Chapter 1

Starting with MATLAB

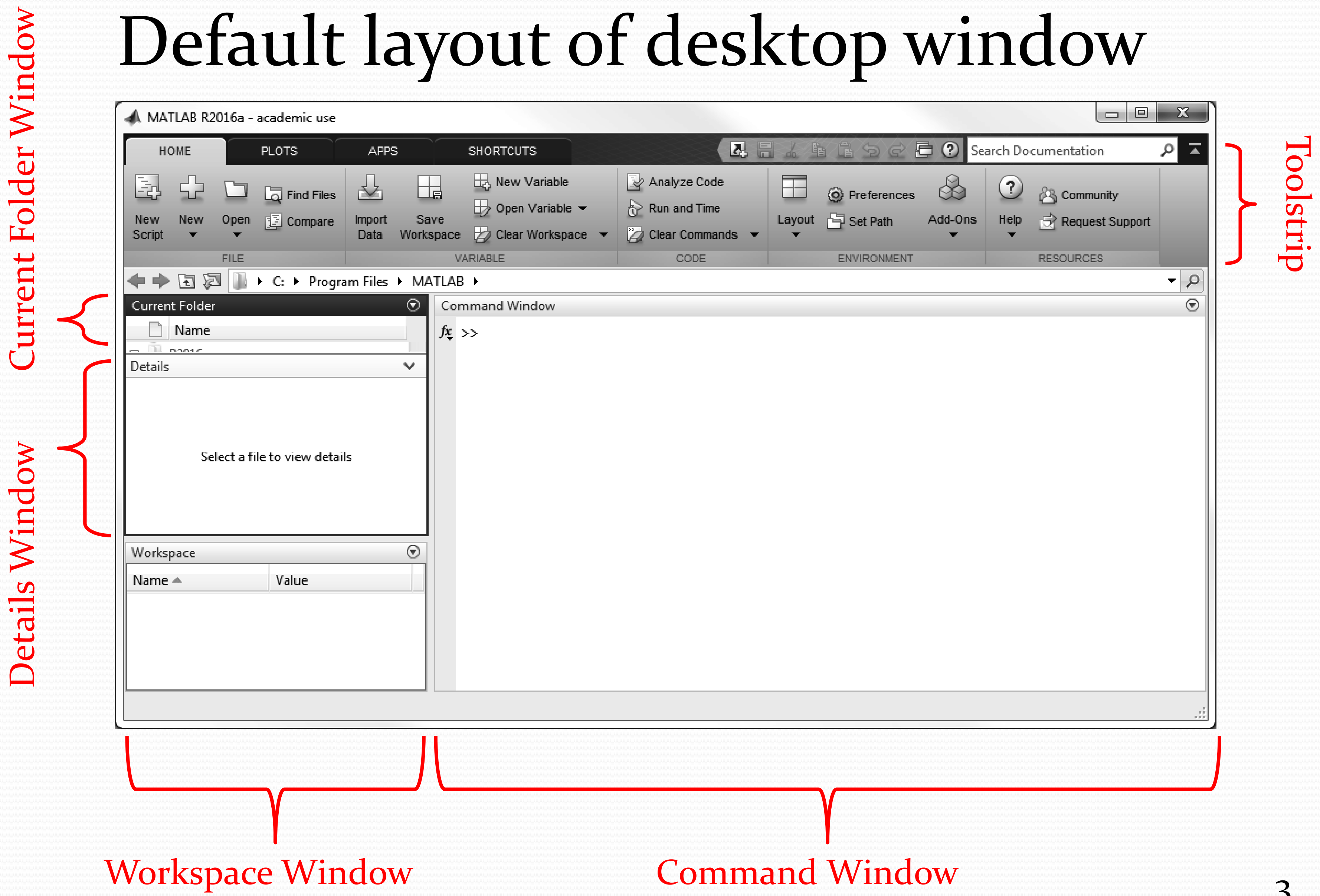
MATLAB An Introduction With Applications, 6th Edition
Dr. Amos Gilat
The Ohio State University

Slide deck by
Dr. Greg Reese
Miami University

This chapter describes:

- MATLAB windows
- Command Window in detail
- How to do basic arithmetic
- Create basic variables
- Introductory script files

Default layout of desktop window

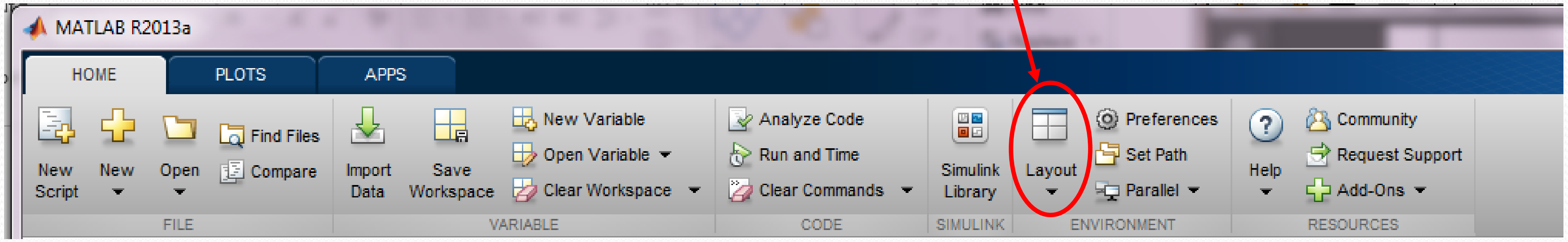


Window	Purpose
Command Window	Main window, enters variables, runs programs.
Figure Window	Contains output from graphic commands.
Editor Window	Creates and debugs script and function files.
Help Window	Provides help information.
Command History Window	Logs commands entered in the Command Window.
Workspace Window	Provides information about the variables that are stored.
Current Folder Window	Shows the files in the current folder.

Often easier to just show Command History Window. To close all other windows:

- Click on down-arrow button in top right of windows, select Close

  or
- From tool strip, select Layout, then Command Window Only



Icons

- **Window Action icon** – the icon showing a down arrow with a circle around it, i.e., It is in the upper, right corner of most MATLAB windows
- **Help icon** – the question-mark icon ( ) in the Resources Section of the desktop toolbar
- **Layout icon** – ( ) in the Environment Section of the desktop toolbar

