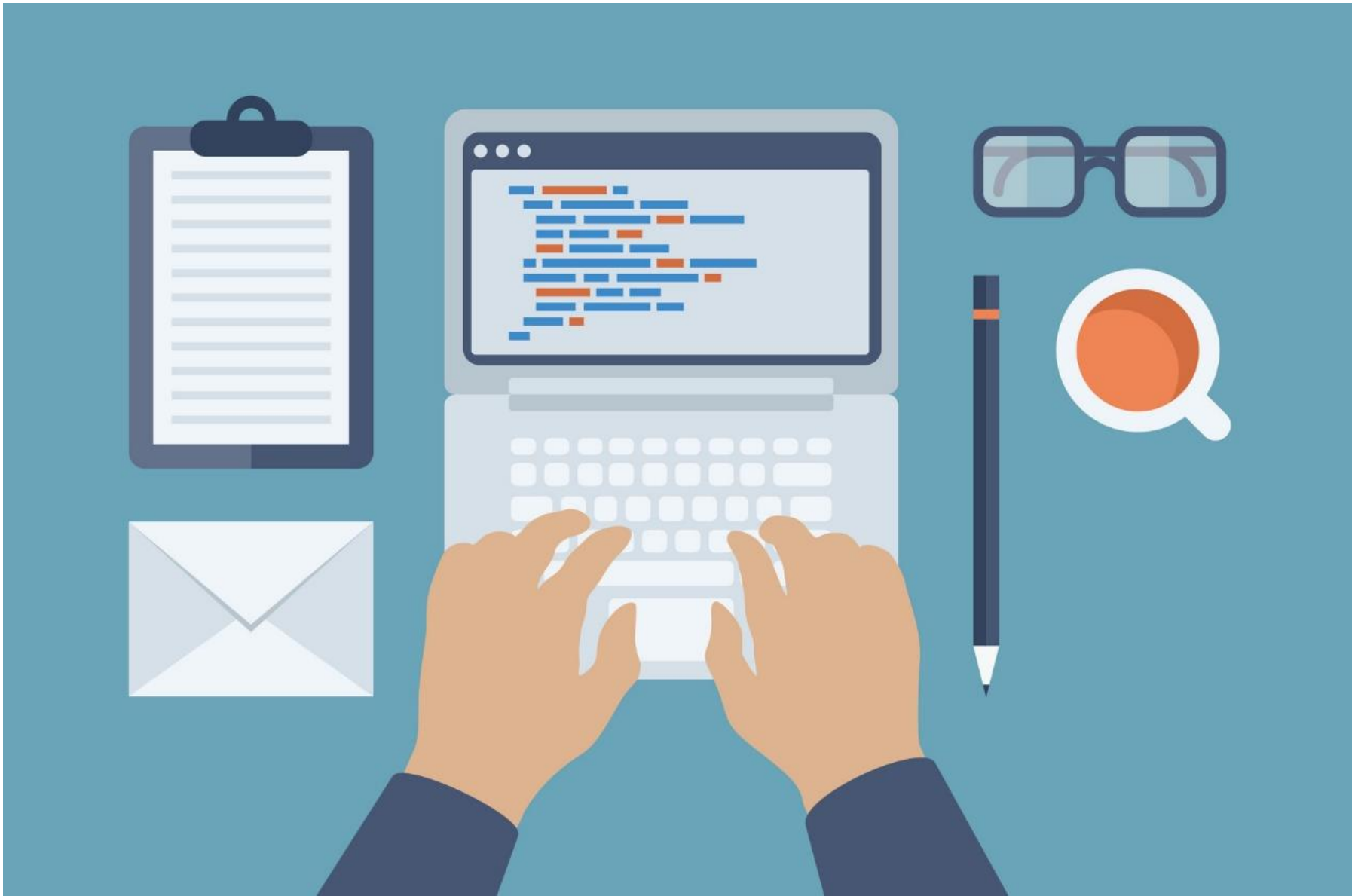
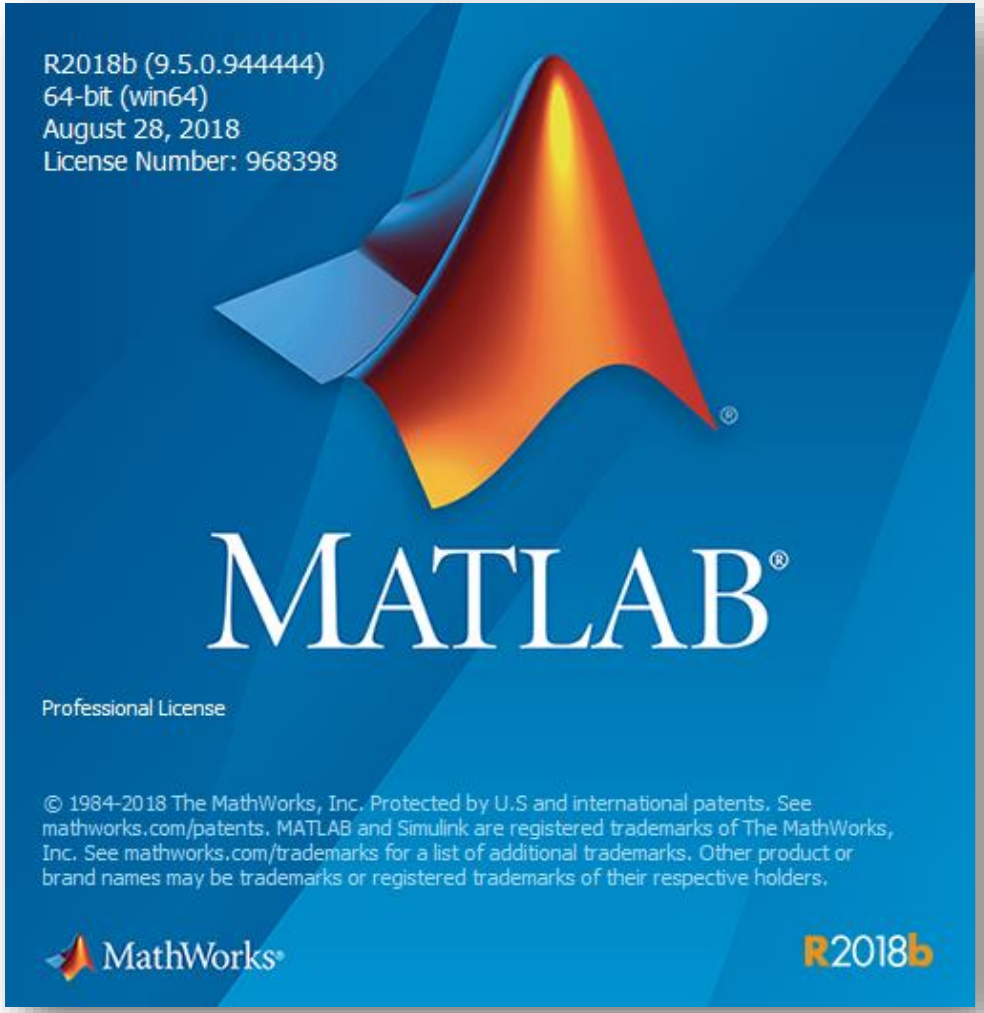
Computer Programming



Programming Languages



TIOBE Index

TIOBE Index for October 2018

The Importance Of Being Earnest Programming Community Index (TIOBE)

Oct 2018	Oct 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.801%	+5.37%
2	2		C	15.376%	+7.00%
3	3		C++	7.593%	+2.59%
4	5	↑	Python	7.156%	+3.35%
5	8	↑	Visual Basic .NET	5.884%	+3.15%
6	4	↓	C#	3.485%	-0.37%
7	7		PHP	2.794%	+0.00%
8	6	↓	JavaScript	2.280%	-0.73%
9	-	⇈	SQL	2.038%	+2.04%
10	16	⇈	Swift	1.500%	-0.17%
11	13	↑	MATLAB	1.317%	-0.56%
12	20	⇈	Go	1.253%	-0.10%
13	9	⇊	Assembly language	1.245%	-1.13%
14	15	↑	R	1.214%	-0.47%
15	17	↑	Objective-C	1.202%	-0.31%
16	12	⇊	Perl	1.168%	-0.80%
17	11	⇊	Delphi/Object Pascal	1.154%	-1.03%
18	10	⇊	Ruby	1.108%	-1.22%
19	19		PL/SQL	0.779%	-0.63%
20	18	↓	Visual Basic	0.652%	-0.77%

Reference: https://www.tiobe.com/tiobe-index/

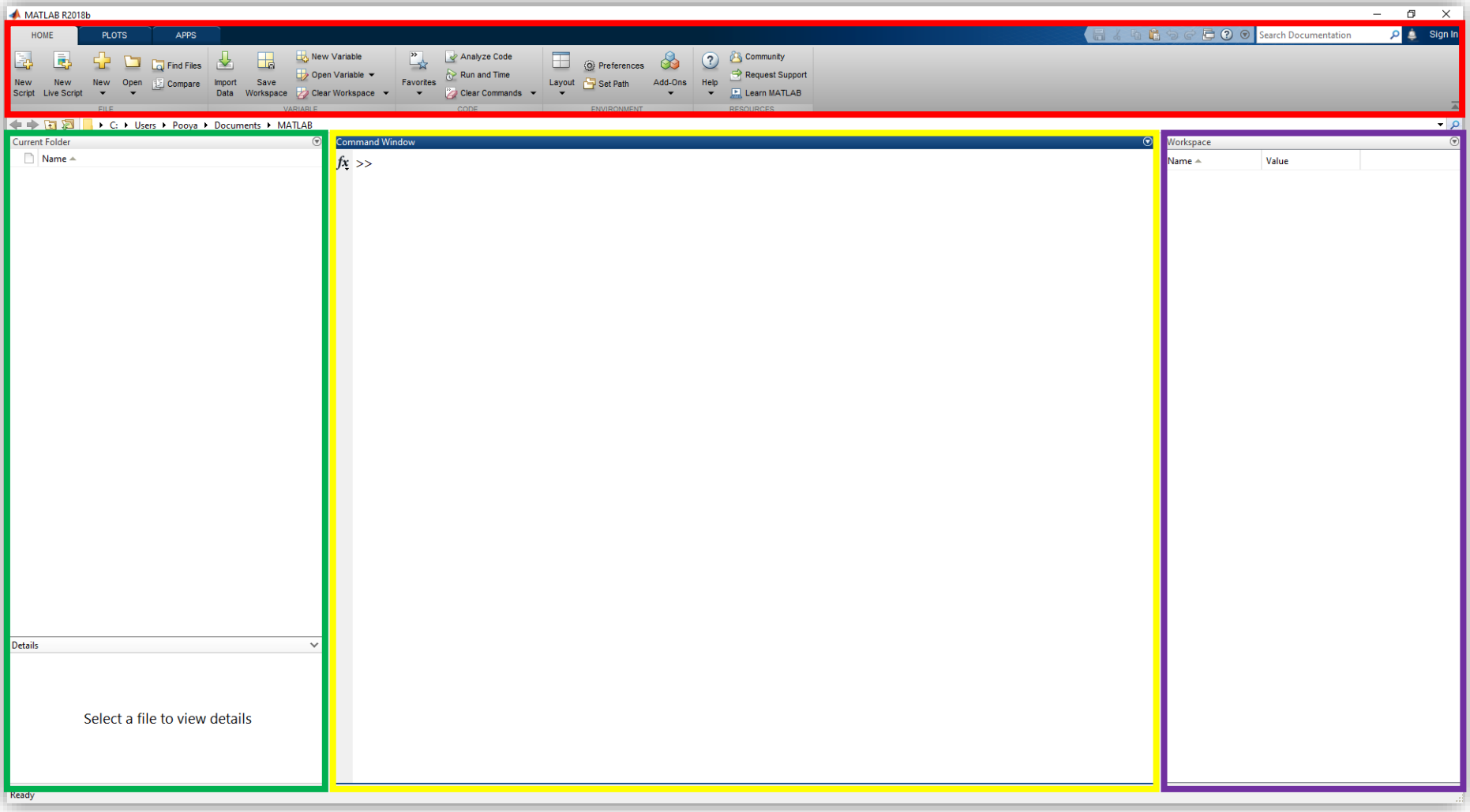
Contents

Chapter 01:

