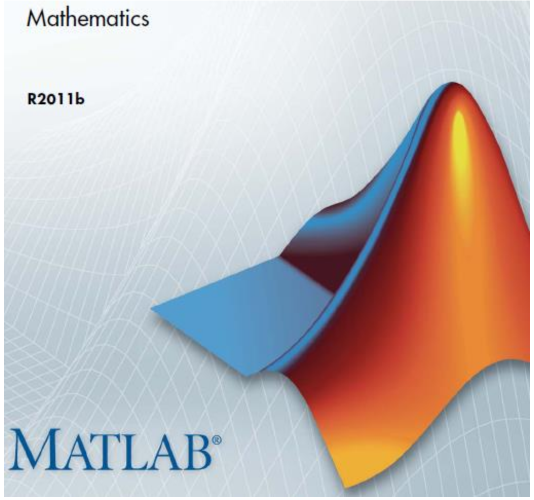
Introduction to MATLAB

Dr.M. Ranjith Kumar

Assistant Professor (Sr. Gr)

Department of Mathematics

Amrita School of Engineering – Chennai



Out Line

This introduction will gives

- Starting a MATLAB
- Matrices and Arrays
- Graphics
- Symbolic Math Toolbox
- Control Flow and if Statement
- Functions

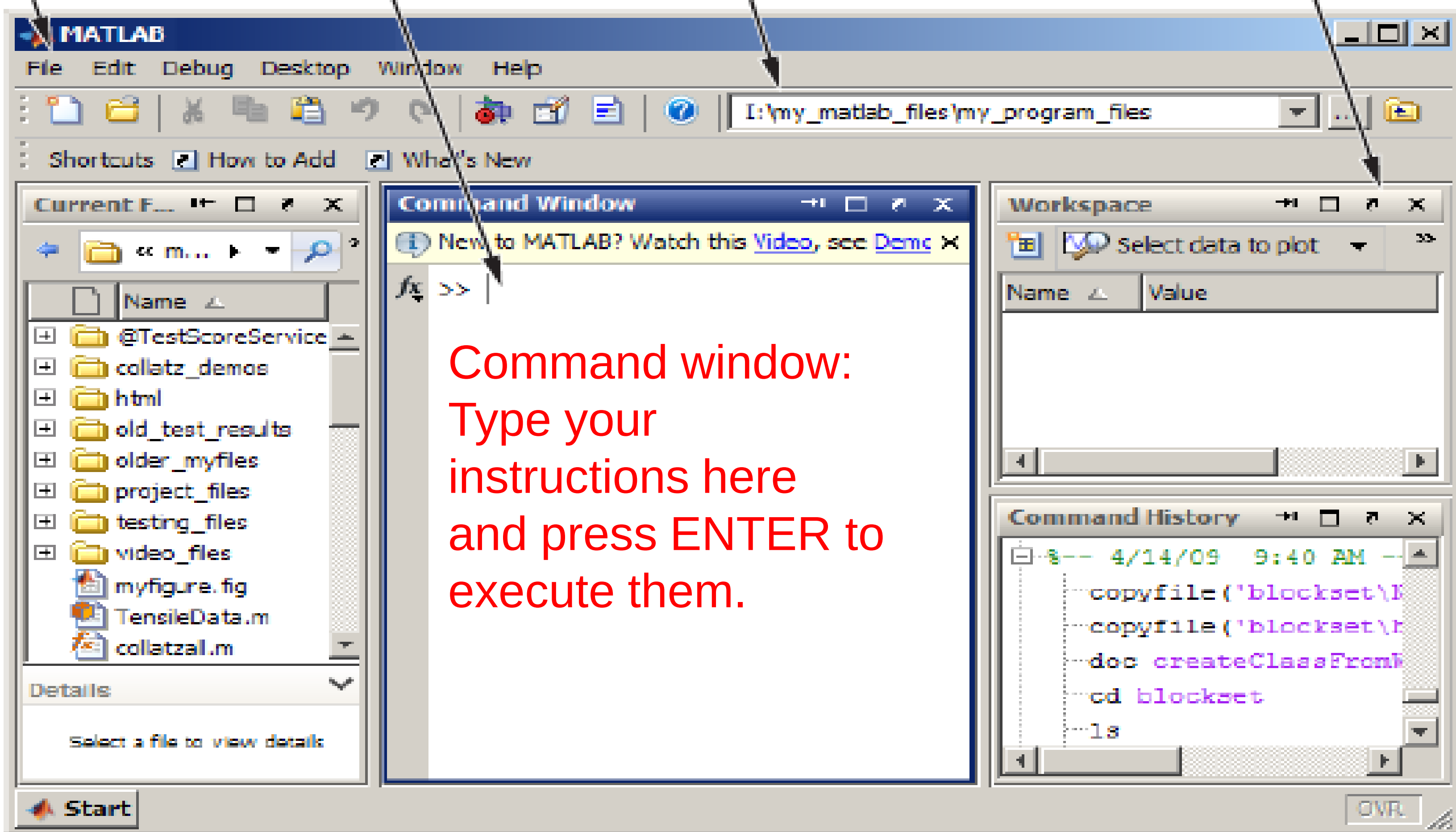
Basic MATLAB Interface

Menus change, depending on the tool you are using.

Enter MATLAB statements at the prompt.

View or change the current folder.

Move, minimize, resize, or close a tool.



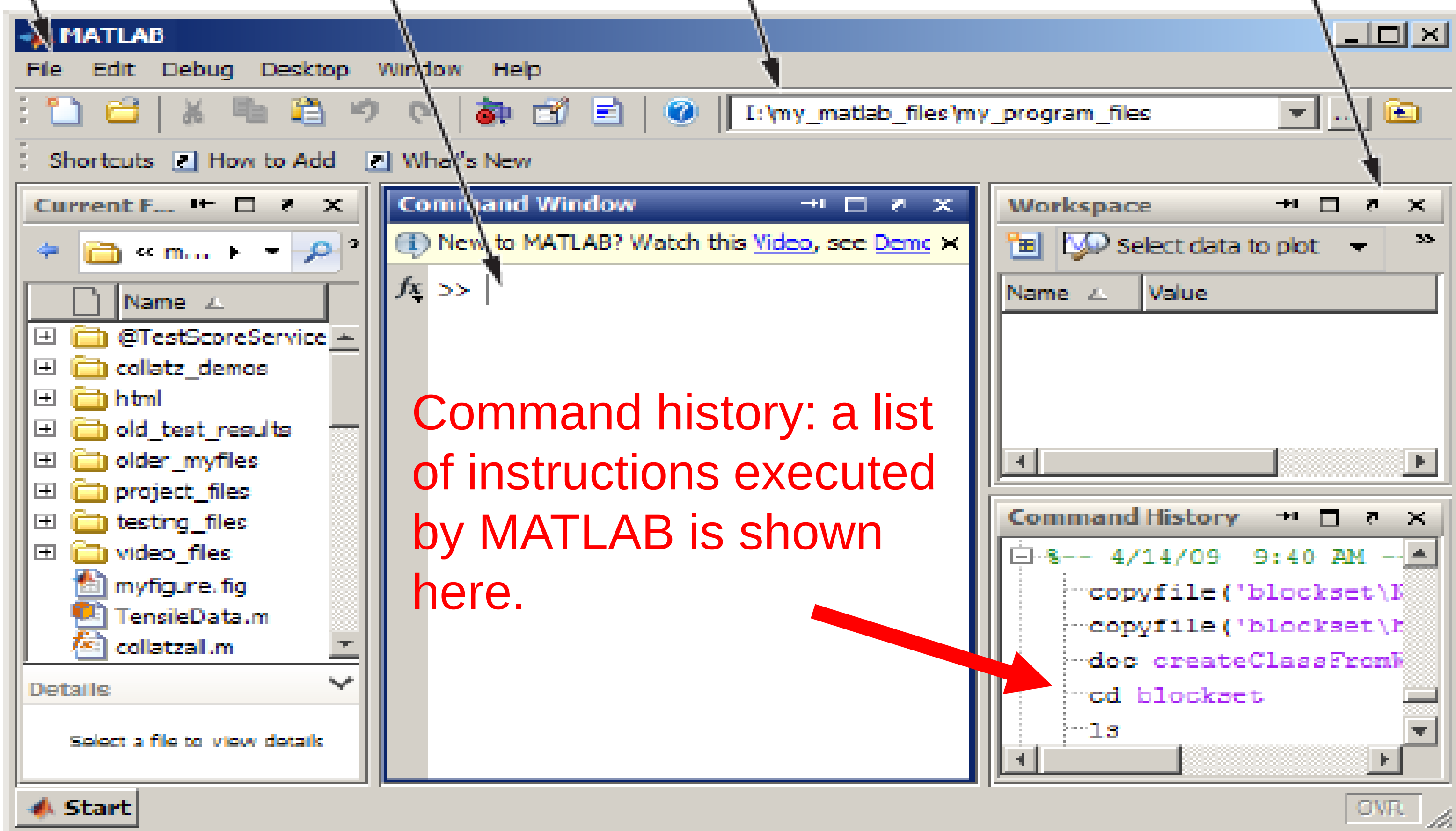
Basic MATLAB Interface

Menus change, depending on the tool you are using.

Enter MATLAB statements at the prompt.

View or change the current folder.

Move, minimize, resize, or close a tool.



