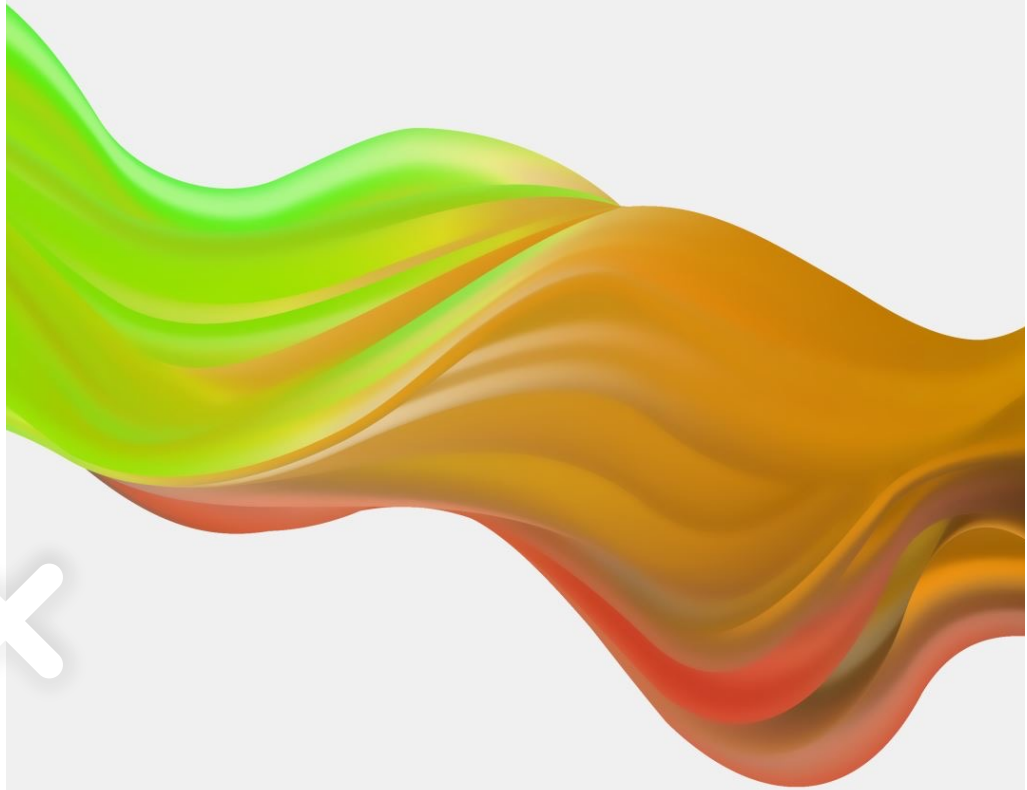
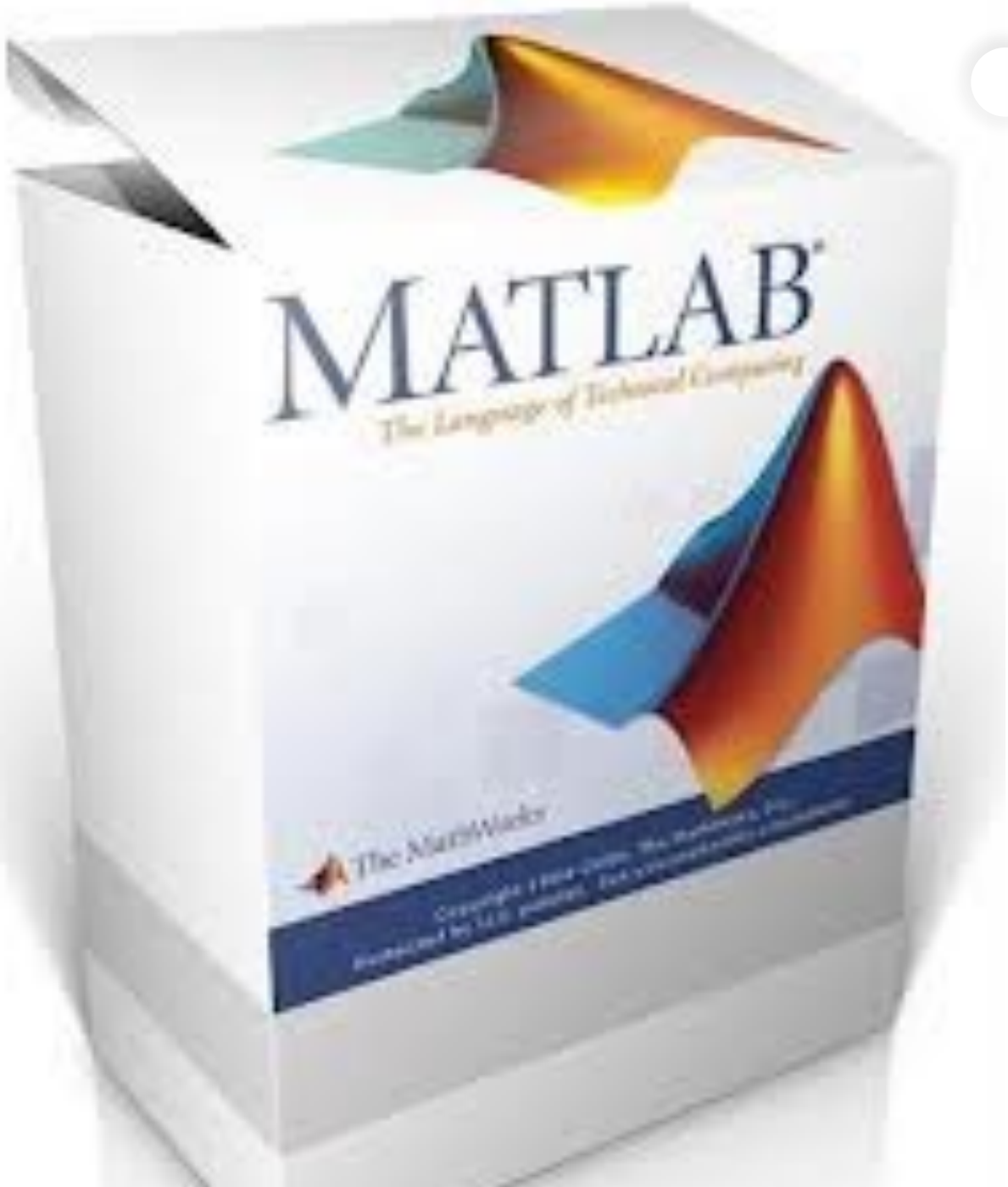
Computer Programming

MENG2020



REVIEW



Objectives

Understand what MATLAB is and why it is widely used in engineering and science

Start the MATLAB program and solve simple problems in the command window

Release Numbers

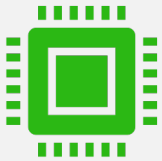
MATLAB is updated regularly

The Mathworks packages their software in groups, called releases, New releases are issued twice a year in the spring and in the fall

Release 2022 a includes	MATLAB 9.12.0	A number of specialized “toolboxes”

About our lecture

What Is MATLAB?



MATLAB is both a powerful computational environment and a programming language that easily handles matrix and complex arithmetic.

It is a large software package that has many advanced features built in, and it has become a standard tool for many working in science or engineering disciplines.

Among other things, it allows easy plotting in both two and three dimensions.

What Is MATLAB?

MATLAB has two different methods for executing commands: interactive mode and batch mode.

In interactive mode, commands are typed (or cut and pasted) into the *command window*.

In batch mode, a series of commands is saved in a text file with a ***.m*** extension. The batch commands in a file are then executed by typing the name of the file at the MATLAB command prompt.

The MATLAB Language

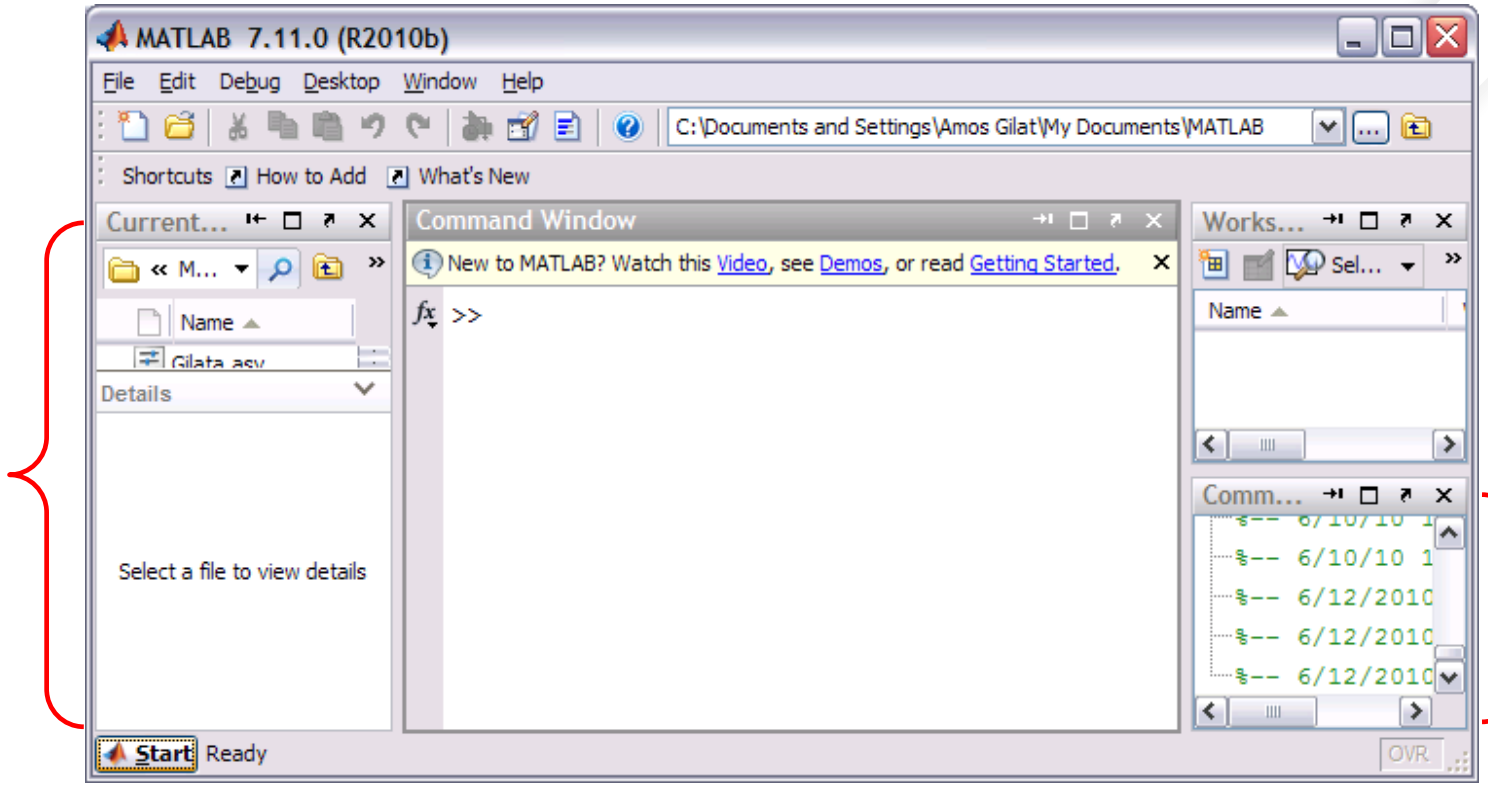
MATLAB is an interpreted language. Commands are executed line by line.

It supports two basic types: characters like 'a' (16 bits) and doubles like 8.25 (64 bits).

Most of the time MATLAB works with arrays or matrices of real numbers or characters.

Other types are also supported: complex, symbolic (if installed), 16-bit and 8-bit integers, etc...

The MATLAB Desktop



Current Folder Window

Workspace Window

Command History Window

Command Window

The Figure Window

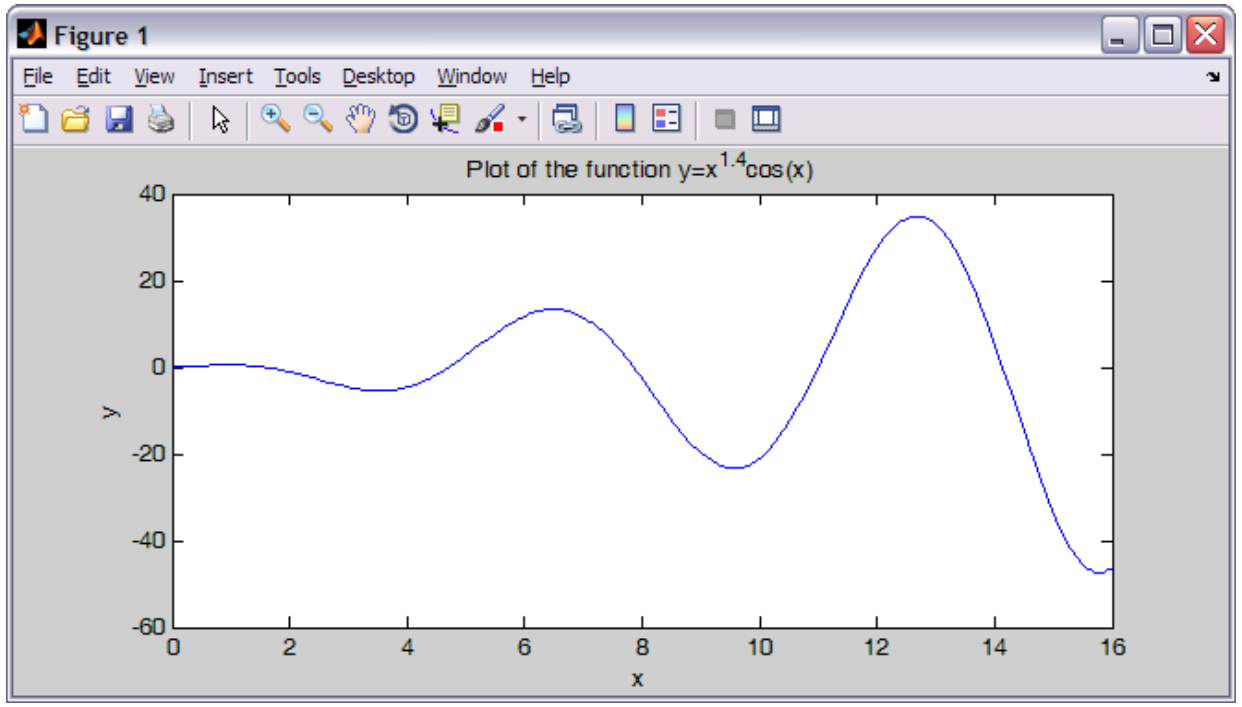
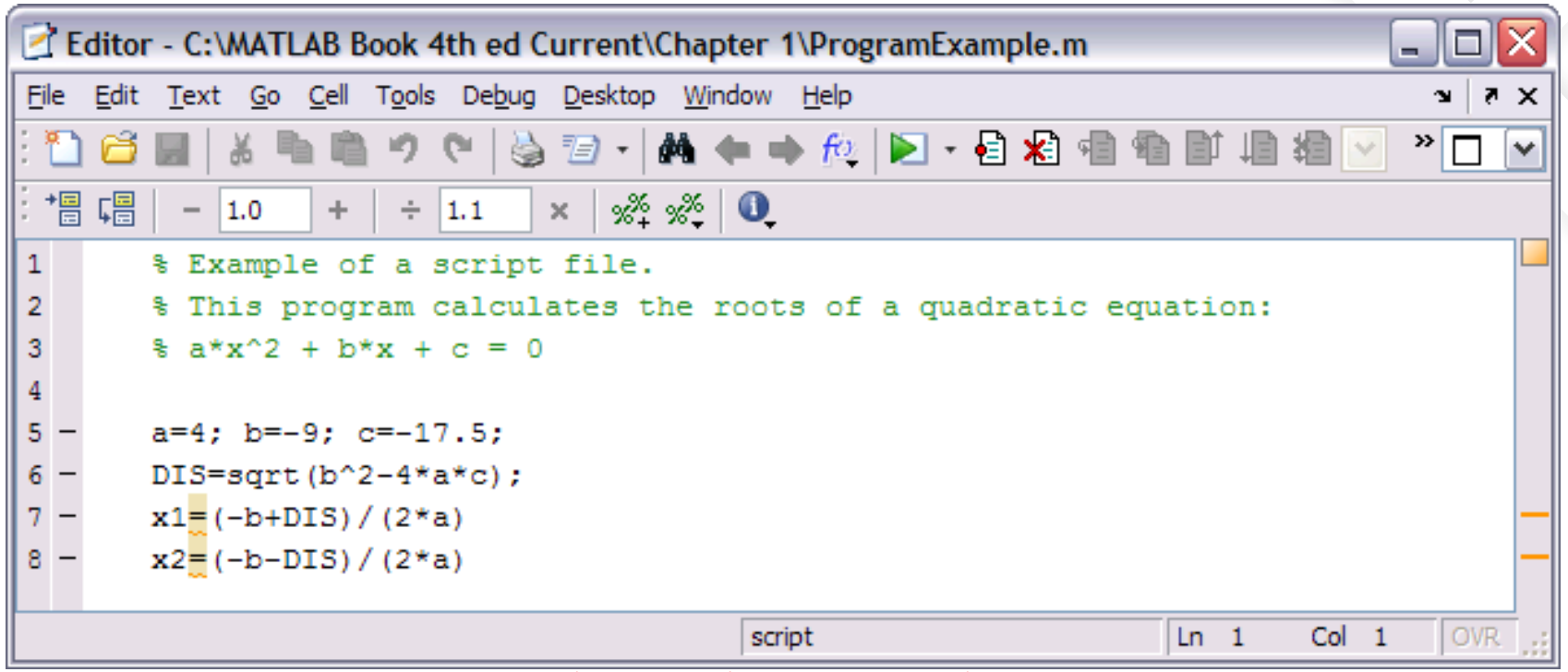