Starting with MATLAB

Chapter 01

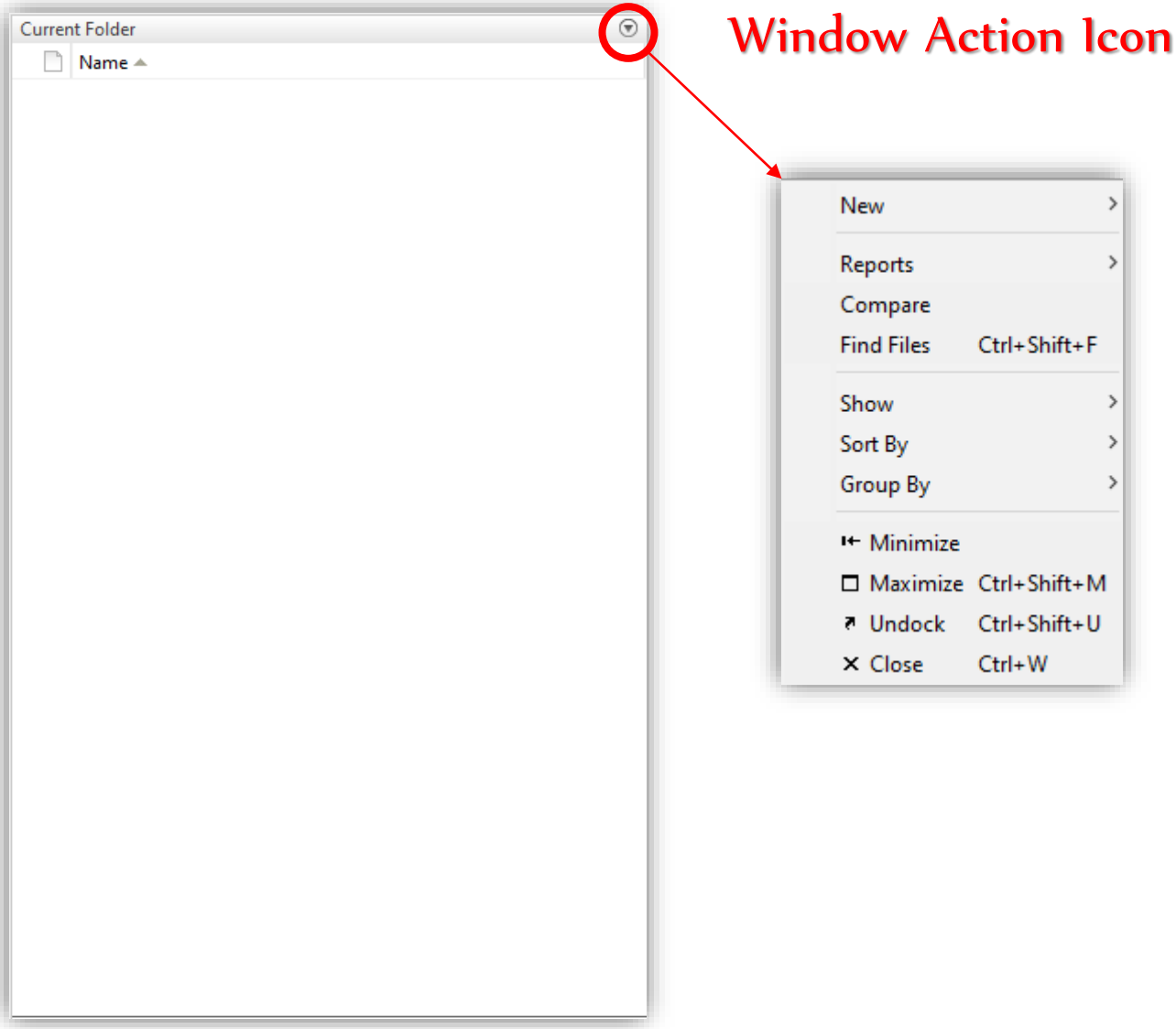
Starting MATLAB, MATLAB Windows



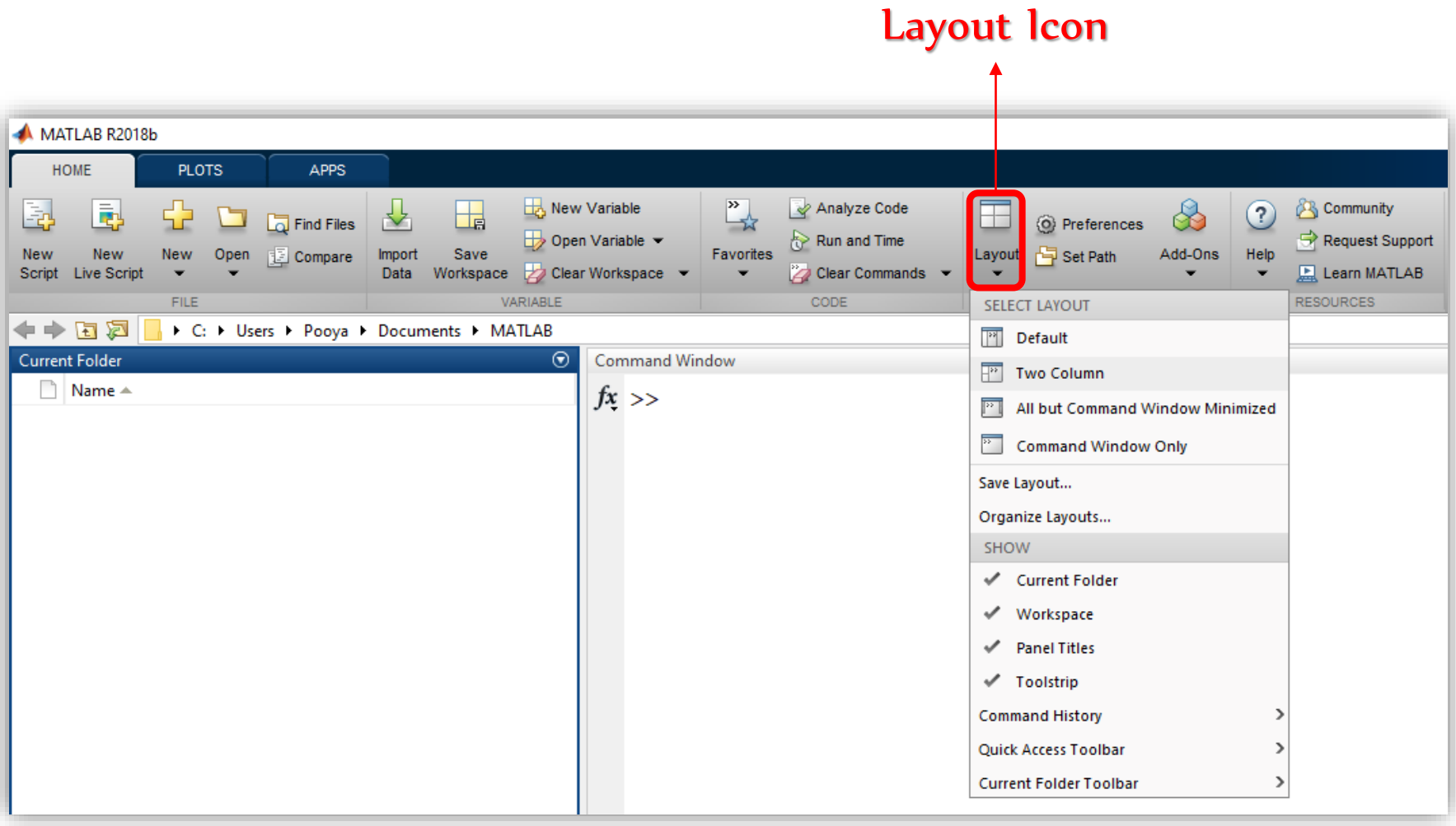
Starting MATLAB, MATLAB Windows

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

Starting MATLAB, MATLAB Windows

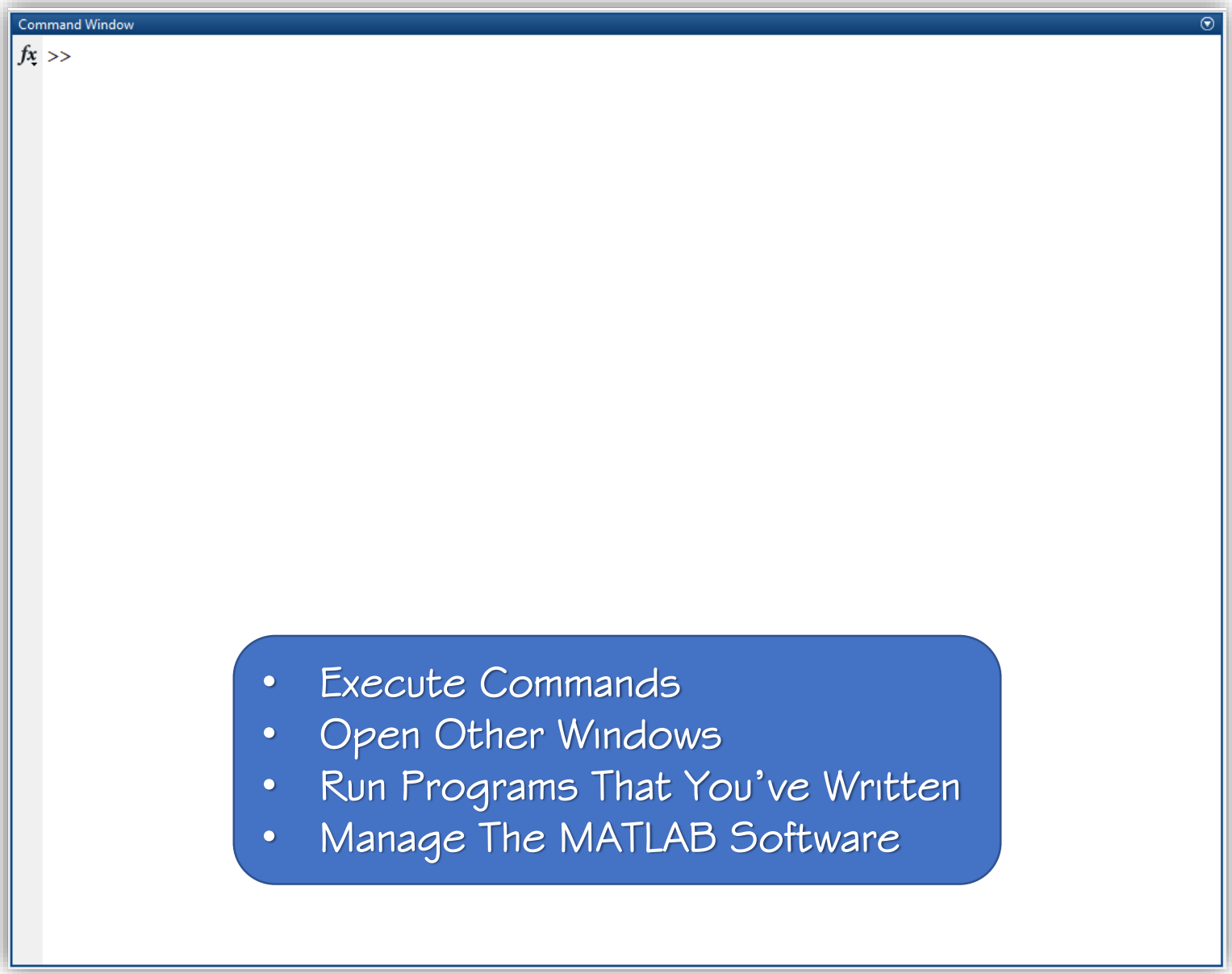


Starting MATLAB, MATLAB Windows

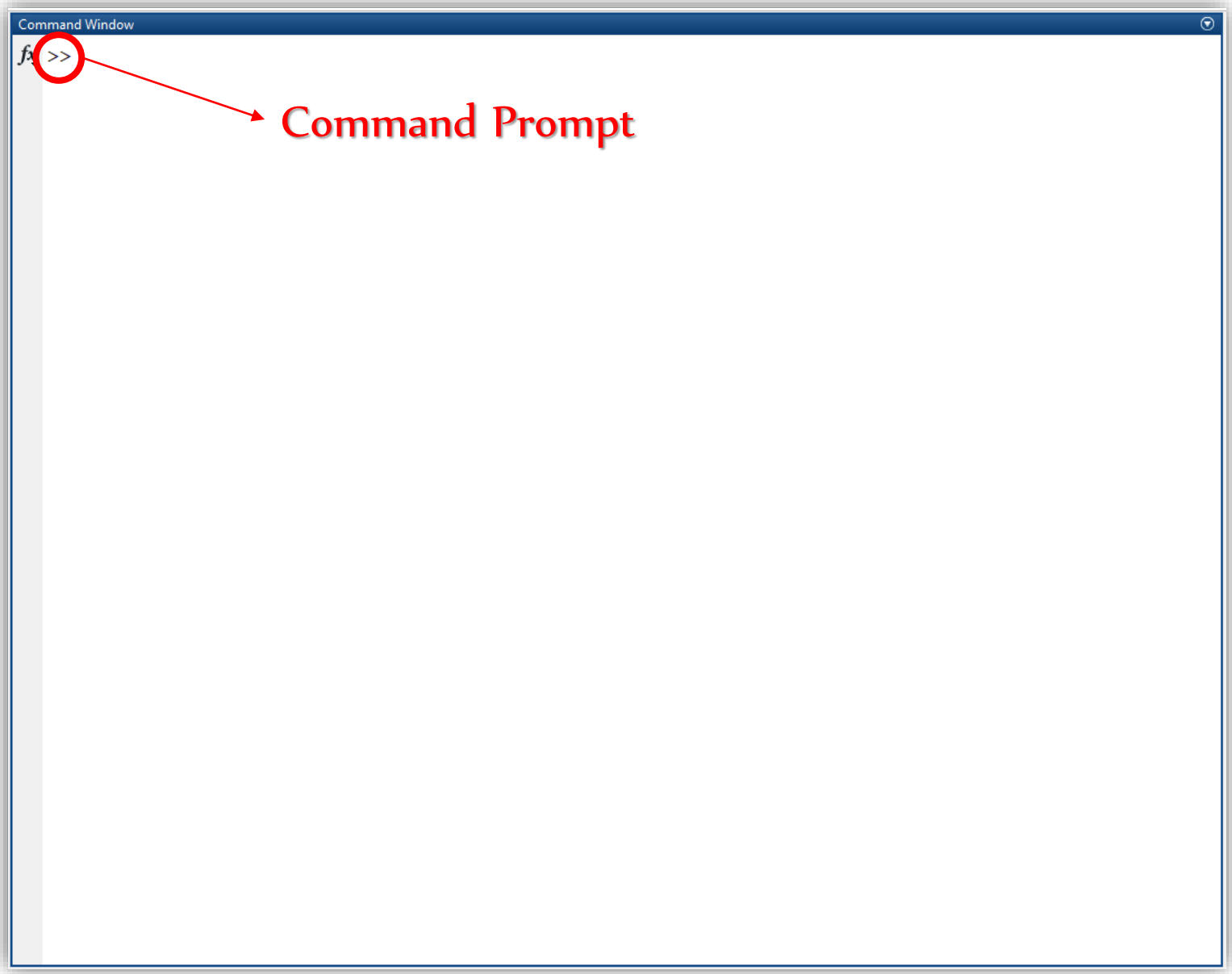


Chapter 01

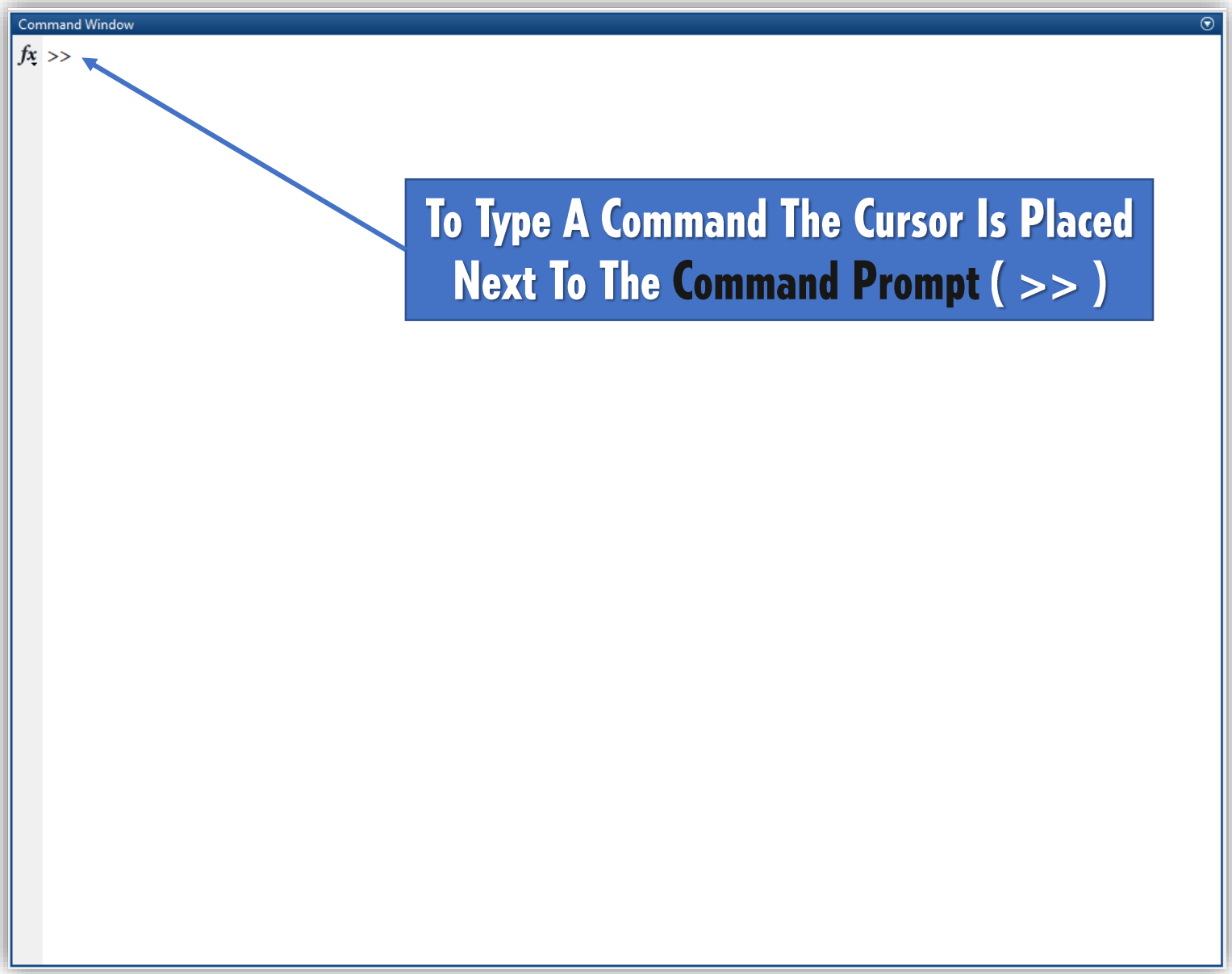
Working In The Command Window



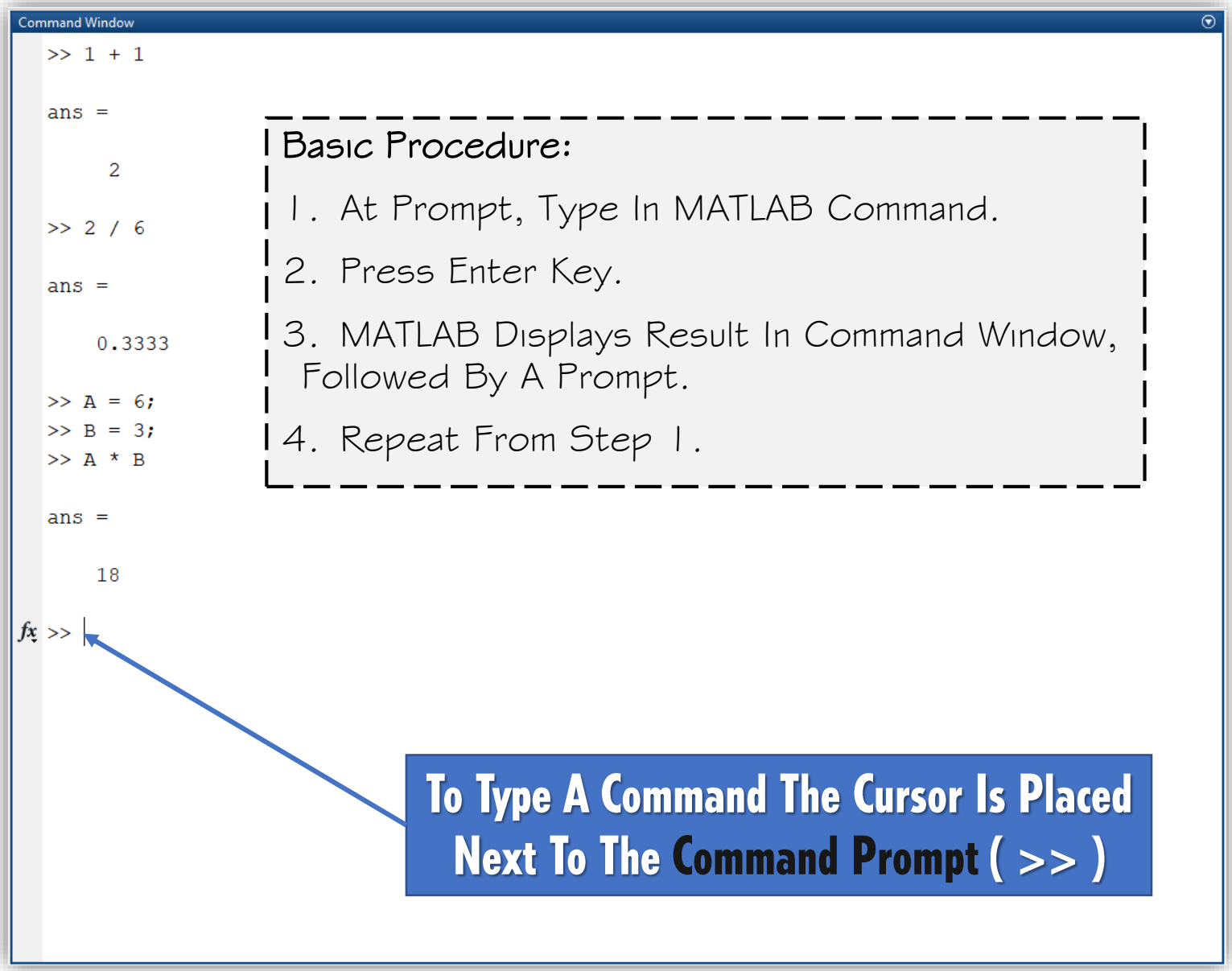
Working In The Command Window



Working In The Command Window

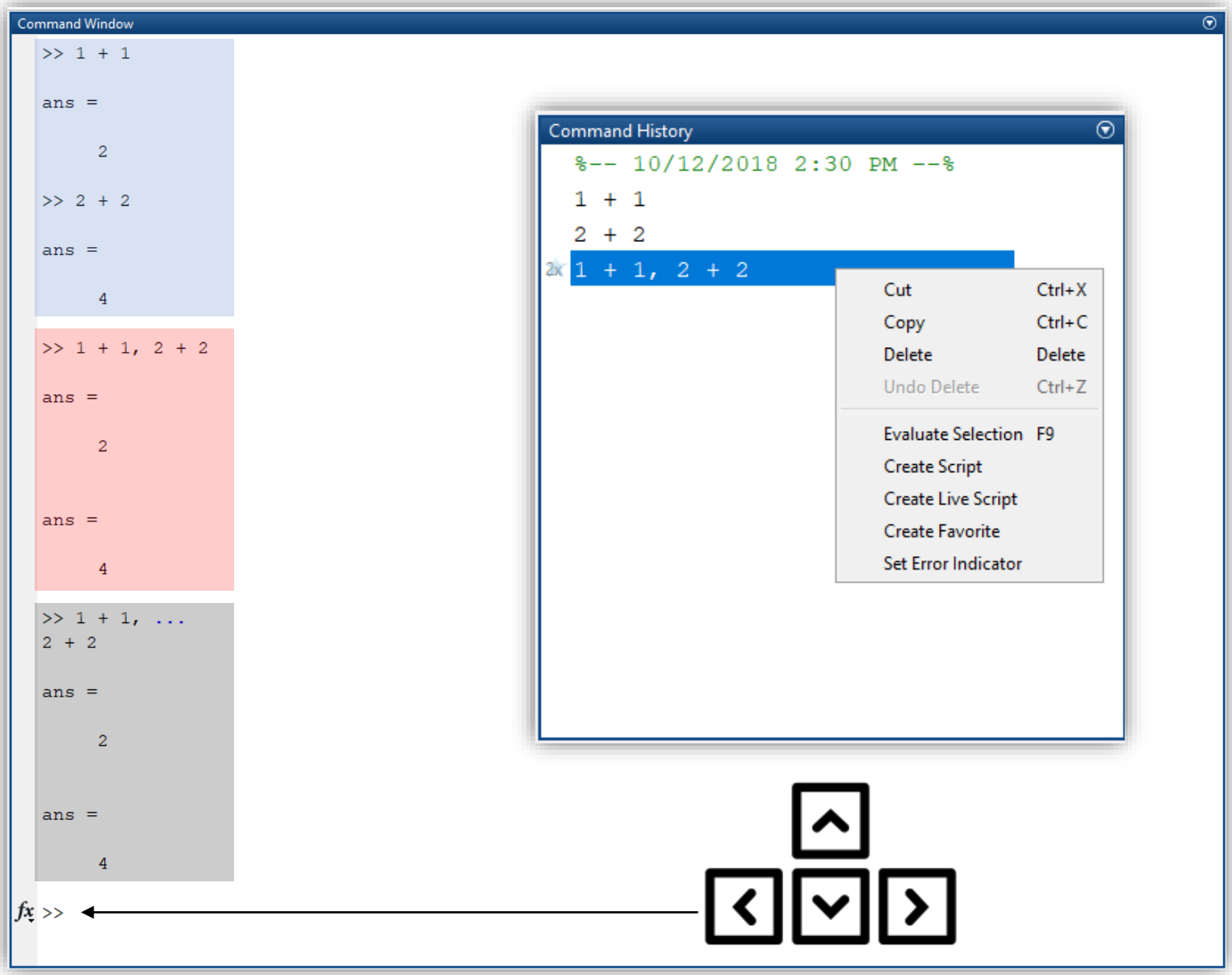


Working In The Command Window



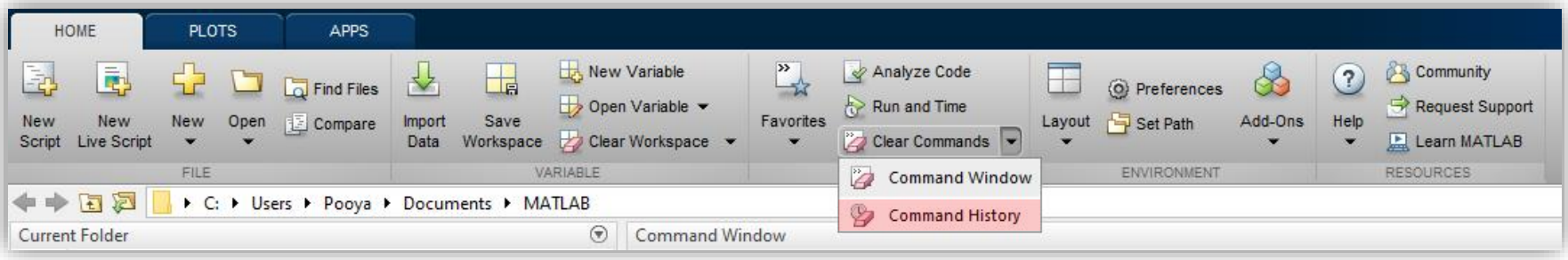
Chapter 01

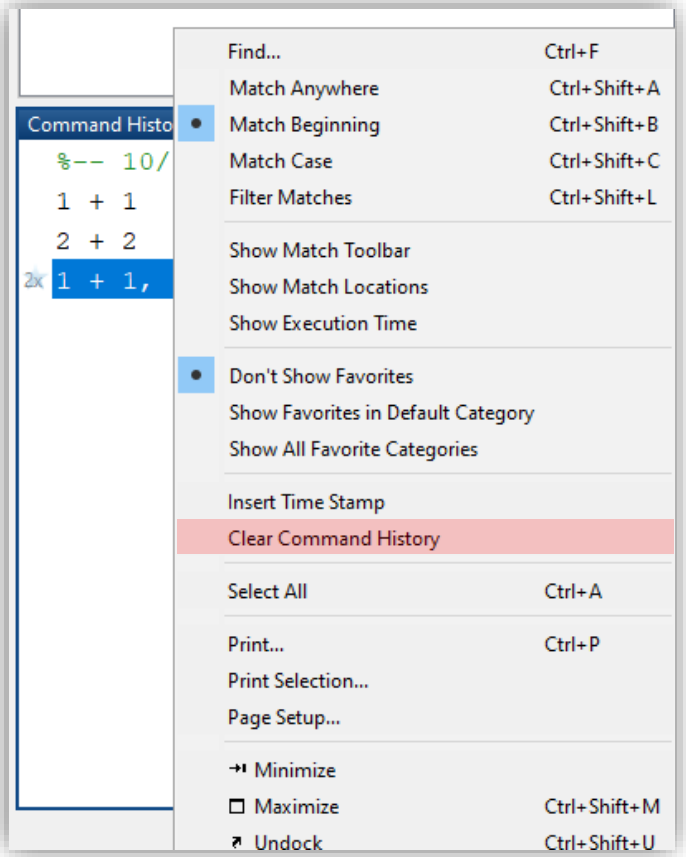
Working In The Command Window



Chapter 01

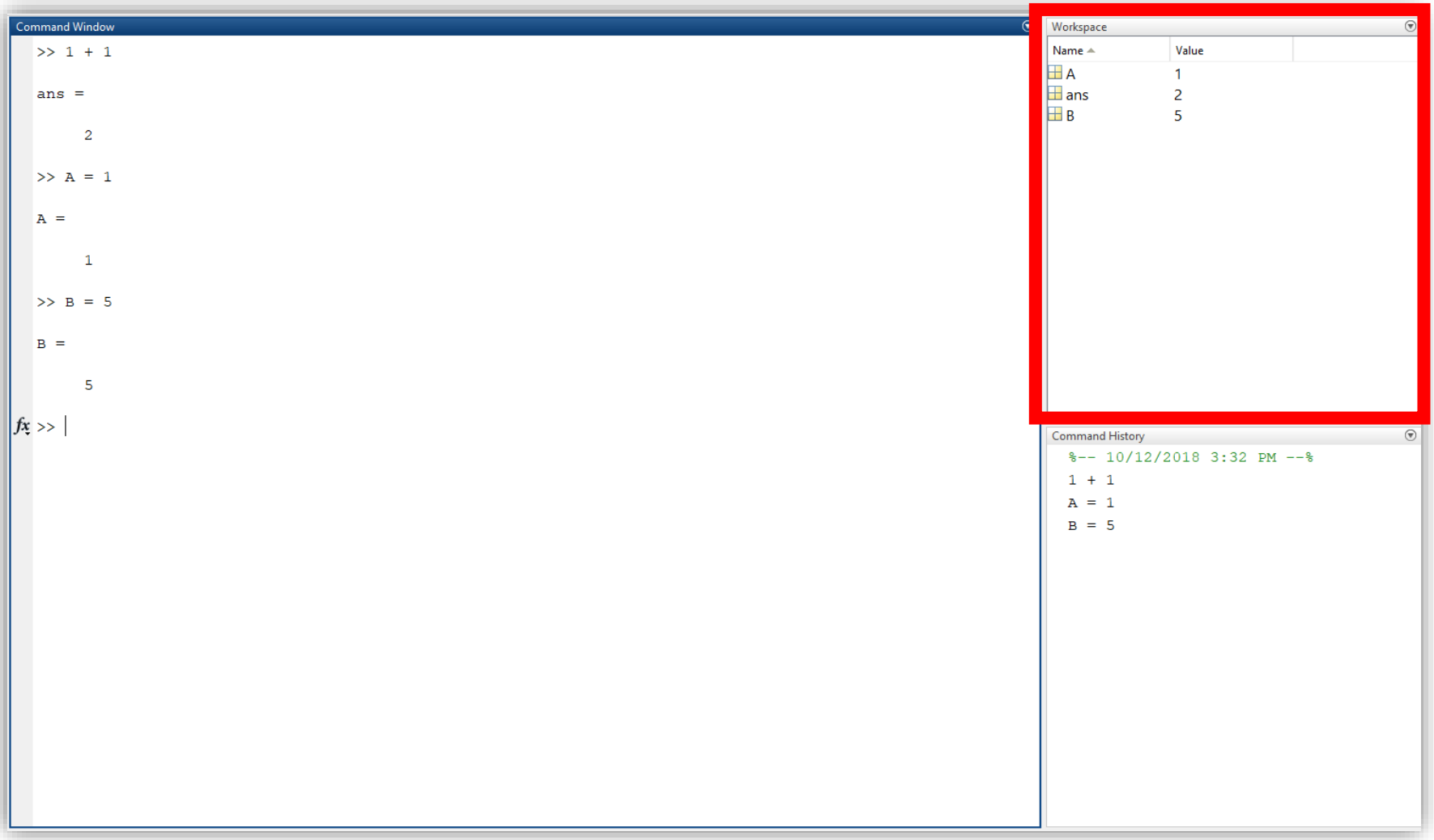
Working In The Command Window





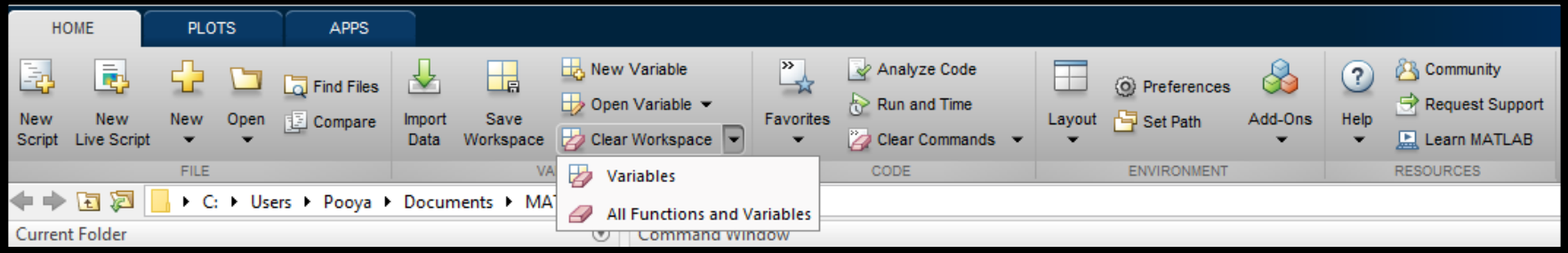
Chapter 01

Working In The Command Window



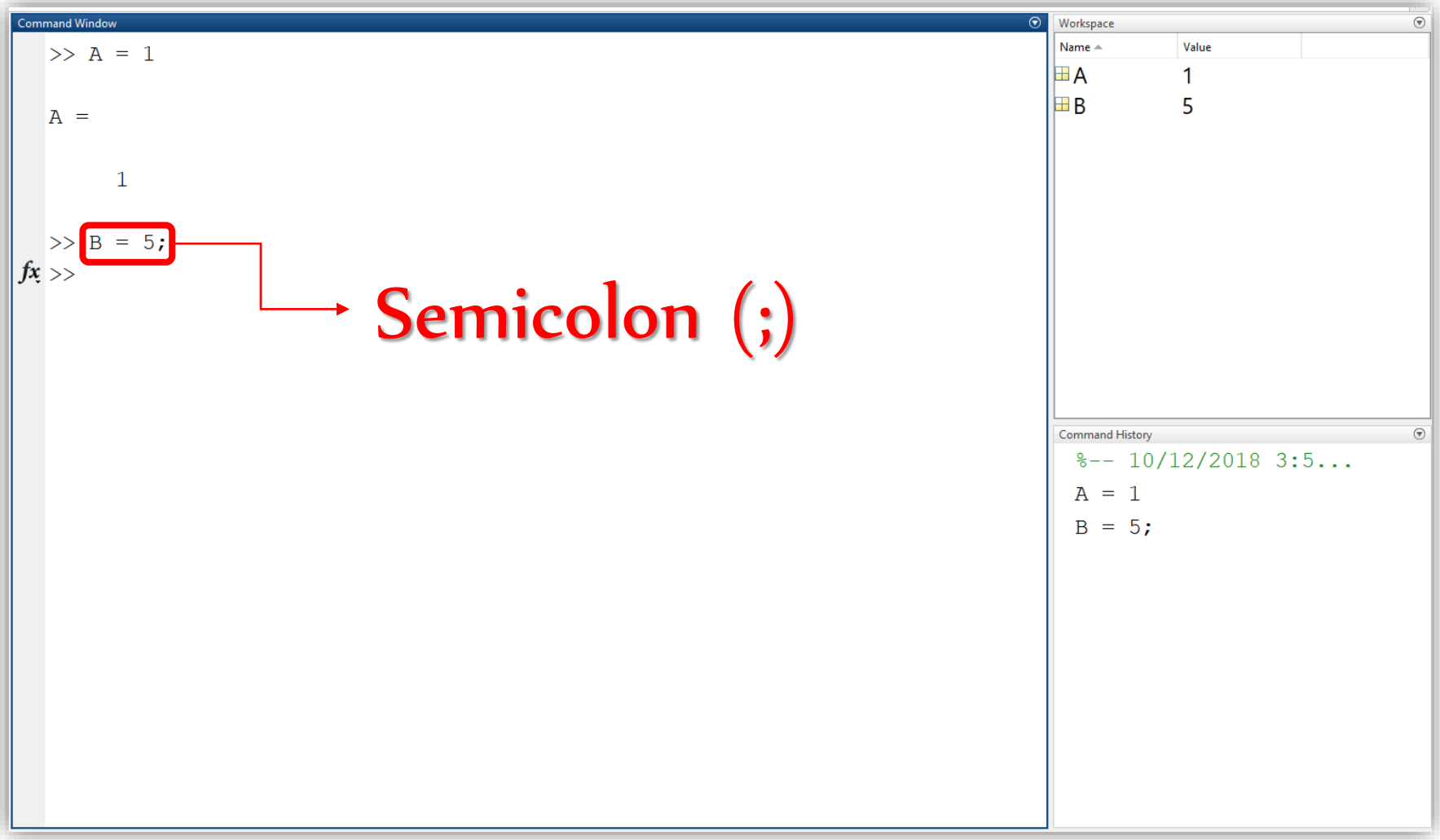
Chapter 01

Working In The Command Window



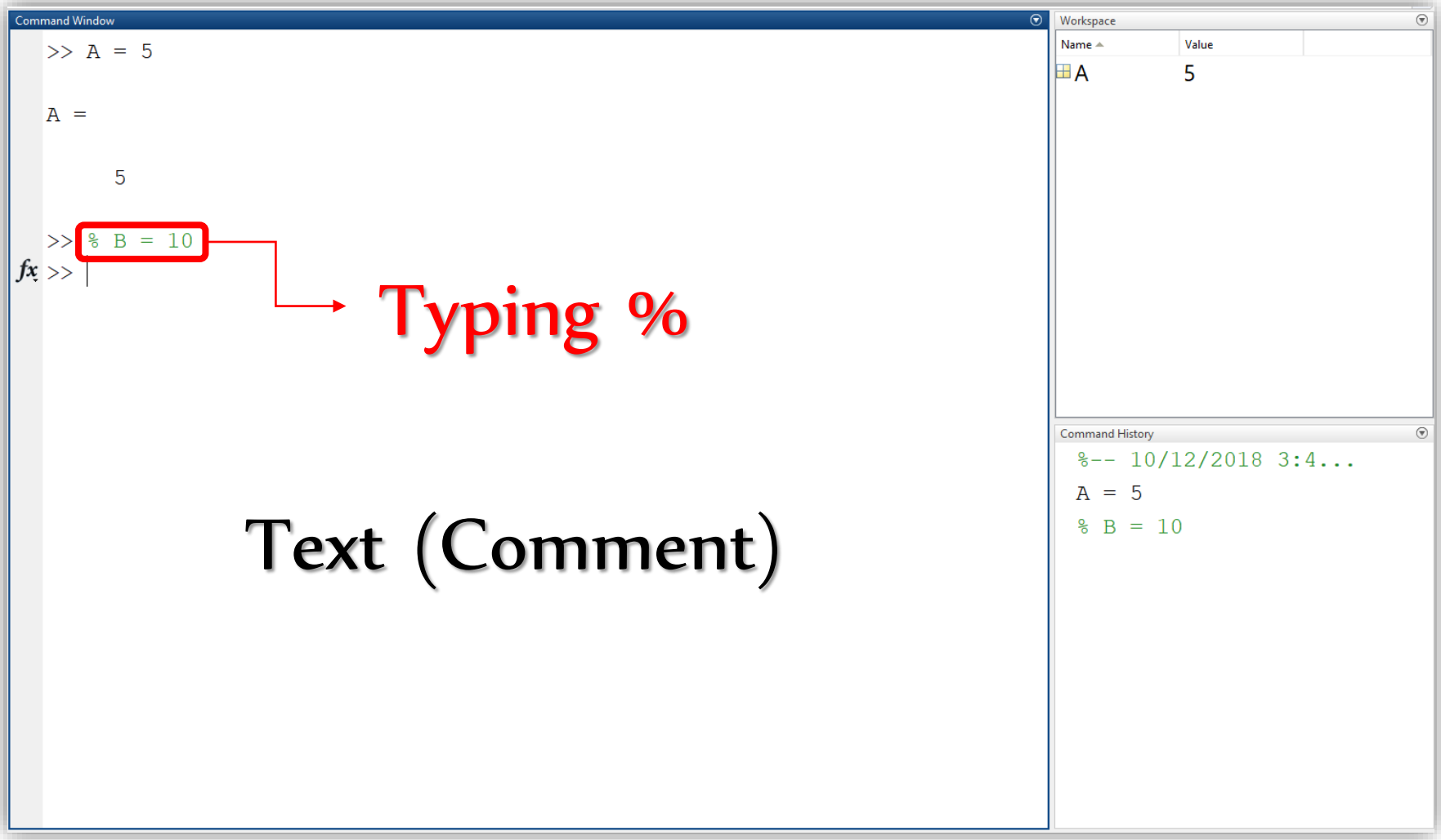
Chapter 01

Working In The Command Window



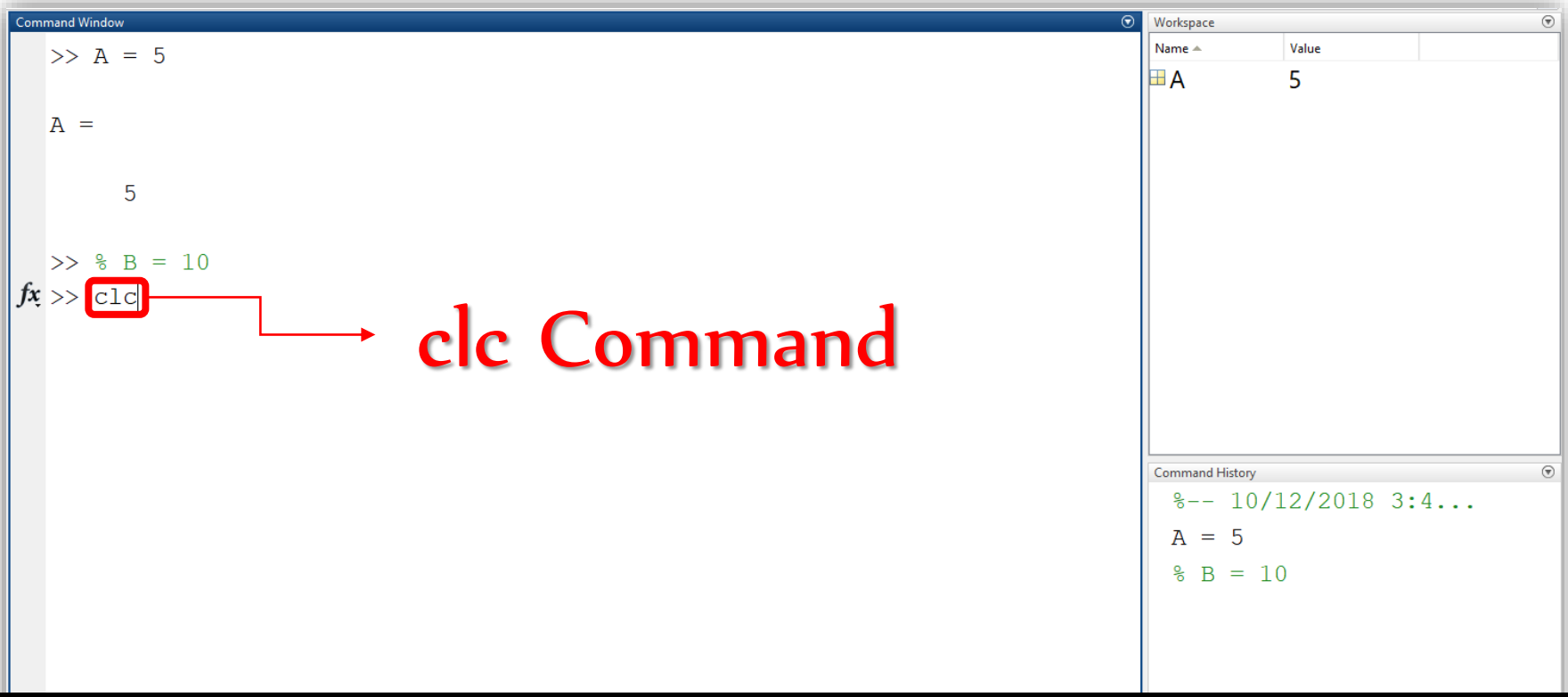
Chapter 01

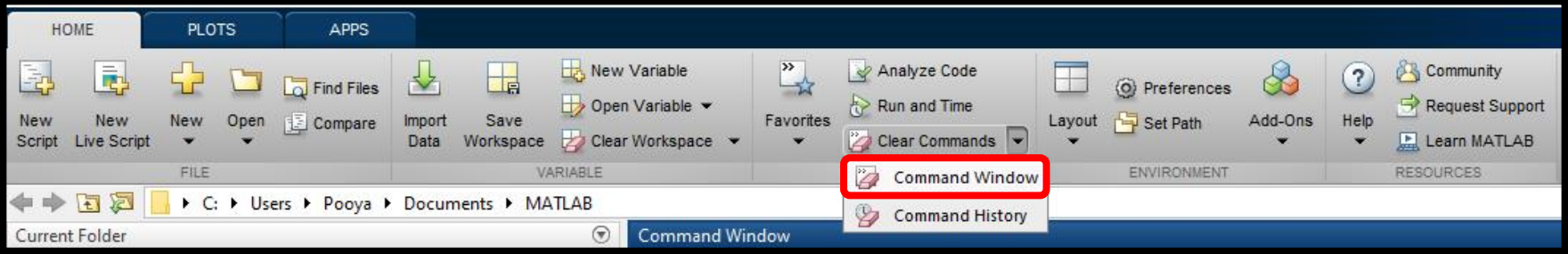
Working In The Command Window



Chapter 01

Working In The Command Window





Arithmetic Operations With Scalars

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 53 = 125)

Arithmetic Operations With Scalars

Order Of Precedence

Precedence	Mathematical Operation
First	Parentheses. For Nested Parentheses, The Innermost Are Executed First.
Second	Exponentiation.
Third	Multiplication, Division (Equal Precedence).
Fourth	Addition And Subtraction.

Arithmetic Operations With Scalars


```
>> 7+8/2
ans =
    11
>> (7+8)/2
ans =
    7.5000
>> 4+5/3+2
ans =
    7.6667
>> 5^3/2
ans =
   62.5000
>> 27^(1/3)+32^0.2
ans =
     5
>> 27^1/3+32^0.2
ans =
    11

>> 0.7854-(0.7854)^3/(1*2*3)+0.785^5/(1*2*3*4*5)...
-(0.785)^7/(1*2*3*4*5*6*7)

ans =
    0.7071
>>
```

Type and press **Enter**.

8/2 is executed first.

Type and press **Enter**.

7+8 is executed first.

5/3 is executed first.

5^3 is executed first, /2 is executed next.

1/3 is executed first, 27^(1/3) and 32^0.2 are executed next, and + is executed last.

27^1 and 32^0.2 are executed first, /3 is executed next, and + is executed last.

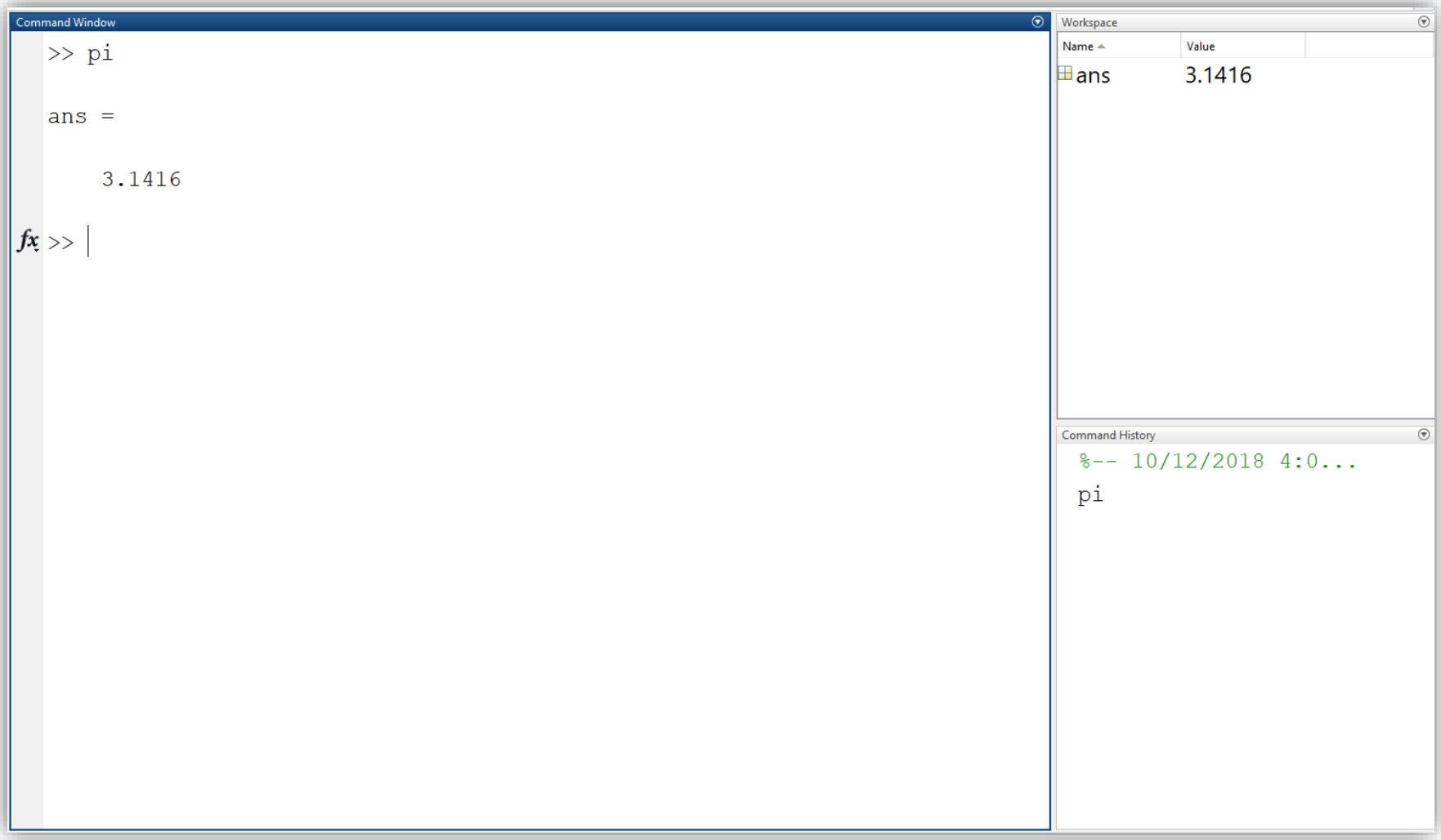
Type three periods ... (and press **Enter**) to continue the expression on the next line.

The last expression is the first four terms of the Taylor series for sin(π/4).

Chapter 01

Display Formats

The User Can Control The Format In Which MATLAB Displays Output On The Screen.



Display Formats

The User Can Control The Format In Which MATLAB Displays Output On The Screen.

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+001`