Simple calculations

Command Window

```
>> a=3; b=5; a+b, c=a*b, d1=a/b, d2=a\b

ans =

     8


c =

    15


d1 =

    0.6000


d2 =

    1.6667
```

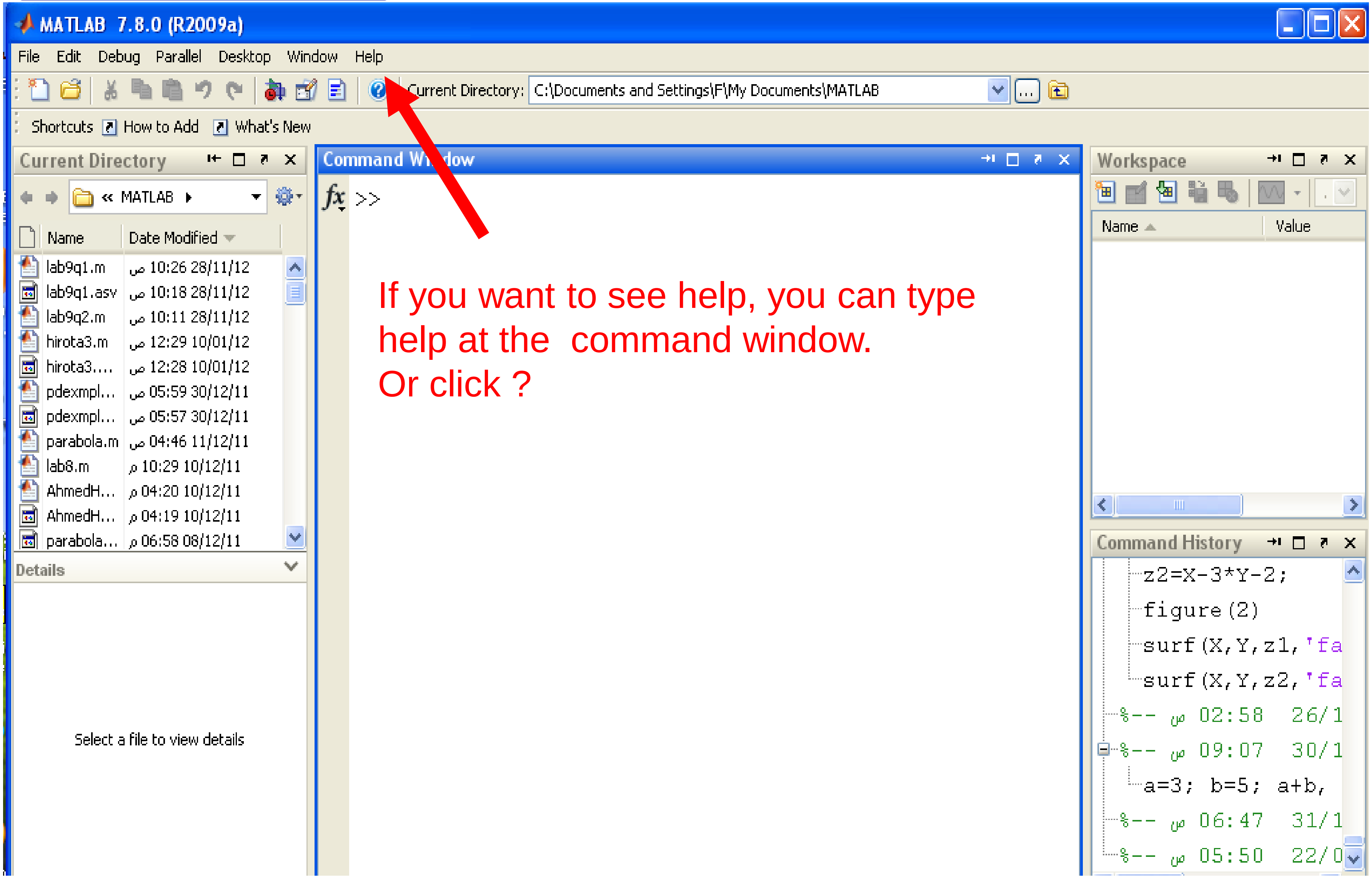
Some special variables and constants are:

ans	Most recent answer.
eps	Floating point relative accuracy.
pi	π
inf	∞
NaN	Not a number.

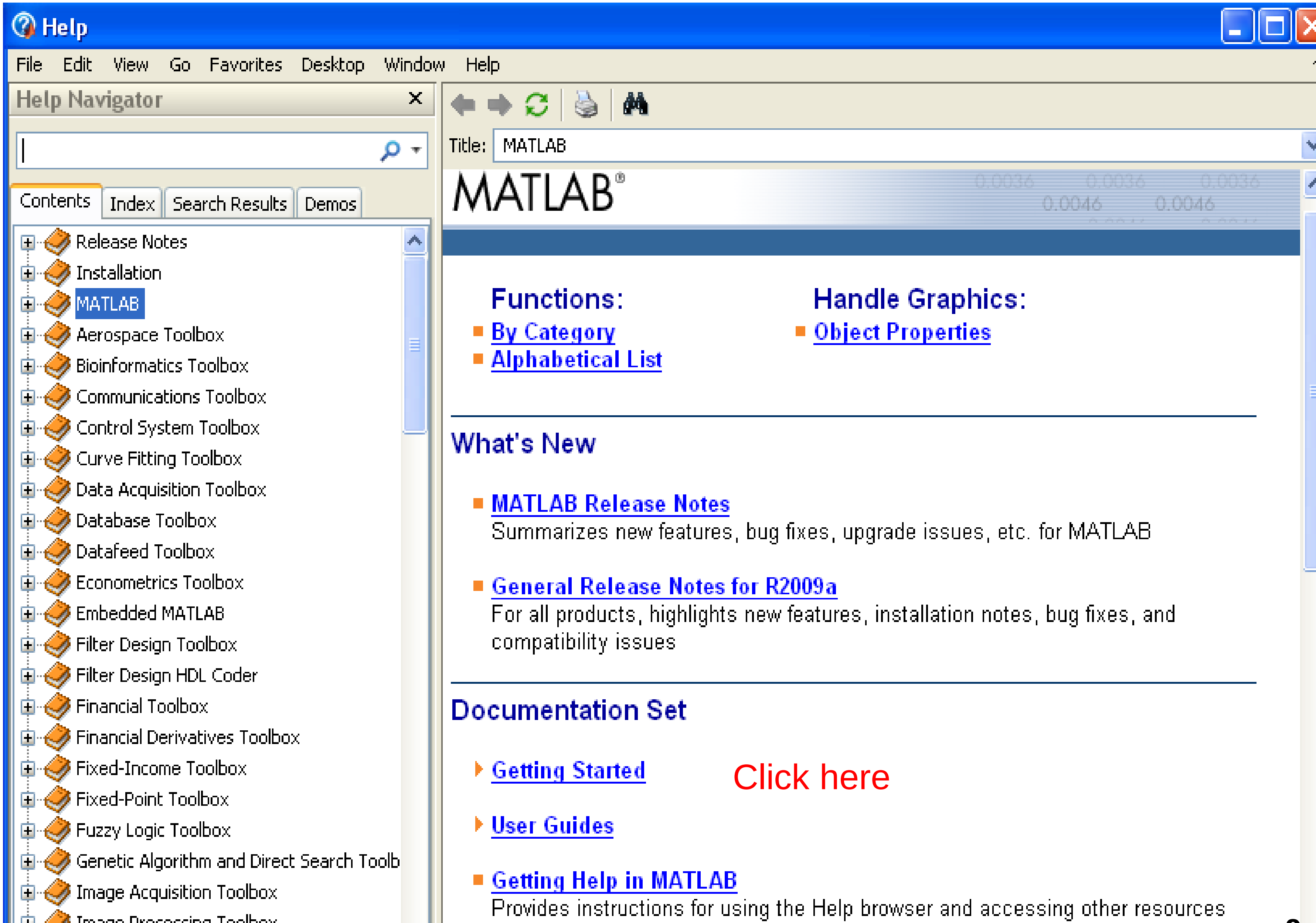
Some Common Mathematical Function:

abs(x)	Absolute value	sqrt(x)	Square root
sin(x)	Sine	asin(x)	Inverse sine
cos(x)	Cosine	acos(x)	Inverse cosine
tan(x)	Tangent	atan(x)	Inverse tangent
log(x)	Natural logarithm	exp(x)	Exponential
log10(x)	Base 10 logarithm	sign(x)	Sign (or) Signum function

