:
>>x=5;
>>x1=2;
```

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

MATLAB BASICS

Variables and Arrays

- **Array:** A collection of data values organized into rows and columns, and known by a single name.



Row 1

Row 2

Row 3

Row 4

arr(3,2)

Col 1 Col 2 Col 3 Col 4 Col 5

MATLAB BASICS

Arrays

- The fundamental unit of data in MATLAB
- Scalars are also treated as arrays by MATLAB (1 row and 1 column).
- Row and column indices of an array start from 1.
- Arrays can be classified as **vectors** and **matrices**.

MATLAB BASICS

- **Vector:** Array with one dimension
- **Matrix:** Array with more than one dimension
- **Size** of an array is specified by the number of rows and the number of columns, with the number of rows mentioned first (For example: n x m array).

  Total number of elements in an array is the product of the number of rows and the number of columns.

Array, Matrix

- a vector `x = [1 2 5 1]`

```
x =
     1     2     5     1
```

- a 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
```

Long Array, Matrix

- `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
```

- `B = [1:4; 5:8]`

```
x =
     1      2      3     4
     5      6      7     8
```

General Functions

- whos: List current variables and their size
- clear: Clear variables and functions from memory
- cd: Change current working directory
- dir: List files in directory
- pwd: Tells you the current directory you work in
- echo: Echo commands in M-files
- format: Set output format (long, short, etc.)

MATLAB BASICS

Changing the data format

```
>> value = 12.345678901234567;
    format short           → 12.3457
    format long            → 12.34567890123457
    format short e         → 1.2346e+001
    format long e          → 1.234567890123457e+001
    format short g         → 12.346
    format long g          → 12.3456789012346
    format rat             → 1000/81
```

MATLAB BASICS

Initializing with Built-in Functions

- zeros(n)
- zeros(n,m)
- zeros(size(arr))
- ones(n)
- ones(n,m)
- ones(size(arr))
- eye(n)
- eye(n,m)
- length(arr)
- size(arr)

```
>> a = zeros(2);
>> b = zeros(2, 3);
>> c = [1, 2; 3, 4];
>> d = zeros(size(c));
```

Generating Vectors 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
```

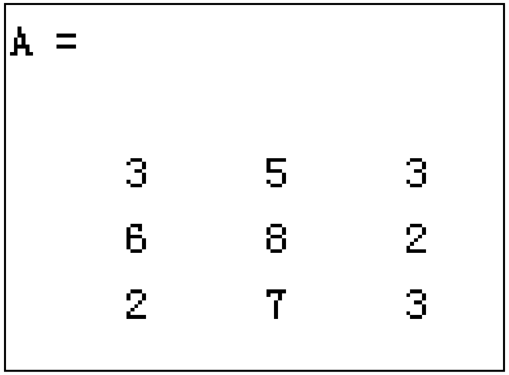
- rand(M,N) MxN matrix of uniformly distributed random numbers on (0,1)

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

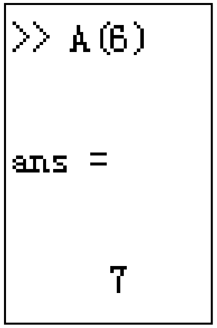
Matrix Index

- The matrix indices begin from 1 (not 0 (as in C))
- The matrix indices must be positive integer

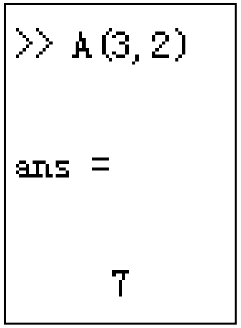
Given:



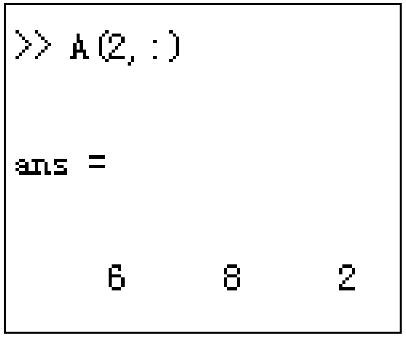
```
A =


     3     5     3
     6     8     2
     2     7     3
```



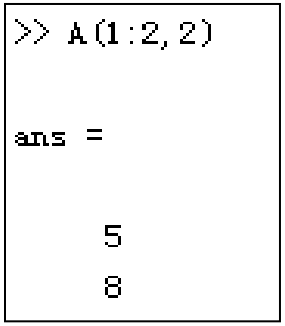
```
>> A(6)

ans =

     7
```



```
>> A(3,2)

ans =

     7
```



```
>> A(2,:)

ans =

     6     8     2
```



```
>> A(1:2,2)

ans =

     5
     8
```

A(-2), A(0)

Error: ??? Subscript indices must either be real positive integers or logicals.