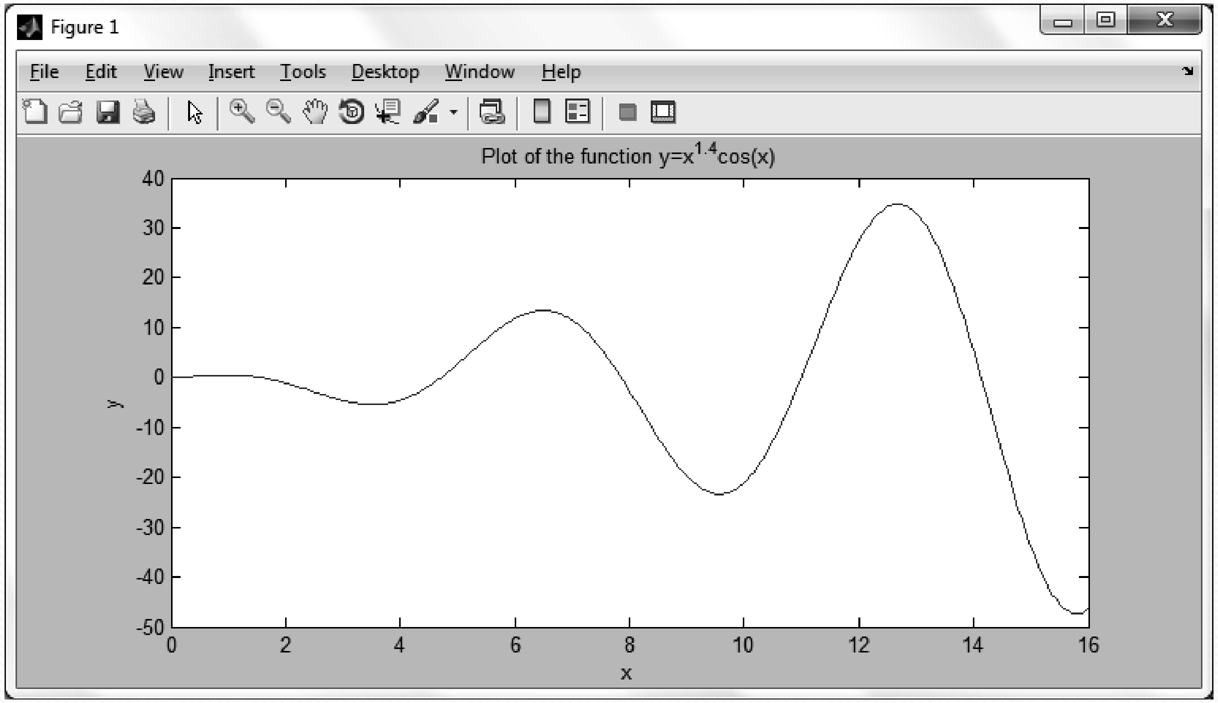
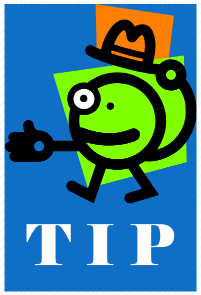
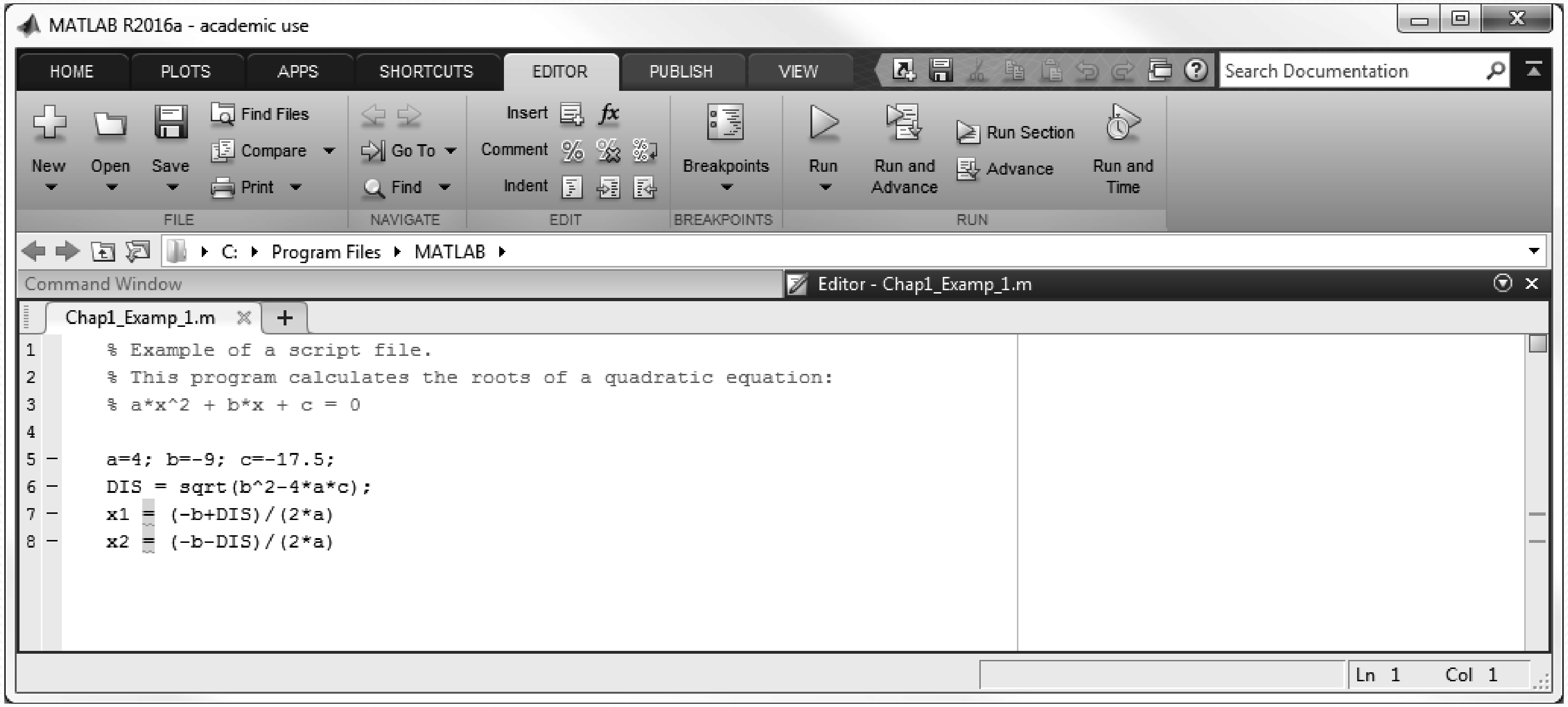


Figure 1
File Edit View Insert Tools Desktop Window Help
Plot of the function $y=x^{1.4}\cos(x)$
40
30
20
10
0
-10
-20
-30
-40
-50
y
0
2
4
6
8
10
12
14
16
x

Figure Window opens automatically after any command that draws a graph



If you don't see a figure window open up, look on the task bar for a black, program bar and click it