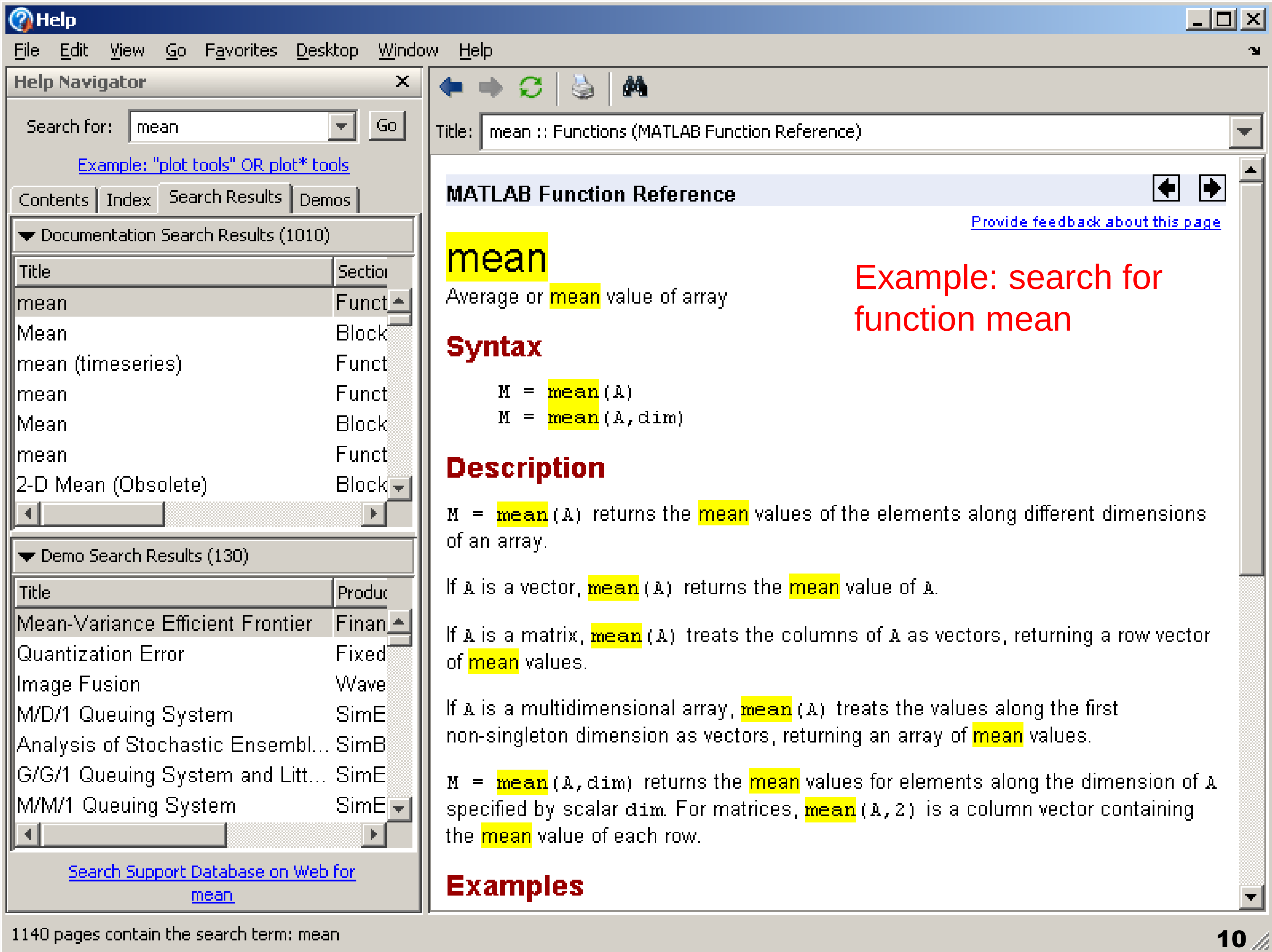
To get help



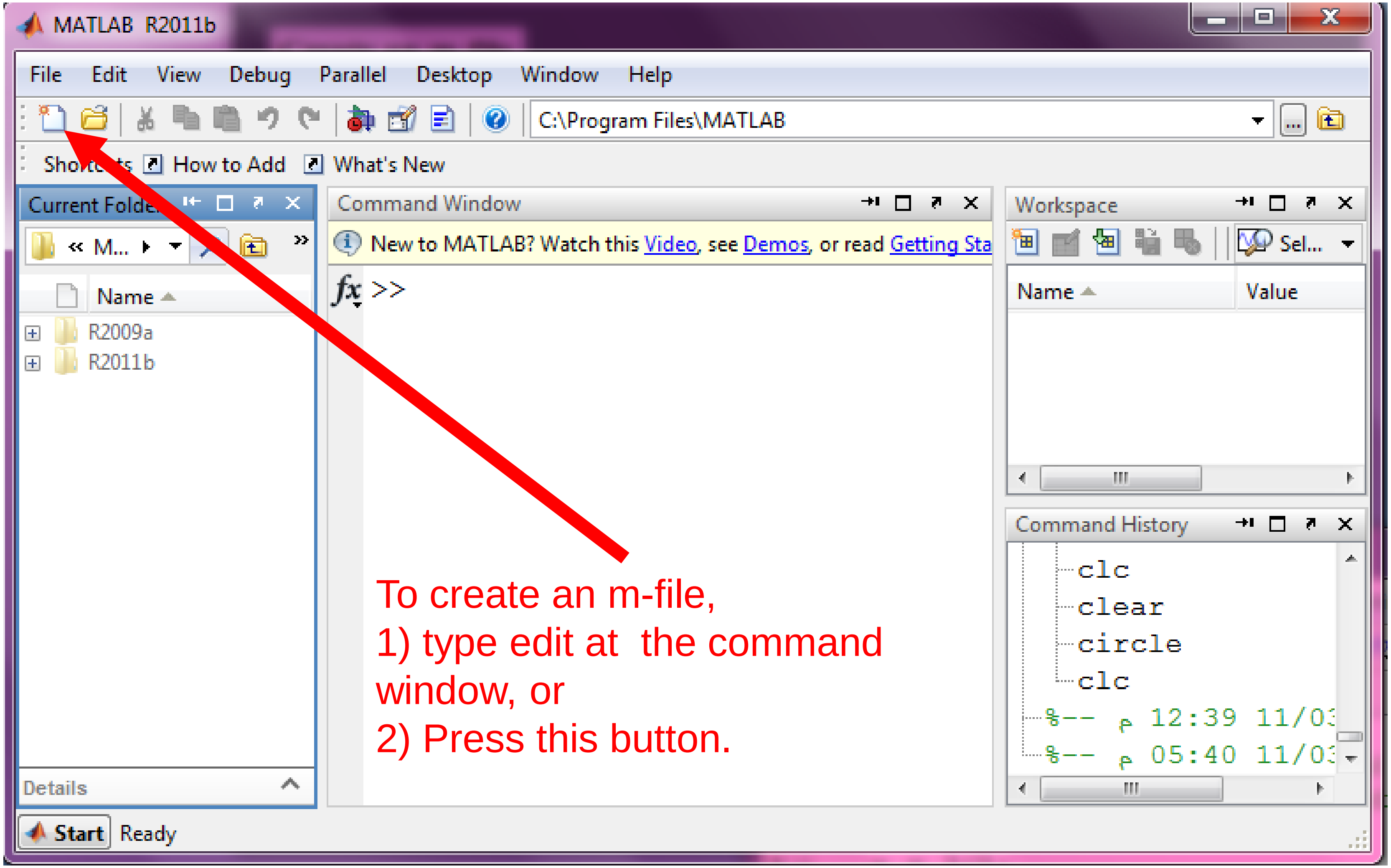
If you want to see help, you can type help at the command window.
Or click ?


Help
File Edit View Go Favorites Desktop Window Help
Help Navigator
Title: MATLAB
Contents
Index
Search Results
Demos
Release Notes
Installation
MATLAB
Aerospace Toolbox
Bioinformatics Toolbox
Communications Toolbox
Control System Toolbox
Curve Fitting Toolbox
Data Acquisition Toolbox
Database Toolbox
Datafeed Toolbox
Econometrics Toolbox
Embedded MATLAB
Filter Design Toolbox
Filter Design HDL Coder
Financial Toolbox
Financial Derivatives Toolbox
Fixed-Income Toolbox
Fixed-Point Toolbox
Fuzzy Logic Toolbox
Genetic Algorithm and Direct Search Toolb
Image Acquisition Toolbox
Image Processing Toolbox
MATLAB®
Functions:
By Category
Alphabetical List
Handle Graphics:
Object Properties
What's New
MATLAB Release Notes
Summarizes new features, bug fixes, upgrade issues, etc. for MATLAB
General Release Notes for R2009a
For all products, highlights new features, installation notes, bug fixes, and compatibility issues
Documentation Set
Getting Started
Click here
User Guides
Getting Help in MATLAB
Provides instructions for using the Help browser and accessing other resources