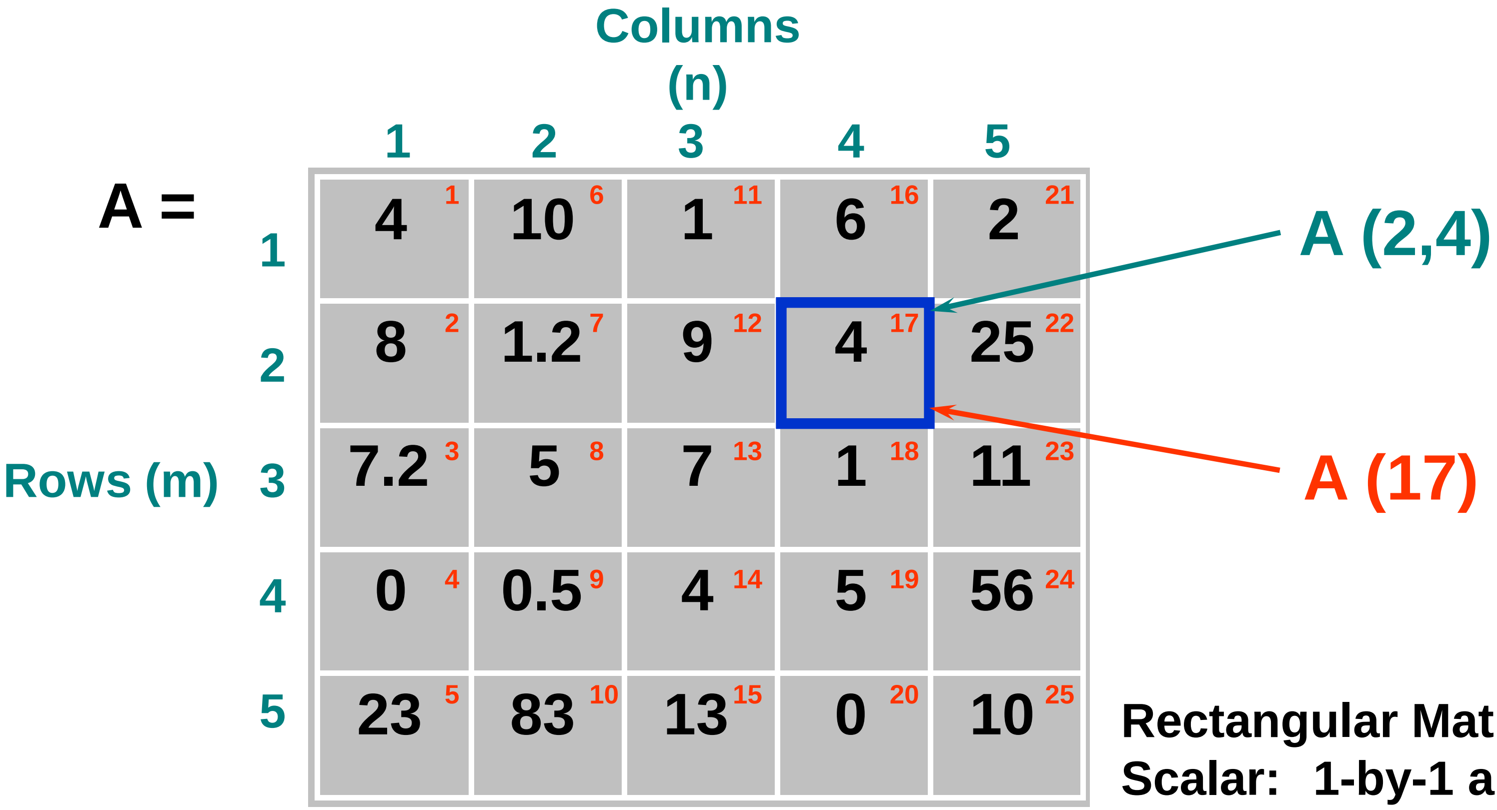
A(4,2)
Error: ??? Index exceeds matrix dimensions.

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.
```

The Matrix in MATLAB



A =

Rows (m) \ Columns (n)	1	2	3	4	5
1	4 (1)	10 (6)	1 (11)	6 (16)	2 (21)
2	8 (2)	1.2 (7)	9 (12)	4 (17)	25 (22)
3	7.2 (3)	5 (8)	7 (13)	1 (18)	11 (23)
4	0 (4)	0.5 (9)	4 (14)	5 (19)	56 (24)
5	23 (5)	83 (10)	13 (15)	0 (20)	10 (25)

A (2,4)

A (17)

Rectangular Matrix:

Scalar: 1-by-1 array

Vector: m-by-1 array
1-by-n array

Matrix: m-by-n array

Operators (arithmetic)

+	addition
-	subtraction
*	multiplication
/	division
^	power
‘	complex conjugate transpose

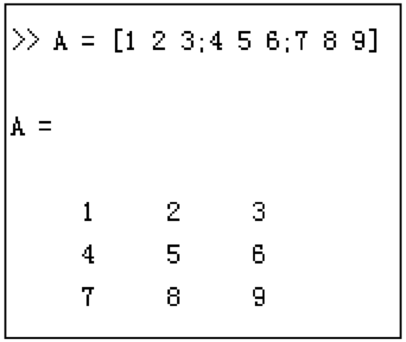
MATLAB BASICS

- *variable_name = expression*;

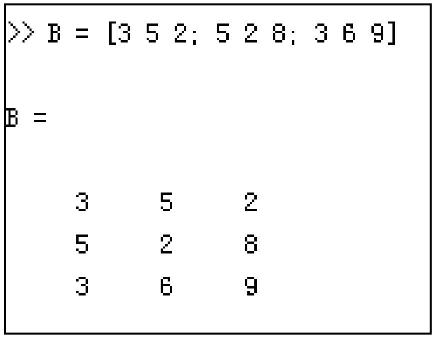
– addition	a + b	→	a + b
– subtraction	a - b	→	a - b
– multiplication	a x b	→	a * b
– division	a / b	→	a / b
– exponent	a^b	→	a ^ b

Matrices Operations

Given A and B:

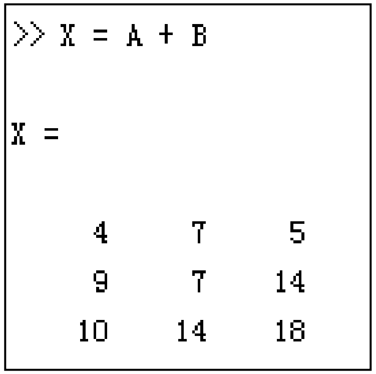