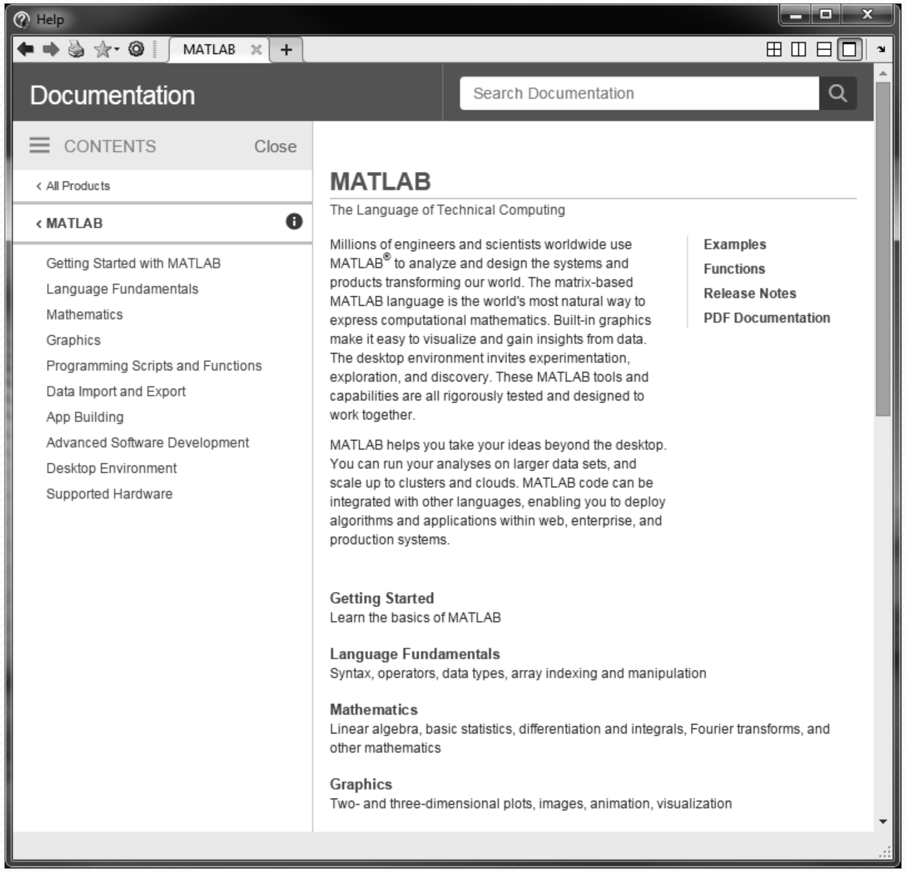


```
% Example of a script file.
% This program calculates the roots of a quadratic equation:
% a*x^2 + b*x + c = 0

a=4; b=-9; c=-17.5;
DIS = sqrt(b^2-4*a*c);
x1 = (-b+DIS)/(2*a)
x2 = (-b-DIS)/(2*a)
```

Use Editor Window to write and debug MATLAB scripts. Open with `edit` command

Get Help Window by clicking on Help icon (question mark) in tool strip


Help

MATLAB

Documentation

Search Documentation

CONTENTS Close

‹ All Products

‹ MATLAB

Getting Started with MATLAB
Language Fundamentals
Mathematics
Graphics
Programming Scripts and Functions
Data Import and Export
App Building
Advanced Software Development
Desktop Environment
Supported Hardware

MATLAB

The Language of Technical Computing

Millions of engineers and scientists worldwide use MATLAB® to analyze and design the systems and products transforming our world. The matrix-based MATLAB language is the world's most natural way to express computational mathematics. Built-in graphics make it easy to visualize and gain insights from data. The desktop environment invites experimentation, exploration, and discovery. These MATLAB tools and capabilities are all rigorously tested and designed to work together.

MATLAB helps you take your ideas beyond the desktop. You can run your analyses on larger data sets, and scale up to clusters and clouds. MATLAB code can be integrated with other languages, enabling you to deploy algorithms and applications within web, enterprise, and production systems.

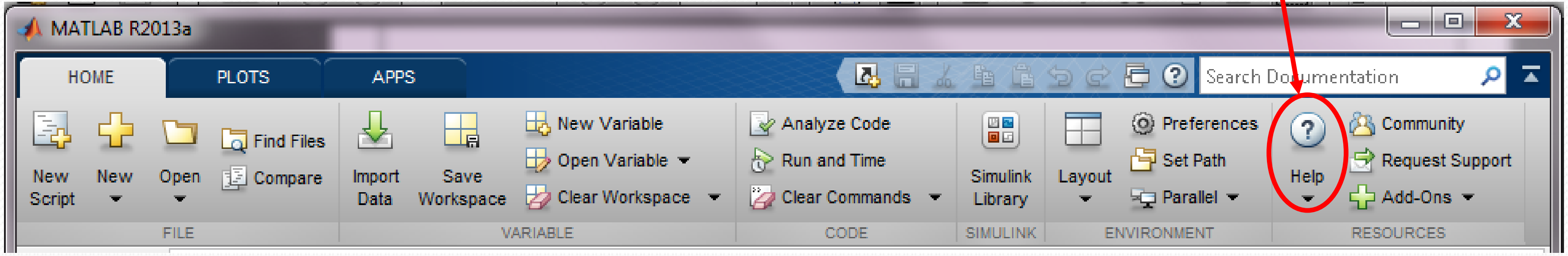
Examples
Functions
Release Notes
PDF Documentation

Getting Started
Learn the basics of MATLAB

Language Fundamentals
Syntax, operators, data types, array indexing and manipulation

Mathematics
Linear algebra, basic statistics, differentiation and integrals, Fourier transforms, and other mathematics

Graphics
Two- and three-dimensional plots, images, animation, visualization



More Fun with Windows

- To reopen a window, click the Layout icon and then click on the window name
- To get the default window layout (shown before) click the Layout icon, then click Default

More Fun with Windows

Undocking a window means removing it from the main MATLAB window and then being able to move it independently. To undock a window:

- Drag the window's title bar until the cursor is outside the MATLAB window, then release the cursor
  or
- Click on the Window Action icon, then click on Undock

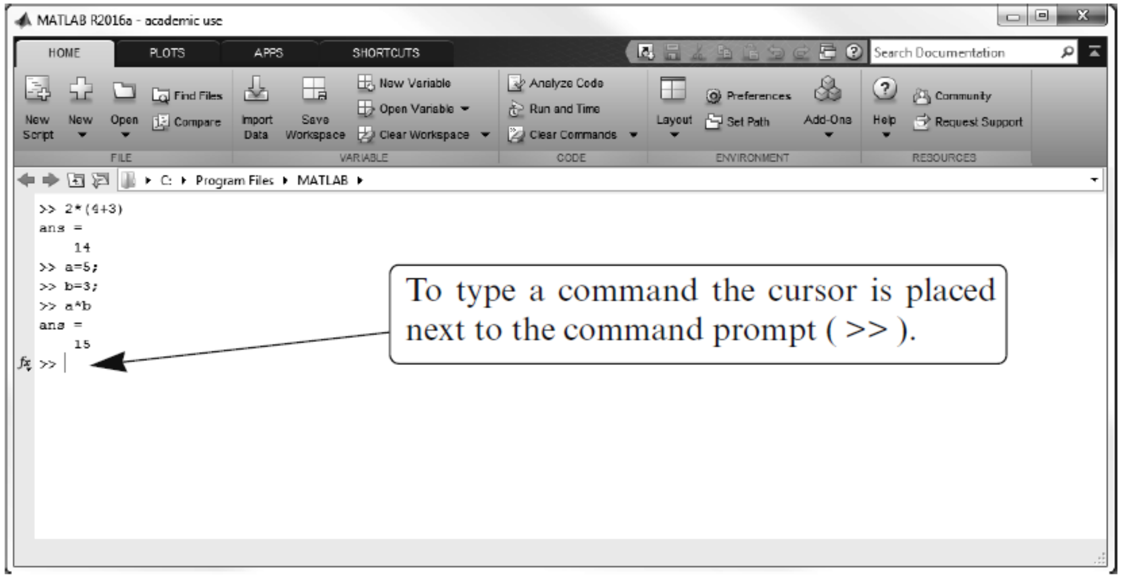
More Fun with Windows

To dock a window:

- Click on the Window Action icon, then click on Dock

Command Window is MATLAB's main window. Use it to:

- Execute commands
- Open other windows
- Run programs that you've written
- Manage the MATLAB software



```
>> 2*(4+3)
ans =
     14
>> a=5;
>> b=3;
>> a*b
ans =
     15
>> |
```

To type a command the cursor is placed next to the command prompt (>>).