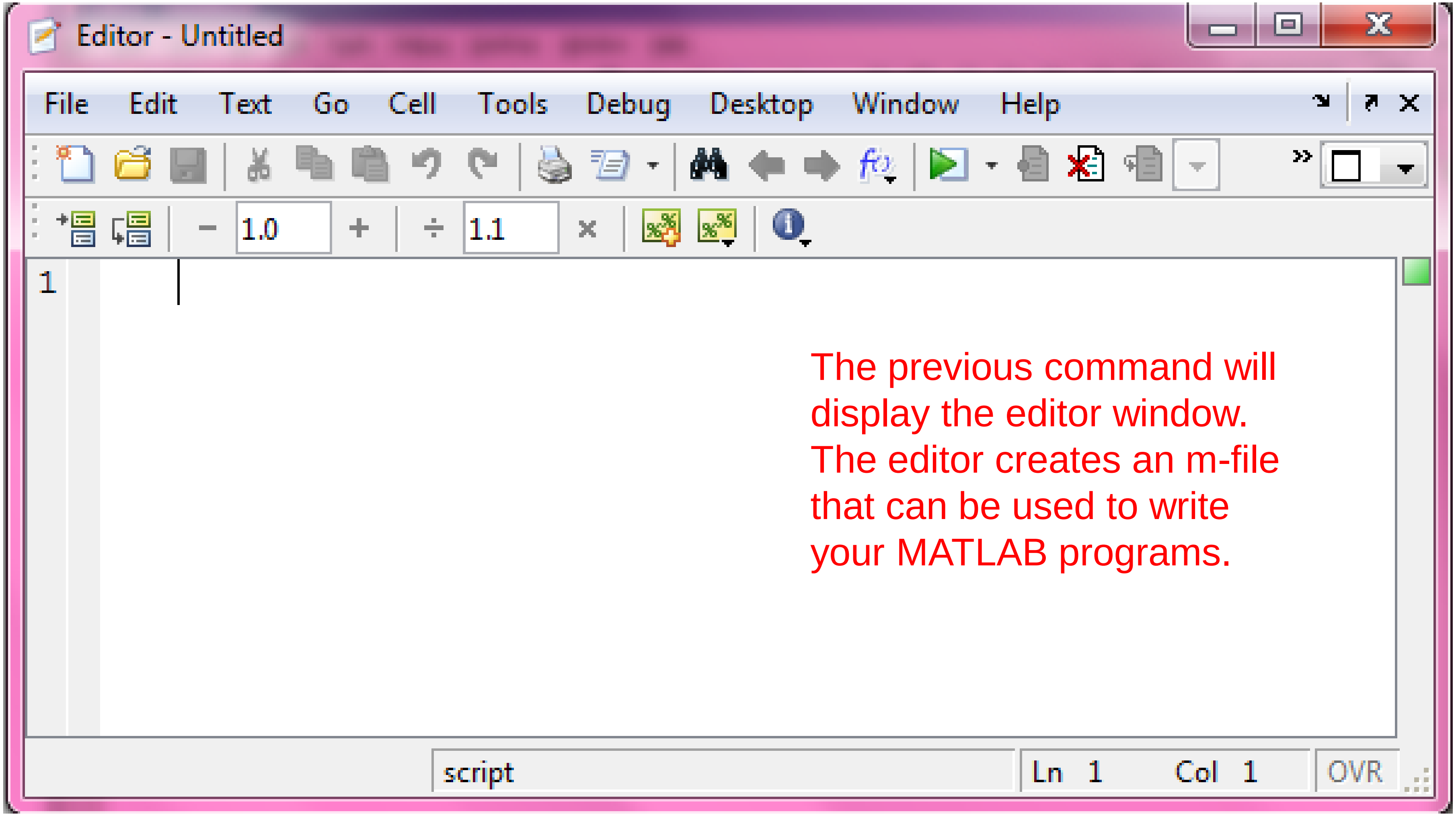
Example: search for function mean



Create an m-file

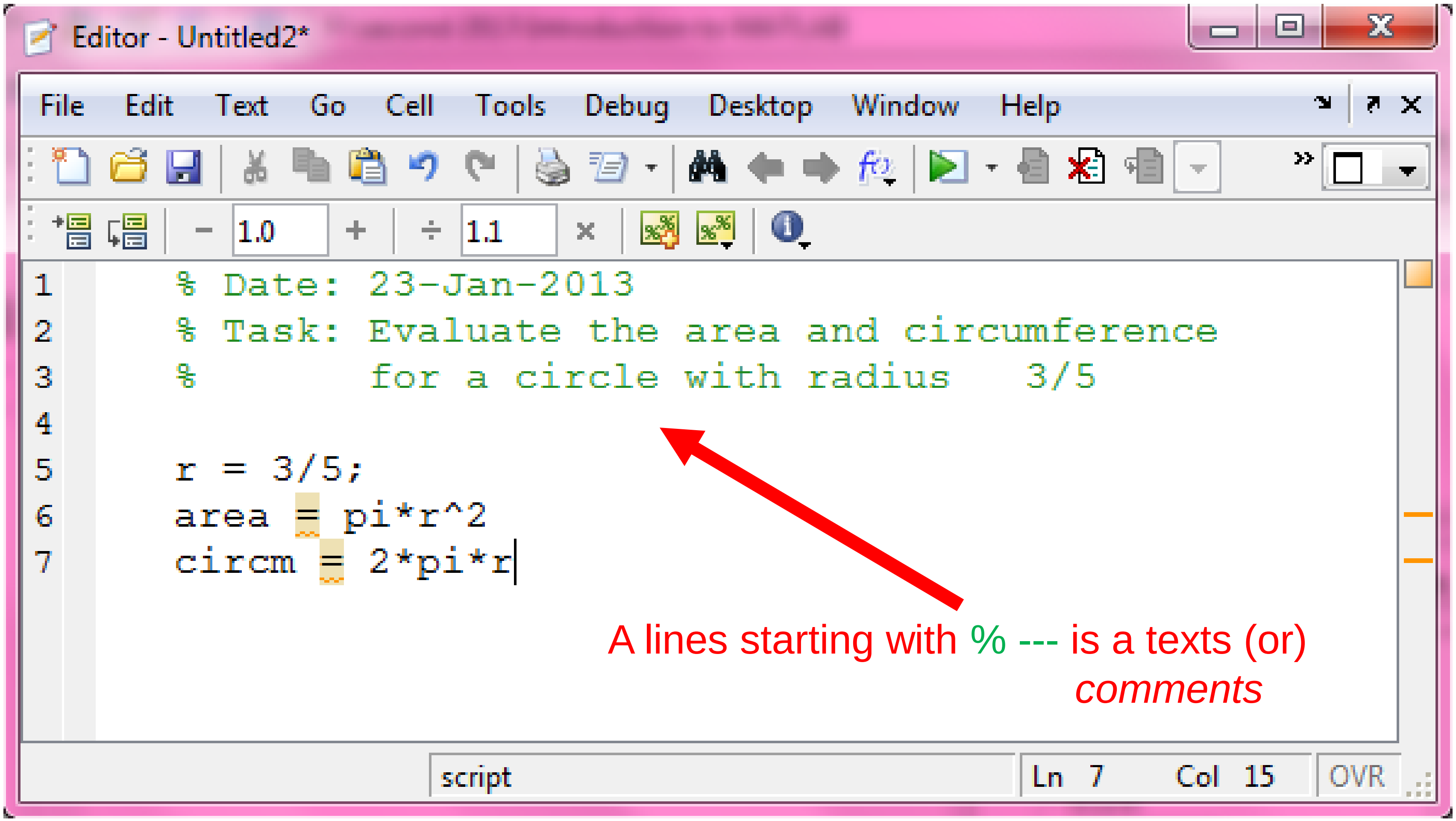


Create an m-file



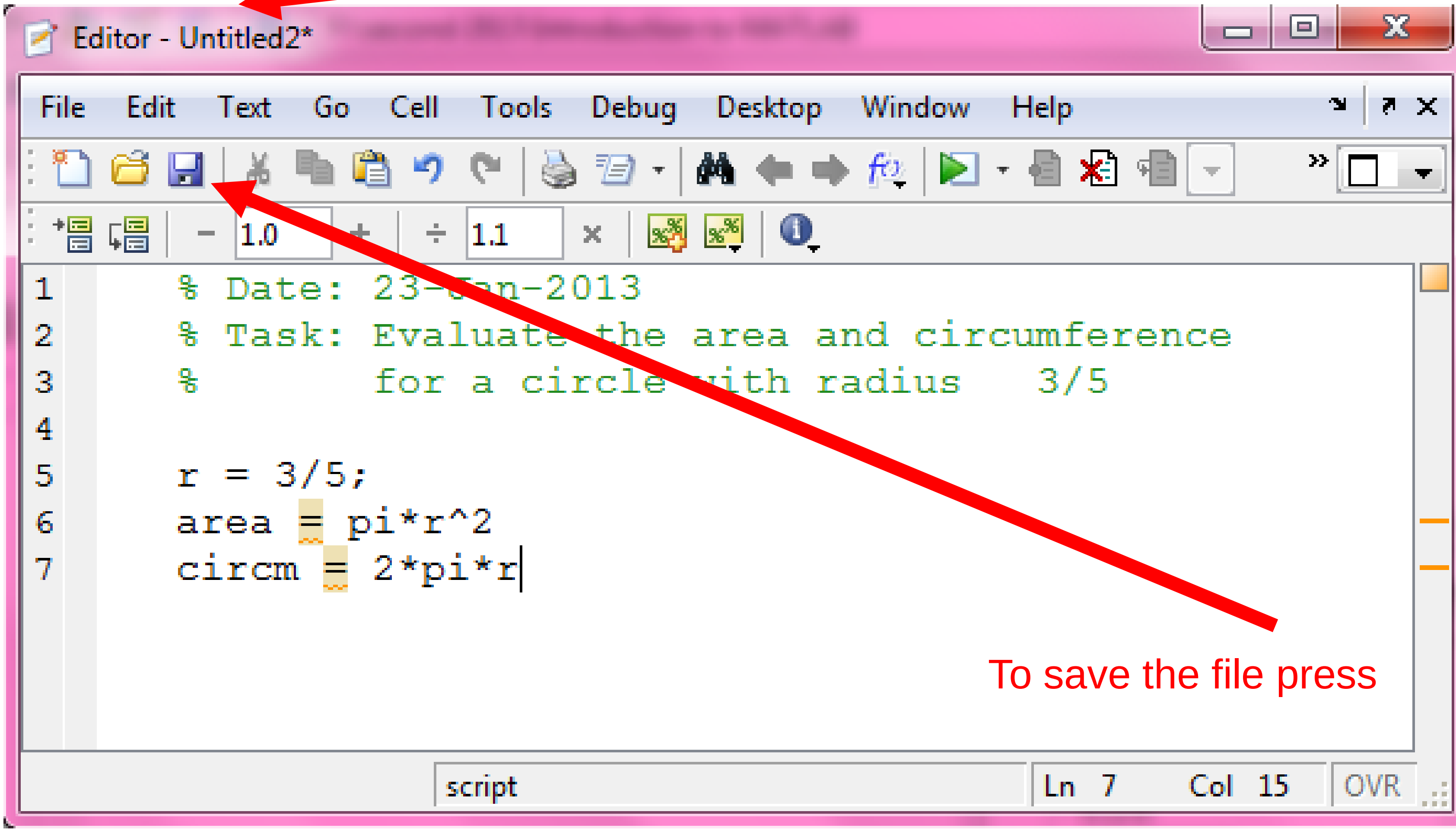
The previous command will display the editor window. The editor creates an m-file that can be used to write your MATLAB programs.