`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`).	

Elementary Math Built-in Functions

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**.

Elementary Math Built-in Functions

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**.

Outputs = **Name** (Inputs)

Y = **sqrt** (X)

Elementary Math Built-in Functions

$$Y = \text{sqrt}(X)$$

```
>> sqrt(64)
ans =
     8
>> sqrt(50+14*3)
ans =
    9.5917
>> sqrt(54+9*sqrt(100))
ans =
    12
>> (15+600/4)/sqrt(121)
ans =
    15
>>
```

Argument is a number.

Argument is an expression.

Argument includes a function.

Function is included in an expression.

Elementary Math Built-in Functions

Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example

`sqrt(x)`	Square root.	`>> sqrt(81)`
		`ans =`
		`9`

`nthroot(x,n)`	Real *n*th root of a real number *x*. (If *x* is negative *n* must be an odd integer.)	`>> nthroot(80,5)`
		`ans =`
		`2.4022`

`exp(x)`	Exponential (e^x).	`>> exp(5)`
		`ans =`
		`148.4132`

Elementary Math Built-in Functions

Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example

`abs(x)`	Absolute value.	`>> abs(-24)`
		`ans =`
		`24`

`log(x)`	Natural logarithm.
	Base *e* logarithm (ln).
		`>> log(1000)`
		`ans =`
		`6.9078`

`log10(x)`	Base 10 logarithm.	`>> log10(1000)`
		`ans =`
		`3.0000`

`factorial(x)`	The factorial function *x*!
	(*x* must be a positive integer.)
		`>> factorial(5)`
		`ans =`
		`120`

Elementary Math Built-in Functions

Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example

`sin(x)`
`sind(x)`
	Sine of angle x (x in radians).
	Sine of angle x (x in degrees).
		`>> sin(pi/6)`
		`ans =`
		`0.5000`

`cos(x)`
`cosd(x)`
	Cosine of angle x (x in radians).
	Cosine of angle x (x in degrees).
		`>> cosd(30)`
		`ans =`
		`0.8660`

`tan(x)`
`tand(x)`
	Tangent of angle x (x in radians).
	Tangent of angle x (x in degrees).
		`>> tan(pi/6)`
		`ans =`
		`0.5774`

`cot(x)`
`cotd(x)`
	Cotangent of angle x (x in radians).
	Cotangent of angle x (x in degrees).
		`>> cotd(30)`
		`ans =`
		`1.7321`

Elementary Math Built-in Functions

Some Commonly Used Elementary MATLAB Mathematical Built-in Functions

Function	Description	Example

`round(x)`	Round to the nearest integer.	`>> round(17/5)`
		`ans =`
		`3`

`fix(x)`	Round toward zero.	`>> fix(13/5)`
		`ans =`
		`2`

`ceil(x)`	Round toward infinity.	`>> ceil(11/5)`
		`ans =`
		`3`

`floor(x)`	Round toward minus infinity.	`>> floor(-9/4)`
		`ans =`
		`-3`

`rem(x,y)`	Returns the remainder after x is divided by y.	`>> rem(13,5)`
		`ans =`
		`3`

`sign(x)`	Signum function. Returns 1 if $x > 0$, -1 if $x < 0$, and 0 if $x = 0$.	`>> sign(5)`
		`ans =`
		`1`

Defining Scalar Variables

The Assignment Operator

Variable_name = A numerical value, or a computable expression

```
>> x=15
x =
     15