```
>> A = [1 2 3;4 5 6;7 8 9]

A =

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



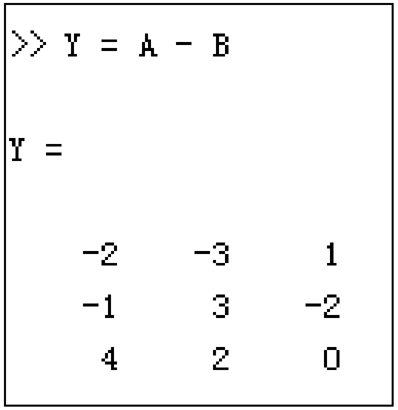
```
>> B = [3 5 2; 5 2 8; 3 6 9]

B =

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

Addition



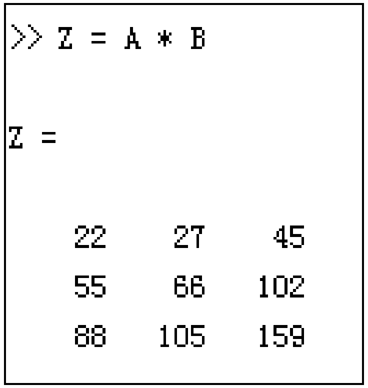
```
>> X = A + B

X =

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

Subtraction



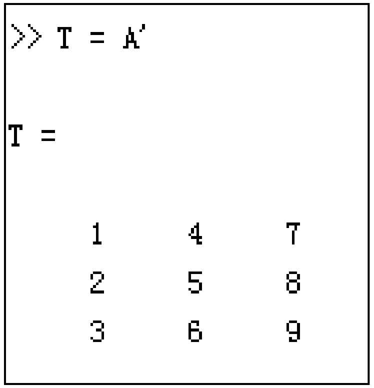
```
>> Y = A - B

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
```

Operators (Element by Element)

.* element-by-element multiplication

./ element-by-element division

.^ element-by-element power

The use of “.” – “Element” Operation

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

```
x = A(1,:)

x=
    1   2   3
```

```
y = A(3 ,:)

y=
    3  4  -1
```

```
b = x .* y

b=
     3  8 -3
```

```
c = x . / y

c=
   0.33   0.5   -3
```

```
d = x .^2

d=
      1    4    9
```

```
K= x^2
Erorr:
 ??? Error using ==> mpower  Matrix must be square.
B=x*y
Erorr:
??? Error using ==> mtimes Inner matrix dimensions must agree.
```

MATLAB BASICS

Special Values

- pi: π value up to 15 significant digits
- i, j: sqrt(-1)
- Inf: infinity (such as division by 0)
- NaN: Not-a-Number (division of zero by zero)
- clock: current date and time in the form of a 6-element row vector containing the year, month, day, hour, minute, and second
- date: current date as a string such as *16-Feb-2004*
- eps: epsilon is the smallest difference between two numbers
- ans: stores the result of an expression

MATLAB BASICS

The disp(*array*) function

```
>> disp( 'Hello' )
Hello
>> disp(5)
     5
>> disp( [ 'Bilkent ' 'University' ] )
Bilkent University
>> name = 'Alper';
>> disp( [ 'Hello ' name ] )
Hello Alper
```

MATLAB BASICS

The num2str() and int2str() functions

```
>> d = [ num2str(16) '-Feb-' num2str(2004) ];
>> disp(d)
16-Feb-2004
>> x = 23.11;
>> disp( [ 'answer = ' num2str(x) ] )
answer = 23.11
>> disp( [ 'answer = ' int2str(x) ] )
answer = 23
```

MATLAB BASICS

The fprintf(*format*, *data*) function

– %d	integer
– %f	floating point format
– %e	exponential format
– %g	either floating point or exponential format, whichever is shorter
– \n	new line character
– \t	tab character

MATLAB BASICS

```
>> fprintf( 'Result is %d', 3 )
Result is 3
>> fprintf( 'Area of a circle with radius %d is %f', 3, pi*3^2 )
Area of a circle with radius 3 is 28.274334
>> x = 5;
>> fprintf( 'x = %3d', x )
x =    5
>> x = pi;
>> fprintf( 'x = %0.2f', x )
x = 3.14
>> fprintf( 'x = %6.2f', x )
x =    3.14
>> fprintf( 'x = %d\ny = %d\n', 3, 13 )
x = 3
y = 13
```

MATLAB BASICS

Data files

- save *filename var1 var2* …

```
>> save myfile.mat  x  y                  → binary
>> save myfile.dat  x  –ascii             → ascii
```

- load *filename*

```
>> load myfile.mat                        → binary
>> load myfile.dat  –ascii                → ascii
```

size(A) - size vector

sum(A) - columns sums vector

sum(sum(A)) - all the elements sum

Visualization and Graphics

- **plot(x,y), plot(x,sin(x))** - plot 1-D function
- **figure , figure(k)** - open a new figure
- **hold on, hold off** - refreshing
- **mesh(x_ax,y_ax,z_mat)** - view surface
- **contour(z_mat)** - view z as top. map
- **subplot(3,1,2)** - locate several plots in figure
- **axis([xmin xmax ymin ymax])** *-* change axes
- **title(‘figure title’)** - add title to figure