Basic procedure

1. At prompt (>>), type in MATLAB command
2. Press ENTER key
3. MATLAB displays result in Command Window, followed by a prompt
4. Repeat from step 1

Notes on Command Window

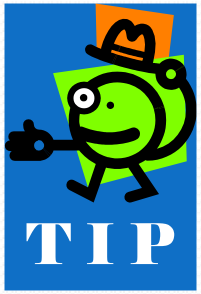
- To start a command, make sure cursor is next to prompt
- MATLAB won't respond until you press ENTER
  - It then executes only last command
  - Commands before last one may still be visible, but MATLAB doesn't execute them

- Can type several commands in same line by putting a comma between commands
  - Hard to read, so don't do this often
- If command too long to fit on line, can continue to next line by typing ellipsis (3 periods, i.e., ...) and then pressing ENTER

When cursor is in bottom command line:

- ← key moves cursor one character to left
- → key moves cursor one character to right
- ↑ key recalls preceding command
- ↓ key recalls command that follows one being displayed, i.e., undoes ↑

- PAGE-UP key moves up to previous commands in a window-size at a time
- PAGE-DOWN key moves down to previous commands in a window-size at a time
- BACKSPACE key deletes character to left of cursor
- DELETE key deletes character to right of cursor



To quickly execute a previous command but with minor changes

1. Recall command with up- and down-arrow keys
2. Use left- and right-arrow keys to move to characters to be altered
3. Use BACKSPACE or DELETE to remove old character, then type new character
4. Press ENTER to execute modified command

Semicolon (;)

- When typed at end of command, suppresses output. (Only prompt displayed at next line)
  - Useful for preventing display of large outputs
  - Used much more in scripts (see Section 1.8)

Percent sign(%)

- When typed at beginning of line, MATLAB treats line as a *comment* and doesn't execute line
  - Used much more in scripts (see Section 1.8)

`clc` command

- Clears Command Window display
- Up and down arrows still bring back previous commands

Command History Window

- Shows previous commands, including ones from previous MATLAB sessions
- Double-clicking on command puts it in Command Window and executes it
- Can drag command to Command Window, make changes in command, then execute it
- To clear one or more commands, select the lines to delete, right click, choose Delete Selection
- To clear entire history, right click, select Clear Command History

In this chapter will only discuss arithmetic with *scalars* (single numbers)

- Can do arithmetic directly on numbers (like a calculator)
- Can store numbers in variables

Symbols for arithmetic are:

Operation	Symbol	Example
Addition	+	5 + 3
Subtraction	–	5 – 3
Multiplication	*	5 * 3
Right division	/	5 / 3
Left division	\	5 \ 3 = 3 / 5
Exponentiation	^	5 ^ 3 (means 5^3 = 125)

Left division rarely used with scalars

Order in which MATLAB does arithmetic

Precedence	Mathematical Operation
First	Parentheses. For nested parentheses, the innermost are executed first.
Second	Exponentiation.
Third	Multiplication, division (equal precedence).
Fourth	Addition and subtraction.

Precedence order

- Same as most calculators
- Same as doing arithmetic by hand
- For multiple operations of same precedence, MATLAB goes left to right
- Can change order by using parentheses

Can use MATLAB as a (very expensive!) calculator

1. Type in mathematical expression
2. Press **Enter** key
3. MATLAB displays answer in Command Window as `ans =` followed by the result

Your display may appear on more than one line and have blank lines between text

Can control display of numbers with `format` command

- Once enter command, format stays the same until another `format` command
- Default format is fixed point with four digits to right of decimal point
    - *fixed-point* means decimal point always between one's-digit and one-tenth's digit
- Format only affects display of numbers. MATLAB always computes and saves numbers in full precision

Some types of formatting

Command	Description	Example

`format short`	Fixed-point with 4 decimal digits for:
	$0.001 \le number \le 1000$
	Otherwise display format `short e`.
		`>> 290/7`
		`ans =`
		`41.4286`

`format long`	Fixed-point with 15 decimal digits for:
	$0.001 \le number \le 100$
	Otherwise display format `long e`.
		`>> 290/7`
		`ans =`
		`41.428571428571431`

`format short e`	Scientific notation with 4 decimal digits.	`>> 290/7`
		`ans =`
		`4.1429e+001`

`format long e`	Scientific notation with 15 decimal digits.	`>> 290/7`
		`ans =`
		
		`4.142857142857143e+0`
		`01`

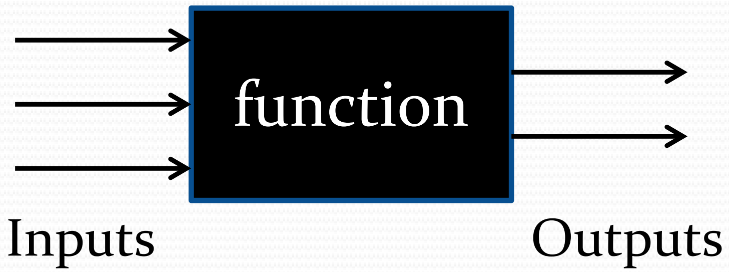
`format short g`	Best of 5-digit fixed or floating point.	`>> 290/7`
		`ans =`
		`41.429`

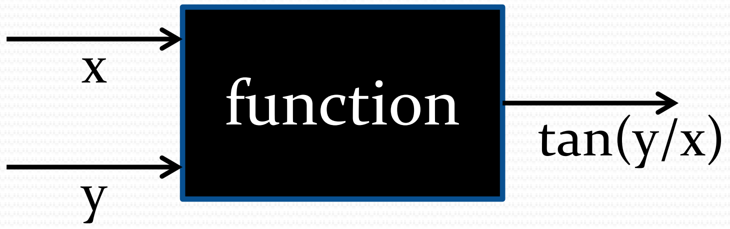
`format long g`	Best of 15-digit fixed or floating point.	`>> 290/7`
		`ans =`
		
		`41.4285714285714`

`format bank`	Two decimal digits.	`>> 290/7`
		`ans =`
		`41.43`

`format compact`	Eliminates blank lines to allow more lines with information displayed on the screen.	
`format loose`	Adds blank lines (opposite of compact).	