Figure Window

The Figure Window opens automatically after any command that draws a graph.

The Editor Window



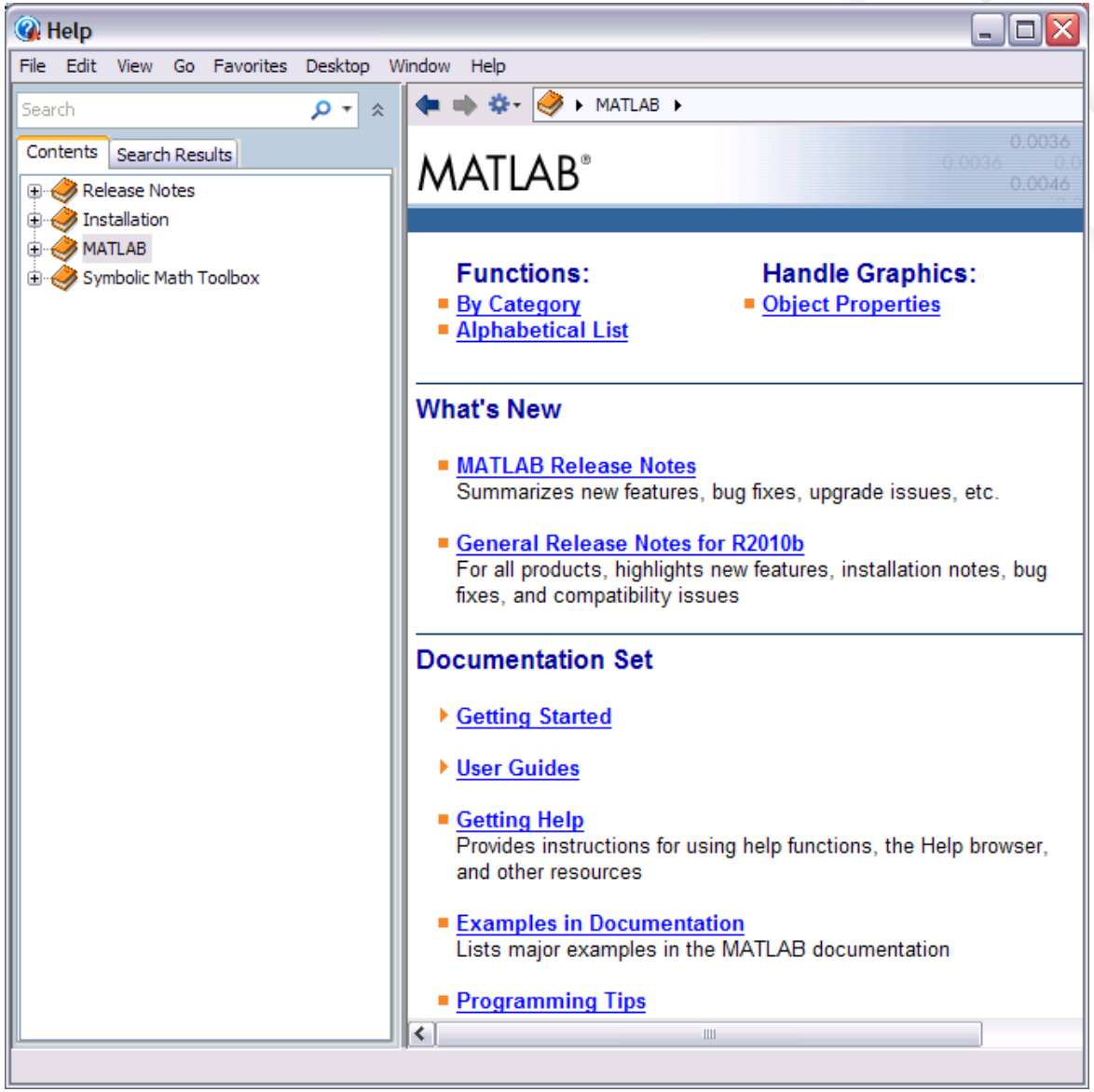
Editor/Debugger Window

Use the editor/debugger window to write and debug MATLAB scripts. Open with the Edit command.

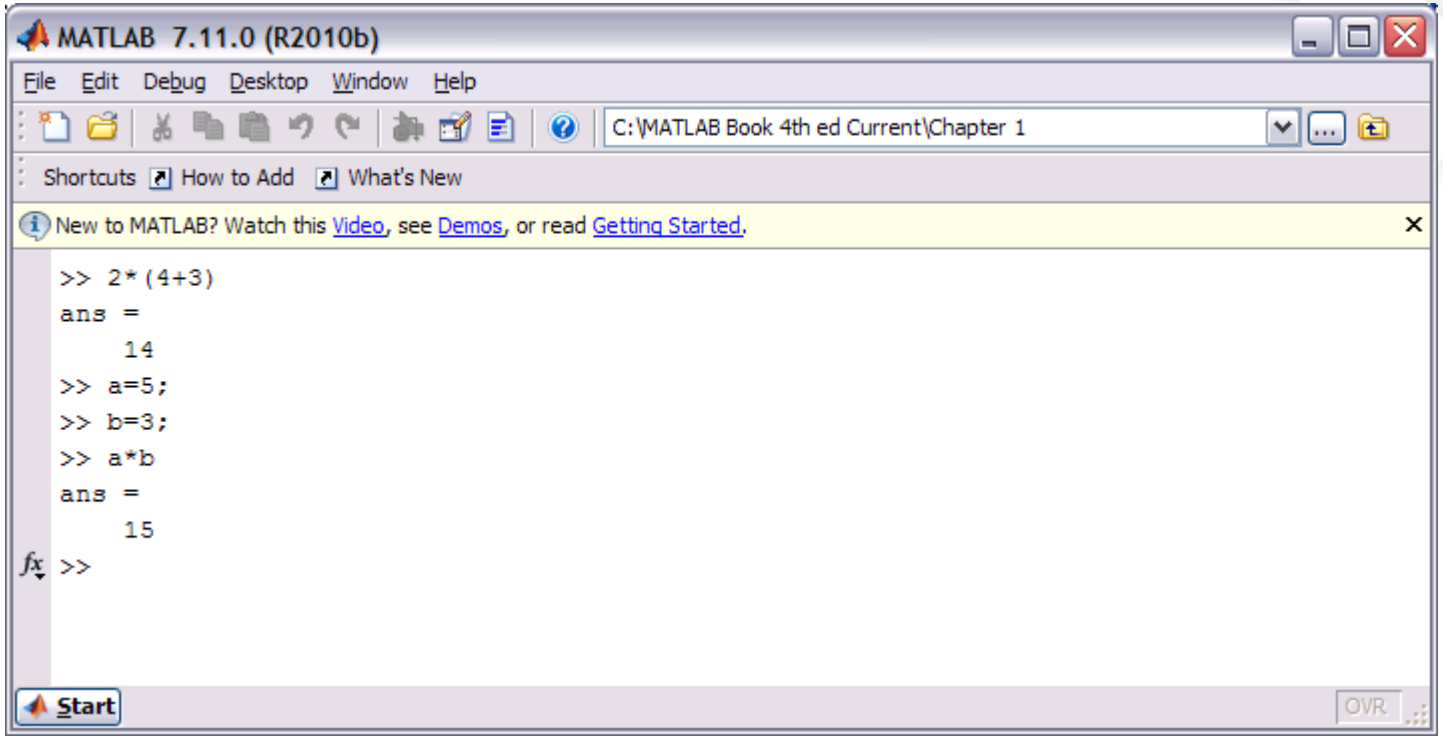
The Help Window

Get the help window from the Help menu in any MATLAB window.

(may vary according to version)



The Command Window



At the prompt (>>), type in a MATLAB command. Press the ENTER key.

MATLAB displays the result in the command window, followed by a prompt.

Repeat the process.

Notes on the Command Window

To start a command, make sure cursor is next to the prompt.

MATLAB won’t respond until you press ENTER. It then executes only the last command.

Commands before the last one may still be visible, but MATLAB doesn’t execute them.

Notes on the Command Window

You can type several commands in the same line by putting a comma between commands. (makes the program harder to read though, so it is not recommended).

If a command is too long to fit on one line, you can continue to the next line by typing ellipsis (3 periods, i.e., ...) and then pressing ENTER.

Navigating the Command Window

When cursor is in the bottom (current) command line:

← key moves the cursor one character to the left.

→ key moves the cursor one character to the right.

↑ key recalls the preceding command

↓ key recalls a previous command if it follows.

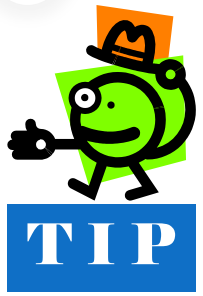
Navigating the Command Window

The PAGE-UP key moves up to previous commands in a window-size at a time.

The PAGE-DOWN key moves down to previous commands in a window-size at a time.

The BACKSPACE (Delete on Mac) key deletes one character to the left of the cursor.

The DELETE (fn-Delete on Mac) key deletes one character to the right of the cursor.



To quickly execute a previous command but with small changes:

Recall the command with the up and down arrow keys.

Use the left and right arrow keys to move to characters to be altered.

Use BACKSPACE (Mac: Delete) or DELETE (Mac: fn-Delete) to remove old characters, then type new characters.

Press ENTER to execute the modified command.

Command Syntax

Semicolon (;)

- When typed at end of a command, it suppresses the output. (only the prompt is displayed at the next line)

Useful for preventing display of large outputs

Used much more in scripts (see Section 1.8 of text book)

Command Syntax

Percent sign (%)

- When typed at beginning of a line, MATLAB treats the line as a *comment* and doesn't execute the line.

Also used much more in scripts.

The clc Command

Clears the command window display.

The up and down arrows still bring back the previous commands.

The Command History Window

Shows the previous commands, including the ones from previous MATLAB sessions.

Double-clicking on a command puts it in the command window and executes it.

You can drag a command to the command window, make changes in it, then execute it.

Arithmetic Operations

In this lesson we will only discuss arithmetic with *scalars* (single numbers).

We can do arithmetic directly on numbers (like a calculator).

Or, we can store numbers in variables (containers).

Arithmetic Operators

Symbols for arithmetic operations 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 to the power of 3 = 125)

Note: Left division is rarely used with scalars.

Arithmetic Operations Precedence Order

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

Arithmetic Operations Precedence Rules

The MATLAB precedence order is:

The same as most calculators.

The same as doing arithmetic by hand.

For multiple operations of same precedence, MATLAB goes left to right.

You can change the order by using parentheses.

Order of Operation

$5*(3+6) = 45$

$5*3+6 = 21$

White space does not matter!!!

$5*3 + 6 = 21$

Adding a space around + and – signs makes the expression more readable

Parentheses

- Use only ( )
- { } and [ ] mean something different
- MATLAB does not assume operators

5 * (3+4) not 5(3+4)

Compute from left to right

$$5*6/6*5 = 25$$

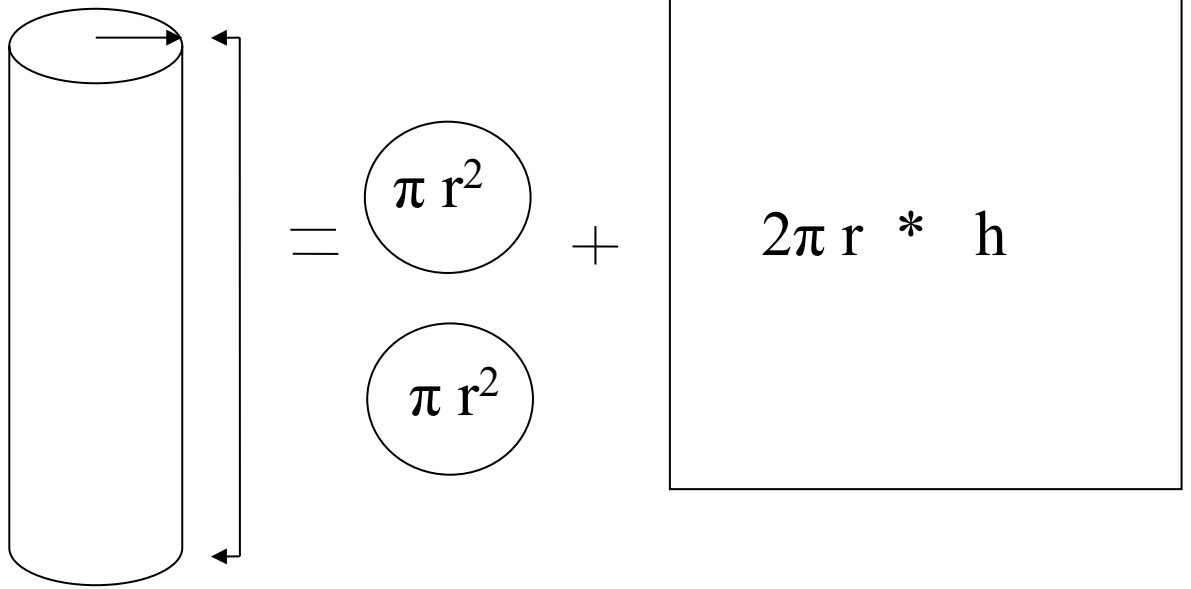
$$5*6/(6*5) = 1$$

EXAMPLE

Find the surface area of a cylinder

r = radius *h = height*

r = 5 *h = 10*



$$SA = 2\pi r^2 + 2\pi rh = 2\pi r(r + h)$$

Using MATLAB as a Calculator

Type in a mathematical expression.