Basic Task: Plot the function sin(x) between 0≤x≤4π

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

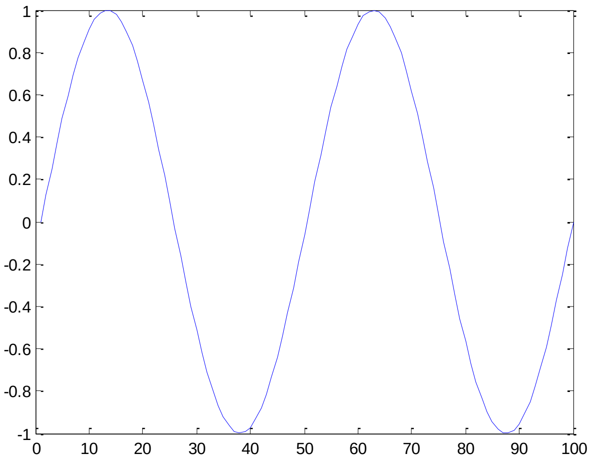
```
>>x=linspace(0,4*pi,100);
```

- Calculate sin(.) of the x-array

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

- Plot the y-array

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



Plot the function $e^{-x/3}\sin(x)$ between $0 \leq x \leq 4\pi$

- 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);
```

- Calculate $e^{-x/3}$ of the x-array

```
>>y1=exp(-x/3);
```

- Multiply the arrays y and y1

```
>>y2=y*y1;
```

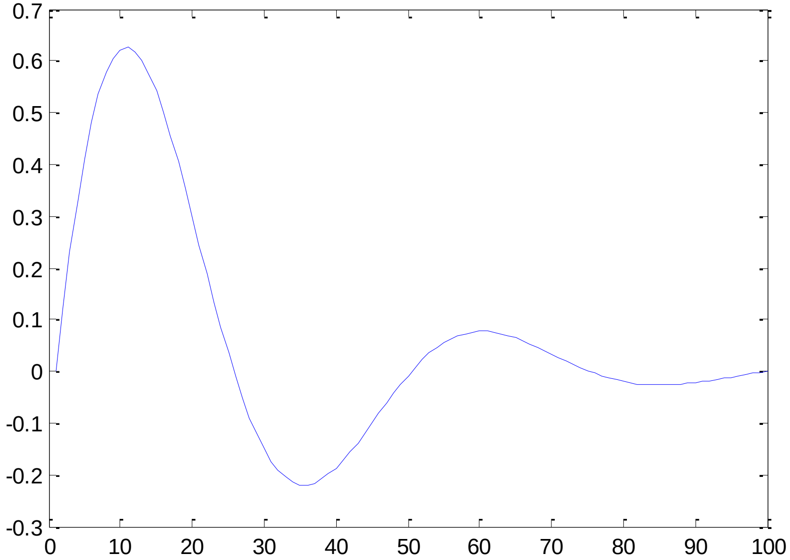
Plot the function $e^{-x/3}\sin(x)$ between $0 \leq x \leq 4\pi$

- Multiply the arrays y and y1 correctly

```
>>y2=y.*y1;
```

- Plot the y2-array

```
>>plot(y2)
```

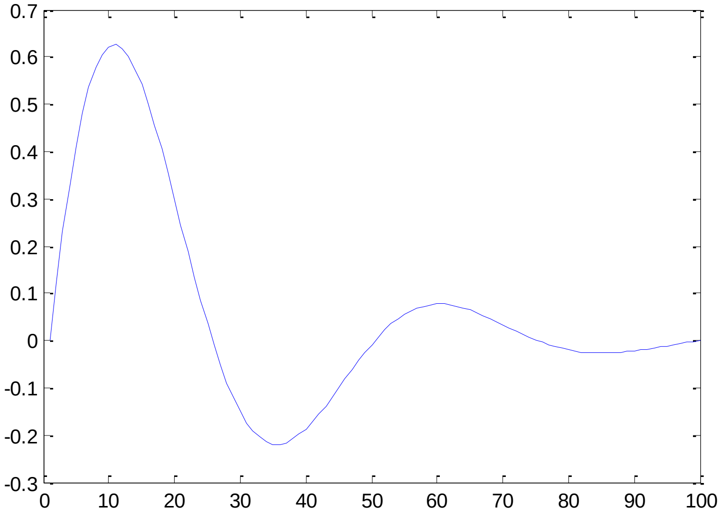


Display Facilities

- plot(.)

Example:

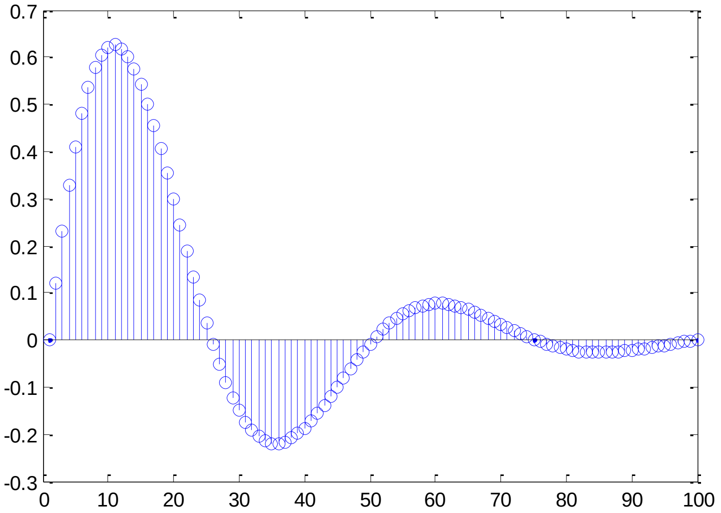
```
>>x=linspace(0,4*pi,100);
>>y=sin(x);
>>plot(y)
>>plot(x,y)
```



- stem(.)

Example:

```
>>stem(y)
>>stem(x,y)
```



Display Facilities

- title(.)

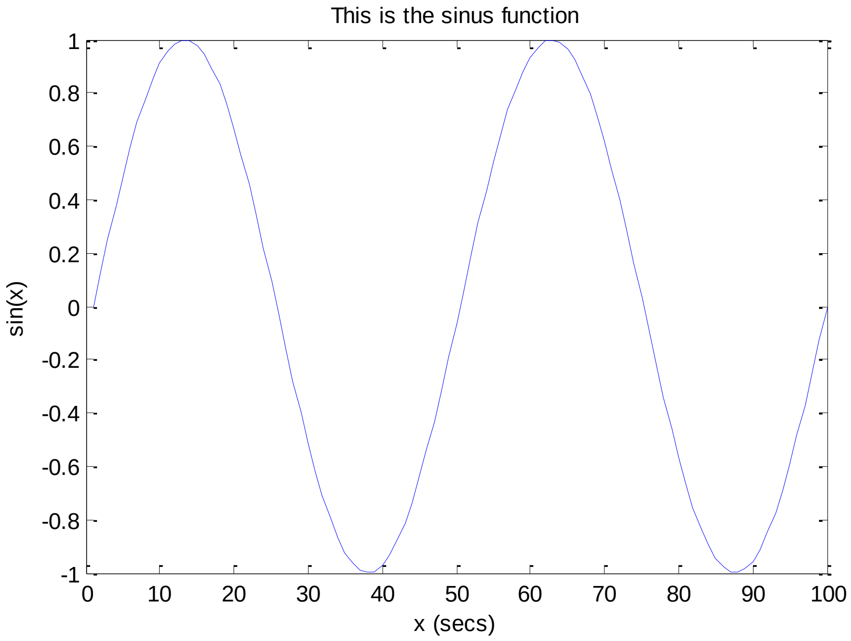
```
>>title(‘This is the sinus function’)
```

- xlabel(.)

```
>>xlabel(‘x (secs)’)
```

- ylabel(.)

```
>>ylabel(‘sin(x)’)
```



MATLAB Basics

Plotting Elementary Functions:

The command subplot can be used to partition the screen so that up to four plots can be viewed simultaneously. See help subplot.

- Example for use of subplot:
- >>% Line plot of a chirp
- >> x=0:0.05:5;
- >> y=sin(x.^2);
- >> subplot(2,2,1), plot(x,y);
- >> % Bar plot of a bell shaped curve
- >> x = -2.9:0.2:2.9;
- >> subplot(2,2,2), bar(x,exp(-x.*x));
- >> % Stem plot
- >> x = 0:0.1:4;
- >> subplot(2,2,3), stem(x,y)
- >> % Polar plot
- >> t=0:.01:2*pi;
- >> subplot(2,2,4), polar(t,abs(sin(2*t).*cos(2*t)));

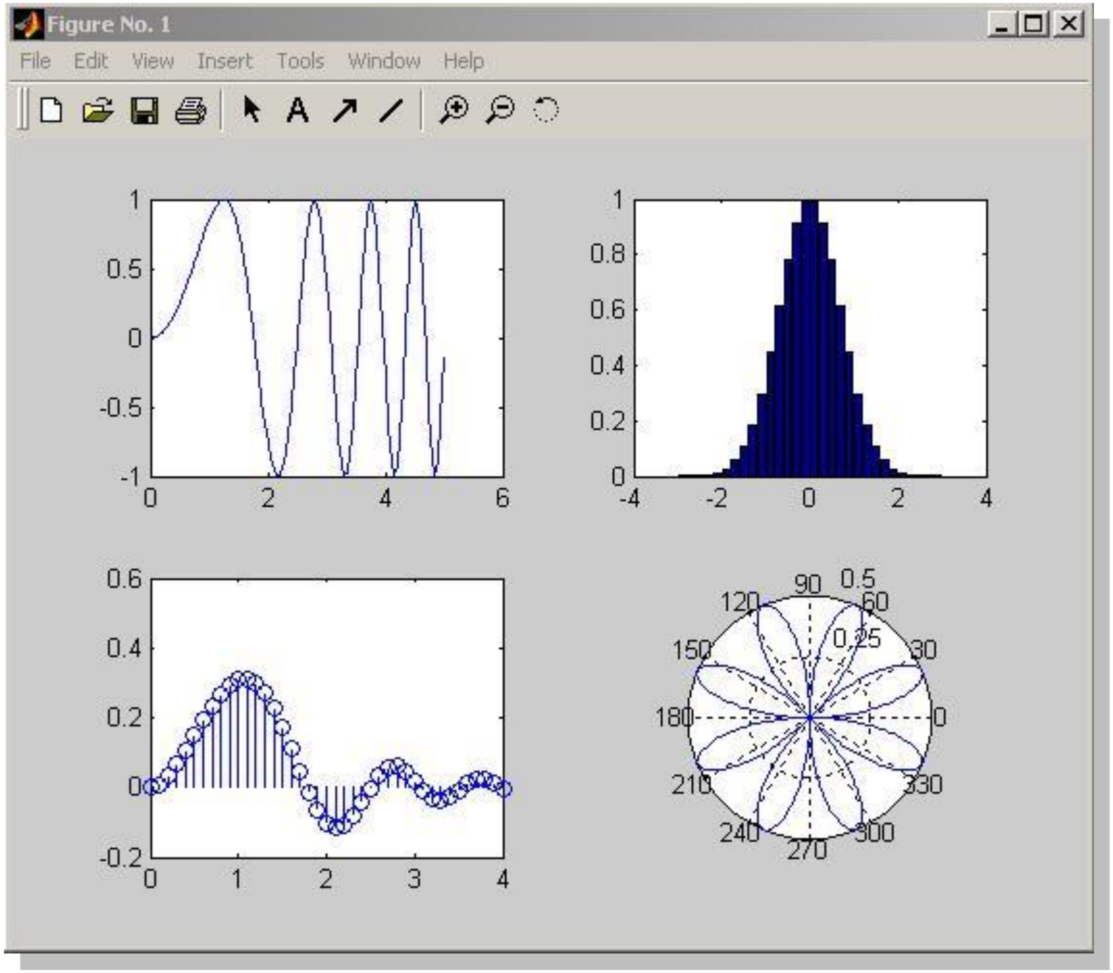
◀ ▶

MATLAB Basics

Plotting

Elementary Functions:

```
>>%Example Subplot
```



Read and Write Images

- I = imread(‘colors.jpg');
  imshow(I);
- Indexed Image:
  - **[x,map] = imread(‘color.png');**

- imwrite(I, ‘newim.jpg’)

Operators (relational, logical)

- == Equal to
- ~= Not equal to
- < Strictly smaller
- > Strictly greater
- <= Smaller than or equal to
- >= Greater than equal to
- & And operator
- | Or operator

Flow Control

- if
- for
- while
- break
- ….

Control Structures

- If Statement Syntax

```
if (Condition_1)
        Matlab Commands
elseif (Condition_2)
        Matlab Commands
elseif (Condition_3)
        Matlab Commands
else
        Matlab Commands
end
```

Some Dummy Examples

```
if ((a>3) & (b==5))
    Some Matlab Commands;
end
```

```
if (a<3)
    Some Matlab Commands;
elseif (b~=5)
    Some Matlab Commands;
end
```

```
if (a<3)
    Some Matlab Commands;
else
    Some Matlab Commands;
end
```

Control Structures

- For loop syntax

```
for i=Index_Array
   Matlab Commands
end
```

Some Dummy Examples

```
for i=1:100
    Some Matlab Commands;
end
```

```
for j=1:3:200
    Some Matlab Commands;
end
```

```
for m=13:-0.2:-21
    Some Matlab Commands;
end
```

```
for k=[0.1 0.3 -13 12 7 -9.3]
    Some Matlab Commands;
end
```

Control Structures

- While Loop Syntax

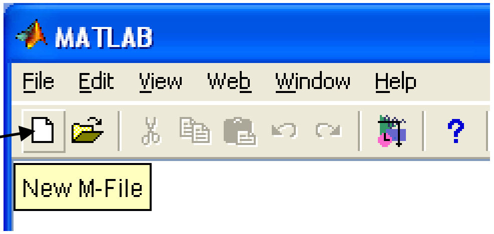
```
while (condition)
   Matlab Commands
end
```

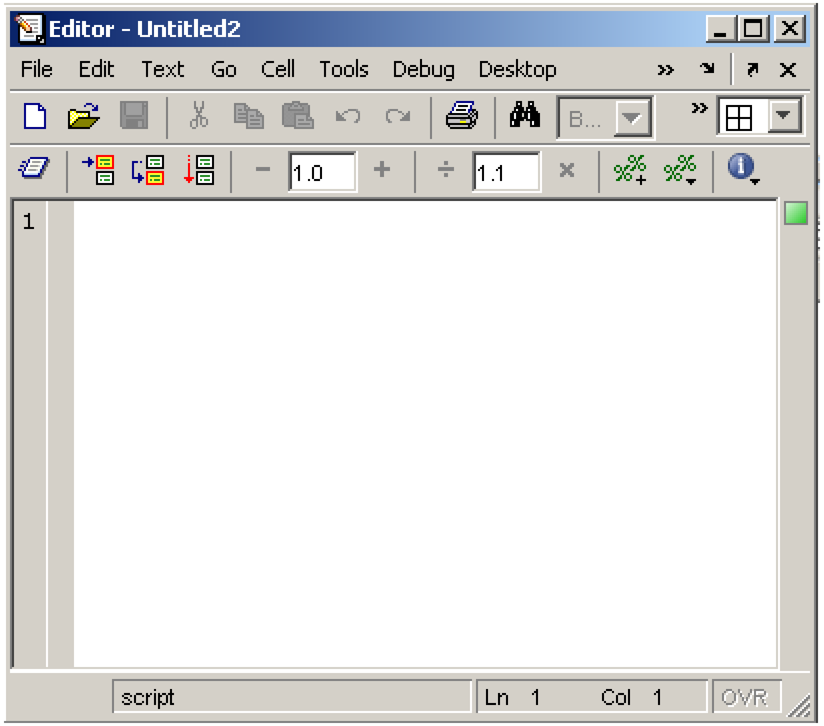
Dummy Example