>> x=3*x-12
x =
     33
>>
```

The number 15 is assigned to the variable x.

MATLAB displays the variable name and its assigned value.

A new value is assigned to x. The new value is 3 times the previous value of x minus 12.

Defining Scalar Variables

The Assignment Operator

THINK OF = AS MEANING "ASSIGN TO" OR "STORE IN" BUT NOT 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"

Defining Scalar Variables

The Assignment Operator

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

The variables a, B, and C are defined but are not displayed, since a semicolon is typed at the end of each statement.

The value of the variable C is displayed by typing the name of the variable.

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

The variable B is not displayed because a semicolon is typed at the end of the assignment.

Defining Scalar Variables

The Assignment Operator

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

A value of 72 is assigned to the variable ABB.

A new value of 9 is assigned to the variable ABB.

The current value of the variable is displayed when the name of the variable is typed and the **Enter** key is pressed.

```
>> x=0.75;
>> E=sin(x)^2+cos(x)^2
E =
     1
>>
```

??? Undefined function or variable 'x'

Defining Scalar Variables

Rules About Variable Names

A variable can be named according to the following rules:

- Must begin with a letter.
- Can be up to 63 characters long.
- Can contain letters, digits, and the underscore character.
- Cannot contain punctuation characters (e.g., period, comma, semicolon).
- MATLAB is case-sensitive: it distinguishes between uppercase and lowercase letters. For example, `AA`, `Aa`, `aA`, and `aa` are the names of four different variables.
- No spaces are allowed between characters (use the underscore where a space is desired).
- Avoid using the name of a built-in function for a variable (i.e., avoid using `cos`, `sin`, `exp`, `sqrt`, etc.). Once a function name is used to for a variable name, the function cannot be used.

Defining Scalar Variables

Predefined Variables and Keywords

```
break case catch classdef continue else elseif end for
function global if otherwise parfor persistent return
spmd switch try while
```

`ans`	A variable that has the value of the last expression that was not assigned to a specific variable (see Tutorial 1-1). If the user does not assign the value of an expression to a variable, MATLAB automatically stores the result in `ans`.
`pi`	The number π.
`eps`	The smallest difference between two numbers. Equal to 2^(–52), which is approximately 2.2204e–016.
`inf`	Used for infinity.
`i`	Defined as $\sqrt{-}$, which is: 0 + 1.0000i.
`j`	Same as i.
`NaN`	Stands for Not-a-Number. Used when MATLAB cannot determine a valid numeric value. Example: 0/0.

