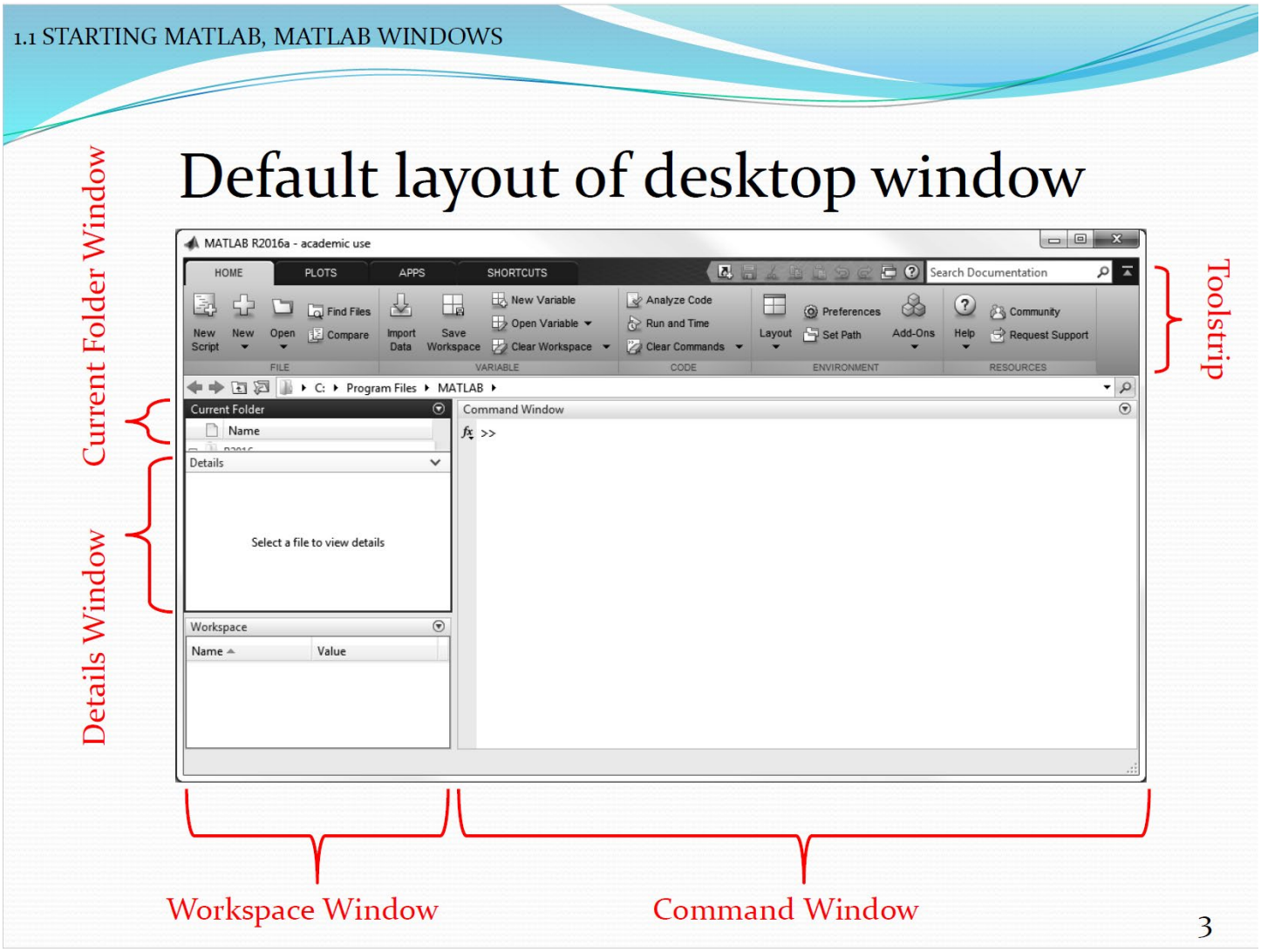
Mrs. Christine Deeb

Chapter 1: Starting with MATLAB

1.1 Starting MATLAB, MATLAB Windows

Open MATLAB on your computer. The MATLAB desktop will pop-up in its default layout.