```
while  ((a>3) & (b==5))
     Some Matlab Commands;
end
```

Use of M-File

Click to create a new M-File

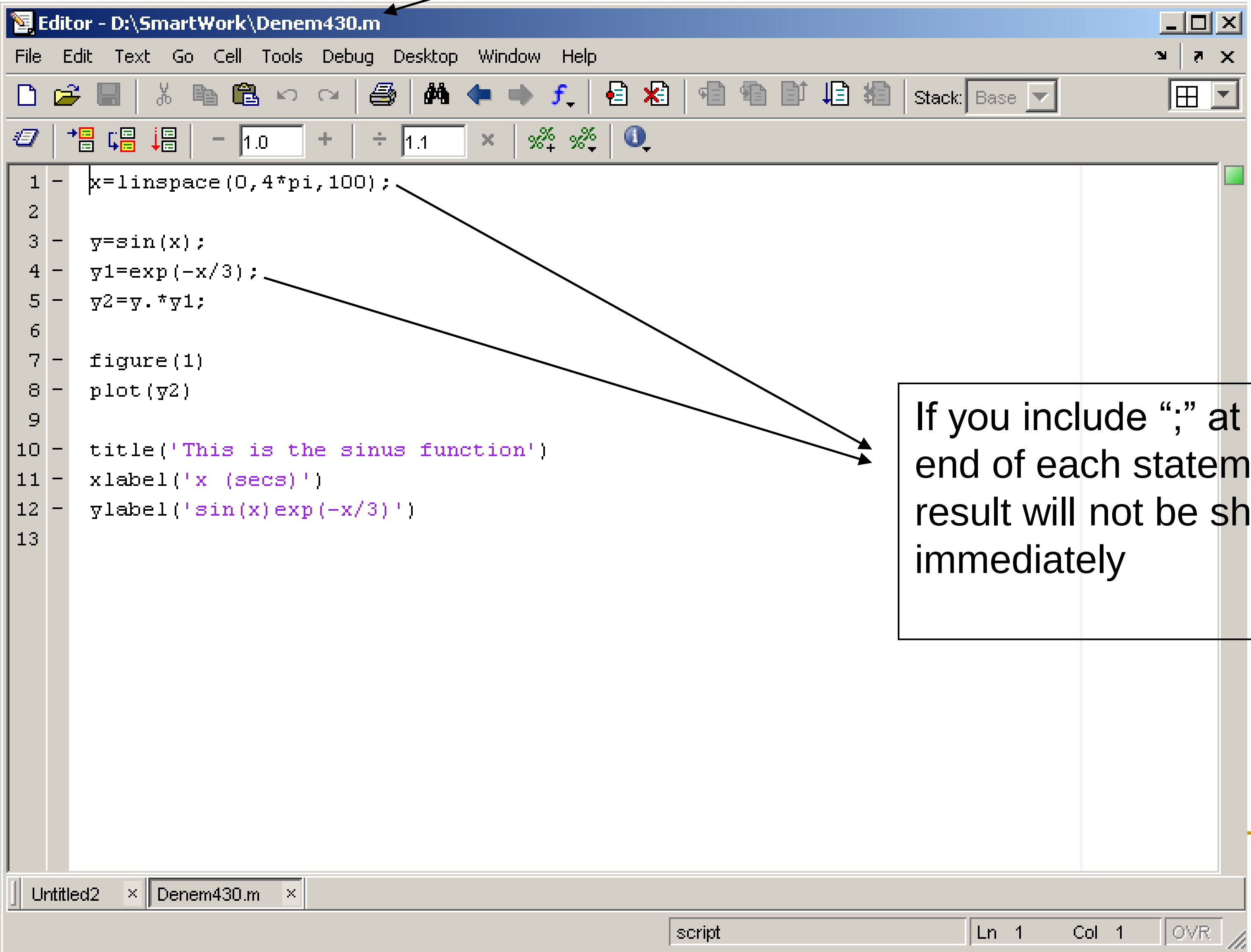




- Extension “.m”
- A text file containing script or function or program to run

Use of M-File

Save file as *Denem430*.m


Editor - D:\SmartWork\Denem430.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
Untitled2
Denem430.m
script
Ln 1
Col 1
OVR

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

y=sin(x);
y1=exp(-x/3);
y2=y.*y1;

figure(1)
plot(y2)

title('This is the sinus function')
xlabel('x (secs)')
ylabel('sin(x)exp(-x/3)')
```

If you include ";" at the end of each statement, result will not be shown immediately

Writing User Defined Functions

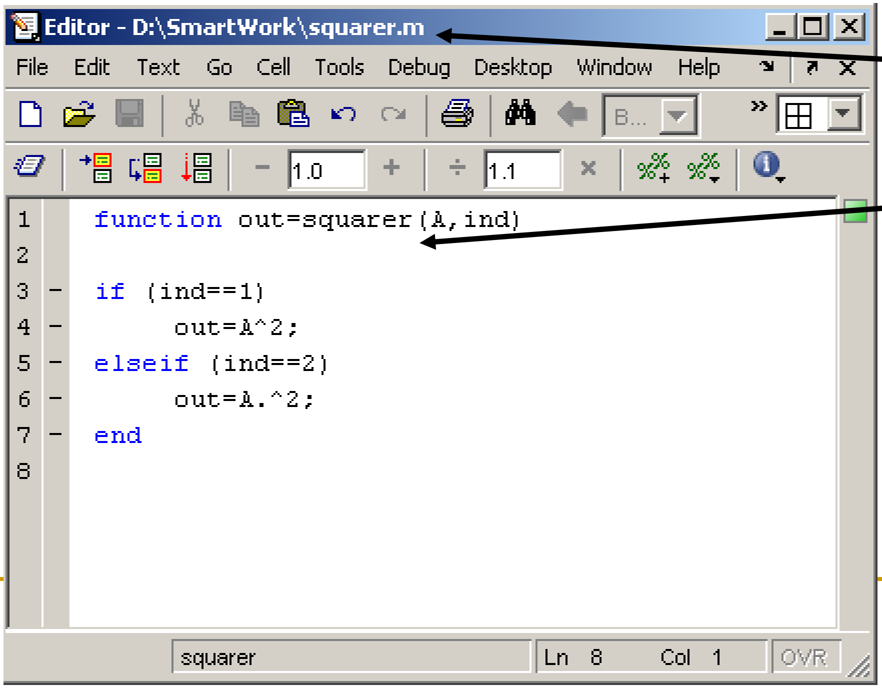
- Functions are m-files which can be executed by specifying some inputs and supply some desired outputs.
- The code telling the Matlab that an m-file is actually a function is

```
function out1=functionname(in1)
function out1=functionname(in1,in2,in3)
function [out1,out2]=functionname(in1,in2)
```