MATLAB expressions can include functions. You can think of a *function* as a black box that, in general, takes inputs, does some computations with them, and produces outputs.





A function

- Has a name
- Can have zero or more *arguments* (inputs)
- Can produce zero or more outputs

```
y = sqrt( x )
```

output name argument

A function’s arguments can be

- Numbers
- Variables (explained in next section 1.6)
- Expressions involving numbers, variables, or functions

`sqrt(64)` Argument is a number

`sqrt(a)` Argument is the variable “a”

```
atan( y/sqrt(3^2+y^2) )
```

Argument to arctan function is an expression that has a number (`3`), a variable (`y`), and a function (`sqrt`)

Elementary math functions

- `sqrt(x)` – square root
- `nthroot(x,n)` – nth real root
- `exp(x)` – e^x
- `abs(x)` – absolute value
- `log(x)` – natural log (base *e*)
- `log10(x)` – log base 10
- `factorial(x)` – *x!*

See Table 1-3 in book for details

Trigonometric functions

- `sin(x)` – sine (`x` in radians)
- `sind(x)` – sine (`x` in degrees)
- `cos(x)` – cosine (`x` in radians)
- `cosd(x)` – cosine (`x` in degrees)
- `tan(x)` – tangent (`x` in radians)
- `tand(x)` – tangent (`x` in degrees)
- `cot(x)` – cotangent (`x` in radians)
- `cotd(x)` – cotangent (`x` in degrees)

See Table 1-4 in book for details

Inverse trigonometric functions

- `asin(x), acos(x), atan(x), acot(x)`
  (x in radians)
- `asind(x), acosd(x), atand(x), acotd(x)` (x in degrees)

Hyperbolic trigonometric functions

- `cosh(x)` – $(e^x + e^{-x})/2$
- `sinh(x)` – $(e^x - e^{-x})/2$
- `tanh(x)` – $(e^x - e^{-x})/(e^x + e^{-x})$
- `coth(x)` – $(e^x + e^{-x})/(e^x - e^{-x})$

Rounding functions

- `round(x)` – round to nearest integer
- `fix(x)` – round toward zero
- `ceil(x)` – round toward infinity
- `floor(x)` – round toward minus infinity
- `rem(x,y)` – remainder after `x` is divided by `y` (also called *modulus*)
- `sign(x)` – returns 1 if `x` is positive, -1 if `x` is negative, zero if `x` is zero

See Table 1-5 in book for details

A *variable* is a name that is assigned a numerical value

- Once assigned, can use variable in expressions, functions, and MATLAB statements and commands
- Can *read* the variable (get its value)
- Can *write to* the variable (set its value)

= (equals sign) is MATLAB's *assignment operator*. It evaluates the expression on its right side and stores the resulting value in the variable on its left side

```
>> a = 3
```
Create the variable called "a" and store the value 3 in it

```
a =

     3
```
MATLAB acknowledges that it has created "a" and set it to 3

EXAMPLE

```
>> a = 3
```
Make a variable and store a number in it

```
a =

     3

>> b = 10*a + 5
```
Make a variable and store the value of an expression made up of a variable, numbers, and addition and multiplication

```
b =

     35
```

Think of = as meaning "assign to" or "store in" but <u>not</u> meaning "equals"!

Why?

$x = x + 6$ has no meaning in math because it implies that $0 = 6$

`x = x + 6` is perfectly fine in MATLAB because it means "take whatever is in `x`, add `6` to that and store the result back into `x`"

EXAMPLE

```
>> x = 3;
```

; at end prevents MATLAB from displaying value of x

```
>> x = x + 6
```

takes what's in x (3), adds 6 to it to get 9, then stores 9 back into x

```
x =

     9
```

now x's value is 9

```
>> x = 2 * x
```

takes what's in x (9), multiplies it by 2 to get 18, then stores 18 back into x

```
x =

    18
```

now x's value is 18

A variable must have a value before you use it in an expression

```
>> x = 3;
>> x+2
ans =
     5
>> x + y % assume y undefined
??? Undefined function or
variable 'y'
```

To find out the value of a variable, just type it and press ENTER

```
>> x = 3;
>> y = 10 * x;
>> z = y ^ 2;
>> y
y =
      30
>> z
z =
      900
```

Can do multiple assignments on one line by separating with a comma or semicolon. If semicolon, no display for that assignment

```
>> a=12, B=4; C=(a-B)+40-a/B*10
a =
     12
C =
     18
```

To change the value of a variable, just assign it the new value

```
>> ABB=72;
>> ABB=9;
>> ABB
ABB =
     9
```

You must define a variable (give it a value) before you can use it in an argument of a function

```
>> sqrt( x ) % assume x undefined
??? Undefined function or
variable 'x'
>> x = 144;
>> sqrt( x )
x =
    12
```

A variable name

- Must begin with a letter
- Can be up to 63 characters long
- Can contain letters, digits, and underscores (_)
- Can't contain punctuation, e.g., period, comma, semicolon

Avoid using the name of a built-in function as the name of a variable, e.g., don't call a variable `exp` or `sqrt`

MATLAB is *case-sensitive*, and does not consider an upper-case letter in a variable name to be the same as its lower-case counterpart, e.g., `MTV`, `MTv`, `mTV`, and `mtv` are four different variable names

A variable name cannot contain a space. Two common alternatives:

1. Use an underscore in place of a space, e.g., `speed_of_light`
2. Capitalize the first letter of all but first word, e.g., `speedOfLight` (This is known as *camel case*!)

A *keyword* is a word that has special meaning to MATLAB

- There are 20 keywords
  - `break case catch classdef continue else elseif end for function global if otherwise parfor persistent return spmd switch try while`
- Appear in blue when typed in the Editor Window
- Can't be used as variable names

MATLAB has pre-defined variables for some common quantities

`pi`	the number π
`eps`	the smallest difference between any two numbers in MATLAB
`inf` or `Inf`	infinity
`i`	$\sqrt{-1}$
`j`	$\sqrt{-1}$ (same as `i`) but commonly used instead of `i` in electrical engineering

More pre-defined variables

`ans`	the value of the last expression that was not assigned to a variable
`NaN` or `nan`	not-a-number. Used to express mathematically undefined values, such as `0/0`

You can redefine (change) the values of the predefined variables, but don't. You'll cause confusion

- Exceptions are `i` and `j`, which are often used as loop variables (see Section 6.4)

Some commands for managing variables

Command	**Outcome**
`clear`	Removes all variables from memory
`clear x y z`	Removes only variables `x`, `y`, and `z` from memory
`who`	Displays a list of the variables currently in memory
`whos`	Displays a list of the variables currently in memory and their size, together with information about their bytes and class (see Section 4.1)

So far, have run MATLAB commands by typing in single command, pressing ENTER, getting MATLAB's result, and then repeating this process for next command

- Not practical for calculations involving more than a few commands. Can use up and down arrow keys to avoid lots of typing, but still not practical