Let's do the following!

1. Change the layout so that the command window is the only visible window:

   In the "Environment" tab, click "Layout" → click "Command Window Only"

2. Change the view to the two-column layout:

   In the "Environment" tab, click "Layout" → click "Two Column"

3. Change the layout so that the command window is the only visible window.

Table 1: MATLAB Windows and their Purpose

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.

1.2 Working in the Command Window

The **Command Window** is MATLAB's main window. We can use it to:

- Execute commands.
- Open other windows.
- Run programs written by the user.
- Manage the MATLAB software.

Table 2: How to use the Command Window:

1. At prompt (>>), type in MATLAB command.
2. Press ENTER key.
3. MATLAB displays the result in the Command Window followed by a prompt on the next line.

Example 1: In the command window, type: `2 + 6 / 2` then hit enter. What is the exact output?

Table 3: Notes about Working in the Command Window

- Once you press ENTER, MATLAB will execute only the last command.
- You can type several commands in the same line by putting commas between the commands.
- If a command is too long to fit on one line, you can continue to the next line by typing ellipsis (...) and pressing ENTER.
- The ← and → keys move the cursor left and right, respectively.
- The ↑ and ↓ recall preceding commands.
- A semicolon (;) at the end of a command suppresses the output.
- A percent sign (%) at the beginning of a line causes the command to be treated as a comment.
- `clc` clears the command window display.

Example 2: Let's say we were trying to calculate (2+6) / 2, but we forgot the parentheses the first time. Place your cursor a new line.

- Click the up arrow on your keyboard (↑).
- Locate the command you want to edit by clicking the up arrow as many times as necessary.
  - If you hit enter, it will re-execute the command instantly, not giving you a chance to edit.
  - If you hit the "←" or "→" key on your keyboard, the command will be there for you to edit.
- Add the parentheses around the `2 + 6`, then hit enter. Did the output change? What is it now?

Example 3: Let's see what the following commands do!

- Type: `clc` then hit enter. What happens?
- Type: `%2 + 3` then hit enter. What happens?
- Type: `2+3, 2*3; 2^3`. What is the output? Which commands were displayed, if any?
- Type: `2+3, 2*3, 2^3, ...`(hit enter) `2^4` then hit enter. Notice that the "..." allows you to continue your command on the next line.

Note: We rarely work directly in the command window because it is not a file that you can save and come back to edit later.

1.3 Arithmetic Operations with Scalars

The simplest way to use MATLAB is as a calculator.

Table 4: Symbols of Arithmetic Operation

Operation	Symbol	Example
Addition	+	$5 + 3$
Subtraction	−	$5 - 3$
Multiplication	*	$5 * 3$
Right Division	/	$5/3$
Left Division	\	$5\backslash 3 = 3/5$
Exponentiation	^	5^3 means 5^3

Table 5: MATLAB Order of Operations (PEMDAS)

Precedence	Operation
First	Parentheses are executed (innermost first).
Second	Exponents are executed.
Third	Multiplication and division are executed in order from left to right
Fourth	Addition and subtraction are executed in order from left to right.

Example 4: Write one line in the command window to calculate $\left(5-\frac{19}{7}+2.5^3\right)^2$:

Solution:
```
>> (5 - 19/7 + 2.5^3)^2
    ans=320.7937
```

1.4 Display Formats

We can control the display of numbers using the `format` command.

Table 6: Display Formats

Command	Description	Example
`format short`	Fixed-decimal format with 4 digits after the decimal point.	`31.4286`
`format long`	Fixed-decimal format with 15 digits after the decimal point.	`31.428571428571427`
`format short e`	Scientific notation with 4 digits after the decimal point.	`3.1429e+01`
`format long e`	Scientific notation with 15 digits after the decimal point.	`3.142857142857143e+01`
`format short g`	Fixed-decimal or scientific notation, whichever is more compact, with a ***total of*** 5 digits	`31.429`
`format long g`	Fixed-decimal or scientific notation, whichever is more compact, with a ***total of*** 15 digits	`31.4285714285714`
`format bank`	2 digits after the decimal point.	`31.43`
`format rational`	Ratio of small integers.	`220/7`