Useful 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

Script Files

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

Better Way:

A. Save All Commands In A File.
B. With One Command In Command Window, Tell MATLAB To Run All Commands In File.

Script Files

Notes About Script Files

i. A Script File Is A Sequence Of MATLAB Commands, Also Called A Program.

ii. **When A Script File Runs, MATLAB Executes The Commands In The Order They Are Written, Just As If They Were Typed In The Command Window.**

iii. When A Script File Has A Command That Generates An Output, The Output Is Displayed In The Command Window.

iv. **Using A Script File Is Convenient Because It Can Be Edited And Executed Many Times.**

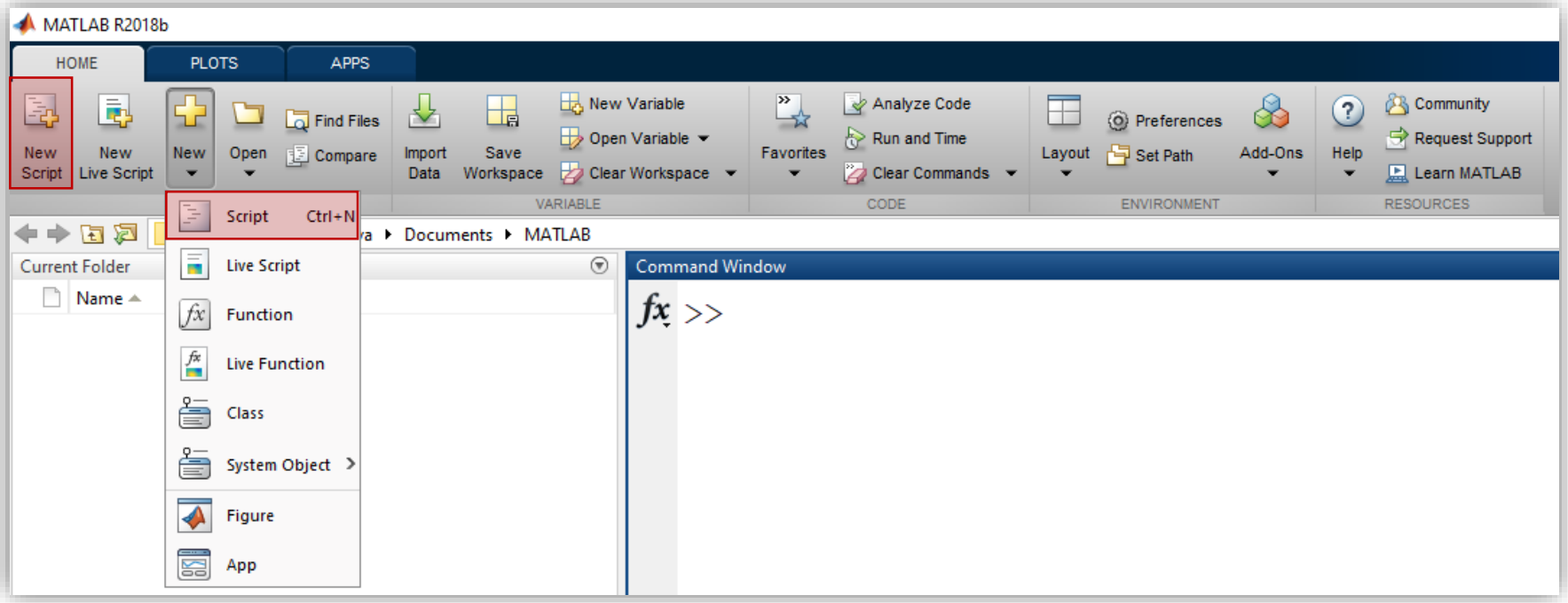
v. Script Files Can Be Typed And Edited In Any Text Editor And Then Pasted Into The MATLAB Editor.

vi. **Script Files Are Also Called M-files Because The Extension .M Is Used When They Are Saved.**