Better way

- Save all commands in a file
- With one command in Command Window, tell MATLAB to run all commands in file

Will use script files to do this

A *script file* is a sequence of MATLAB commands, also called a program

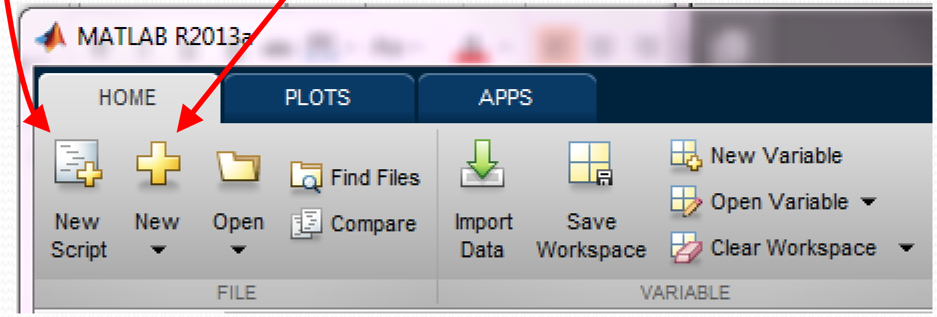
- When a script file runs (is *executed*), MATLAB performs the commands in the order they are written, just as if they were typed in the Command Window
- When a script file has a command that generates an output (e.g. assignment of a value to a variable without semicolon at the end), the file displays the output in the Command Window

- Using a script file is convenient because it can be edited (corrected and/or changed) and executed many times
- Script files can be typed and edited in any text editor and then pasted into the MATLAB editor
- Script files are also called *M-files* because the extension .m is used when they are saved

Use the Editor Window to work with script files

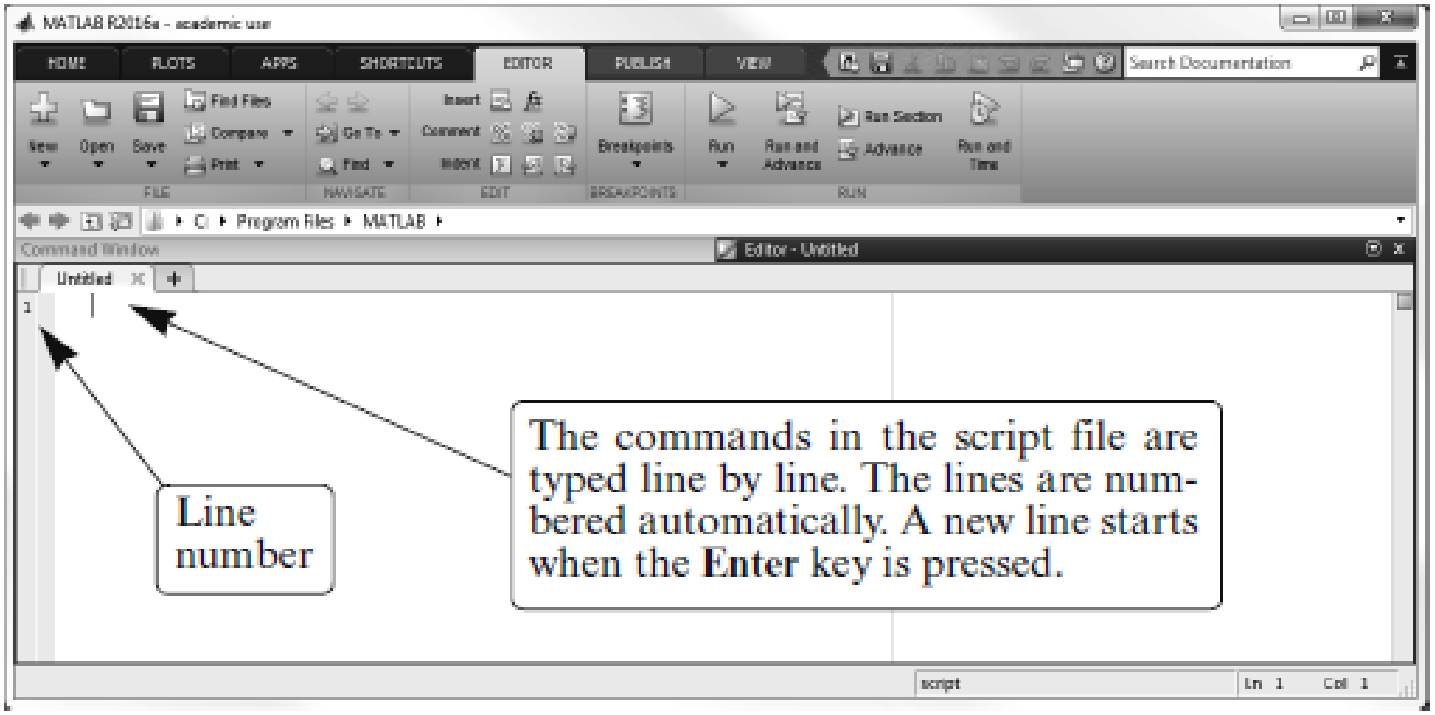
Can open window and create file two ways

1. Click on New Script icon
2. Click on New icon, select Script
3. In the Command Window, type `edit` and then press ENTER