`format compact`	Suppress excess blank lines to show more output on a single screen.	`>> 220/7`
		`ans =`
		`220/7`

`format loose`	Add blank lines to make output more readable.	`>> 220/7`
		
		`ans =`
		
		`220/7`

Table 7: Additional Notes about Format

- The default format for numerical values is "short." The default format for the space is "loose."
- The format only affects ***display*** of numbers. MATALB always computes and saves numbers in full precision.
- Once you enter a `format` command, the format stays the same until another `format` command.
- To go back to the default, you may use the command `format default`.

1.5 Elementary Math Built-In Functions

In addition to basic arithmetic, MATLAB includes built-in functions. A built-in function has a name, one or more arguments (i.e., inputs) in parentheses, and produces one or more outputs. For example,

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

- `sqrt` is the name of the function.
- `x` is the argument of the function.
  - The argument can be a number; variable (explained in next section); or an expression involving numbers, variables, or functions.
- `y` is the output of the function.

Table 8: Elementary Math Functions

Function	Description
`sqrt(x)`	Calculates the square root of a real number x: $\sqrt{x}$
`nthroot(x,n)`	Calculates the real nth root of a real number x: $\sqrt[n]{x}$ If x is negative, n must be an odd integer.
`exp(x)`	Calculates the exponential function: e^x.
`abs(x)`	Calculates the absolute value: $\|x\|$.
`log(x)`	Calculates the natural logarithm of the input: $\ln(x)$.
`log10(x)`	Calculates common logarithm (i.e., log base 10): $\log(x)$.
`factorial(x)`	Calculates the factorial of a positive integer x: $x!$

Note: Use `pi` for π.

Example 5: Calculate $\sqrt{87}$ in the command window.

```
>> sqrt(87)
ans =
    9.3274
```

Example 6: Calculate $\sqrt{87}$ to 15 decimal places in the command window.

```
>> format long, sqrt(87)
ans =
   9.327379053088816
```

Example 7: Calculate $(36.1 - 2.25\pi)\left(e^{2.3} + \sqrt[3]{20}\right)$ in the command window.

```
>> (36.1-2.25*pi)*(exp(2.3)+nthroot(20,3))
ans =
  368.3680
```

Table 9: Trigonometric Math Functions

Function	Description
`sin(x)` `sind(x)`	Calculates the sine of an angle x in radians. Calculates the sine of an angle x in degrees.
`cos(x)` `cosd(x)`	Calculates the cosine of an angle x in radians. Calculates the cosine of an angle x in degrees.
`tan(x)` `tand(x)`	Calculates the tangent of an angle x in radians. Calculates the tangent of an angle x in degrees.
`cot(x)` `cotd(x)`	Calculates the cotangent of an angle x in radians. Calculates the cotangent of an angle x in degrees.

Table 10: Additional Trigonometric Functions

Function	Description
`asin(x)` `asind(x)`	Calculates the inverse sine of an angle x in radians. Calculates the inverse sine of an angle x in degrees.
`acos(x)` `acosd(x)`	Calculates the inverse cosine of an angle x in radians. Calculates the inverse cosine of an angle x in degrees.
`atan(x)` `atand(x)`	Calculates the inverse tangent of an angle x in radians. Calculates the inverse tangent of an angle x in degrees.
`acot(x)` `acotd(x)`	Calculates the inverse cotangent of an angle x in radians. Calculates the inverse cotangent of an angle x in degrees.
`sinh(x), cosh(x),` `tanh(x), coth(x)`	Hyperbolic trig functions.