Create an m-file



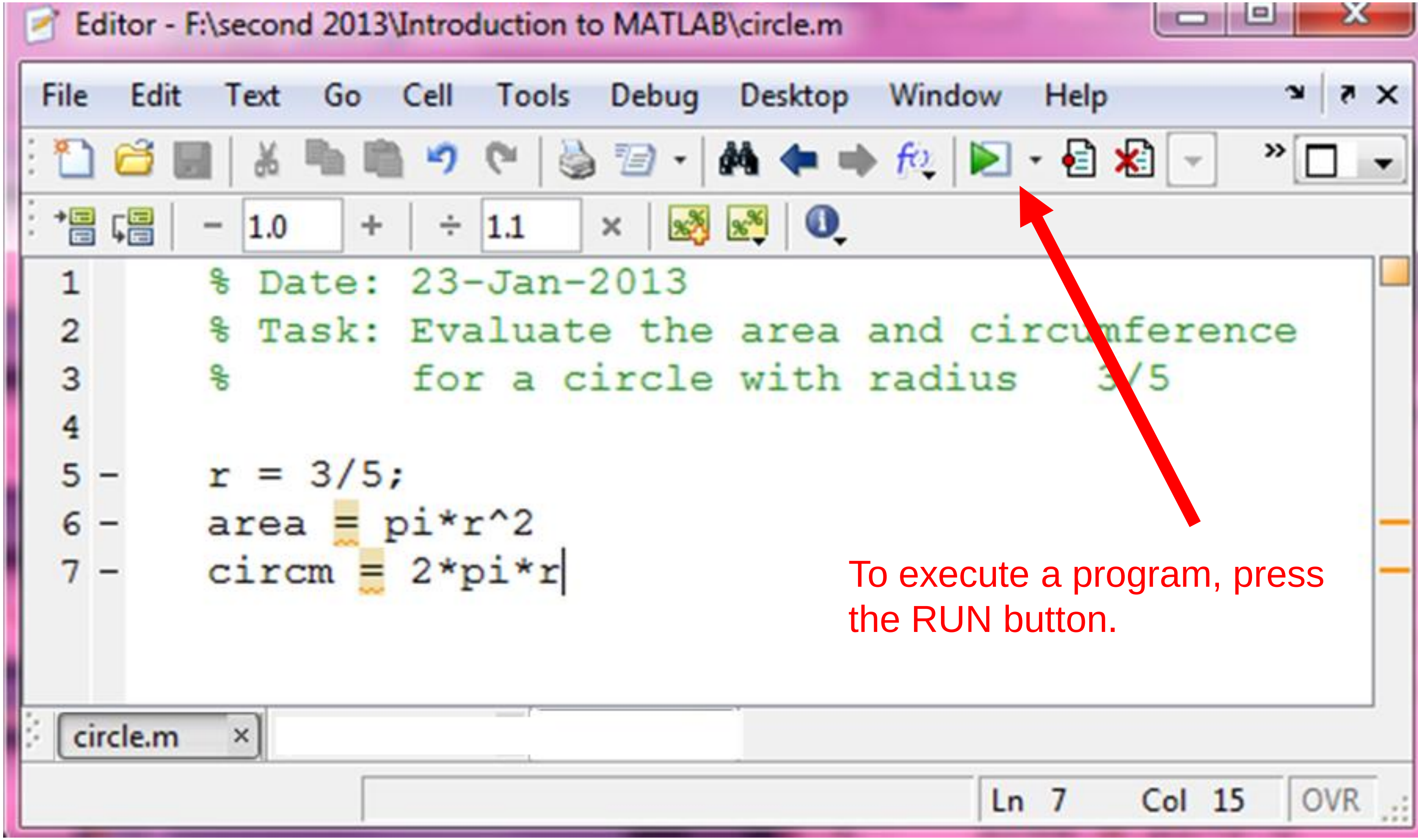
Save an m-file

Note the file name will be changed …



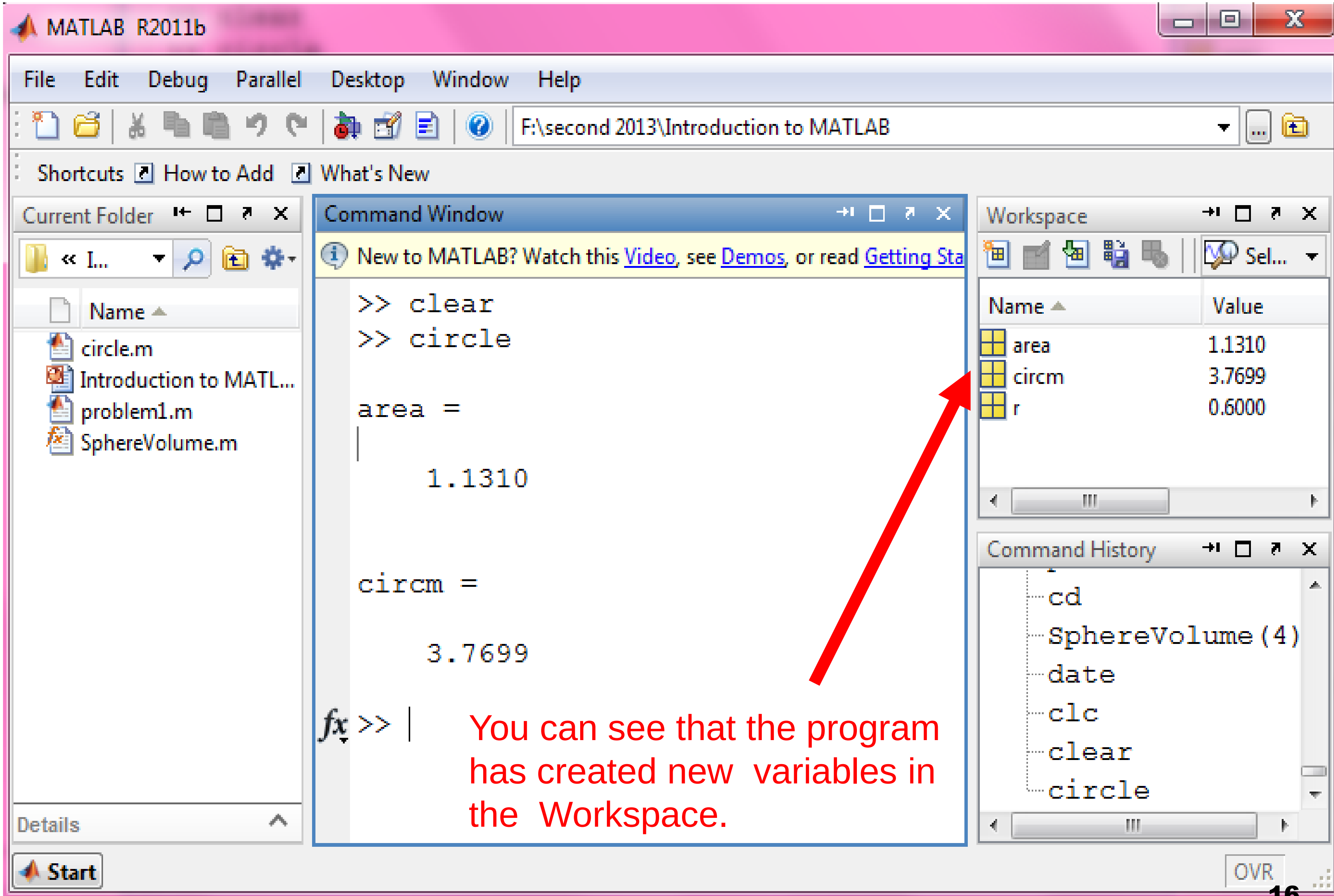
To save the file press

Run an m-file



To execute a program, press the RUN button.

Run an m-file



You can see that the program has created new variables in the Workspace.

Matrices and Arrays

Simple Arrays

```
>> V = [1  3,  sqrt(5)]
V =
   1.0000    3.0000   2.2361


>> A = [1  2  3;4  5  6;7  8 0]
A =
    1     2     3
    4     5     6
    7     8     0
```

Arrays operators

+	Addition
-	Subtraction
*	Multiplication
*.	Element-by-element multiplication
/	division
/.	Element-by-element division
\	left division
\.	Element-by-element left division
^	power
^.	Element-by-element power
'	array transpose
'.	Unconjugated array transpose

Arrays operators Example

A = [1 2 3;4 5 6;7 8 0]

A =

1	2	3
4	5	6
7	8	0

```
A*A  =
     30     36     15
     66     81     42
     39     54     69
```

```
A.*A  =
      1      4      9
     16     25     36
     49     64      0
```

Special Matrices and Vectors

Function	Description
zeros	All zeros
ones	All ones
rand	Uniformly distributed random elements
randne	Normally distributed random elements
Eye(n)	Returns the n-by-n identity matrix.

```
Z = zeros(2,4)
Z =
      0     0     0     0
      0     0     0     0
```

```
F = 5*ones(3,3)
F =
      5     5     5
      5     5     5
      5     5     5
```

Basic Operations and Descriptive Statistics

Function	Description
brush	Interactively mark, delete, modify, and save observations in graphs
cumprod	Cumulative product
cumsum	Cumulative sum
linkdata	Automatically update graphs when variables change
prod	Product of array elements
sort	Sort array elements in ascending or descending order
sortrows	Sort rows in ascending order
sum	Sum of array elements
corrcoef	Correlation coefficients
max	Largest elements in array
mean	Average or mean value of array
median	Median value of array

Basic Operations and Descriptive Statistics

Function	Description
cov	Covariance matrix
min	Smallest elements in array
mode	Most frequent values in array
std	Standard deviation
var	Variance

Example

```
A = [1 2 3; 3 3 6; 4 6 8; 4 7 7];  B = [2,7,8,3];
prod(B)
ans =
   336

mean(A)                                 % mean of each column
ans =
    3.0000    4.5000    6.0000

mean(A,2)                               % mean of each row
ans =
     2
     4
     6
     6
```

Polynomials

Representing Polynomials

Consider the equation $p(x) = x^3 - 2x - 5$

To enter this polynomial into MATLAB, use

p = [1 0 -2 -5];

Polynomial Functions

Function	Description
conv	Multiply polynomials
deconv	Divide polynomials
poly	Polynomial with specified roots
polyder	Polynomial derivative
polyfit	Polynomial curve fitting
polyval	Polynomial evaluation
polyvalm	Matrix polynomial evaluation
residue	Partial-fraction expansion (residues)
roots	Find polynomial roots

Examples

To evaluate p at x=5, use

```
polyval(p,5)
 ans =
          110

X = [2 4 5; -1 0 3; 7 1 5];
Y = polyvalm(p,X)
Y =
   377   179   439
   111    81   136
   490   253   639
```

Examples

To find the characteristic polynomial of a matrix

A = 1 1 1 1

1 2 3 4

1 3 6 10

1 4 10 20

Use

>>p=poly(A)

ans =

p=1X5

1 -29 72 -29 1

The characteristic polynomial is

p(x)=$x^4 - 29x^3 + 72x^2 - 29x + 1$

Examples

Roots of Polynomials $p(x) = x^3 - 2x - 5$

To find the roots p:

```
>>r = roots(p)

r =
   2.0946
  -1.0473 + 1.1359i
  -1.0473 - 1.1359i
```