Press the **Enter** key.

MATLAB displays the answer in the command window as ans = followed by the result.
Example:

```
>> 2 * 3
        ans = 6
```

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

Formatting the Display

You can control the display of numbers with the **format** command.

Once a command is entered, the format stays the same until another **format** command is issued.

The default format is fixed point with four digits to the right of the decimal point. *fixed-point* means decimal point always between one's-digit and one-tenth's digit.

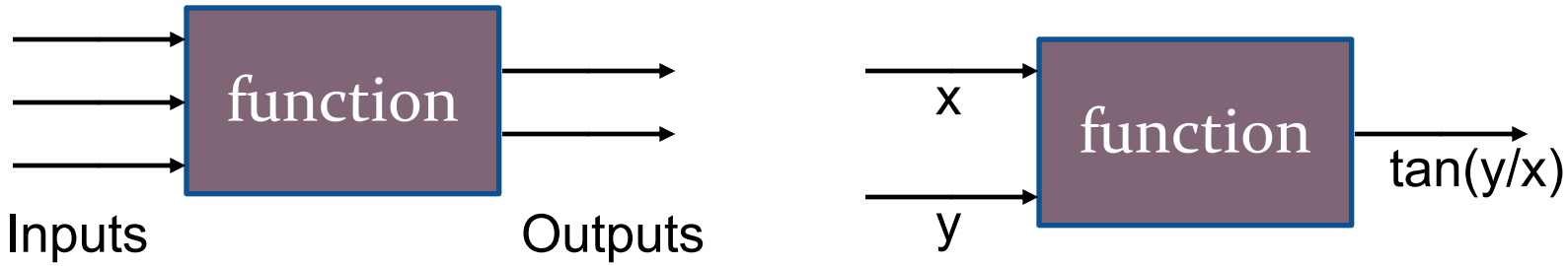
The **format** command only affects the display of numbers. MATLAB always computes and saves numbers in full precision.

MATLAB Display Formats

Matlab command	Display	Comments
Format short	3.1416	default
Format long	3.14159265358979	16 digit
Format short e	3.1416e+000	5 digit plus exponent
Format long e	3.14159265358979e+000	16 digit plus exponent
Format bank	3.14	2 decimal places
Format +	+	positive
Format rat	355/113	ratio
Format hex	400921fb54442d18	hexadecimal

MATLAB 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.



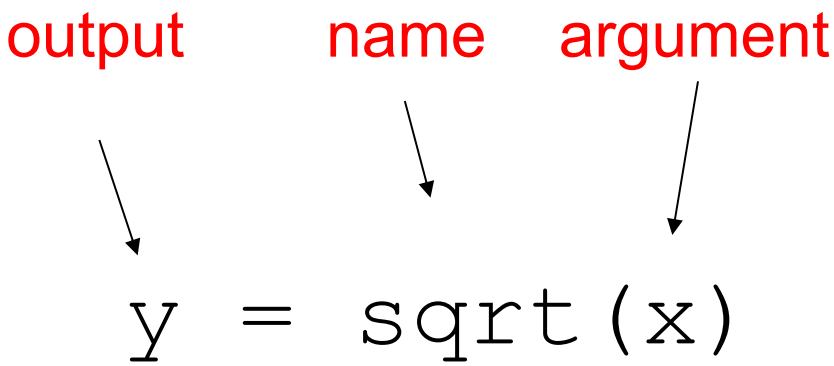
MATLAB Built-In Functions

A function has a name.

It can have zero or more *arguments* (inputs).

It can produce zero or more outputs.

This is a typical use of such functions:



MATLAB Built-In Functions

A function's arguments can be numbers, variables (containers or receptacles), or expressions involving numbers, variables, or other functions.

Examples:

```
sqrt(64)   argument is the number 64

sqrt(a)    argument is the variable named a

atan(y/sqrt(3^2+y^2))
argument is an expression that contains numbers (2
and 3), a variable (y), and a function call (sqrt)
```

Elementary Math Functions

sqrt(x) – square root

nthroot(x,n) – *n*th real root

exp(x) – *e to the power of x*

abs(x) – absolute value

log(x) – natural log (base *e*)

log10(x) – log base 10

factorial(x) – *x!*

See text book for more details.

Elementary Trig 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 text book for more details.

$$\text{angle in degrees} = \text{angle in radians} \cdot \frac{180^\circ}{\pi}$$

$$1 \text{ rad} = 1 \cdot \frac{180^\circ}{\pi} \approx 57.2958^\circ$$

Rounding Functions

`round(x)` – round to nearest integer

`fix(x)` – round toward zero (transforms a double into an integer)

`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 text book for more details.

What Is a Variable?

What Is a Variable?

A *variable* is a name that is assigned a value. Think of it as a container or receptacle that has a name and contains some value.

Once assigned with a meaningful value, you can use variables in expressions, functions, and MATLAB statements and commands.

Assigning Values to Variables

= (the equal 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

Assigning Values to Variables

= (the equal 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

```
Q?

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

Assigning Values to Variables

= (the equal 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

```
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
a =

3
```

Assignment Operator

Example:

```
>> a = 3
```

Create a variable and store a number in it

```
a =

3
```

```
>> b = 10 * a + 5
```

Create a variable and store the value of an expression made up of a variable, numbers, and addition and multiplication

```
b =

35
```

How Assignment Works

Think of **=** as meaning “assign to” or “store in” but it is not meaning “equals”!

Why? A left arrow would make more sense but that symbol does not exist on keyboards! So we use = instead.

x = *x* + *6* has no meaning in math because it implies that *0* = *6.* But x = x + 6 is perfectly fine in MATLAB because it means “take whatever is in x (the x on the right side), add 6 to that and store the result back into x (the x on the left side which really is the same container (variable) named x)”.

How Assignment Works

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

But x = x + 6 is perfectly fine in MATLAB because it means “take whatever is in x (the x on the right side), add 6 to that and store the result back into x (the x on the left side which really is the same container (variable) named x)”.

Assignment Operator Examples

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 the value of x 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 the value of x is 18

Assignment Operator Warning

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

```
>> x = 3;

>> x + 2

ans =

5
```

Q?

```
>> x + y
```

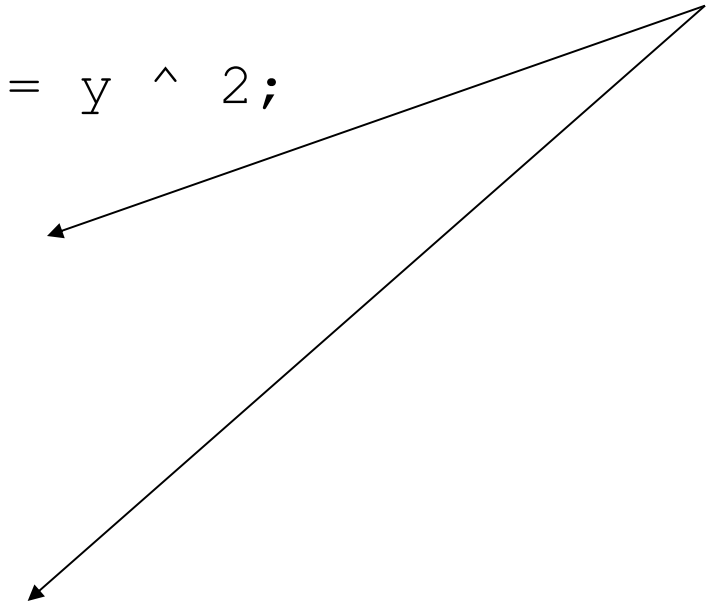
Assignment Operator Warning


```
>> x + y % assume y is undefined

??? Undefined function or
variable 'y'
```

This is a comment (follows %, ignored by MATLAB)

The Value of a Variable

```
>> x = 3;

>> y = 10 * x;

>> z = y ^ 2;

>> y

y =

30

>> z

z =

900
```



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

Multiple Assignments

You can do multiple assignments on one line by separating them with a comma or semicolon.

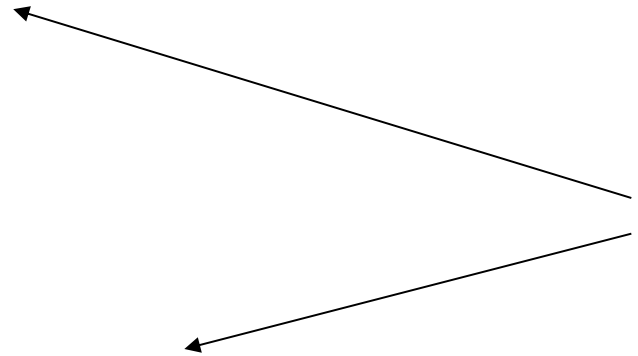
If you use a semicolon, the resulting value is not displayed but the assignment happens nonetheless.

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

a =

12

C =

18
```

Changing the Value of a Variable

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

```
>> ABB=72;

>> ABB=9;

>> ABB

ABB =

9
```



Only the last value is kept!
One value at a time in a scalar variable. The original value of 72 is gone!

r=5

area=pi*r^2

r=10

surface_area=4*pi*r^2

r=2

volume=4/3*pi*r^3

Workspace Area

The **workspace** area shows all the variables and their content.

We can modify the variables on the fly using the variable editor (double-click on a variable to access).

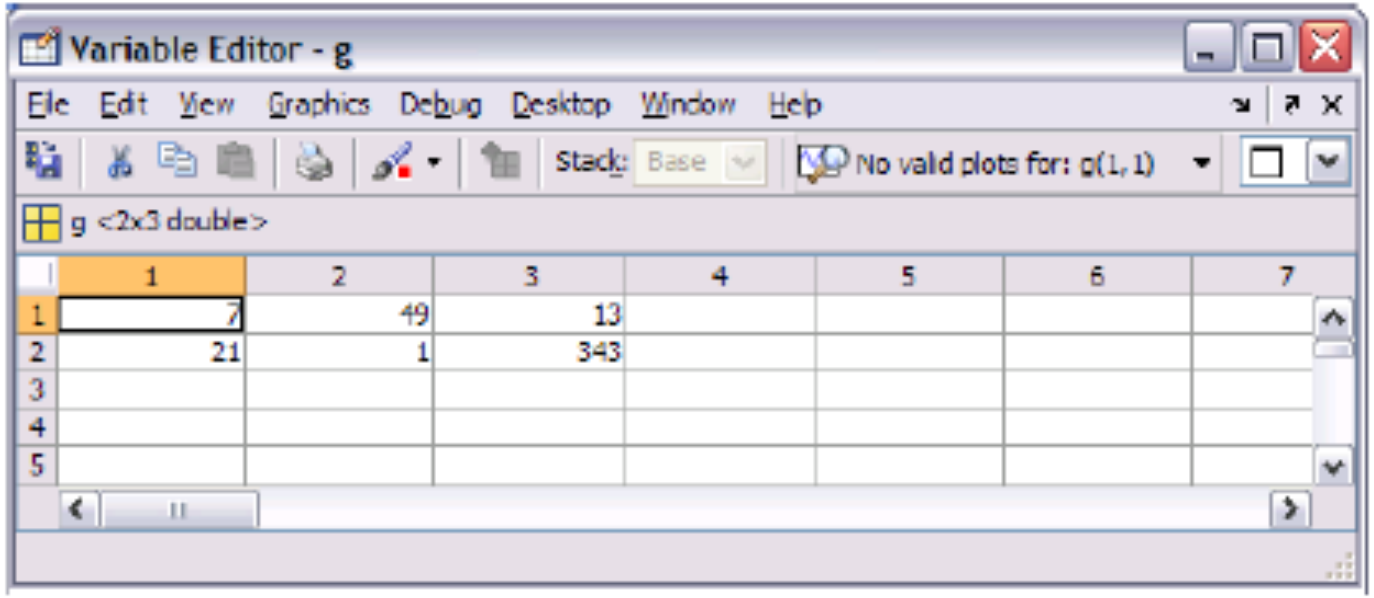


Figure 4-2: The Variable Editor Window.

Variables and Functions

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

An undefined variable is like an empty container. It is useless.

Q?

```
>> sqrt( x )
```

Q?

```
>> sqrt( x ) % assume x undefined

??? Undefined function or variable 'x'

>> x = 144;

>> sqrt( x )

x =

12
```

```
%a)Volume
r=10; %cm
length=15; %cm
d=1; % cm
% Find the volume of each sphere
volume_sphere=4/3*pi*r^3;
% Find the volume of the bar
volume_bar=pi*(d/2)^2*length;
% Combine the components to get the total volume
total_volume=2*volume_sphere +volume_bar
```

Name	Value
area	78.5398
d	1
f_c	1
F_c	1.0343
fs	10
length	15
n	6
r	10
surface_area	1.2566e+03
T	0.1000
total_volume	8.3894e+03
volume	33.5103
volume_bar	11.7810
volume_sphere	4.1888e+03

Variable Name Rules

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.

Variable Name Rules

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,

SPEED, Speed, speed, and SpEEd are four different variable names!

Variable Name Rules

A variable name cannot contain a space. *Speed of light* as a variable name is invalid. Two common alternatives:

Use an underscore in place of a space, ex: *speed_of_light*.

Capitalize the first letter of every word, ex: SpeedOfLight.