Example 8: Calculate $\frac{\sin(0.2\pi)}{\cos(\pi/6)} + \tan(72°)$ in the command window.

```
>> sin(0.2*pi)/cos(pi/6)+tand(72)
ans =
    3.7564
```

Example 9: Given $\int x^2 \cos x \, dx = 2x \cos x + (x^2 - 2) \sin x$, use MATLAB to calculate the definite integral:

$$\int_{\pi/6}^{\pi/3} x^2 \cos x \, dx$$

```
>> (2*(pi/3)*cos(pi/3)+((pi/3)^2-2)*sin(pi/3))...
-(2*(pi/6)*cos(pi/6)+((pi/6)^2-2)*sin(pi/6))
ans =
    0.2209
```

Table 11: Rounding Functions

Function	Description	Example

`round(x)`	Round to the nearest integer.	`>> round(-20/7)`
		`ans =`
		`-3`

`round(x,n)`	Rounds to n digits.	`>> round(-20/7,2)`
		`ans =`
		`-2.8600`

`fix(x)`	Round towards zero.	`>> fix(-20/7)`
		`ans =`
		`-2`

`ceil(x)`	Round toward infinity.	`>> ceil(-20/7)`
		`ans =`
		`-2`

`floor(x)`	Round towards minus infinity.	`>> floor(-20/7)`
		`ans =`
		`-3`

`rem(x,y)`	Returns the remainder after x is divided by y.	`>> rem(-20,7)`
		`ans =`
		`-6`

`sign(x)`	Returns 1 if $x > 0$; -1 if $x < 0$; and 0 if $x = 0$.	`>> sign(-20/7)`
		`ans =`
		`-1`

1.6 Defining Scalar Variables

A variable is a name made of a letter or a combination of several letters (and digits) that is assigned a numerical value. Once a variable is defined, it can be used in mathematical expressions, functions, and commands.

We use the equals sign (=) as "assign to" or "store in".

Variable_name = A numerical value, or a computable expression

```
>> x=5   creates the variable "x" and stores the value 5 in it
x =
     5   MATLAB acknowledges that it has created "x" and set it to 5

>> x+2
ans =
     7
```

In MATLAB, variables must be defined before we can use them in subsequent commands.

```
>> x=5;
>> x+y
Unrecognized function or variable 'y'.
```

Example 10: Define the variables x and y as $x = 6.5$ and $y = 3.8$, then evaluate $(x^2 + y^2)^{2/3} + \frac{xy}{y-x}$.

Solution:

```
>> x=6.5; y=3.8;
>> Z=(x^2+y^2)^(2/3)+(x*y/(y-x))
Z =
    5.6091
```

Table 12: Rules about Variable Names in MATLAB

A variable name

- Must begin with a letter.
- Can be up to 63 characters long.
- Can contain letters, digits, and the underscore (_) character.
- Cannot contain punctuation characters, such as period, comma, semicolon, etc.
- Cannot contain spaces.
- Cannot be the name of a built-in function, such as `sqrt`, `cos`, `round`, `exp`, etc.
- Cannot be the name of a keyword, such as `pi`, `clear`, `clc`, `for`, `else`, etc. See page 19 of the textbook for a full list!
- Is case sensitive, meaning `abc`, `Abc`, `ABC` are all different.

1.7 Managing Variables

The following are commands that can be used to eliminate variables or to obtain information about variables that have been created.

Table 13: Commands for Managing Variables

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

Note: For the remainder of the semester, we will make a habit of starting most files with `clear, clc`.

- `clear` removes all variables from memory.
- `clc` clears the command window display.

Example 11: Let's try the following commands!

```
>> clear,clc
>> y=sqrt(x)
Unrecognized function or variable 'x'.
>> x=121
x =
   121
>> y=sqrt(x)
y =
    11
>> y=y*2
y =
    22
```

Note that this command has redefined the variable y rather than solving the equation $y = y * 2$. This is what we mean when we say the "=" means "assign to" as opposed to "equal to".

```
>> who

Your variables are:

x  y

>> whos
  Name        Size              Bytes  Class     Attributes

  x           1x1                   8  double
  y           1x1                   8  double
```

To figure out the variable's value from the command window, type the variable name and hit enter.

```
>> x
x =
   121
>> y
y =
    22
```

Example 12: Use MATLAB to calculate the area of a triangle with a base of 5 inches and a height of 12 inches by first defining the variables b and h then using them to find the area A.