To find a polynomial that have the roots r:

```
>>p2 = poly(r)

p2 =
    1.0000   -0.0000   -2.0000   -5.0000
```

Examples

Derivatives

To obtain the derivative of the polynomial p = [1 0 -2 -5];

```
>>q = polyder(p)

q =
    3     0    -2
```

Polynomial curve fitting

>>p = polyfit(x,y,n) finds the coefficients of a polynomial p(x) of degree n that fits the data, x(i) , y(i)

```
>>x = 0: 0.1: 2.5; y = erf(x);
>>p = polyfit(x,y,4)

p =
   0.0238   -0.0262   -0.4343    1.2840   -0.0096
```

Linear Algebra Functions

Function	Description	Function 2	Description 3
norm	Matrix or vector norm	inv	Matrix inverse
rank	Matrix rank	subspace	Angle between two subspaces
det	Determinant	linsolve	Solve a system of linear equations
trace	Sum of diagonal elements	lu	LU factorization
null	Null space	chol	Cholesky factorization
orth	Orthogonalization	eig	Eigenvalues and eigenvectors
rref	Reduced row echelon form	expm	Matrix exponential
pinv	Pseudoinverse	funm	Evaluate general matrix function

Examples

$$A=\begin{bmatrix}1 & -2 & 3\\ 4 & 0 & 6\\ 2 & -1 & 3\end{bmatrix}$$

```
A=[1 -2 3;4 0 6; 2 -1,3]
[L,U,P] = lu(A)

A =
     1    -2     3
     4     0     6
     2    -1     3

L =
    1.0000         0         0
    0.2500    1.0000         0
    0.5000    0.5000    1.0000

U =

    4.0000         0    6.0000
         0   -2.0000    1.5000
         0         0   -0.7500

P =
     0     1     0
     1     0     0
     0     0     1
```

Linear system of equations

The system

$$\begin{bmatrix} 1 & 0 & 3 \\ 0 & 5 & 6 \\ 7 & 8 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ -2 \\ 3 \end{bmatrix}$$

can be solved as:

A = [1 0 3;0 5 6;7 8 0];b = [5;-2;3];

x = inv(A)*b
x =
 2.1765
-1.5294
 0.9412

Or

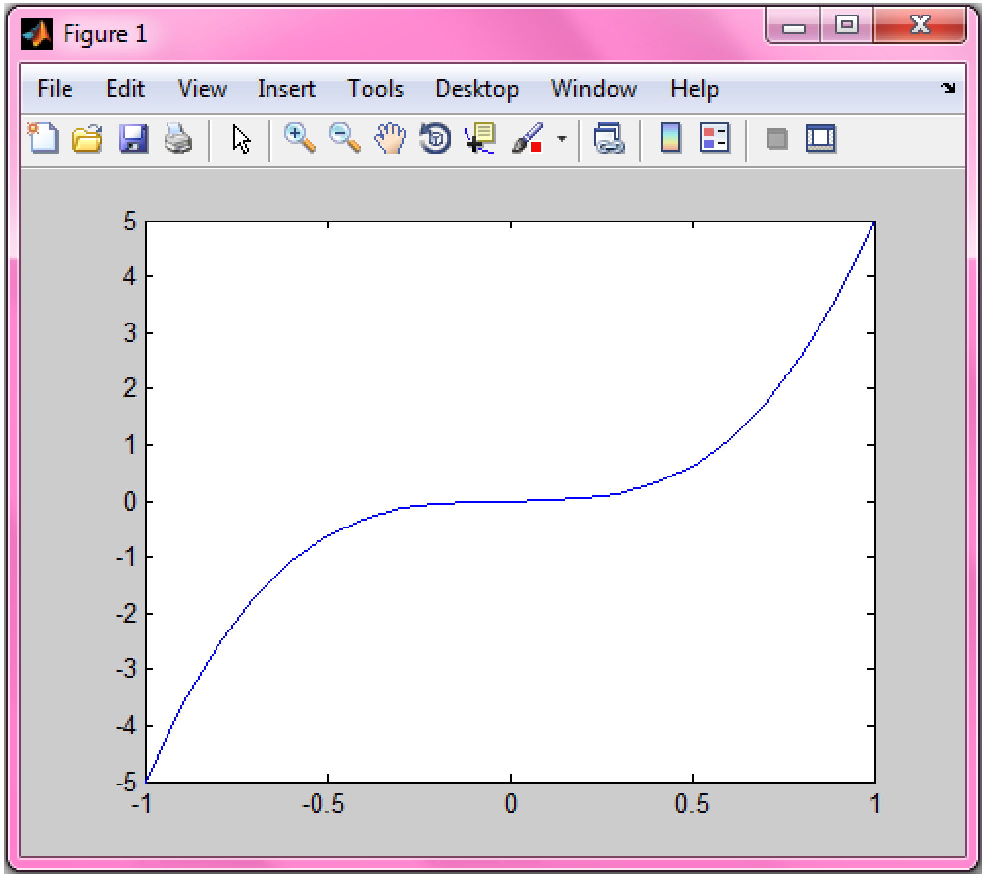
y=linsolve(A,b)
y =
 2.1765
-1.5294
 0.9412

Plot

Two-Dimensional Graphics

Suppose we want to graph the function $y = 5x^3$ over the x domain -1 to 1 .

```
x = -1:0.1:1;
y = 5*x^3;
plot(x,y)
```

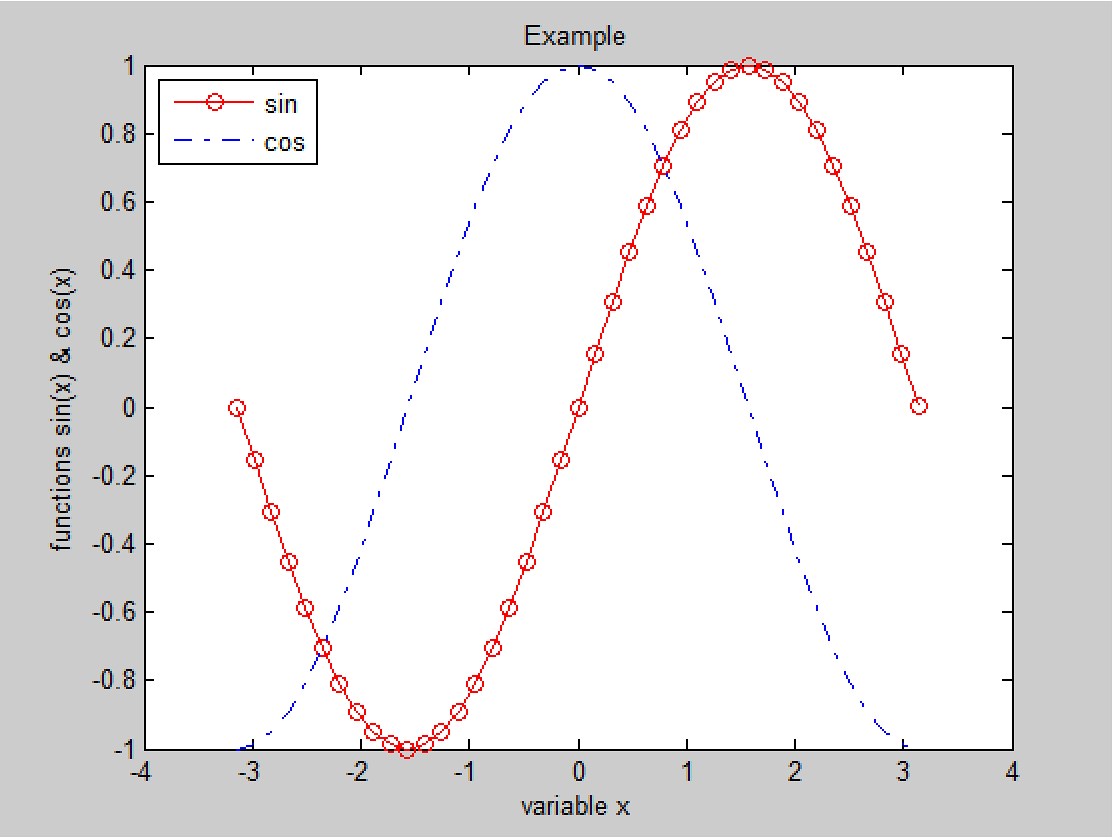


Changing the Appearance

Symbol	Color	Symbol 2	Marker	Symbol 3	Line style
b	blue	.	point	-	solid line
g	green	o	circle	:	dotted lin
r	red	x	x-mark	.-	dash-dot line
c	cyan	+	plus	--	dashed line
m	magenta	*	star		
y	yellow	s	square		
k	black	d	diamond		
w	white	^	triangle		
		>	triangle left		
		<	triangle right		
		P	pentagram		
		h	hexagram		

To add a title and axis labels write

```
x  = -pi:pi/20:pi;
f1 = sin(x);
f2 = cos(x);
plot(x,f1,'-ro',x,f2,'-.b')
xlabel('variable x')
ylabel('functions sin(x) & cos(x)')
title('Example')
legend('sin','cos',2);
```