Variable Name Rules
MATLAB Keywords

A *keyword* or *reserved word* 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
```

They appear in blue when typed in the editor/debugger window (type *iskeyword* to see the list).

They can’t be used as variable names!

Predefined Variables

There are a few predefined variables in MATLAB that you will use often.

Both **i** and **j** are defined to be the square root of -1

pi is defined as the usual 3.1416.

inf is used for infinity

eps is used for the smallest difference between two numbers or 2^(-52) (epsilon).

These variables can be assigned other values but it is not recommended.

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.

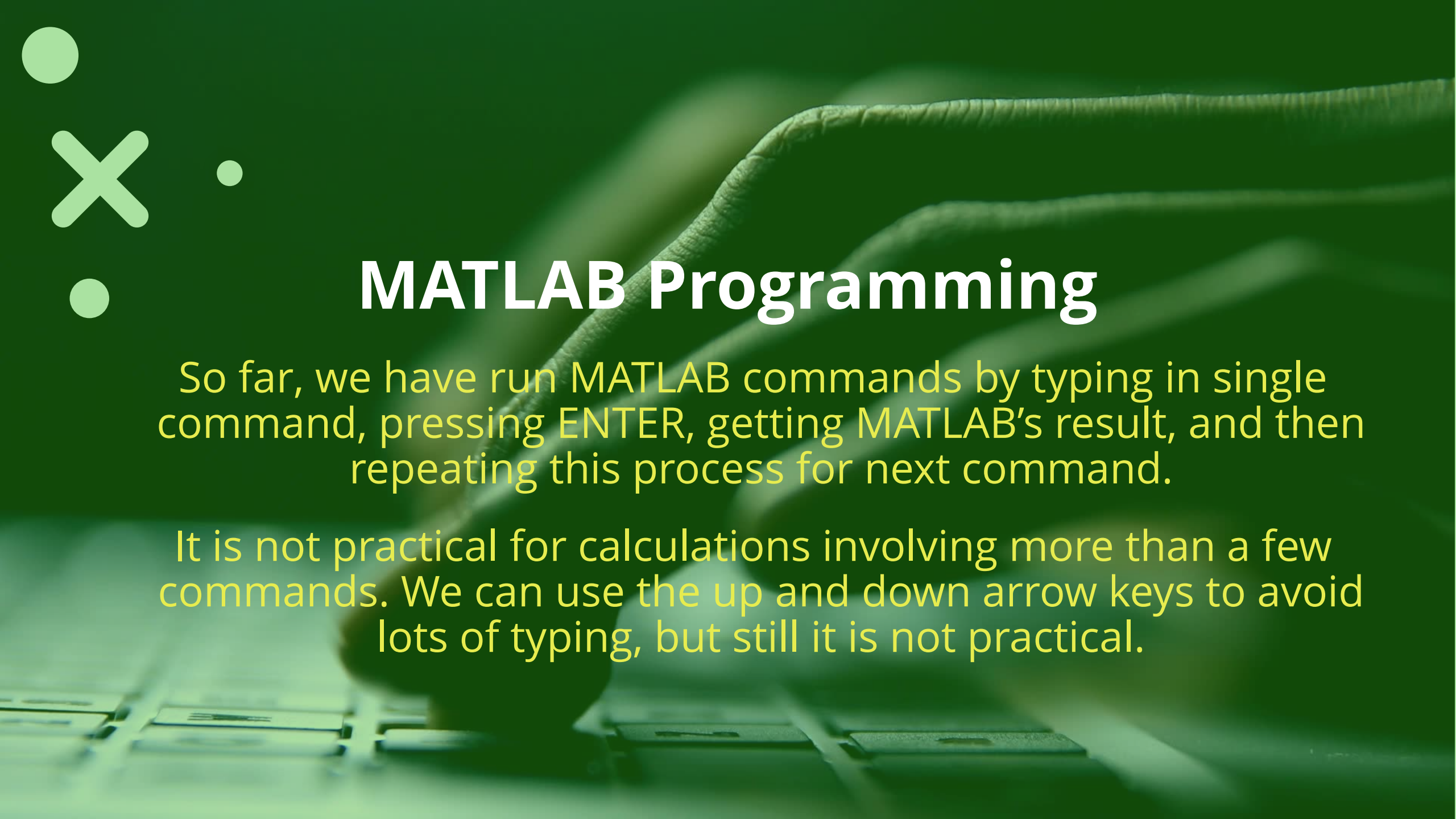
The clear

- You can remove a variable from MATLAB's memory by clearing it (clear x).
- Clearing a predefined variable will restore its default value (clear pi).
- clear by itself removes all variables.

The whos Commands

- **whos** reports the names and dimensions of all variables currently in MATLAB's memory.
- **who** only reports the variable names.

```
>> whos
Name      Size            Bytes  Class
  x         1x1                 8  double array
Grand total is 1 element using 8 bytes
```



MATLAB Programming

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

It is not practical for calculations involving more than a few commands. We can use the up and down arrow keys to avoid lots of typing, but still it is not practical.

MATLAB Programming

There is a better solution:

We save all the commands in a file, and then with one command in the command window, we tell MATLAB to run all the commands in the file.

We will use script files to do this. That is called *programming*.

MATLAB Programming

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 (like the assignment of a value to a variable without a semicolon at the end), the file displays the output in the command window. All similar commands work the same way.

MATLAB Programming

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.

MATLAB Programming

Use the Editor/Debugger Window to work with script files

Can open window and create file two ways

File|New|Script – this means select the File menu, then the New menu item, then the Script menu item	In the Command Window, type **edit** and then press ENTER

Creating and saving a script file

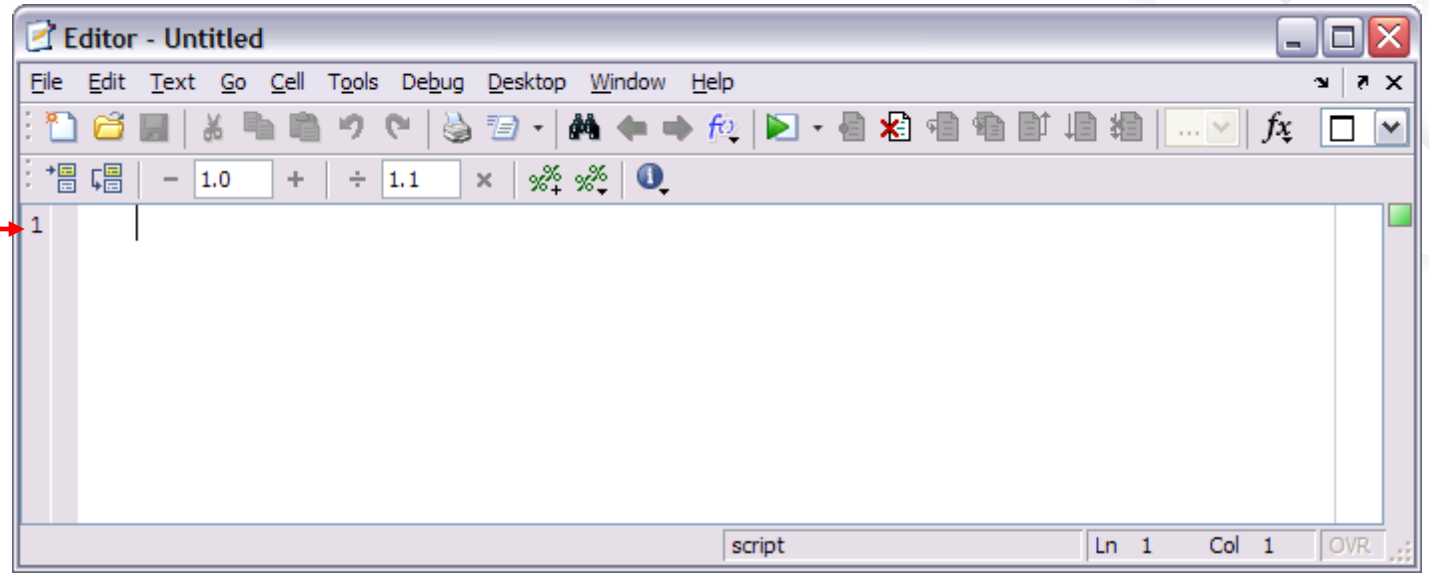


Figure 1-6: The Editor/Debugger Window

Type in the commands line-by-line, pressing ENTER after each one

MATLAB automatically numbers the lines.

Creating and saving a script file

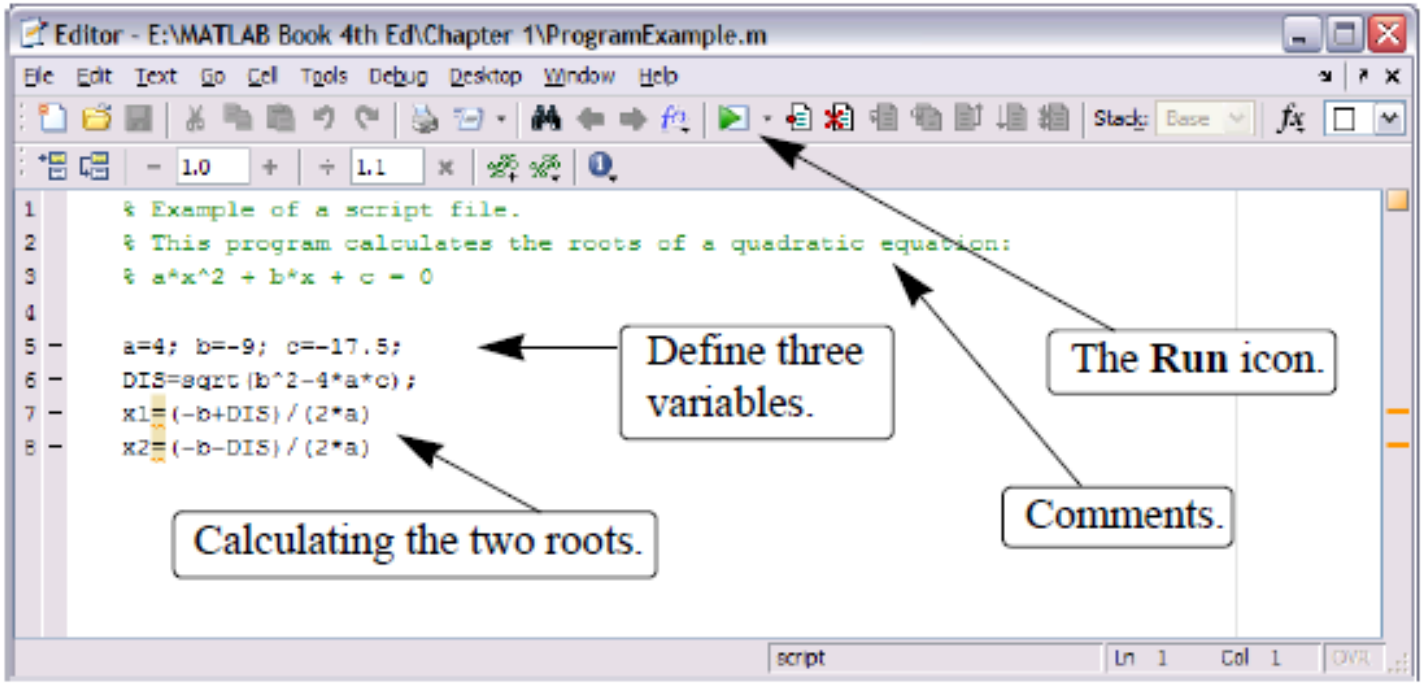


Figure 1-7: A program typed in the Editor/Debugger Window.

Comment lines are lines that start with percent sign (%).

It is common for first few lines to be comments and to briefly explain what commands in file do.

Editor/Debugger Window shows the comment lines in green.

Creating and saving a script file

Before MATLAB can run commands in a file, you must save the file.

If you haven't named file yet, select
File|Save As... , type in a name, press Save.

If you've already named and saved file, just use File|Save.

If you don't add an extension (.xxx) to the file name, MATLAB adds ".m" (if you name your file *fred*, the file will appear as *fred.m* on your disk).

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.

Executing a script file

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) or by typing the file name in the command window and pressing ENTER (if you don't see anything, try moving the command window out of the way).

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).

To reopen an existing file to make changes just use the edit command.

If the file is named *fred.m,* just enter *edit fred*.

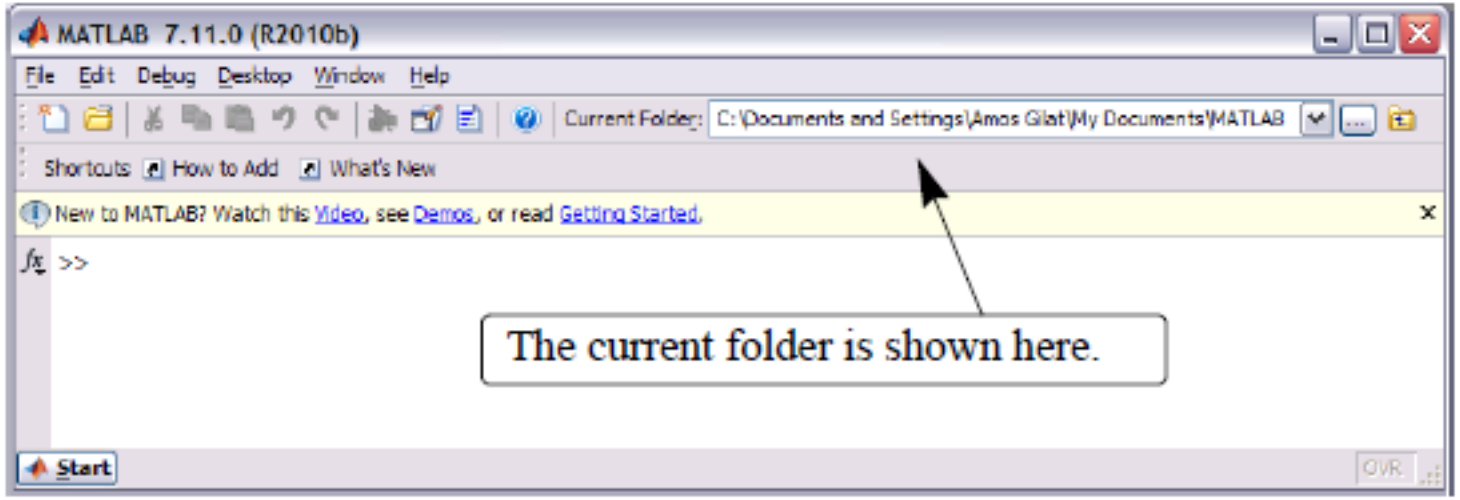


Figure 1-8: The Current folder field in the Command Window.

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

Can see current folder in desktop toolbar in Command Window (Figure 1-8).

Can also display current folder by issuing MATLAB command **`pwd`**.

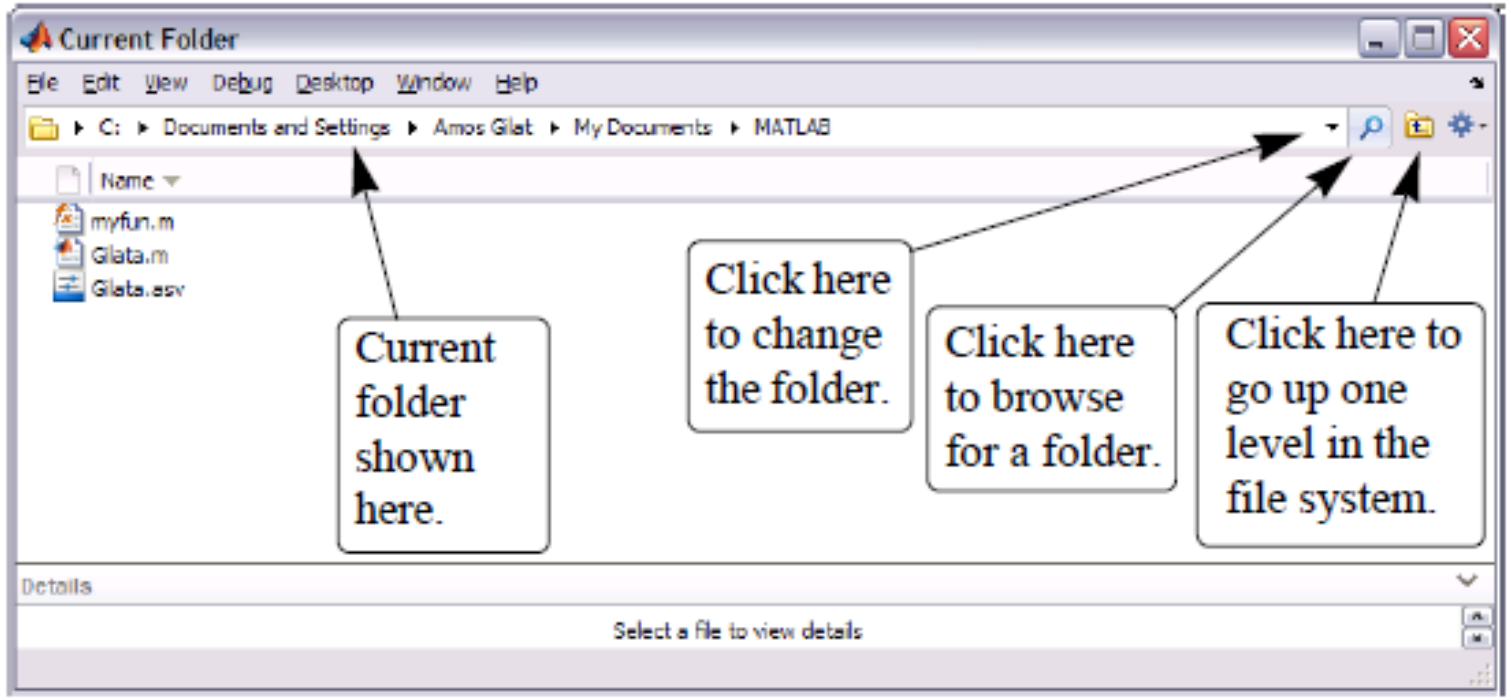


Figure 1-10: The Current Folder Window.

You can change the current folder in Current Folder Window (see Figure 1-10).

Get Current Folder Window from Desktop|Current Folder.

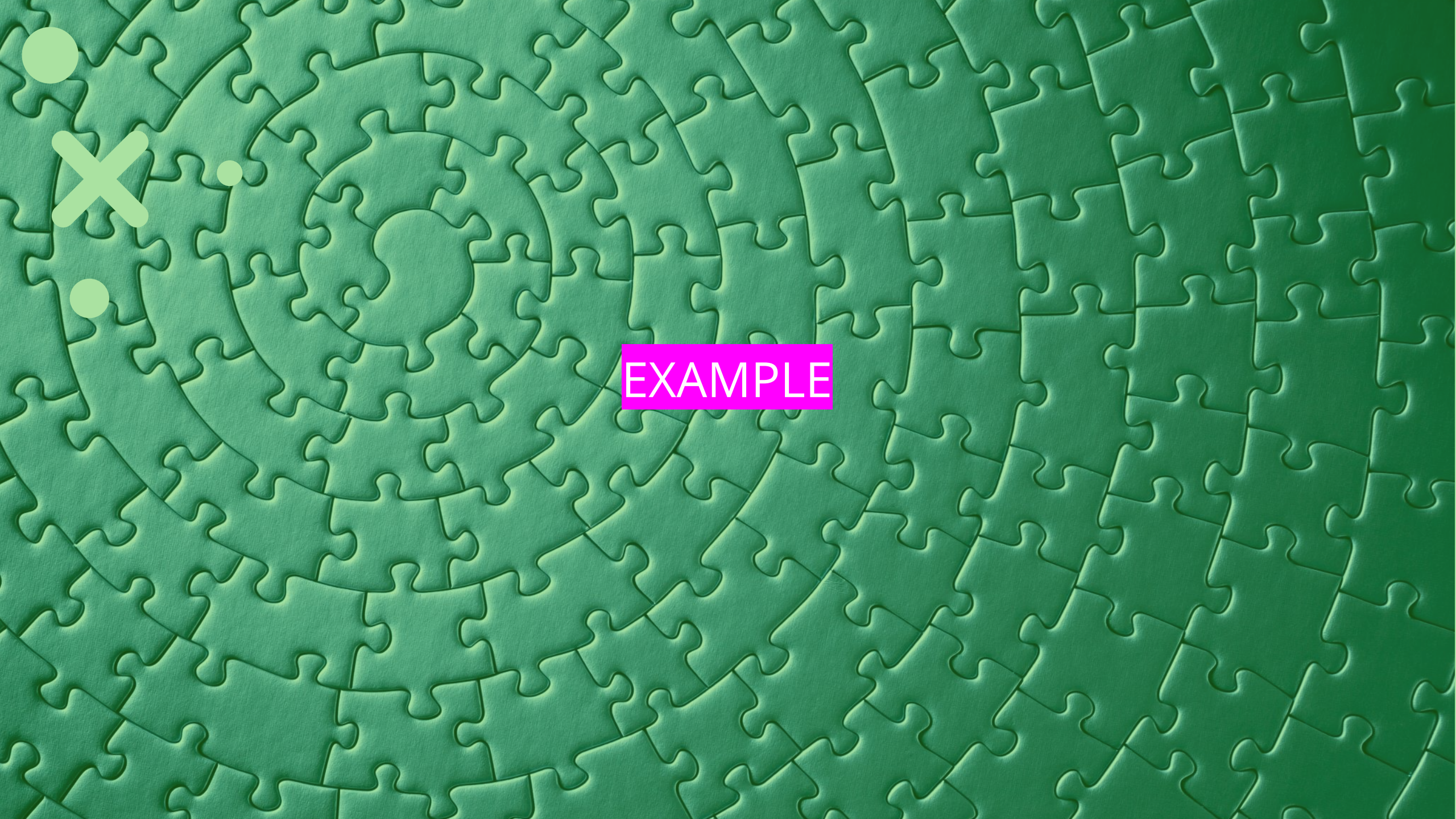
Current Folder

You can change the 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 'F:\slides\Chapter 1'


EXAMPLE

cone.m

File Edit Text Cell Tools Debug Desktop Window Help