Solution:

```
>> clear,clc
>> b=5; h=12; A=0.5*b*h
A =
    30
```

1.8 Script Files

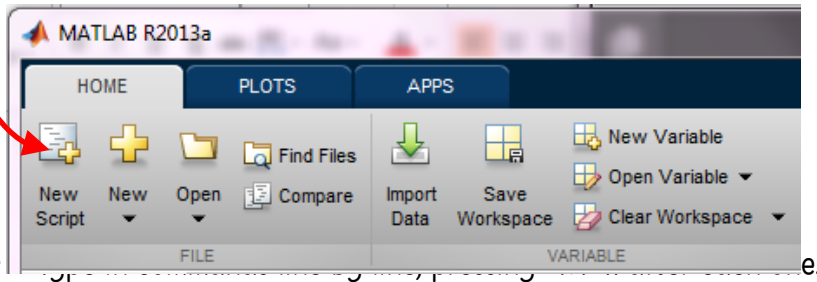
So far all the commands were typed in the Command Window. A better way of executing a series of MATLAB commands is to first create a file with a list of commands (a program), save it, and then run (or execute) the file.

- A **script file**, also called a program, is a sequence of MATLAB commands.
- When a script file runs (is executed), MATLAB executes 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, the output is displayed in the Command Window upon executing the program.
    - Commands in a script file follow the same rules as in the Command Window.
    - To suppress the output of a command, you must end the line with a semi-colon (;).
    - To write comment(s), you must put a percent (%) in the beginning of the line.
    - Using double percents (%%) will allow you to split your file into sections.
- Using a script file is convenient because it can be edited (corrected and/or changed) and executed many times.
- Script files are also called M-files because the extension .m is used when they are saved.

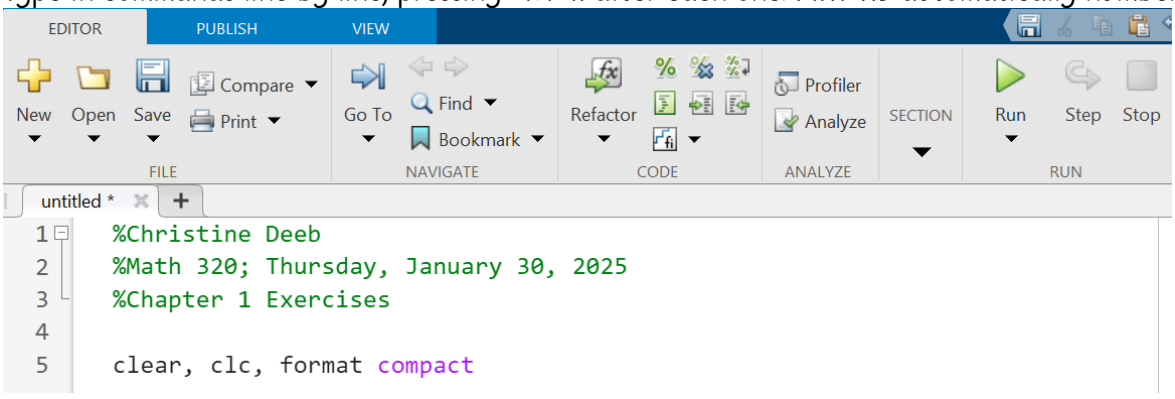
Script file = M-file = Program = Code

Getting Started with Script Files

- Open a new script file by clicking on the New Script icon or typing `edit` in the command window.



- e. MATLAB automatically numbers lines.



- Execute your script file by clicking the green "Run" button in the Toolstrip.

- You must save your file before MATLAB can run the commands.

- Save your script file by clicking on the Save icon. Naming your script file follows many of the same rules as variable names. It must start with a letter, it cannot contain spaces, etc.

- MATLAB will only execute M-files within the current folder or within the current "search path." If you try to execute the script and a MATLAB editor screen pops up, you may need to change the current folder.

- We are going to work with the script file and command window side-by-side like in the figure below. When we run our script file, the output will display in the command window. When you open an M-file on your computer, it will automatically open the MATLAB program.