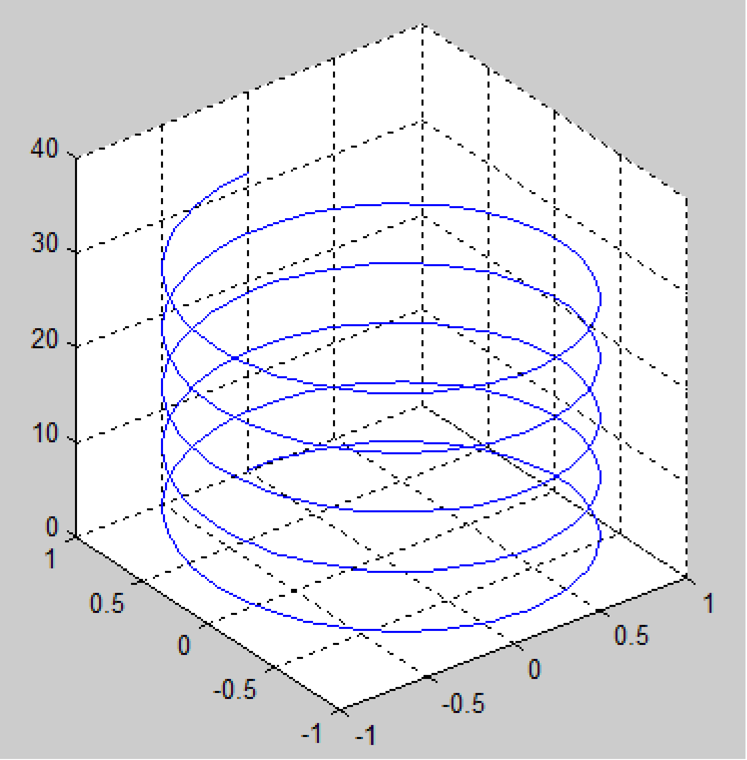


Three-Dimensional Graphics

1. Line plotes

The plot3 function displays a three-dimensional plot of a set of data points. Here is an example of a three-dimensional helix:

```
t = 0:pi/50:10*pi;
plot3(sin(t), cos(t), t)
grid on, axis square
```

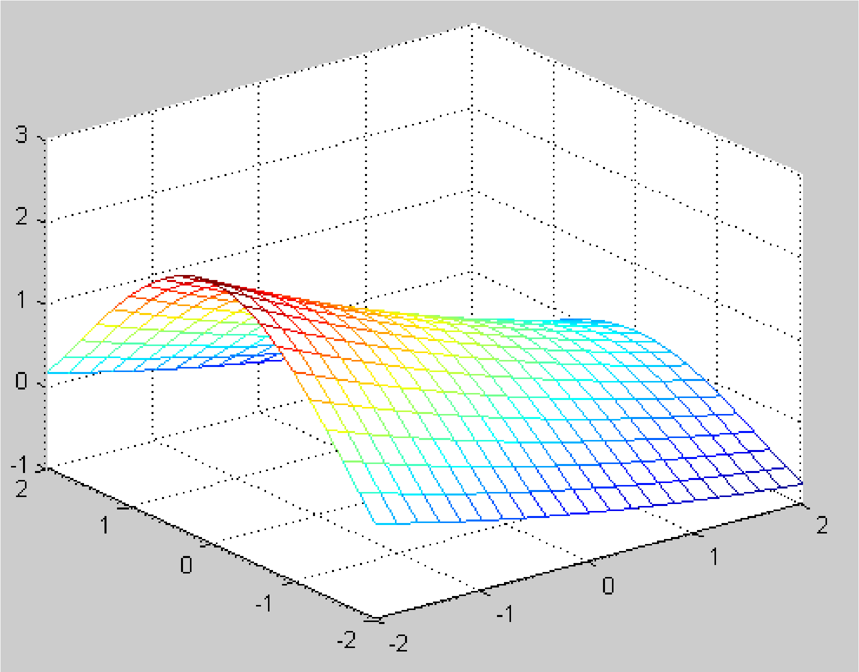


Three-Dimensional Graphics

2. Mesh and Surface Plot

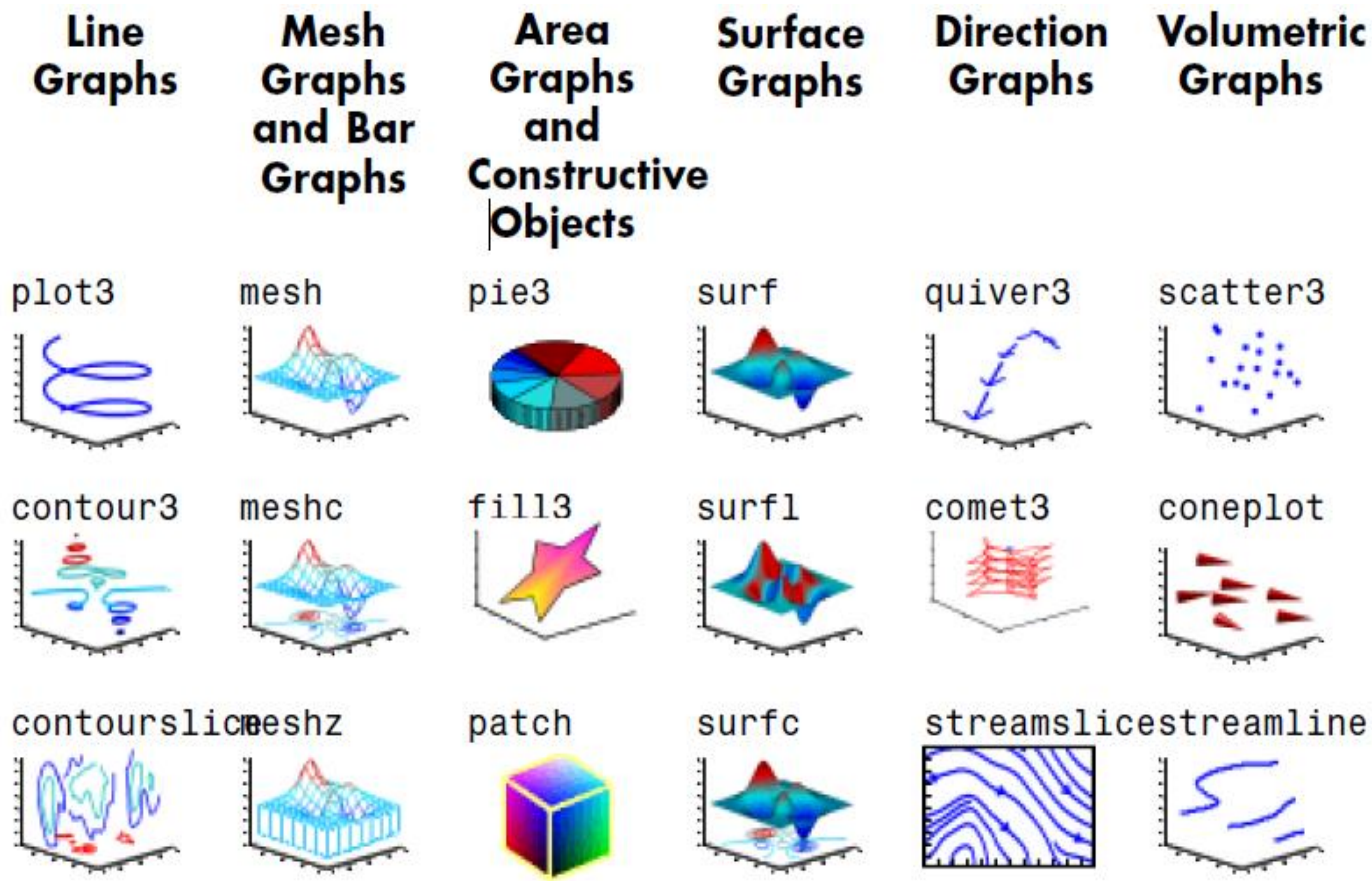
Matlab defines a mesh surface by the z-coordinates of points above a rectangular grid in the xy-plane.
The result looks like a fishing net.
Here is an example:

```
[X,Y] = meshgrid(-2:.2:2);
R = sqrt(X + Y.^2);
Z = cos(R);
mesh(X,Y,Z)
```



Three-Dimensional Graphics

The table below shows some available MATLAB 3-D and volumetric plot functions.



Symbolic Math Toolbox

Using symbolic math toolbox we can do:

- Differentiation
- Integration
- Linear algebraic operations
- Simplification
- Transforms
- Variable-precision arithmetic
- Equation solving
- …………..
- …………..

Examples

```
>>  syms  x;
g  =  x^3  +  6*x^2  +  11*x  +  6;
factor(g)
ans =
(x + 3)*(x + 2)*(x + 1)

>> diff(g)
ans =
3*x^2 + 12*x + 11

>> int(g)
ans =
x^4/4 + 2*x^3 + (11*x^2)/2 + 6*x
```

```
>> solve( 'x^2 + 4*x + 1'  )
ans =
   3^(1/2) - 2
 - 3^(1/2) - 2
```

```
>>  y = dsolve('D2y-2*Dy-3*y=0','y(0)=0','y(1)=1')
y =
  1/(exp(-1)-exp(3))*exp(-t)-1/(exp(-1)-exp(3))*exp(3*t)
```

Control Flow and if Statement

Control flow is extremely powerful; its lets past computations influence future operations. MATLAB has several flow control constructs:

if , switch and case, for, while continue, break, try–catch, return.

- if, else, and elseif

A simple example is to evaluate

$$f(x) = \begin{cases} x^2 & \text{if} \quad x \le 2 \\ x^3 & \text{if} \quad x > 2 \end{cases}$$

at a given points:

```
if        x<=2  f=x^2
else            f=x^3
end
```

Operators

Often we need some relational or logical operators to companion if statement. Operators are shown in the following table:

Relational Operators		Logical Operators	
>	Less than	& &	Logical AND
= >	Less than or equal to	| |	Logical OR
<	Greater than	&	Logical AND for arrays
= <	Greater than or equal to	|	Logical OR for arrays
= =	Equal to	~	Logical NOT
= ~	Not equal to		

for loop

for loop allow a group of commands to be repeated a fixed, predetermined number of times. For example:

```
for   i = 1:4
   x(i) = i^2
end

for  k = 2:5:20,   y = k^3 - 7,   end

for   x = [2 0 3], y = x^3 - 5*x,  end
```

while loop

A while loop evaluates a group of statements an indefinite number of times such as

```
c = 0; i=1;
while c==0
   i=i+2
   s=1/i
   if s<=0.1 c=1
   end
end
```

Bisection Method Example

Here is a complete program, illustrating while, if, else, and end, which uses interval bisection to find a zero of a polynomial:

```
a = 0; fa = -Inf;
b = 3; fb = Inf;
while b-a > eps*b
   x = (a+b)/2;
   fx = x^3-2*x-5;
   if fx == 0
      break
   elseif sign(fx) == sign(fa)
      a = x; fa = fx;
   else
      b = x; fb = fx;
   end
end
x
```

Functions

You will often need to build your own MATLAB functions as you use MATLAB to solve problems.