```
1    %Sample of well-commented M-File for cone.m
2    %Created by: Joe Bushey
3    %ES100-01
4    %Last modified: 1/11/07
5
6 -  m = 4    %radius of cone
7 -  h = 12   %height of cone
8 -  v = (pi.*r.^2.*h)./3     %Calculate volume of cone.
```

script Ln 1 Col 1 OVR

Now execute (or run) the program by pushing F5, or by typing on the command line

>> cone <Enter>

or by clicking the run button. (Note that the **run button** looks like a page with a green triangle in it. It can be found on the toolbar of the edit window.)

If you entered the code as written on the previous slide you will get an error!

What went wrong?

Repair your program (Change r = 4.), save it, and run it again.

MATLAB can be used like a hand calculator to do arithmetic.

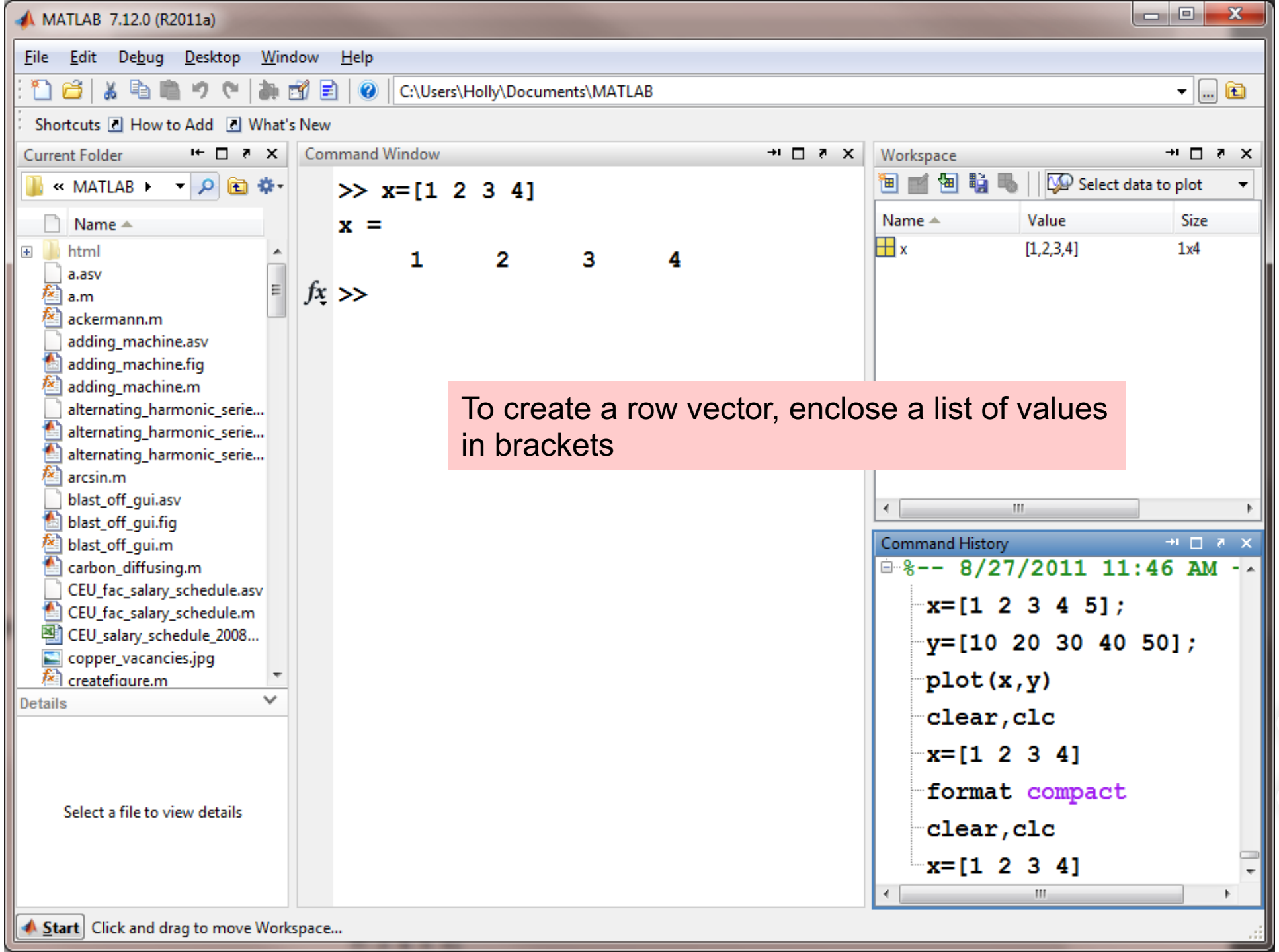
You can define (or assign) variables with numbers and expressions to do calculations as illustrated by the volume-of-cone example.

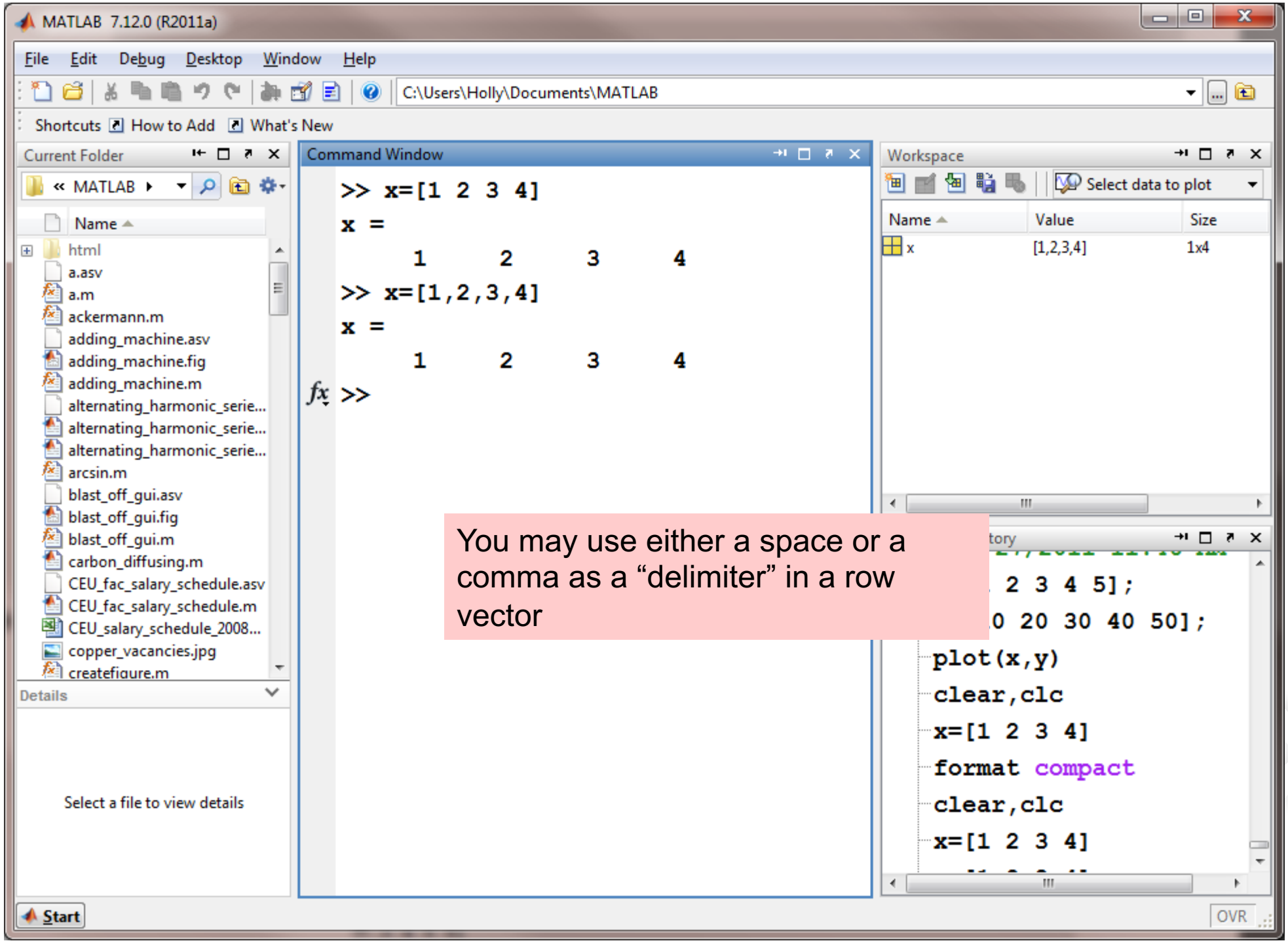
The advantage of saving programs as *M-files* is that you open it, make changes and/or execute it again without having to type it all over again.

This concludes our overview of MATLAB and a taste of things to come!

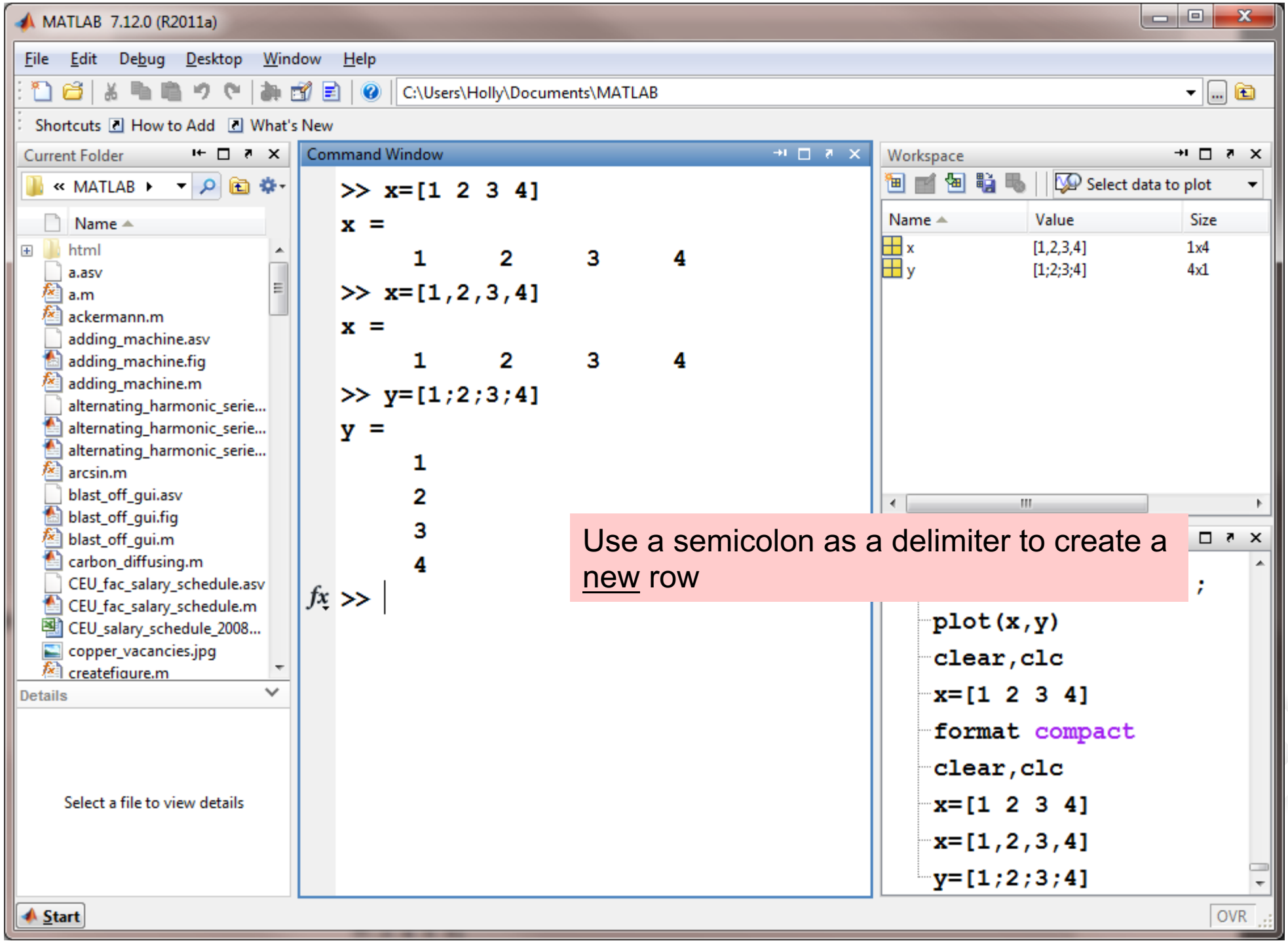
Array Operations

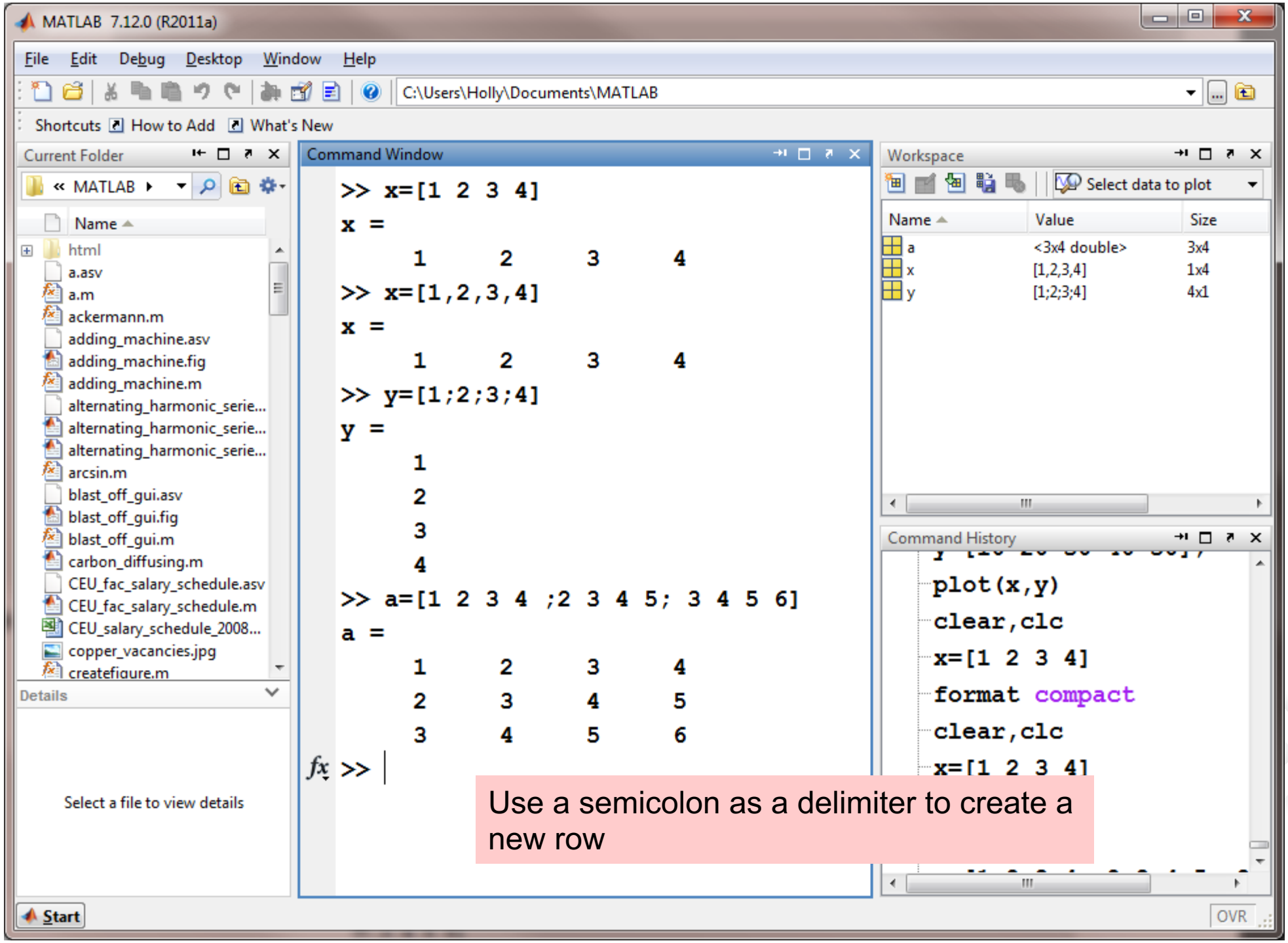
- Using MATLAB as a glorified calculator is OK, but its real strength is in matrix manipulations

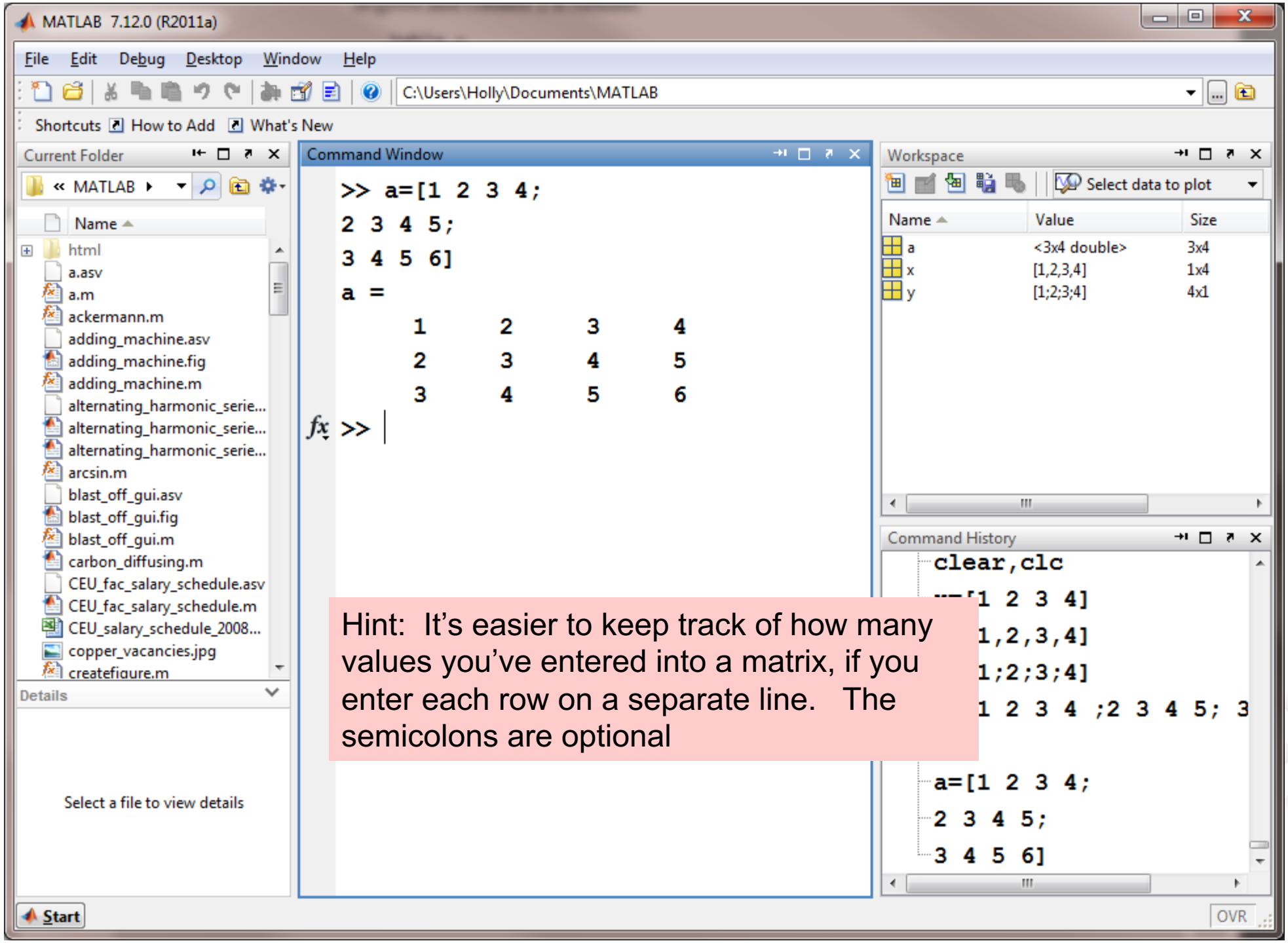

MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> x=[1 2 3 4]
x =
1 2 3 4
>>
To create a row vector, enclose a list of values in brackets
Workspace
Select data to plot
Name Value Size
x [1,2,3,4] 1x4
Command History
%-- 8/27/2011 11:46 AM --
x=[1 2 3 4 5];
y=[10 20 30 40 50];
plot(x,y)
clear,clc
x=[1 2 3 4]
format compact
clear,clc
x=[1 2 3 4]
Start
Click and drag to move Workspace...



You may use either a space or a comma as a “delimiter” in a row vector