Chapter 01

Script Files

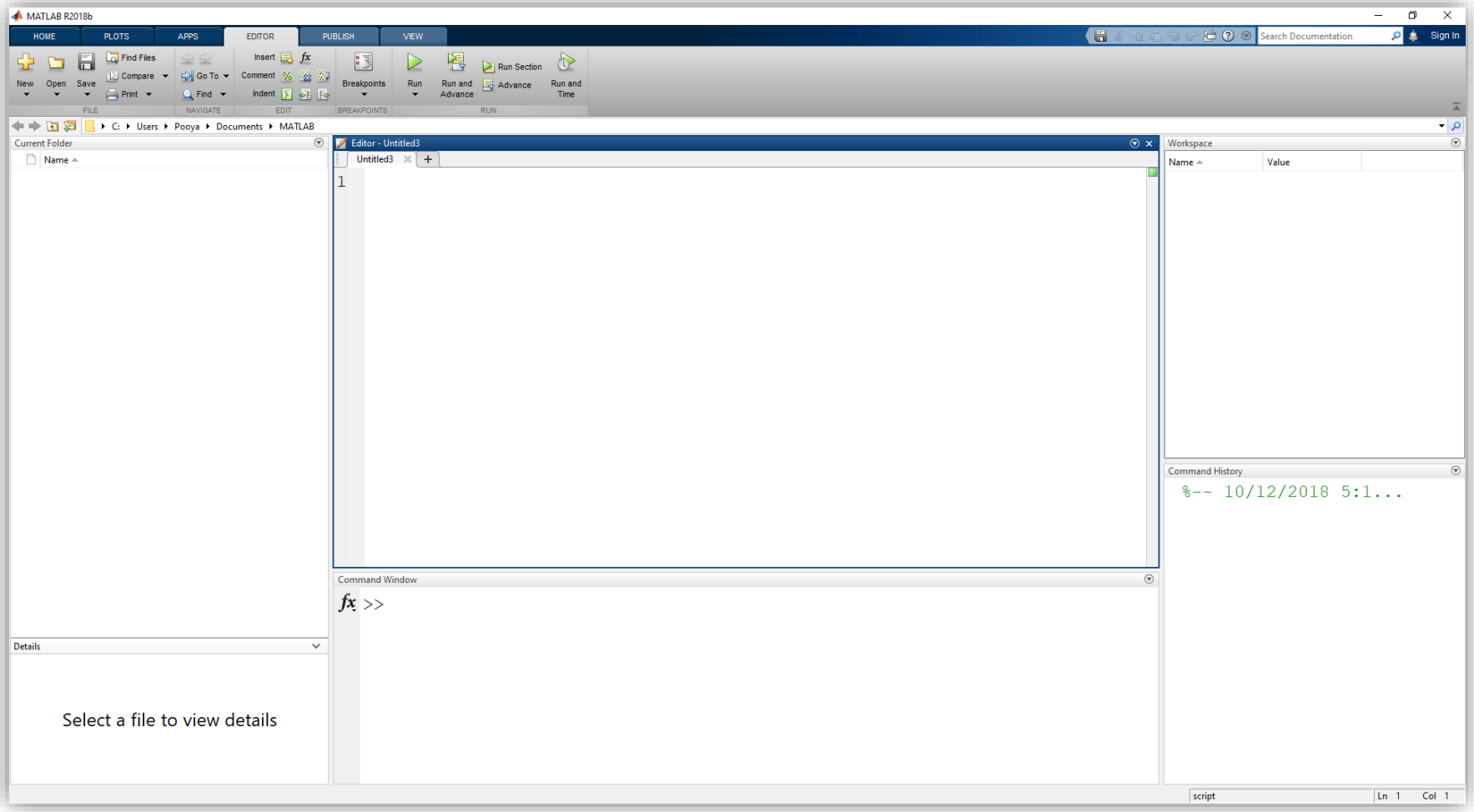
Creating And Saving A Script File



Chapter 01

Script Files

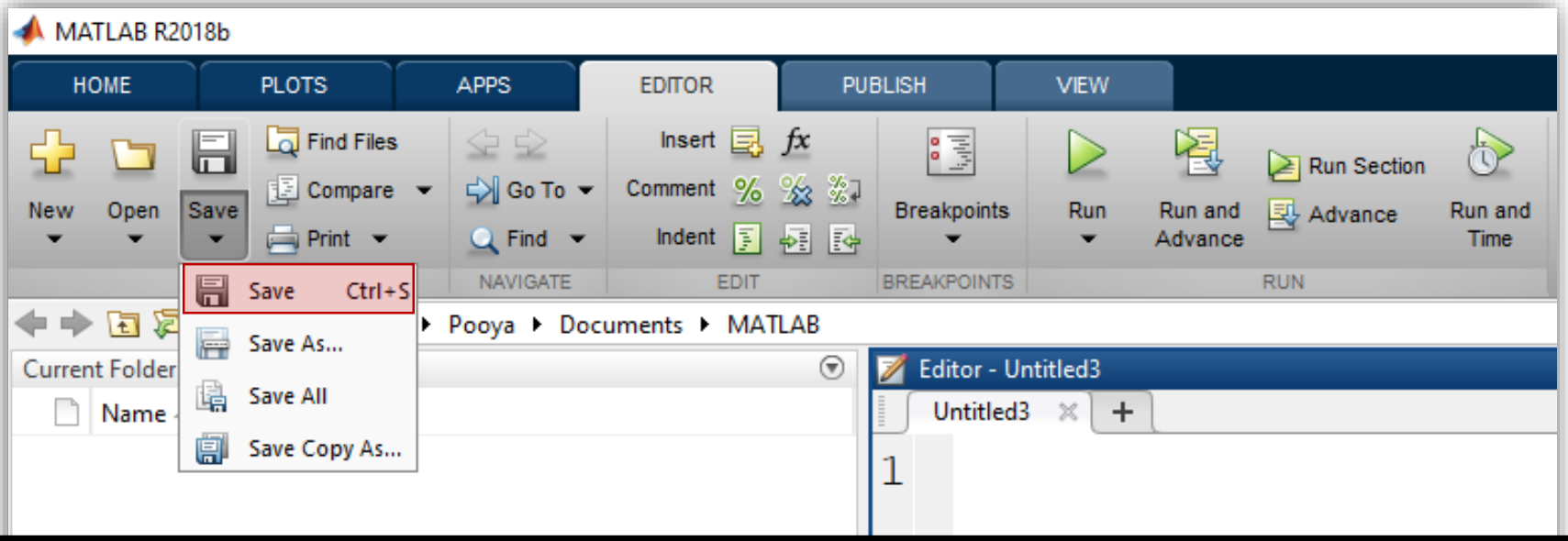
Creating And Saving A Script File

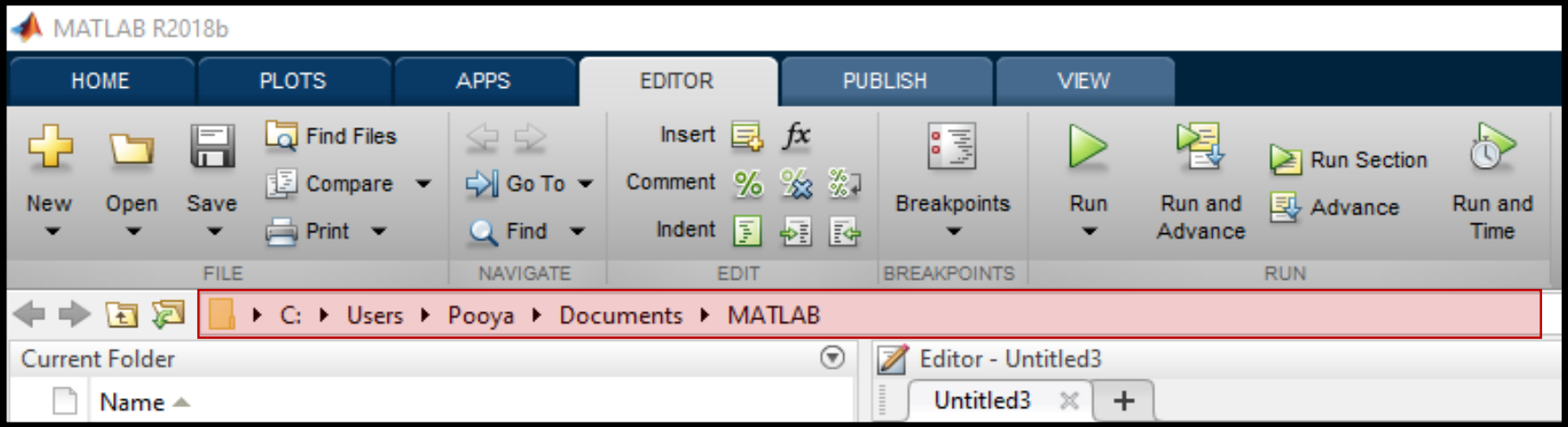


Chapter 01

Script Files

Creating And Saving A Script File

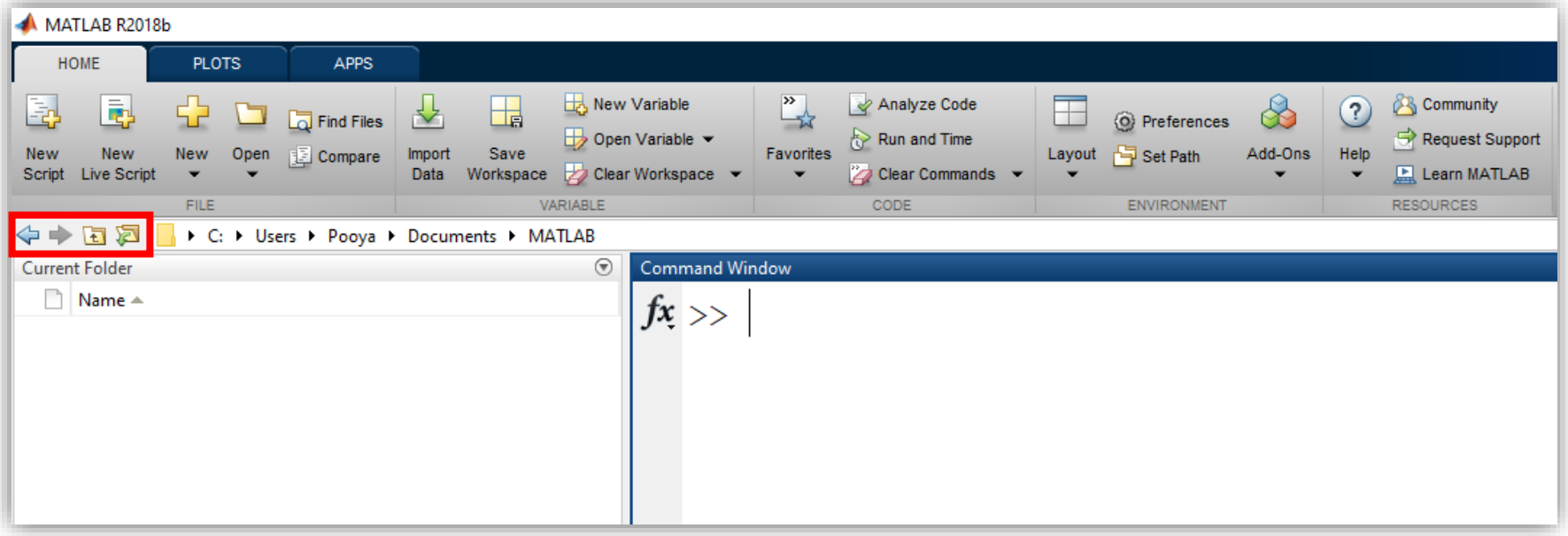




Chapter 01

Script Files

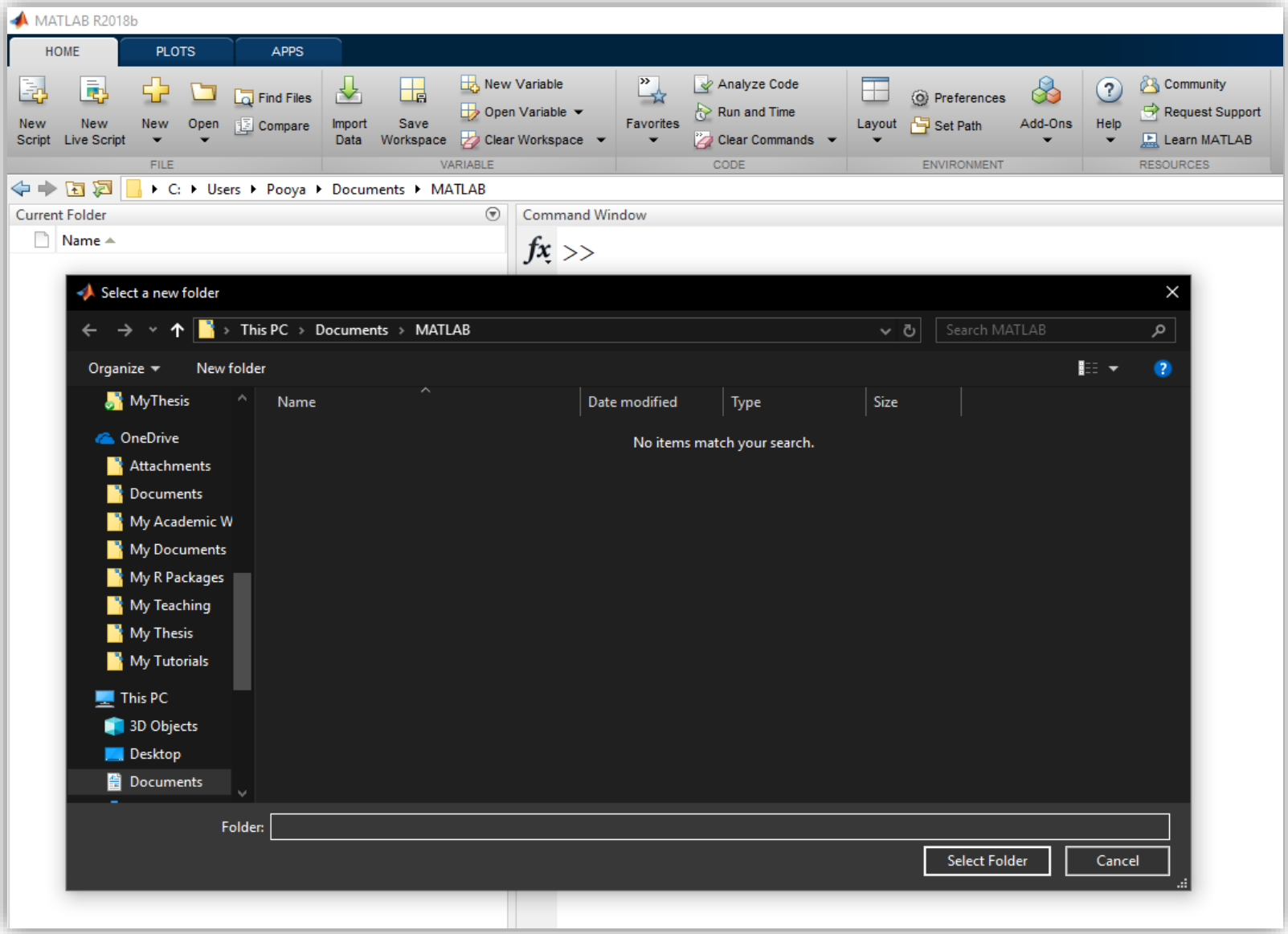
Creating And Saving A Script File



Chapter 01

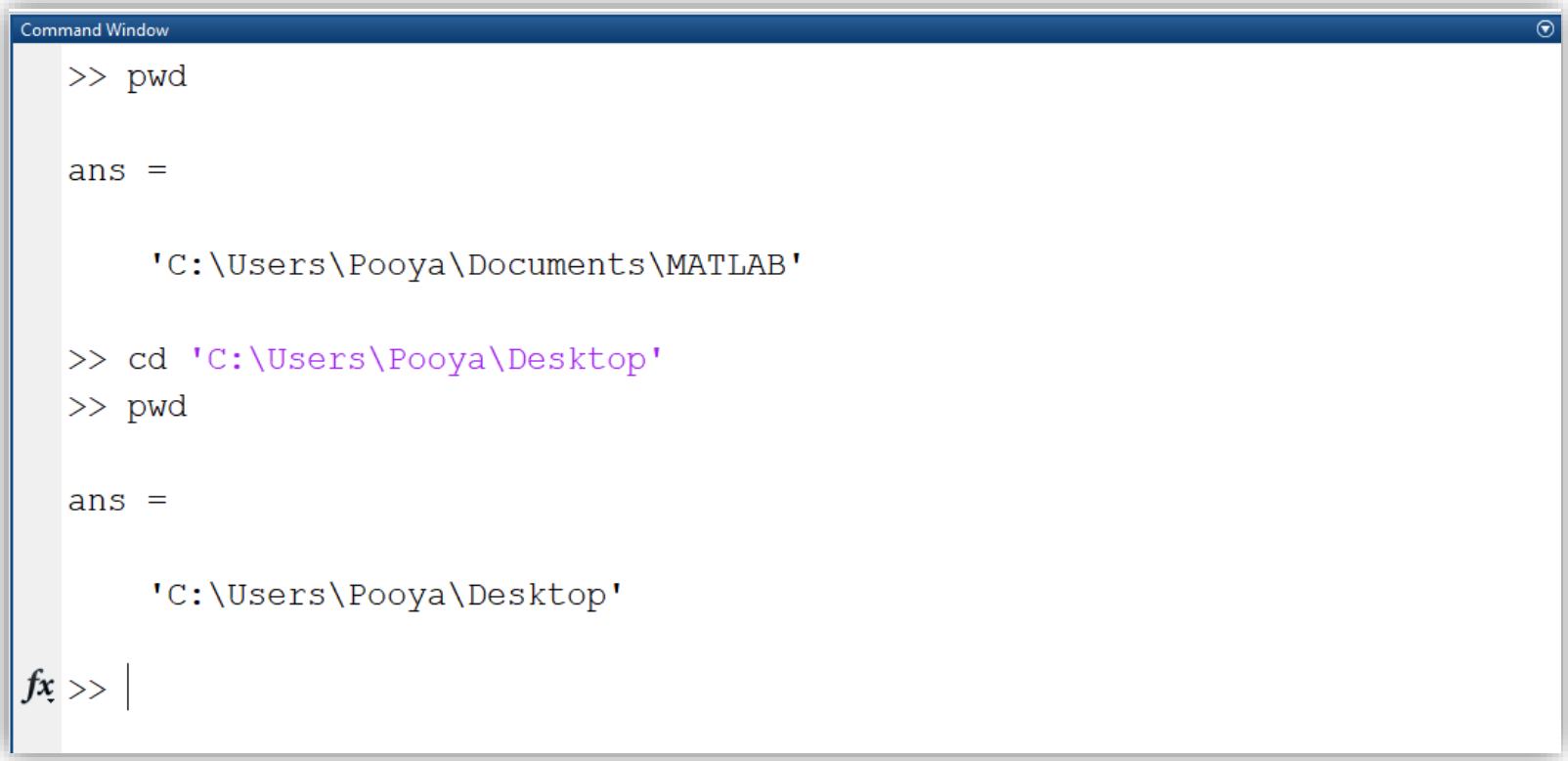
Script Files

Creating And Saving A Script File



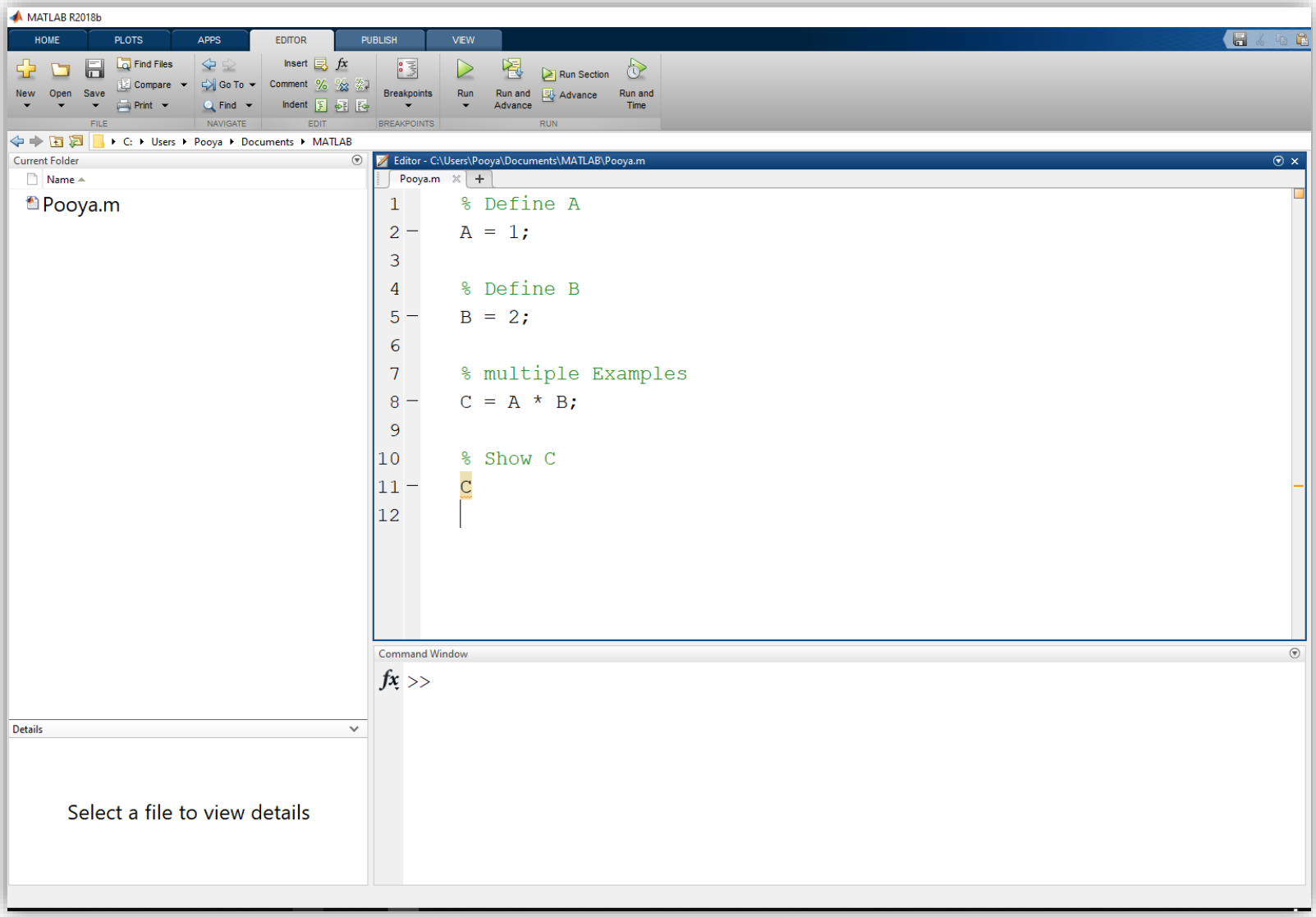
Script Files

Creating And Saving A Script File



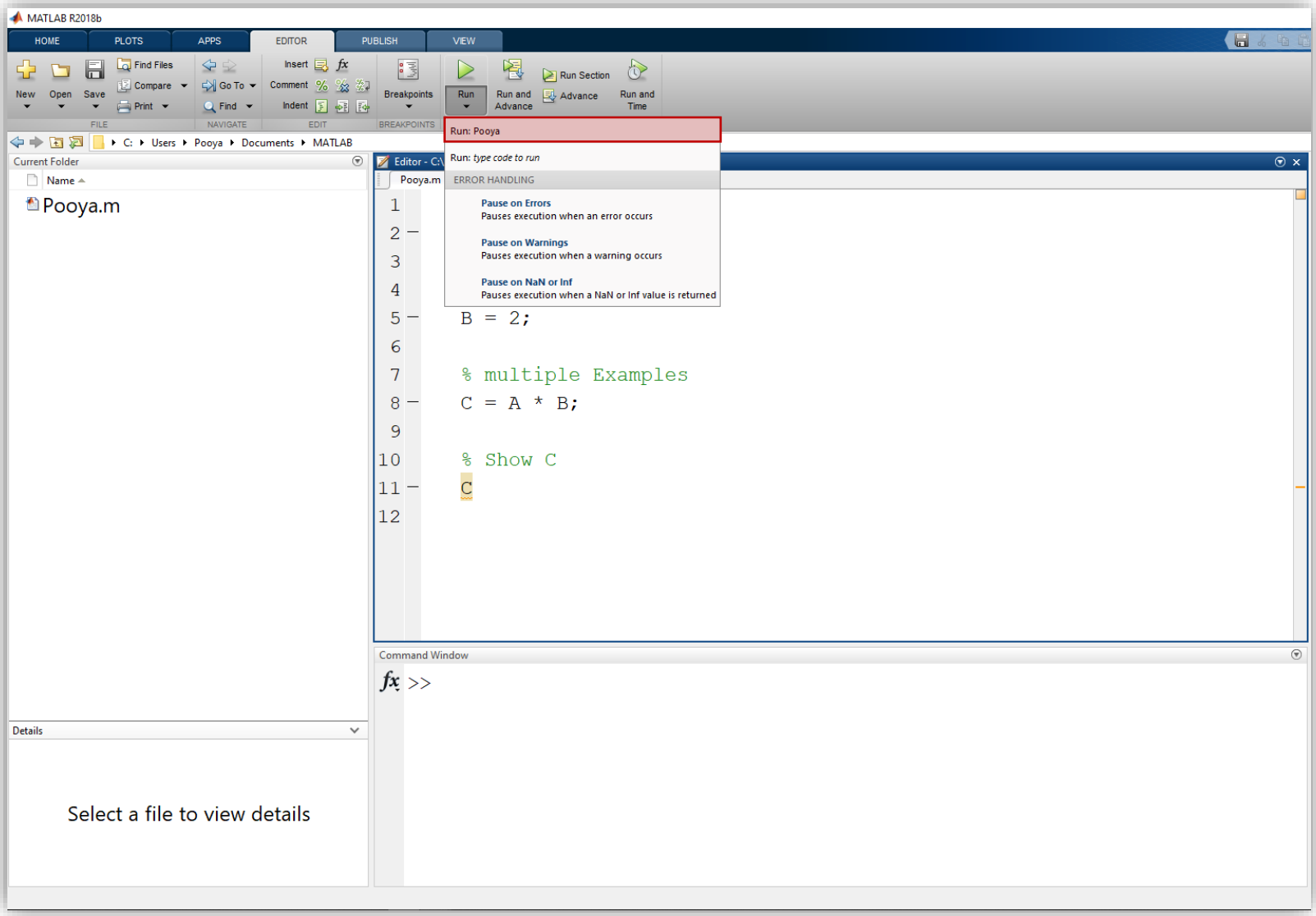
Script Files

Running (Executing) A Script File



Script Files

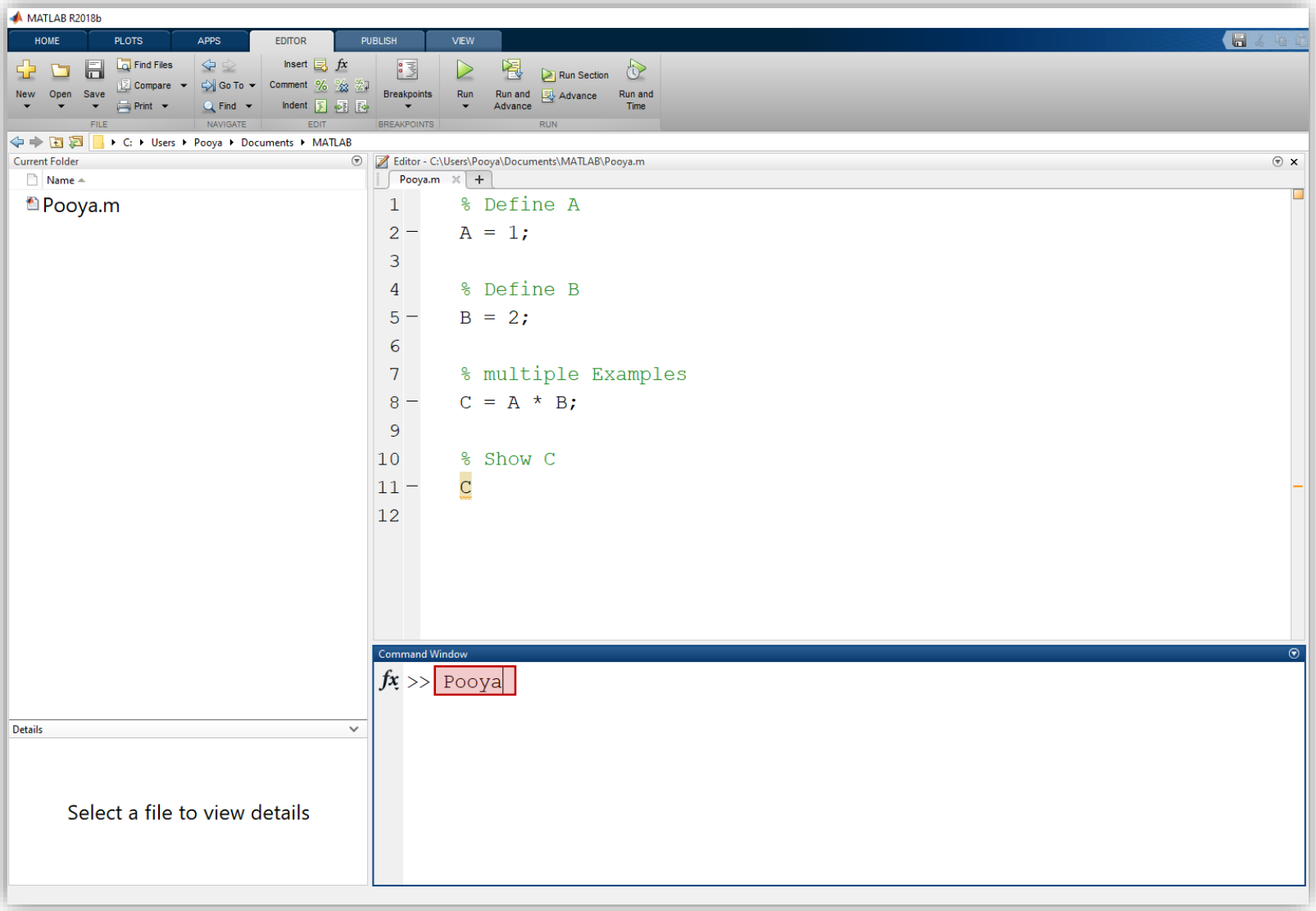
Running (Executing) A Script File



Chapter 01

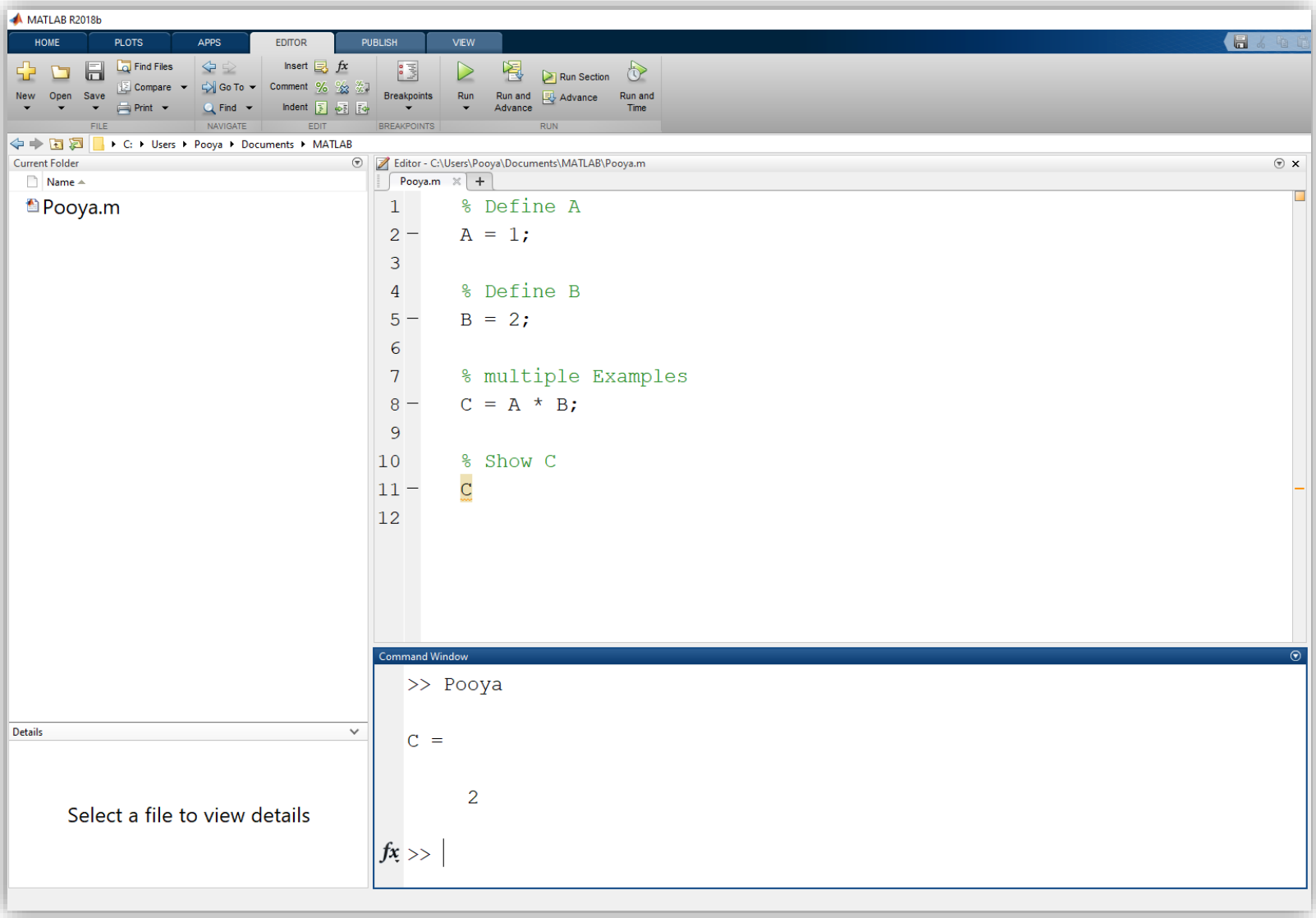
Script Files

Running (Executing) A Script File



Script Files

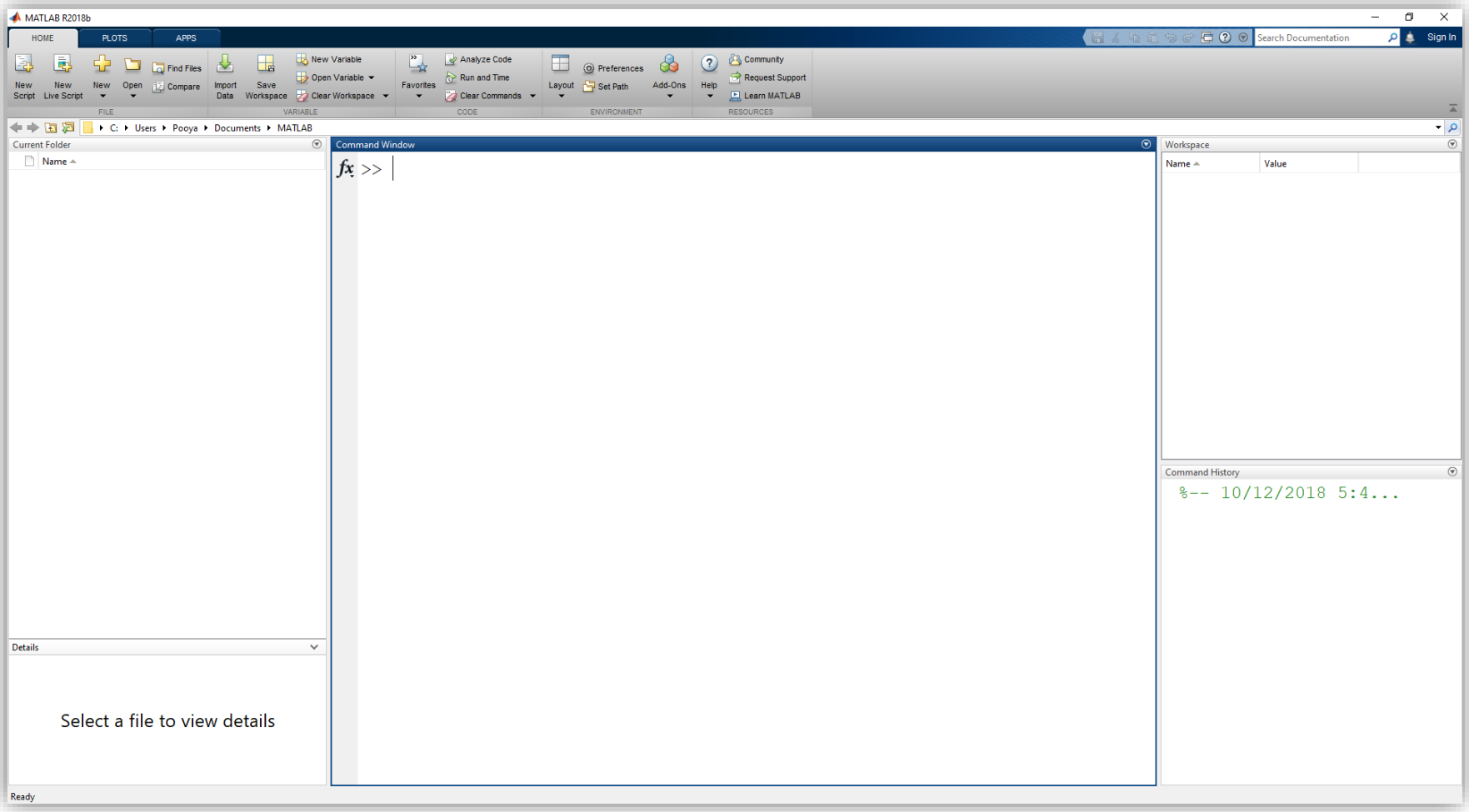
Running (Executing) A Script File