- You should write this command at the beginning of the m-file and you should save the m-file with a file name same as the function name

Writing User Defined Functions

- Examples
  - Write a function : out=squarer (A, ind)
    - Which takes the square of the input matrix if the input indicator is equal to 1
    - And takes the element by element square of the input matrix if the input indicator is equal to 2



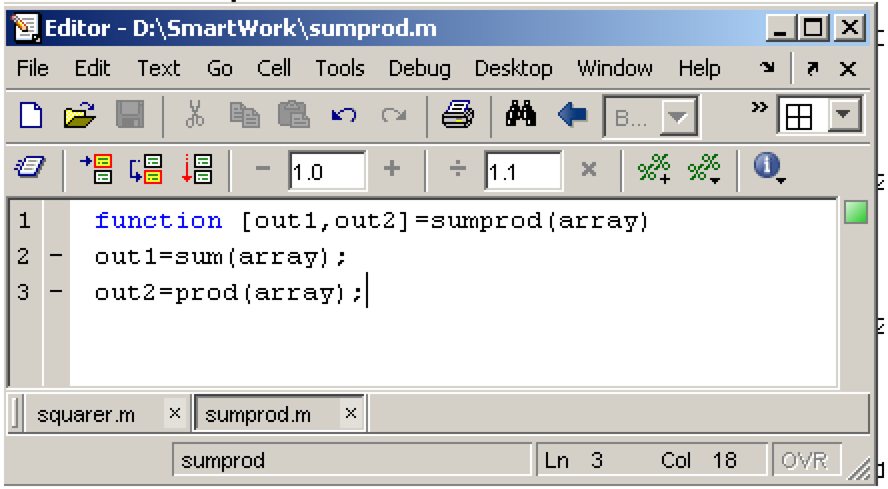
Editor - D:\SmartWork\squarer.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
squarer
Ln 8 Col 1
OVR

```
function out=squarer(A,ind)

if (ind==1)
        out=A^2;
elseif (ind==2)
        out=A.^2;
end

```

Same Name

Writing User Defined Functions

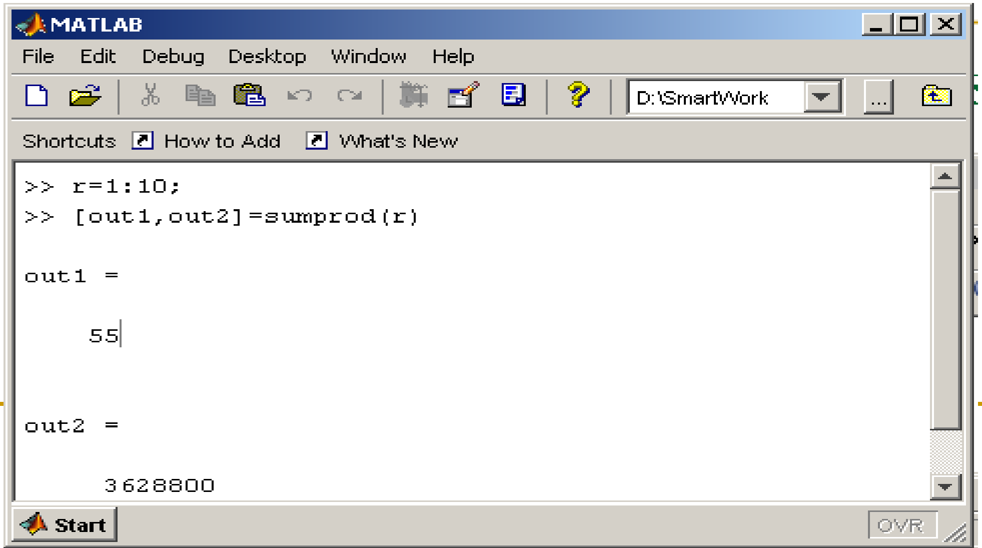
- Another function which takes an input array and returns the sum and product of its elements as outputs



```
function [out1,out2]=sumprod(array)
out1=sum(array);
out2=prod(array);
```

Editor - D:\SmartWork\sumprod.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
squarer.m
sumprod.m
sumprod
Ln 3 Col 18
OVR

- The function sumprod(.) can be called from command window or an m-file as



```
>> r=1:10;
>> [out1,out2]=sumprod(r)

out1 =

    55


out2 =

     3628800
```

MATLAB
File Edit Debug Desktop Window Help
D:\SmartWork
Shortcuts How to Add What's New
Start
OVR

Notes:

- "%" is the neglect sign for Matlab (equaivalent of "//" in C). Anything after it on the same line is neglected by Matlab compiler.
- Sometimes slowing down the execution is done deliberately for observation purposes. You can use the command "pause" for this purpose

```
pause %wait until any key
pause(3) %wait 3 seconds
```

Useful Commands

- The two commands used most by Matlab users are

```
>>help functionname
```

```
>>lookfor keyword
```

Questions

- ?
- ?
- ?
- ?
- ?

Thank You…