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> x=[1 2 3 4]
x =
1 2 3 4
>> x=[1,2,3,4]
x =
1 2 3 4
>> y=[1;2;3;4]
y =
1
2
3
4
>>
Workspace
Select data to plot
Name Value Size
x [1,2,3,4] 1x4
y [1;2;3;4] 4x1
;
plot(x,y)
clear,clc
x=[1 2 3 4]
format compact
clear,clc
x=[1 2 3 4]
x=[1,2,3,4]
y=[1;2;3;4]
Use a semicolon as a delimiter to create a new row
Start
OVR


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> x=[1 2 3 4]
x =
1 2 3 4
>> x=[1,2,3,4]
x =
1 2 3 4
>> y=[1;2;3;4]
y =
1
2
3
4
>> a=[1 2 3 4 ;2 3 4 5; 3 4 5 6]
a =
1 2 3 4
2 3 4 5
3 4 5 6
>>
Workspace
Select data to plot
Name Value Size
a <3x4 double> 3x4
x [1,2,3,4] 1x4
y [1;2;3;4] 4x1
Command History
plot(x,y)
clear,clc
x=[1 2 3 4]
format compact
clear,clc
x=[1 2 3 4]
Use a semicolon as a delimiter to create a new row
Start
OVR



Hint: It's easier to keep track of how many values you've entered into a matrix, if you enter each row on a separate line. The semicolons are optional

Shortcuts

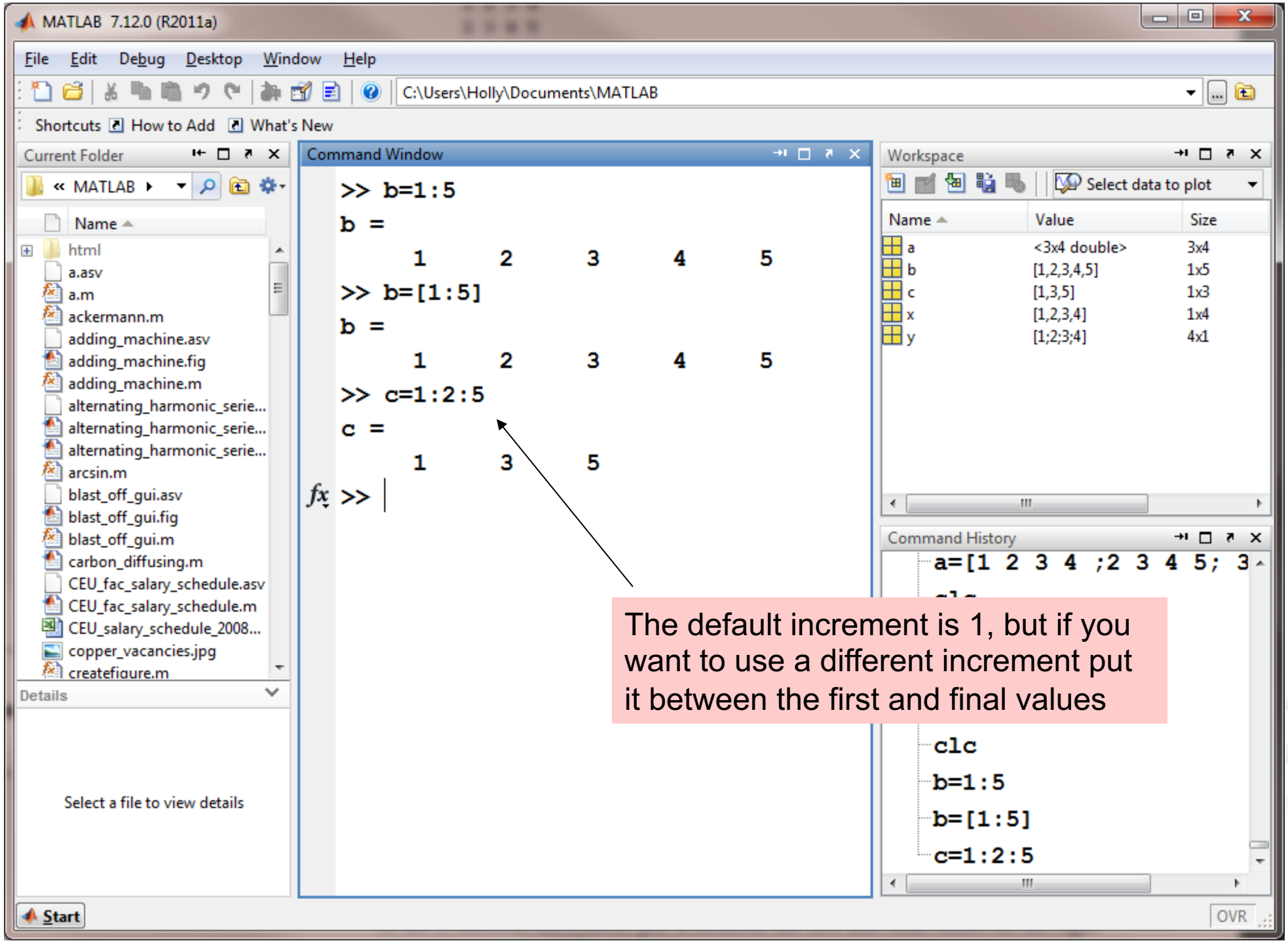
- While a complicated matrix might have to be entered by hand, evenly spaced matrices can be entered much more readily. The command

  b= 1:5

  or the command

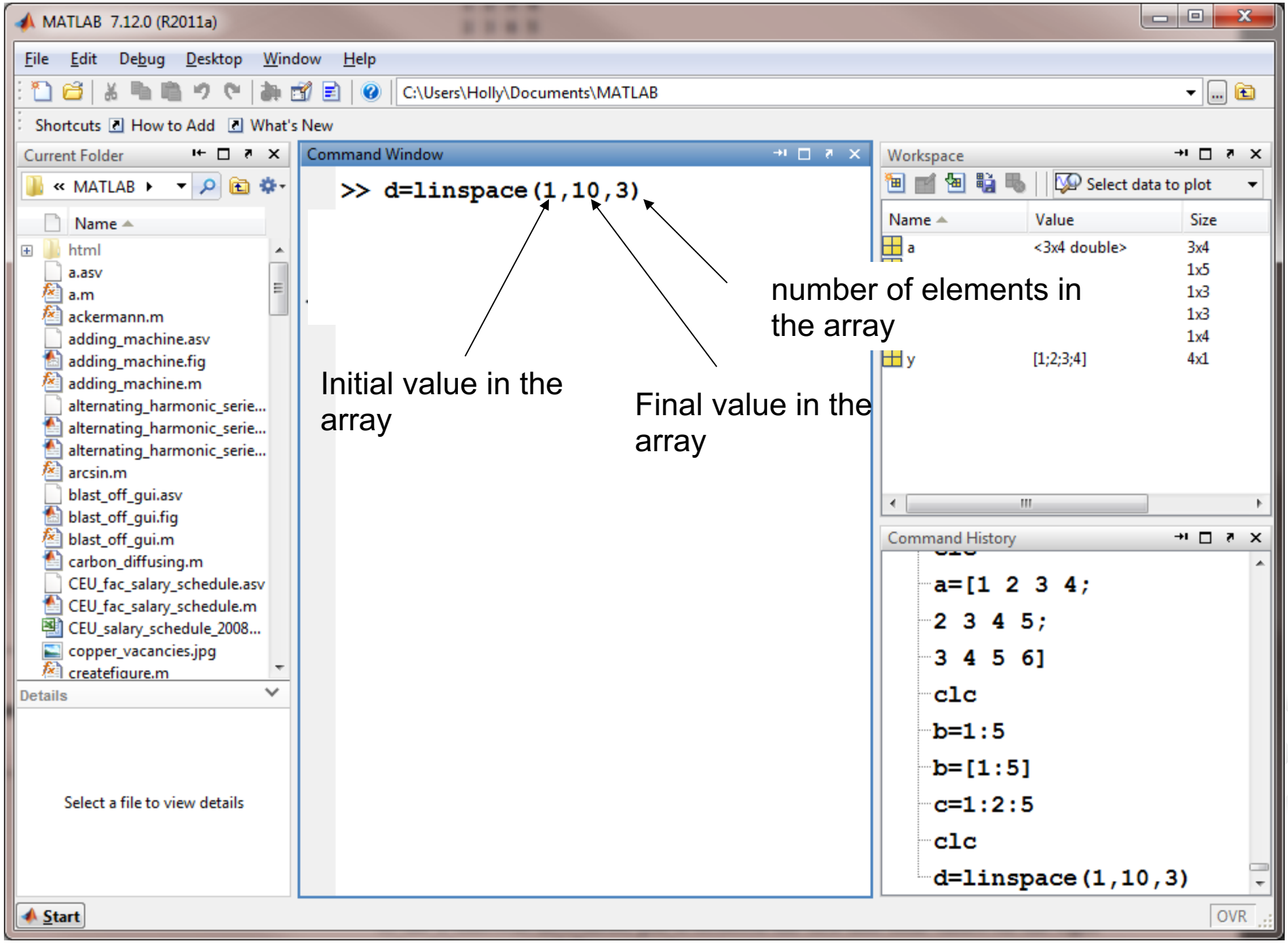
  b = [1:5]

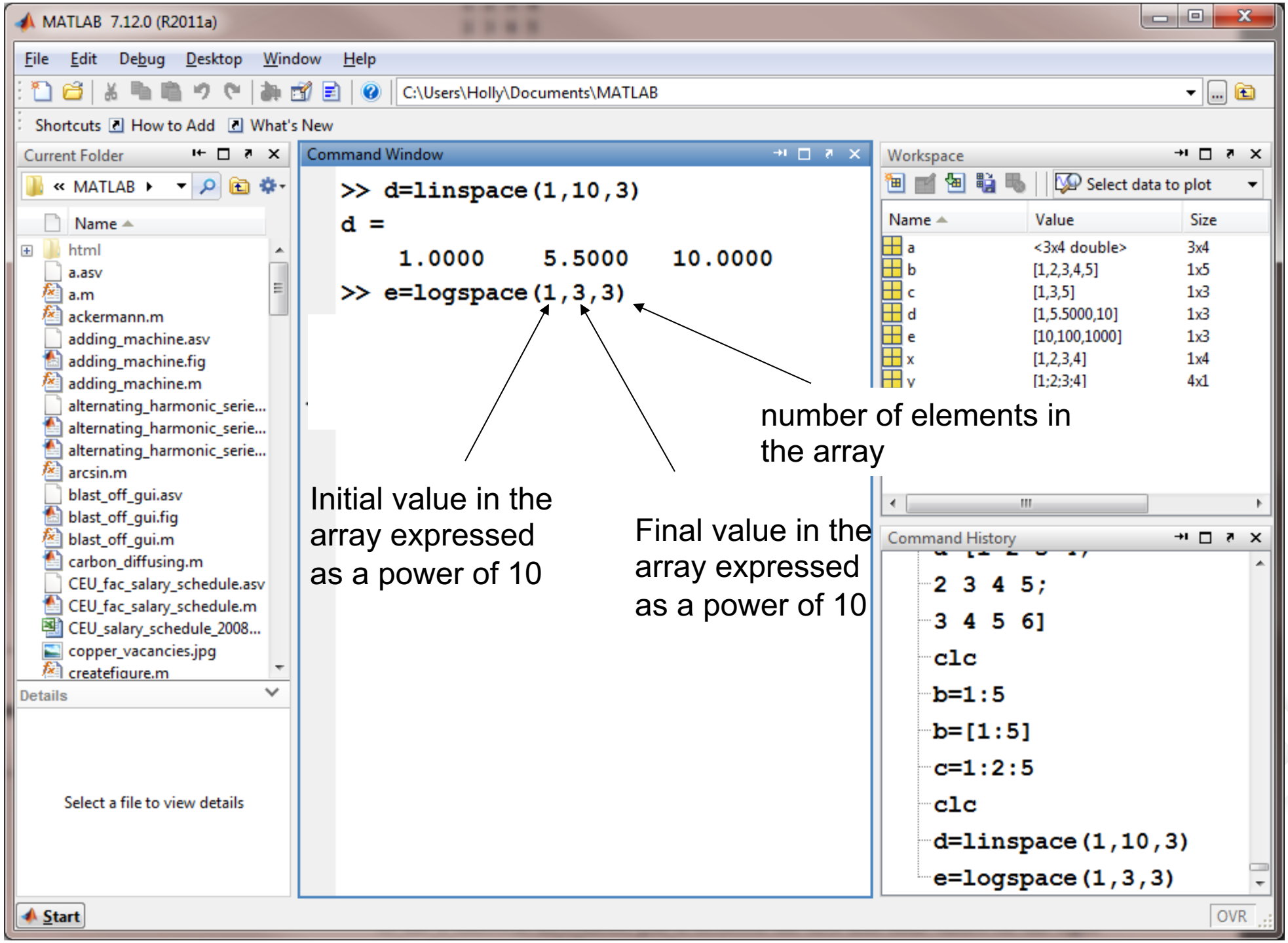
  both return a row matrix

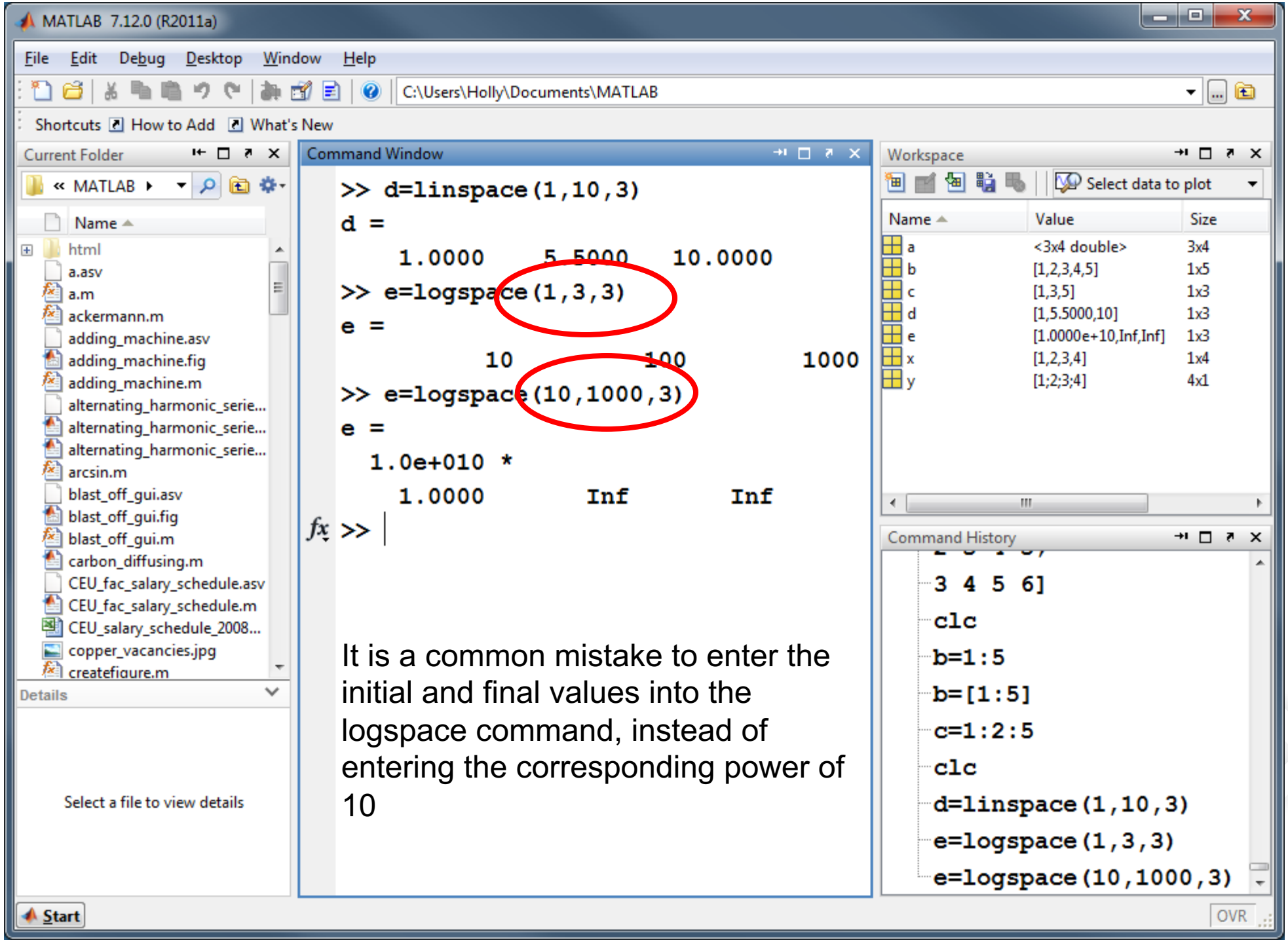

MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> b=1:5
b =
1 2 3 4 5
>> b=[1:5]
b =
1 2 3 4 5
>> c=1:2:5
c =
1 3 5
>>
Workspace
Select data to plot
Name Value Size
a <3x4 double> 3x4
b [1,2,3,4,5] 1x5
c [1,3,5] 1x3
x [1,2,3,4] 1x4
y [1;2;3;4] 4x1
Command History
a=[1 2 3 4 ;2 3 4 5; 3
clc
b=1:5
b=[1:5]
c=1:2:5
The default increment is 1, but if you want to use a different increment put it between the first and final values
Start
OVR

To calculate spacing between elements use...