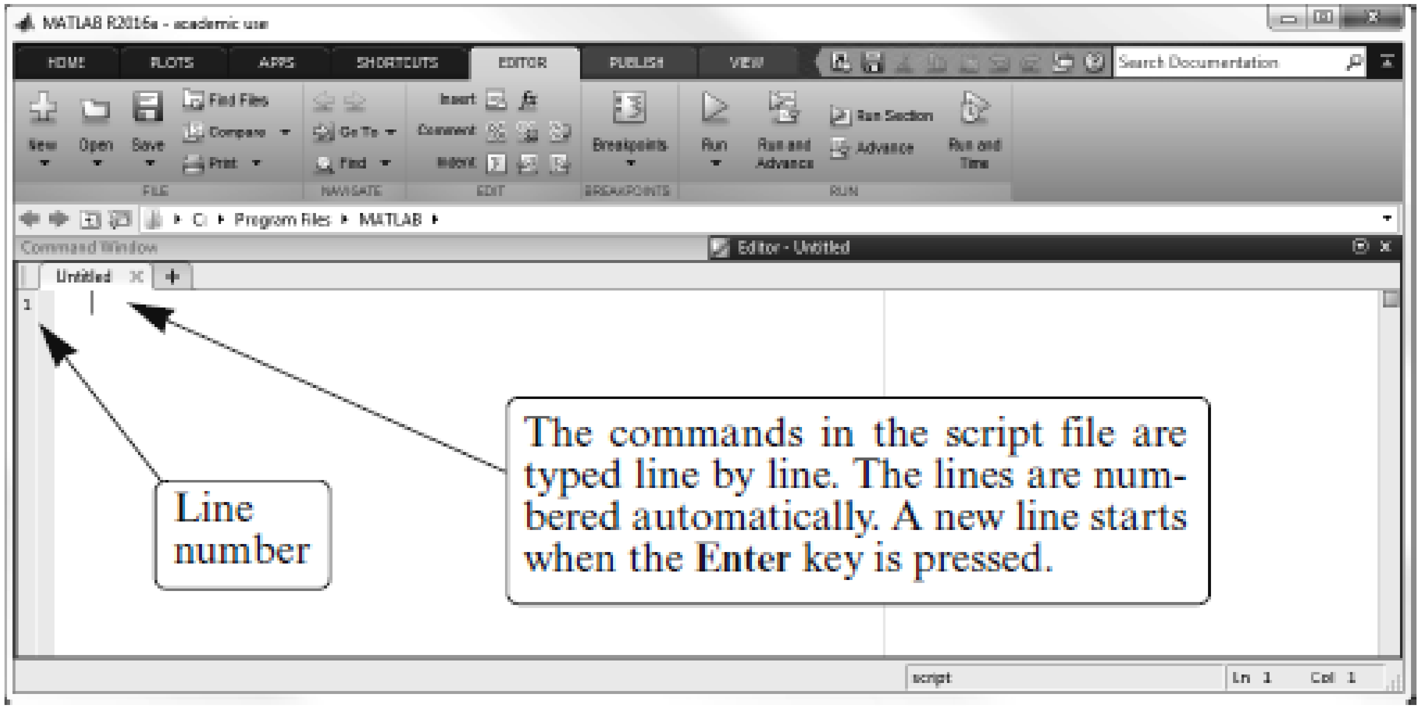


Editor has tool strip on top with three tabs – EDITOR, PUBLISH, VIEW

- MATLAB used most often with EDITOR tab selected

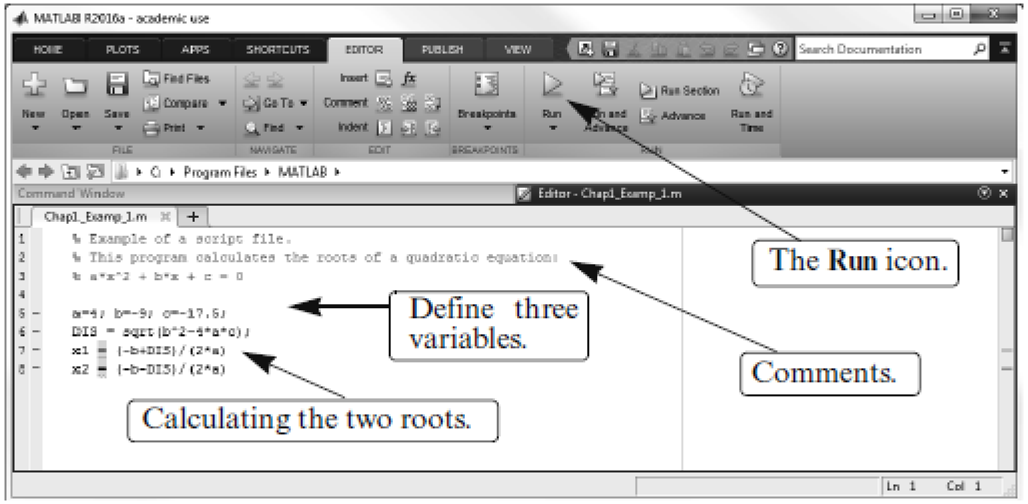


The commands in the script file are typed line by line. The lines are numbered automatically. A new line starts when the **Enter** key is pressed.



The commands in the script file are typed line by line. The lines are numbered automatically. A new line starts when the **Enter** key is pressed.

- Type in commands line by line, pressing ENTER after each one
- MATLAB automatically numbers lines



```
% Example of a script file.
% This program calculates the roots of a quadratic equation:
% a*x^2 + b*x + c = 0

a=4; b=-9; c=-17.5;
DIS = sqrt(b^2-4*a*c);
x1 = (-b+DIS)/(2*a)
x2 = (-b-DIS)/(2*a)
```

Comment lines

- Lines that start with percent sign (%)
- Common for first few lines to be comments and to briefly explain what commands in file do
- Editor Window shows comment lines in green

Before MATLAB can run commands in file, you must save file

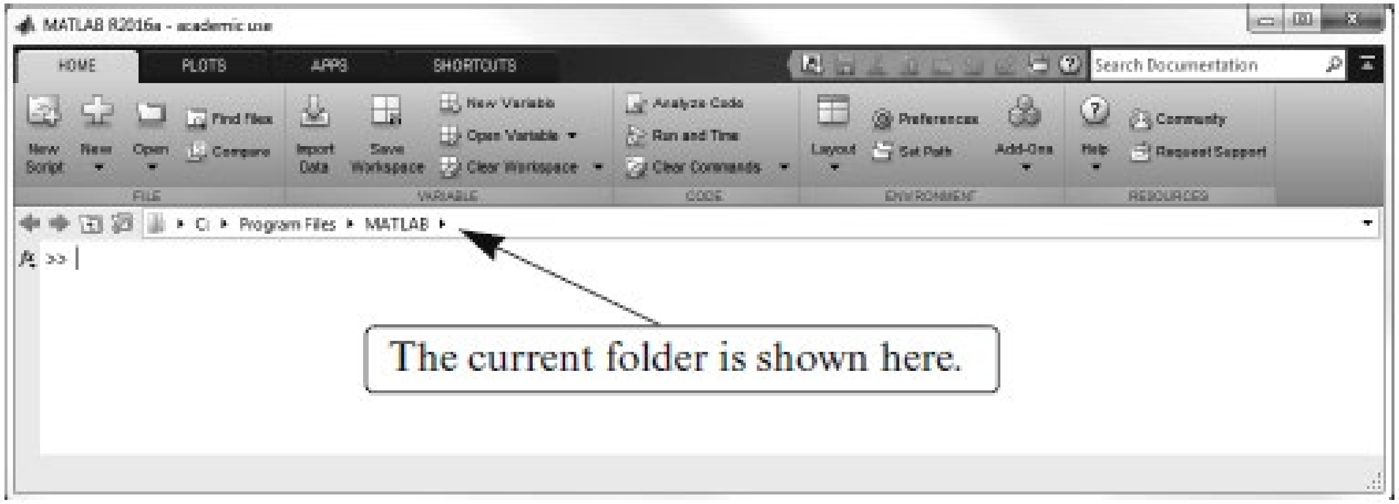
- If you haven't named file yet, click on Save icon, MATLAB brings up Save As dialog box
- If you've already named and saved file, just click on Save icon
- If you don't add an extension (.xxx) to the file name, MATLAB adds ".m"
- Rules for file names are same as rules for function names
- Don't use names of your variables, predefined variables, MATLAB commands, or MATLAB functions

To *execute* a script file means to run all of the commands in it. You can execute a file by

- Pressing the Run icon (a green arrow)
- Typing the file name in the Command Window and pressing ENTER