Chapter 01

Script Files

Running (Executing) A Script File



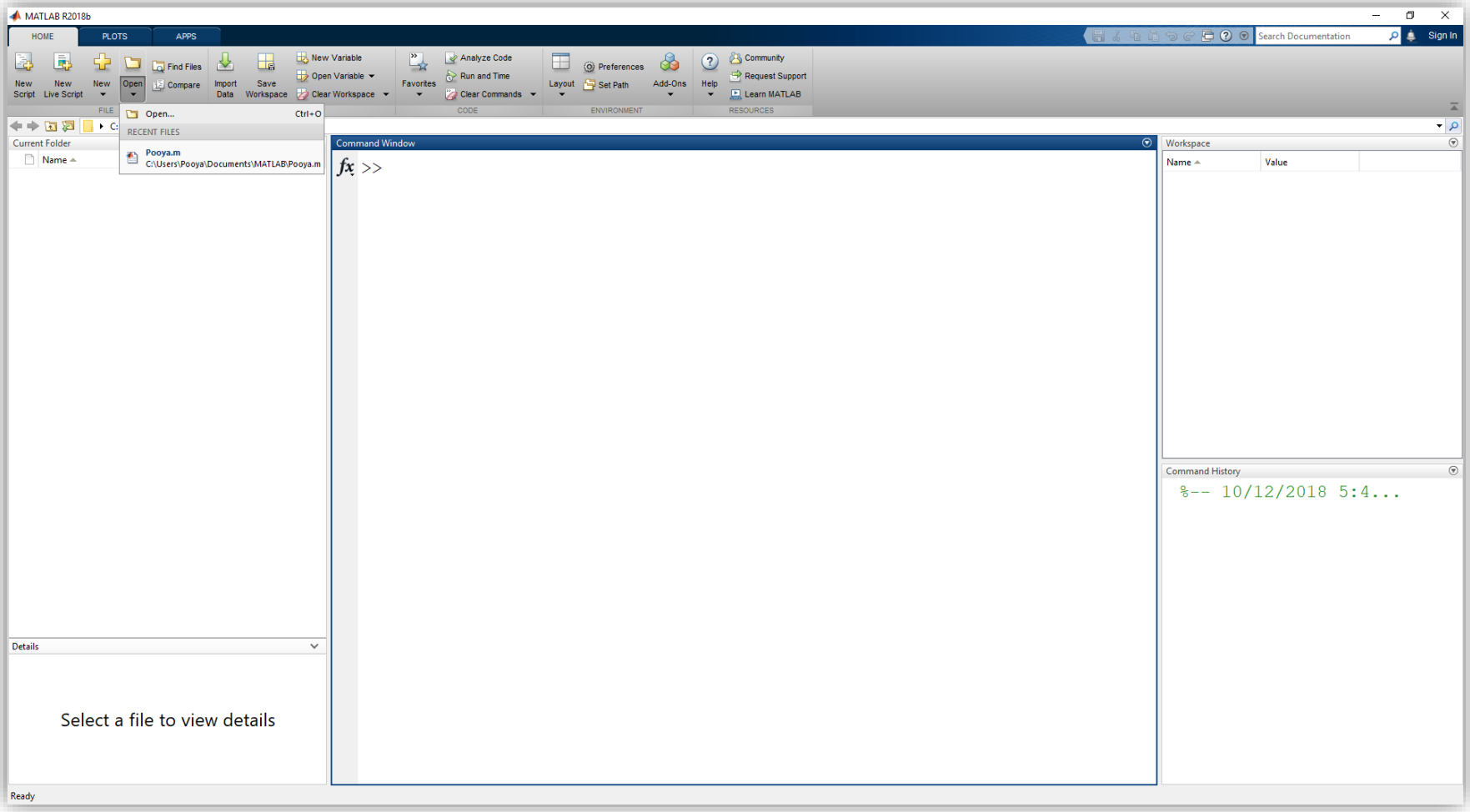
Script Files

Running (Executing) A Script File

Chapter 01

Script Files

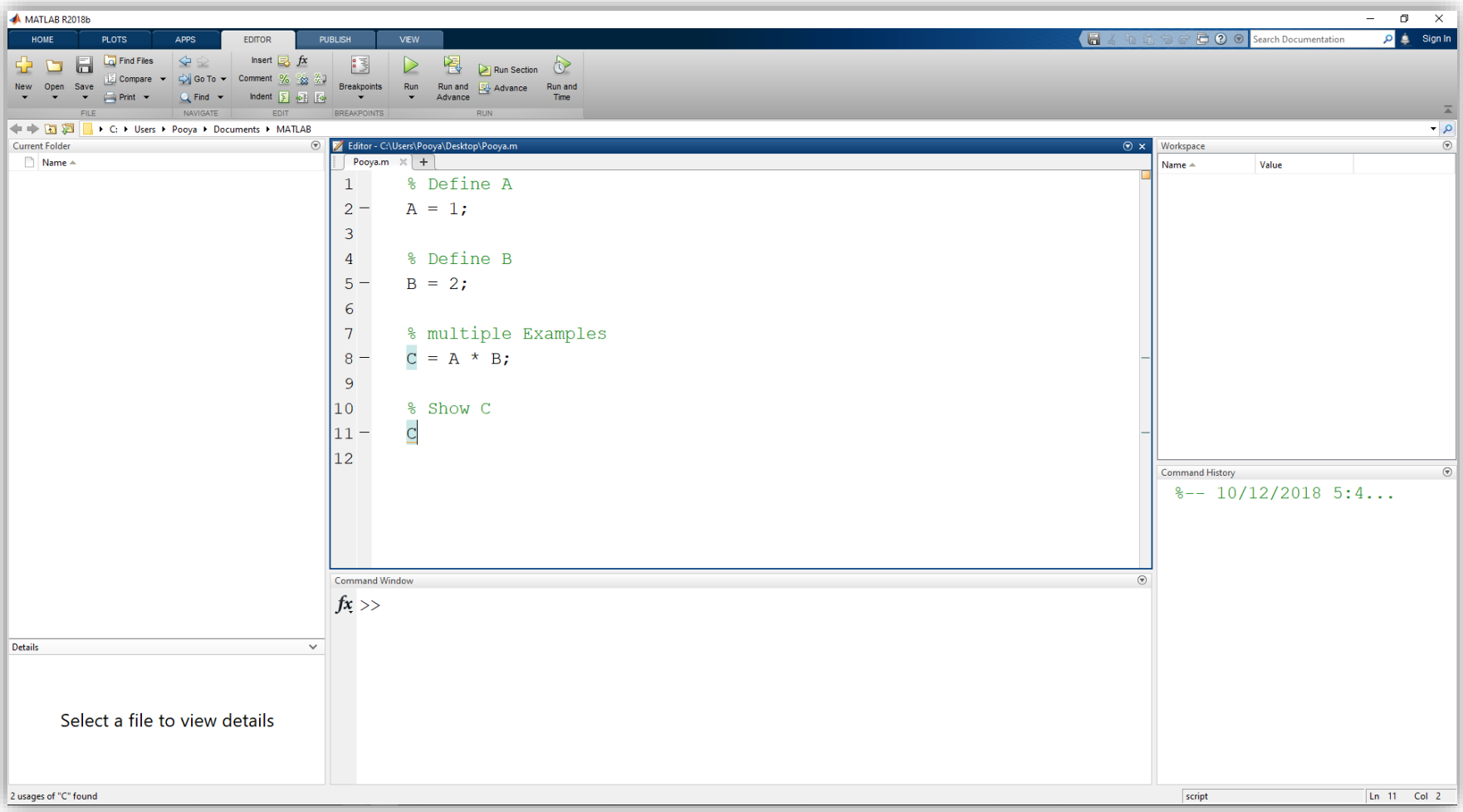
Running (Executing) A Script File



Chapter 01

Script Files

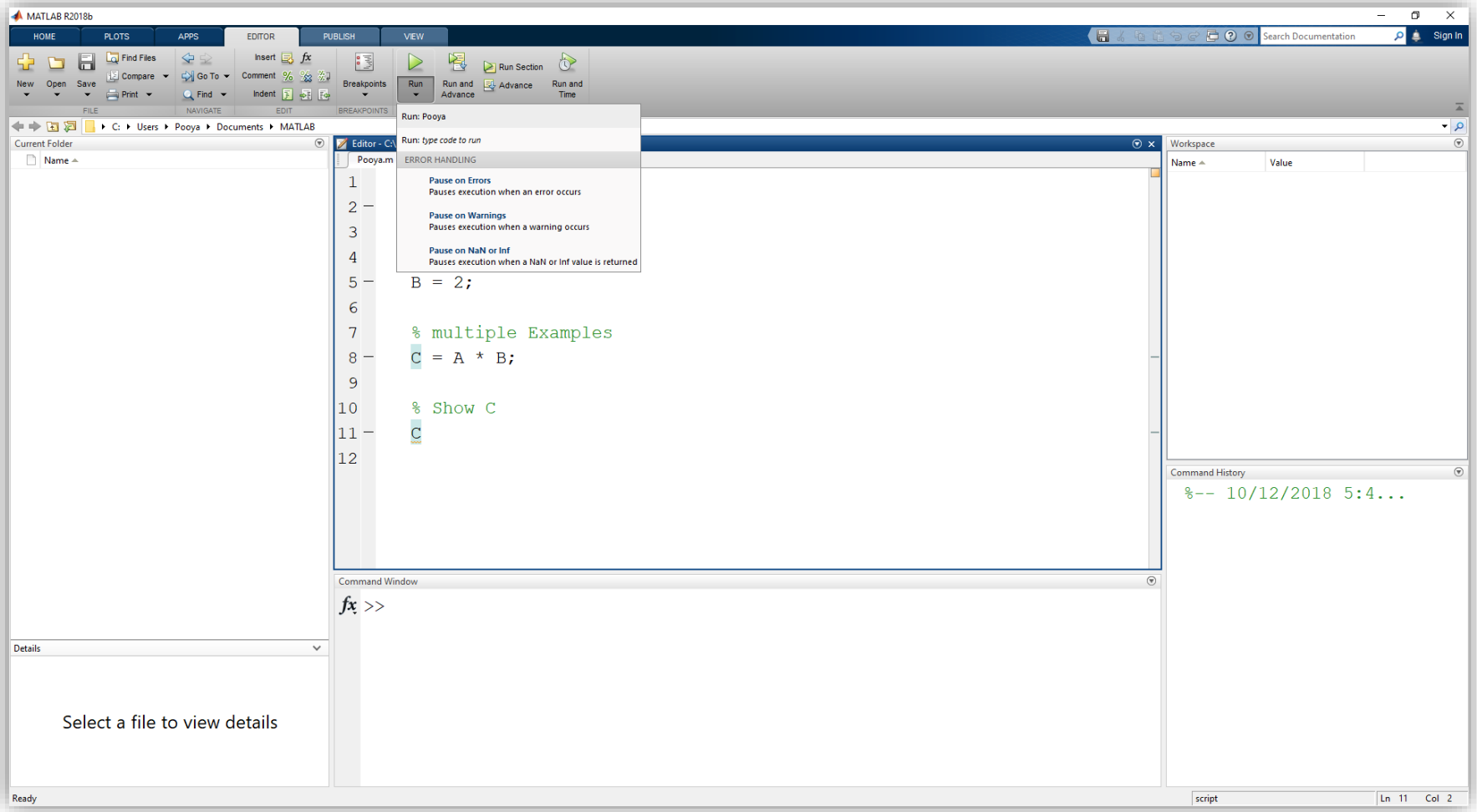
Running (Executing) A Script File



Chapter 01

Script Files

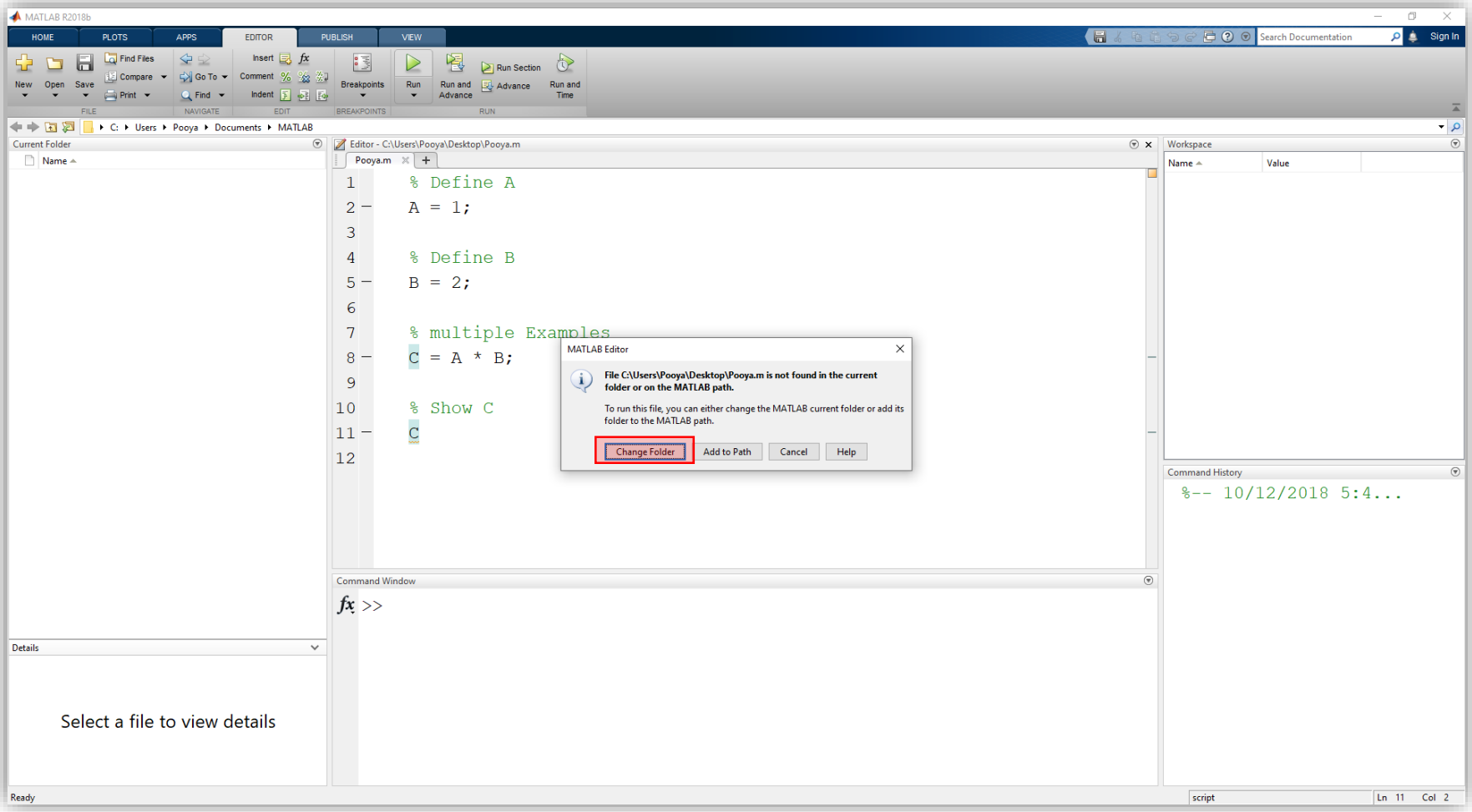
Running (Executing) A Script File



Chapter 01

Script Files

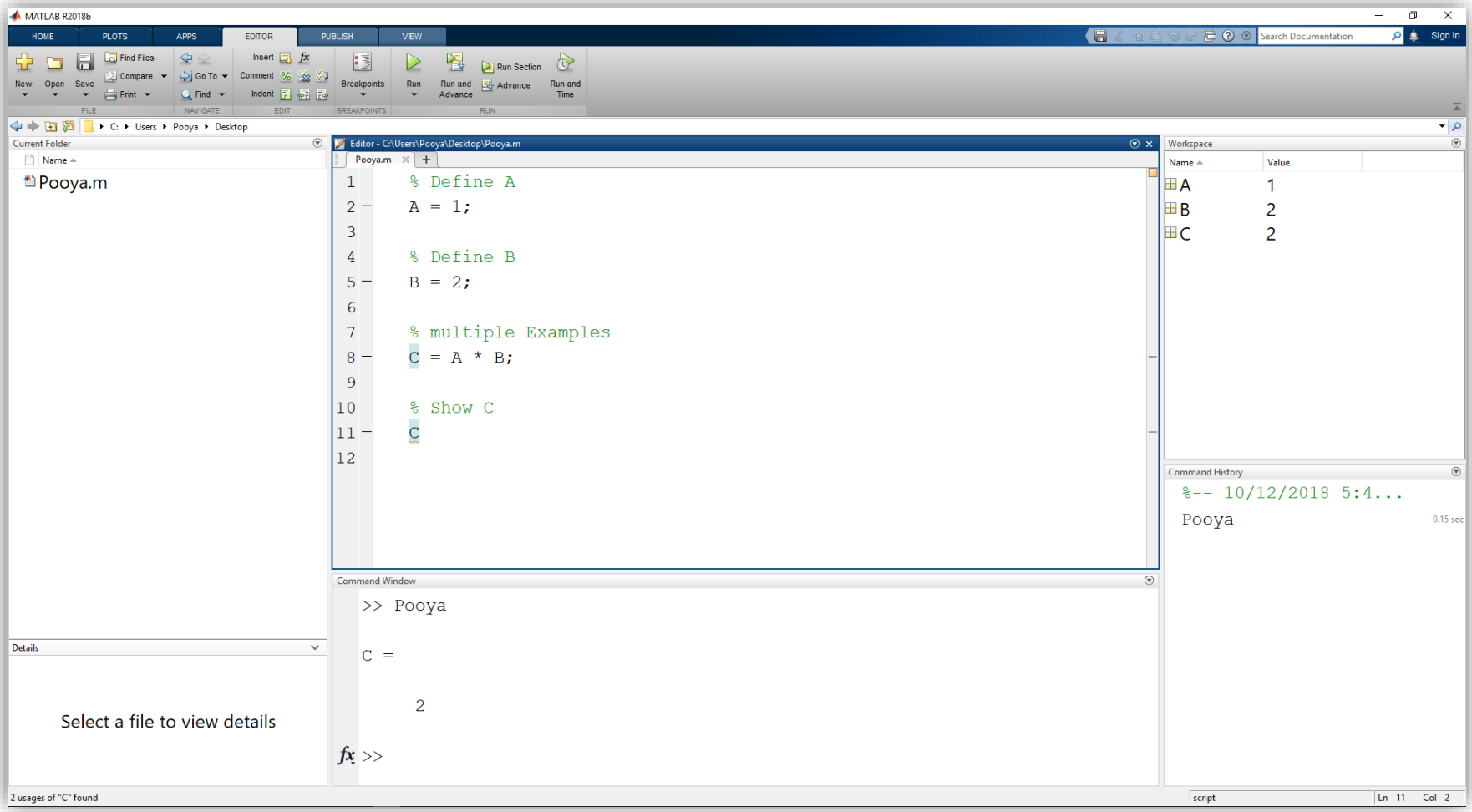
Running (Executing) A Script File



Chapter 01

Script Files

Running (Executing) A Script File



Examples Of MATLAB Applications

Sample Problem 01: HEAT TRANSFER

An object with an initial temperature of T_0 that is placed at time $t = 0$ inside a chamber that has a constant temperature of T_S will experience a temperature change according to the equation

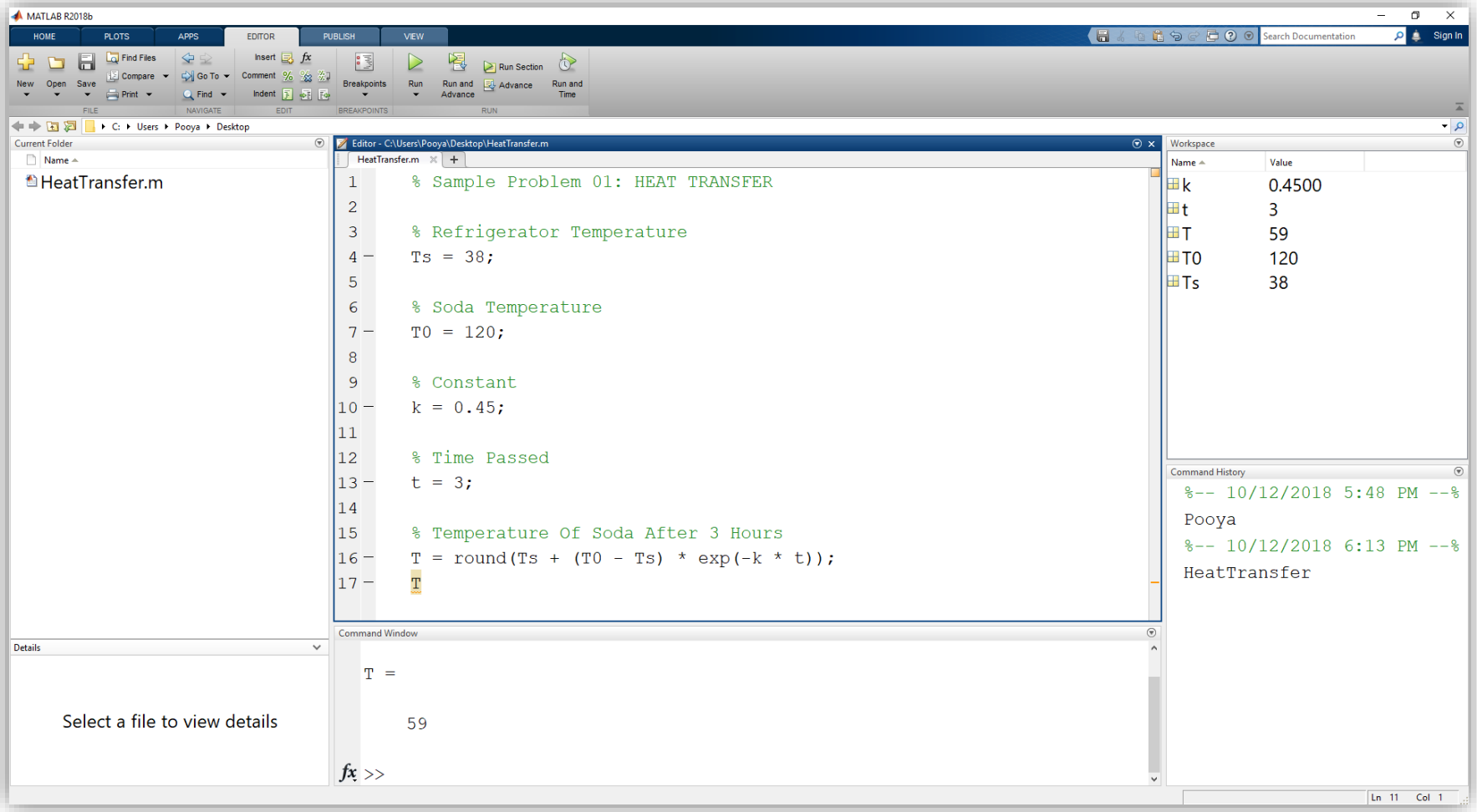
$$T = T_S + (T_0 - T_S)e^{-kt}$$