+ linspace
+ logspace


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> d=linspace(1,10,3)
Initial value in the array
Final value in the array
number of elements in the array
Workspace
Select data to plot
Name
Value
Size
a
<3x4 double>
3x4
1x5
1x3
1x3
1x4
y
[1;2;3;4]
4x1
Command History
a=[1 2 3 4;
2 3 4 5;
3 4 5 6]
clc
b=1:5
b=[1:5]
c=1:2:5
clc
d=linspace(1,10,3)
Start
OVR


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
« MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> d=linspace(1,10,3)
d =
1.0000 5.5000 10.0000
>> e=logspace(1,3,3)
Workspace
Select data to plot
Name Value Size
a <3x4 double> 3x4
b [1,2,3,4,5] 1x5
c [1,3,5] 1x3
d [1,5.5000,10] 1x3
e [10,100,1000] 1x3
x [1,2,3,4] 1x4
y [1;2;3;4] 4x1
Command History
2 3 4 5;
3 4 5 6]
clc
b=1:5
b=[1:5]
c=1:2:5
clc
d=linspace(1,10,3)
e=logspace(1,3,3)
Start
OVR
number of elements in the array
Initial value in the array expressed as a power of 10
Final value in the array expressed as a power of 10



It is a common mistake to enter the initial and final values into the logspace command, instead of entering the corresponding power of 10

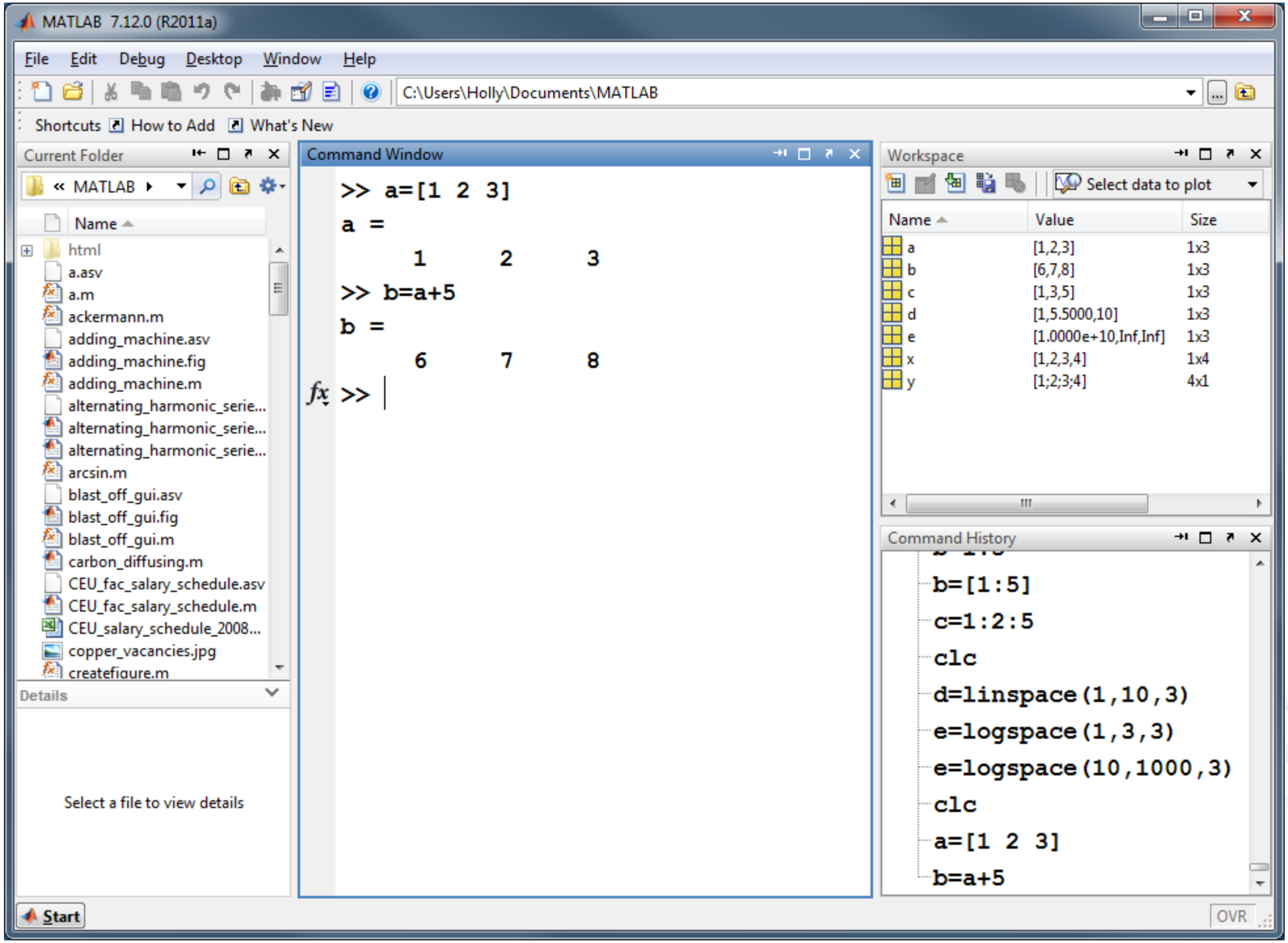
Hint

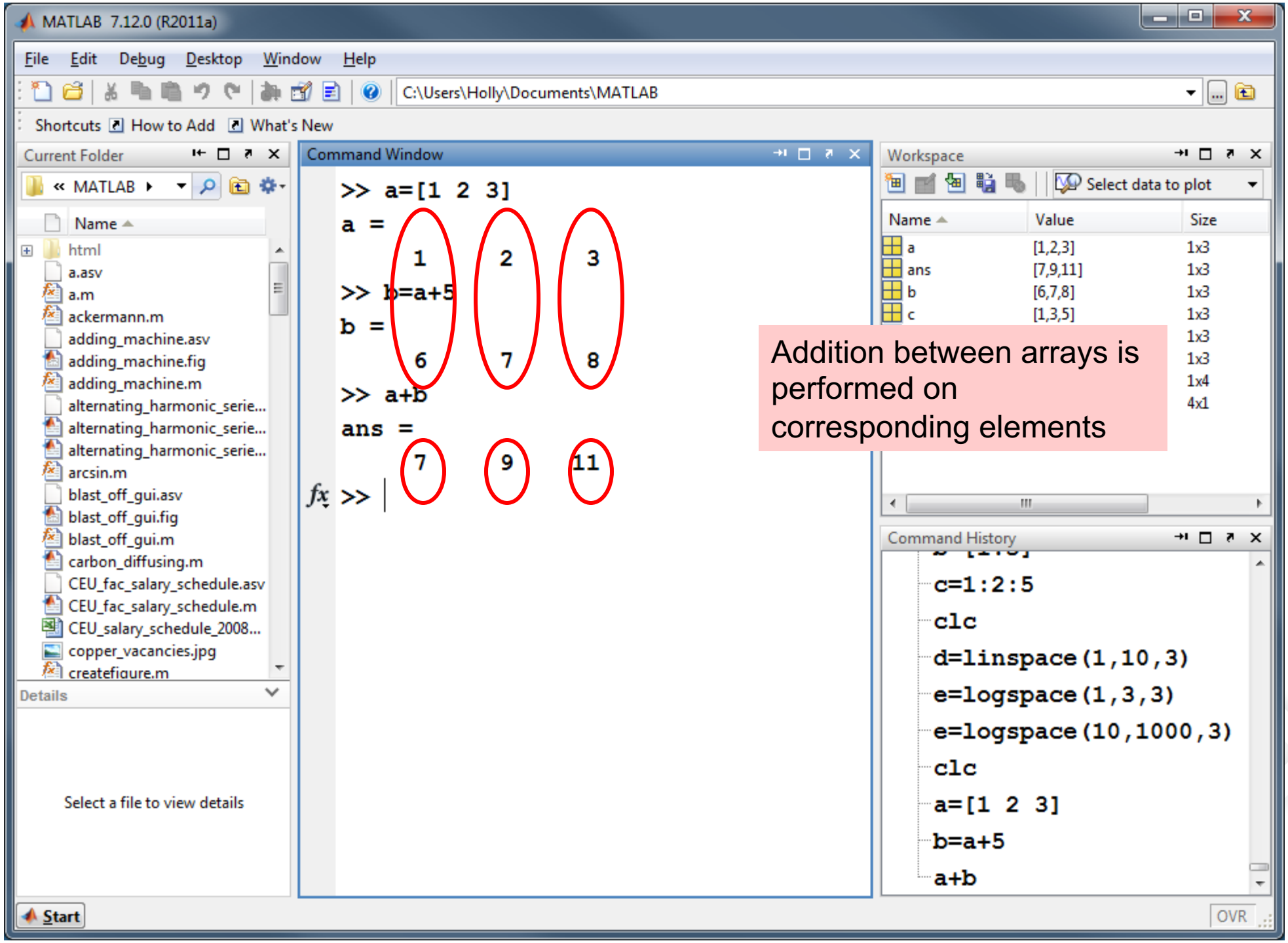
- You can include mathematical operations inside a matrix definition statement.
- For example

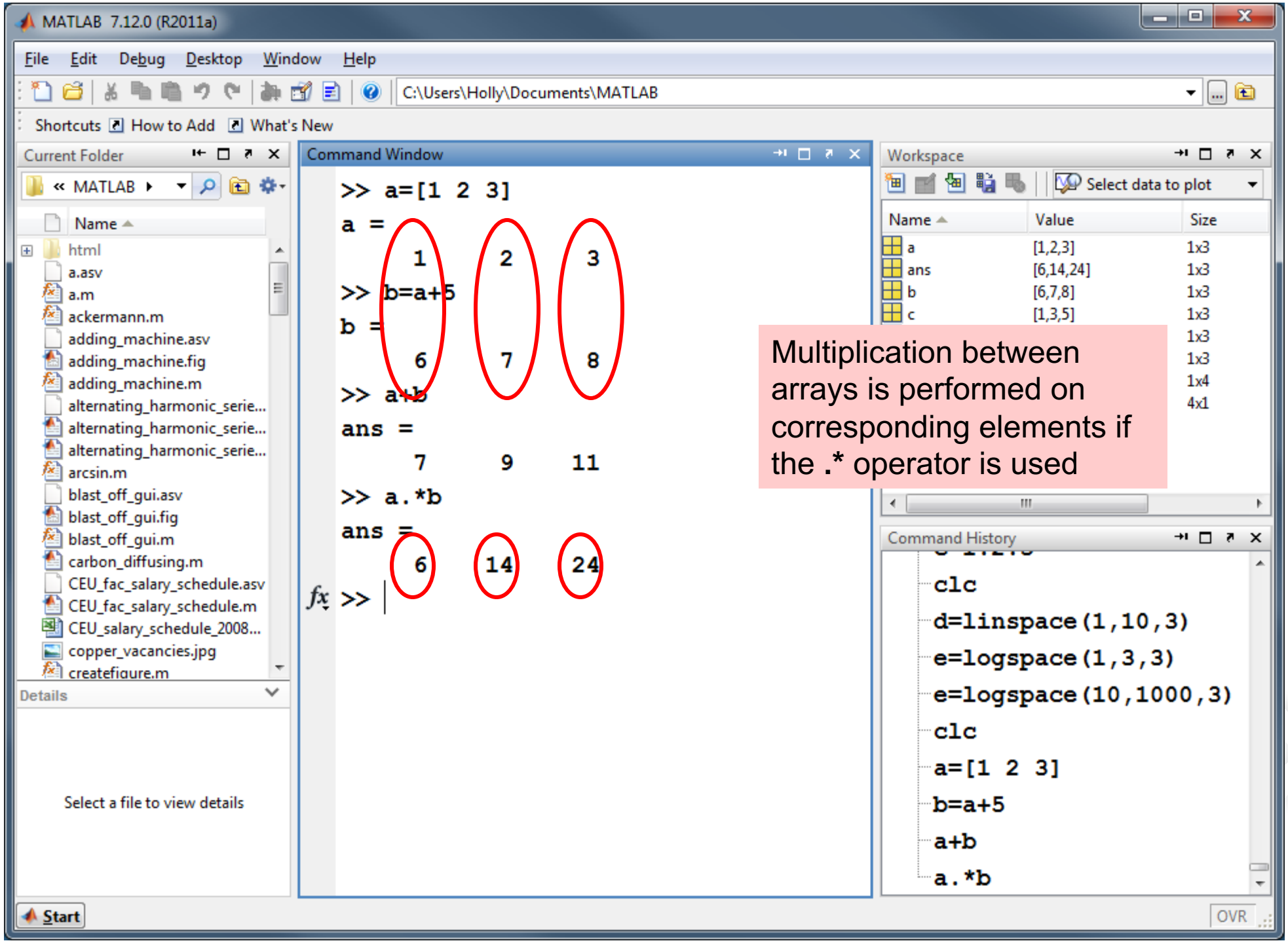
```
a = [0: pi/10: pi]
```

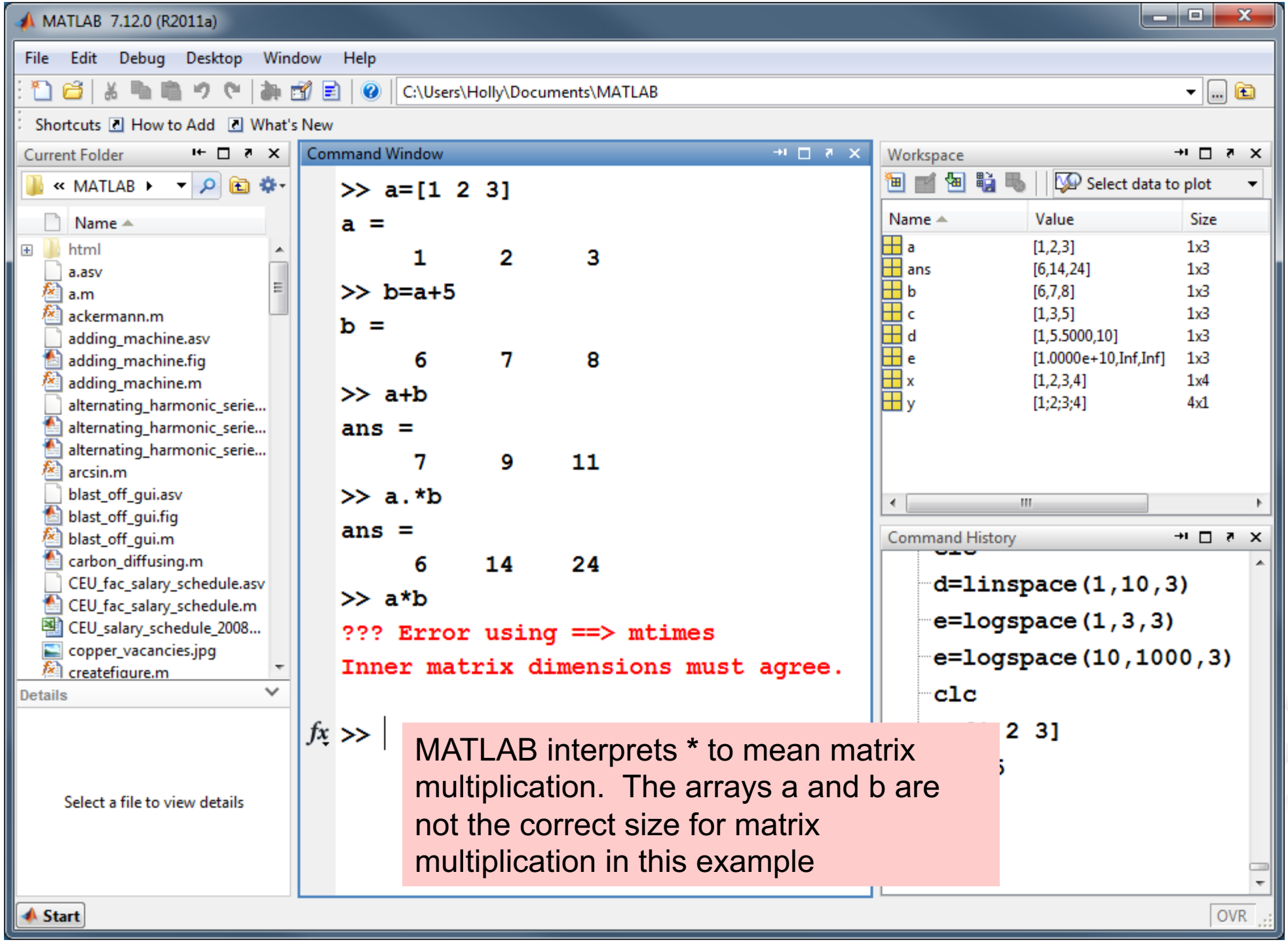
Mixed calculations between scalars and arrays

- Matrices can be used in many calculations with scalars
- There is no confusion when we perform addition and subtraction
- Multiplication and division are a little different
- In matrix mathematics the multiplication operator (*) has a very specific meaning


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> a=[1 2 3]
a =
1 2 3
>> b=a+5
b =
6 7 8
>>
Workspace
Select data to plot