- Inline

g = inline('t^2')

g(3)

f = inline('sin(alpha*x)')

f(3,pi/2)

- function_handle (@)

sqr = @(x) x.^2

Sqr(5)

sqrpy = @(x,y) x.^2+y

srtpy(2,1)

```
function [out1, out2, ...] = funname(in1, in2, ...)
```

Functions

This function calculates the mean and standard deviation of a vector:

```
function [mean,stdev] = stat(x)
n = length(x);
mean = sum(x)/n;
stdev = sqrt(sum((x-mean).^2/n));
```

```
[mean stdev] = stat( [12.7  45.4  98.9  26.6  53/1] )
mean =
   47.3200
stdev =
   29.4085
```

ODE function

Defining an ODE function in an M-file

```
[outputs] = function_handle(inputs)
[t,state] = solver(@dstate,tspan,ICs,options)
```

An array. The solution of the ODE (the values of the state at every time).

Matlab algorithm (e.g., `ode45`, `ode23`)

Handle for function containing the derivatives

Vector that specifiecs the interval of the solution (e.g., `[t0:5:tf]`)

A vector of the initial conditions for the system (row or column)

… Solving first-order ODEs

Example

$$y' = -2ty^2, \qquad y(0) = 1$$

```
function [T,Y] = eq2()
    format long
    tspan = [0 .25 .5 .75 1]; y0 = 1;
    [t1 y1] = ode23(@odeq, tspan, y0);
    [t2 y2] = ode45(@odeq', tspan, y0);
    R = [t1 y1 y2]
end

function dy = odeq(t,y)
    % The m-file for the ODE y' = -2ty^2.
    dy = -2*t*y(1).^2;
end
```

Output

```
R =

                   0   1.00000000000000   1.00000000000000
   0.25000000000000   0.94118221525751 4   0.94117646765649 6
   0.50000000000000   0.80002280597122 2   0.79999999678379 9
   0.75000000000000   0.64001788410486 7   0.63999998775736 3
   1.00000000000000   0.49999658522365 9   0.50000000471194 2
```

Example

$$
\begin{aligned}
y'_1 &= y_2 y_3 & y_1(0) &= 0 \\
y'_2 &= -y_1 y_3 & y_2(0) &= 1 \\
y'_3 &= -0.51 y_1 y_3 & y_3(0) &= 1
\end{aligned}
$$

```matlab
% An example of a nonstiff system
% --------------------------------
function out = odeexample()
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-5]);
[T,Y] = ode45(@rigid,[0 12],[0 1 1],options);
plot(T,Y(:,1),'-',T,Y(:,2),'-.',T,Y(:,3),'.')
end

function dy = rigid(t,y)
dy = zeros(3,1);    % a column vector
dy(1) = y(2) * y(3);
dy(2) = -y(1) * y(3);
dy(3) = -0.51 * y(1) * y(2);
end
```

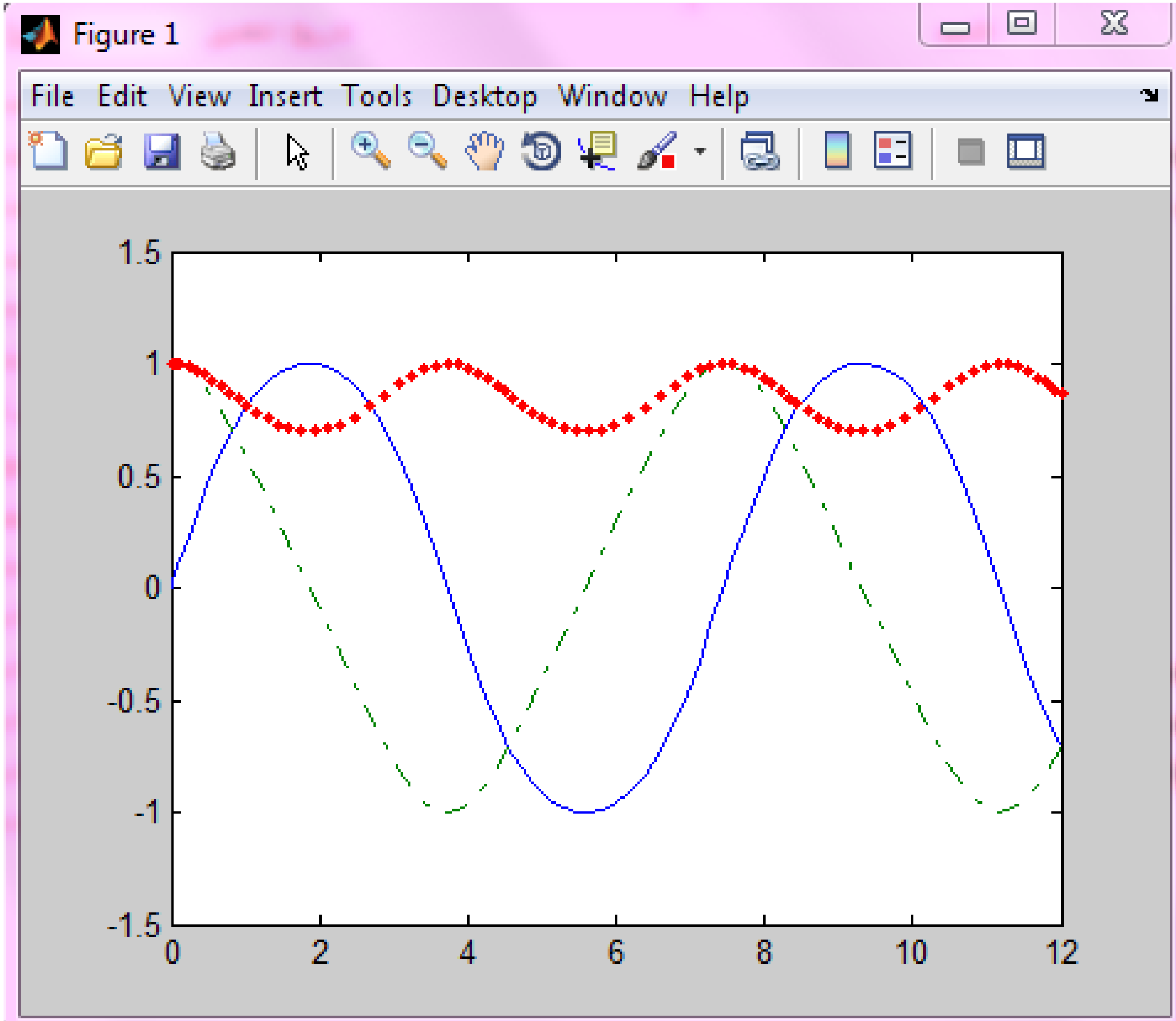


Figure 1
File Edit View Insert Tools Desktop Window Help
1.5
1
0.5
0
-0.5
-1
-1.5
0
2
4
6
8
10
12

Problems

Sample Problems

Problem 1

Write a program that plots the function:

$$y = x^3 + x^2 + 3x + 6$$

for values of x = -50 to 50

Problem 2

Write a program that calculates and displays the volume of a sphere when given the radius. The volume calculation must be performed in a function called Sphere Volume.

The formula for volume is:

$$v = \frac{3}{4}\pi r^2$$

Problem 3 **Matrix-Vector Products**

(a) Create a matrix A=[1 0 -1; 2 1 -1; 3 2 -1].

(b) Create a vector u=[1 2 3]. Is u a row vector (1x3) or a column vector (3x1)?

(c) Create a vector v=[1;2,;3]. Is v a row vector or a column vector?

(d) Try computing A*u and A*v. One works and one doesn't. Which one does? Why does this make sense? (Comment out the one that doesn't work so the script file runs without an error.)

(e) Try computing A.*u. What does this operation do?

(f) Create a 10x10 matrix B of 1's, and a vector w with 10 entries 1 through 10. Calculate Bw.

Problem 4 **Dot Product**

(a) Define the vector x=[-1 0 -1]

(b) We want to calculate the dot product u · x. Try u*x. Why doesn't this work? (After you try it and see the error, comment out the line of code so the script file runs.)

(c) Try u.*x. This doesn't give an error, but it also doesn't calculate the dot product. Why?

(d) The transpose of a vector or matrix flips the matrix over its diagonal. For vectors, this means it makes a row vector into a column vector or vice versa. Compute transpose(u)

(e) Compute x*transpose(u) and u*transpose(x) and transpose(u)*x? Which of these gives the dot product and which doesn't?

(f) Use Google and find another way of calculating the transpose in Matlab. Calculate the transpose of A from problem 3.

Problem 5 **Matrix-Matrix Products**

(a) Define matrices

$$C = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & -6 \end{bmatrix} \; and \; D = \begin{bmatrix} 1 & -2 & 3 \\ 4 & 5 & -6 \end{bmatrix}$$

(b) Calculate C*D and D*C.

(c) Create a matrix I=eye(3). What matrix is this?

(d) Calculate AI and A − I where A is the matrix defined in problem 3

(e) Calculate A.*(5*I) What is 5I and what does the **.*** operation do?

Problem 6 **Matrix Inverses**

(a) Define

$$M = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -1 & 6 \\ 1 & 1 & 2 \end{bmatrix}$$

(b) Calculate inv(M)

(c) Calculate M^(-1)

(d) What's the difference between the two commands?

(e) Solve the system M~x = ~v (~v is defined in Problem 3) using a matrix inverse.

(f) Calculate M\v Then Google ”matrix inverse matlab” and read the ”Tips” section of the Matlab documentation on Matrix inverse. What's the difference between M\x and inv(M)*x

(e) Find Characteristic Polynomial, Eigen values and Eigen vectors of M.