where T is the temperature of the object at time t, and k is a constant. A soda can at a temperature of 120° F (after being left in the car) is placed inside a refrigerator where the temperature is 38° F. Determine, to the nearest degree, the temperature of the can after three hours. Assume $k = 0.45$. First define all of the variables and then calculate the temperature using one MATLAB command.

Chapter 01

Examples Of MATLAB Applications

Sample Problem 01: HEAT TRANSFER



Examples Of MATLAB Applications

Sample Problem 02: TRIGONOMETRIC IDENTITY

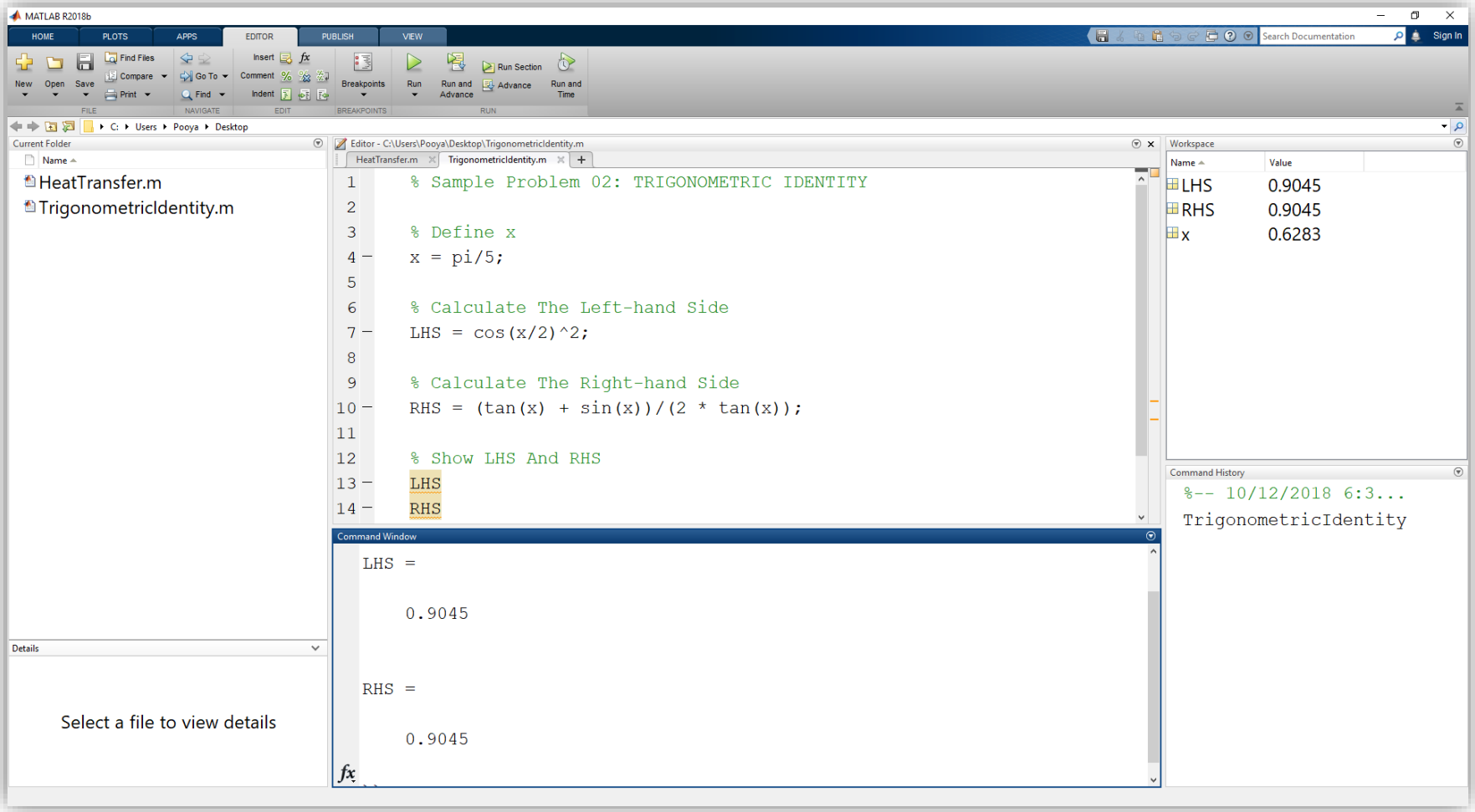
A trigonometric identity is given by:

$$\cos^2\frac{x}{2} = \frac{\tan x + \sin x}{2\tan x}$$

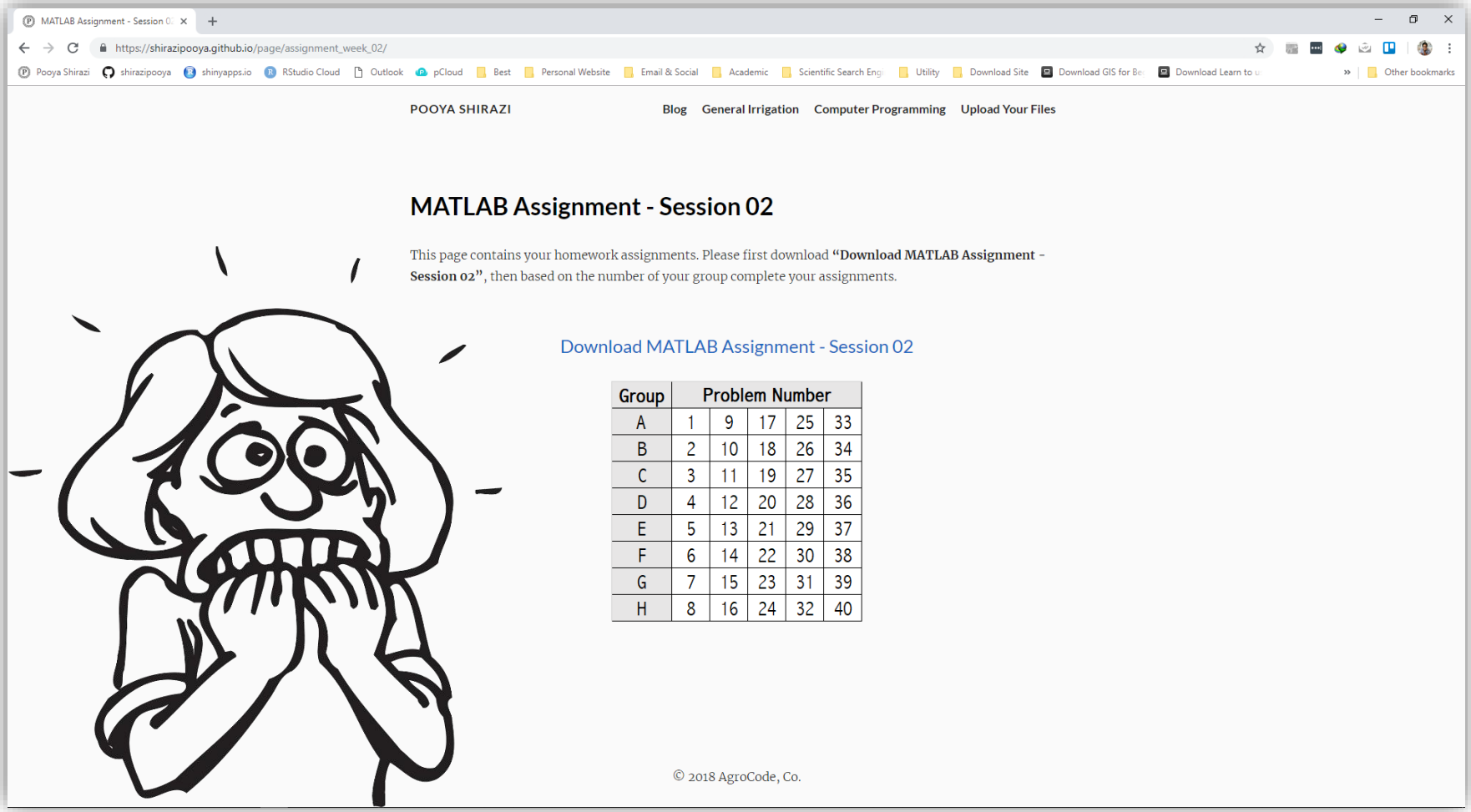
Verify that the identity is correct by calculating each side of the equation, substituting $x = \frac{\pi}{5}$.

Examples Of MATLAB Applications

Sample Problem 02: TRIGONOMETRIC IDENTITY



Assignment



Group	Problem Number				
A	1	9	17	25	33
B	2	10	18	26	34
C	3	11	19	27	35
D	4	12	20	28	36
E	5	13	21	29	37
F	6	14	22	30	38
G	7	15	23	31	39
H	8	16	24	32	40