Name Value Size
a [1,2,3] 1x3
b [6,7,8] 1x3
c [1,3,5] 1x3
d [1,5.5000,10] 1x3
e [1.0000e+10,Inf,Inf] 1x3
x [1,2,3,4] 1x4
y [1;2;3;4] 4x1
Command History
b=[1:5]
c=1:2:5
clc
d=linspace(1,10,3)
e=logspace(1,3,3)
e=logspace(10,1000,3)
clc
a=[1 2 3]
b=a+5
Start
OVR


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
« MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> a=[1 2 3]
a =
1 2 3
>> b=a+5
b =
6 7 8
>> a+b
ans =
7 9 11
>>
Workspace
Select data to plot
Name Value Size
a [1,2,3] 1x3
ans [7,9,11] 1x3
b [6,7,8] 1x3
c [1,3,5] 1x3
1x3
1x3
1x4
4x1
Addition between arrays is performed on corresponding elements
Command History
c=1:2:5
clc
d=linspace(1,10,3)
e=logspace(1,3,3)
e=logspace(10,1000,3)
clc
a=[1 2 3]
b=a+5
a+b
Start
OVR


MATLAB 7.12.0 (R2011a)
File Edit Debug Desktop Window Help
C:\Users\Holly\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
MATLAB
Name
html
a.asv
a.m
ackermann.m
adding_machine.asv
adding_machine.fig
adding_machine.m
alternating_harmonic_serie...
alternating_harmonic_serie...
alternating_harmonic_serie...
arcsin.m
blast_off_gui.asv
blast_off_gui.fig
blast_off_gui.m
carbon_diffusing.m
CEU_fac_salary_schedule.asv
CEU_fac_salary_schedule.m
CEU_salary_schedule_2008...
copper_vacancies.jpg
createfigure.m
Details
Select a file to view details
Command Window
>> a=[1 2 3]
a =
1 2 3
>> b=a+5
b =
6 7 8
>> a+b
ans =
7 9 11
>> a.*b
ans =
6 14 24
>>
Workspace
Select data to plot
Name Value Size
a [1,2,3] 1x3
ans [6,14,24] 1x3
b [6,7,8] 1x3
c [1,3,5] 1x3
1x3
1x3
1x4
4x1
Multiplication between arrays is performed on corresponding elements if the .* operator is used
Command History
clc
d=linspace(1,10,3)
e=logspace(1,3,3)
e=logspace(10,1000,3)
clc
a=[1 2 3]
b=a+5
a+b
a.*b
Start
OVR



MATLAB interprets * to mean matrix multiplication. The arrays a and b are not the correct size for matrix multiplication in this example