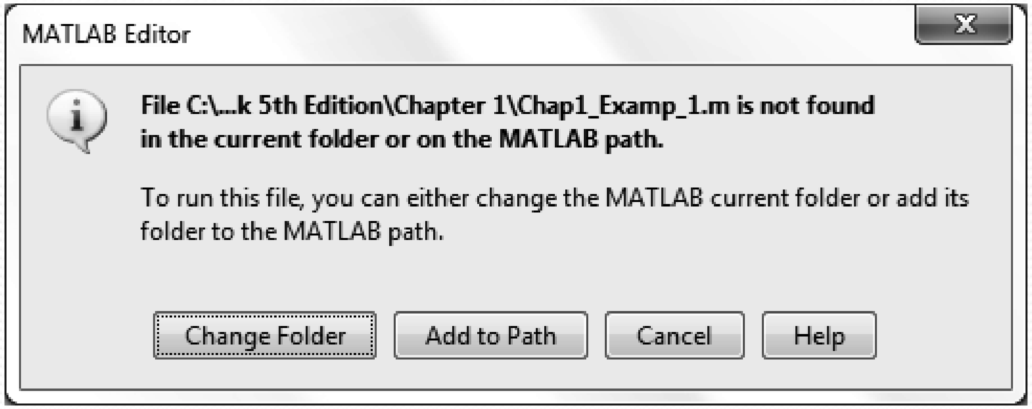
MATLAB will execute file if it is in MATLAB's current folder or if the file's folder is in the search path (explained next)

1.8.4 Current Folder



The *current folder* is the folder that MATLAB checks first when looking for your script file

- Can see current folder in desktop toolbar
- Can also display current folder by issuing MATLAB command `pwd`

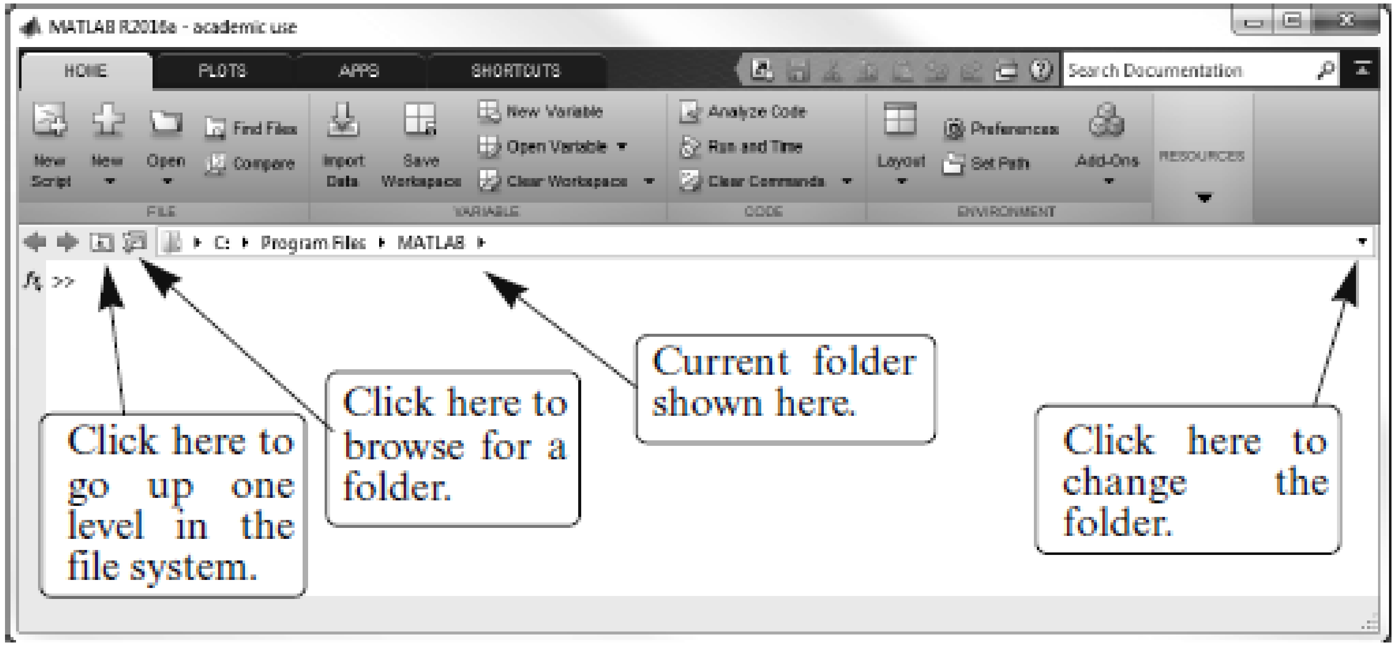

MATLAB Editor

File C:\...k 5th Edition\Chapter 1\Chap1_Examp_1.m is not found in the current folder or on the MATLAB path.

To run this file, you can either change the MATLAB current folder or add its folder to the MATLAB path.

If you try to run the script by clicking on the Run icon and the current folder is not the folder where the script file is saved, you'll get the prompt shown.

- Either change the current folder or add the script folder to the MATLAB path



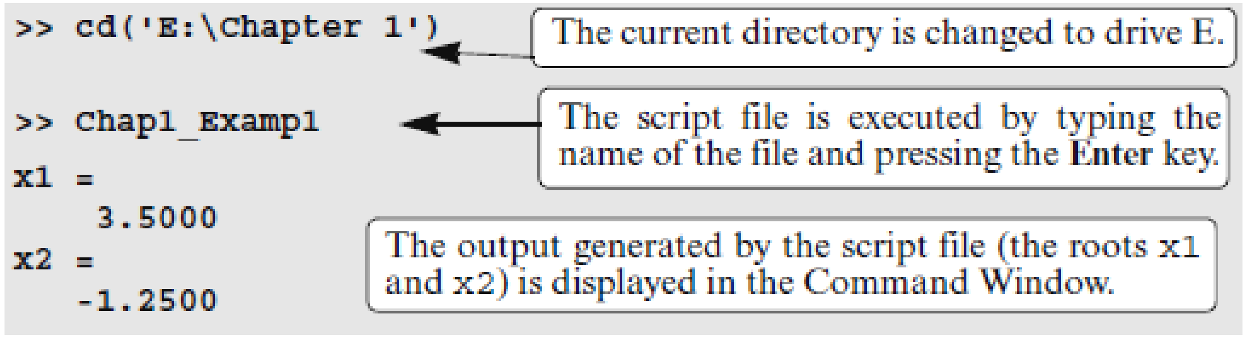
Can change current folder in Current Folder Window

- To show Current Folder Window, click on Layout icon in desktop, then select Current Folder

Can change current folder from command line using `cd` command, space, new folder name in single quote marks, ENTER, i.e.,

```
>> cd 'new folder'
```

For example,



```
>> cd('E:\Chapter 1')

>> Chap1_Examp1
x1 =
    3.5000
x2 =
   -1.2500
```

The current directory is changed to drive E.

The script file is executed by typing the name of the file and pressing the **Enter** key.

The output generated by the script file (the roots x1 and x2) is displayed in the Command Window.