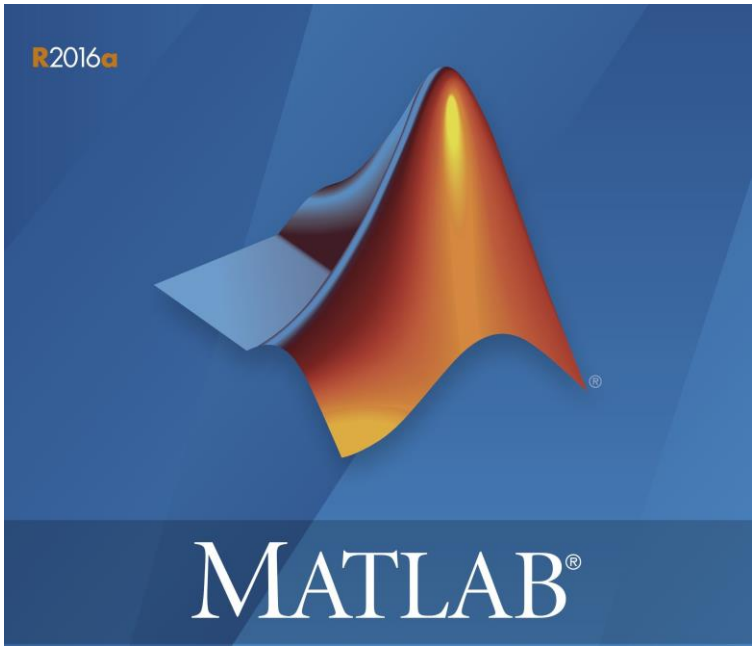
MATLAB

a short primer



Ovidiu Ivanov
'Gheorghe Asachi' Technical University,
Electrical Engineering Faculty
Iasi, Romania
www.tuiasi.ro

Part Two

Getting input data for our programs and saving results

How to define, save and load data
Some useful statements of general purpose

Defining data for your programs

- MATLAB stores numerical and text data into variables
  - You can define a variable with a name and a value
  - You can compute values for variables using numbers, mathematical operators, other variables, mathematical functions
  - You can import variables or values into variables from external files

- In MATLAB, you don't need to declare the variables and their types before using them
- MATLAB sets the type according to what you write
- You can define a variable anywhere in your program
- A variable is accessible if you can see it in the Workspace

Most common variable types

- integer
- double
- char
- string
- array of integer or float – vector or matrix
- cell and cell arrays

Variable types in MATLAB

Numeric Types

Integer and floating-point data

Numeric classes in MATLAB® include signed and unsigned integers, and single-precision and double-precision floating-p
cannot change the default type and precision.) You can choose to store any number, or array of numbers, as integers or
double precision.

All numeric types support basic array operations, such as subscripting, reshaping, and mathematical operations.

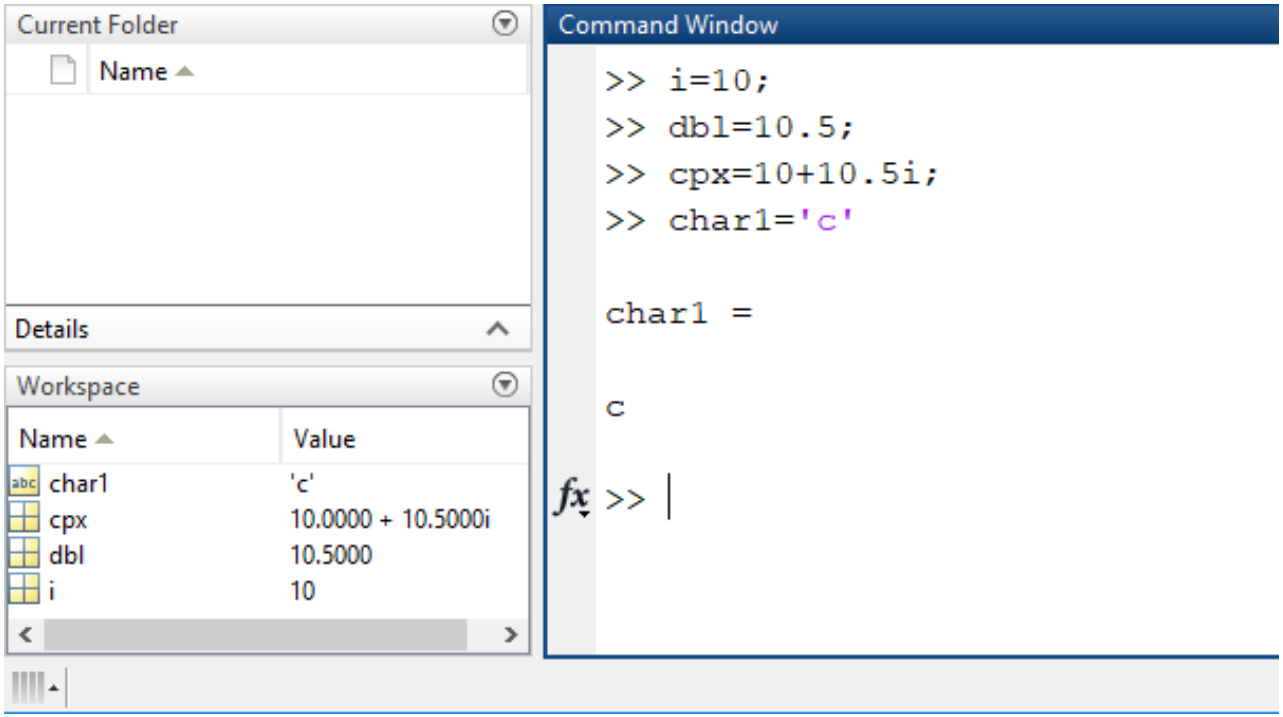
Functions

`double`	Double-precision arrays
`single`	Single-precision arrays
`int8`	8-bit signed integer arrays
`int16`	16-bit signed integer arrays
`int32`	32-bit signed integer arrays
`int64`	64-bit signed integer arrays
`uint8`	8-bit unsigned integer arrays
`uint16`	16-bit unsigned integer arrays
`uint32`	32-bit unsigned integer arrays
`uint64`	64-bit unsigned integer arrays

`cast`	Cast variable to different data type
`typecast`	Convert data types without changing underlying data

`isinteger`	Determine if input is integer array
`isfloat`	Determine if input is floating-point array

Defining integer, double, char and complex variables



- You can use small or capital letters, numbers and underscore
- The variable name must start with a letter
- If you don't terminate a line with semicolon, the variable name and value are displayed on the screen.

Defining vectors and matrices

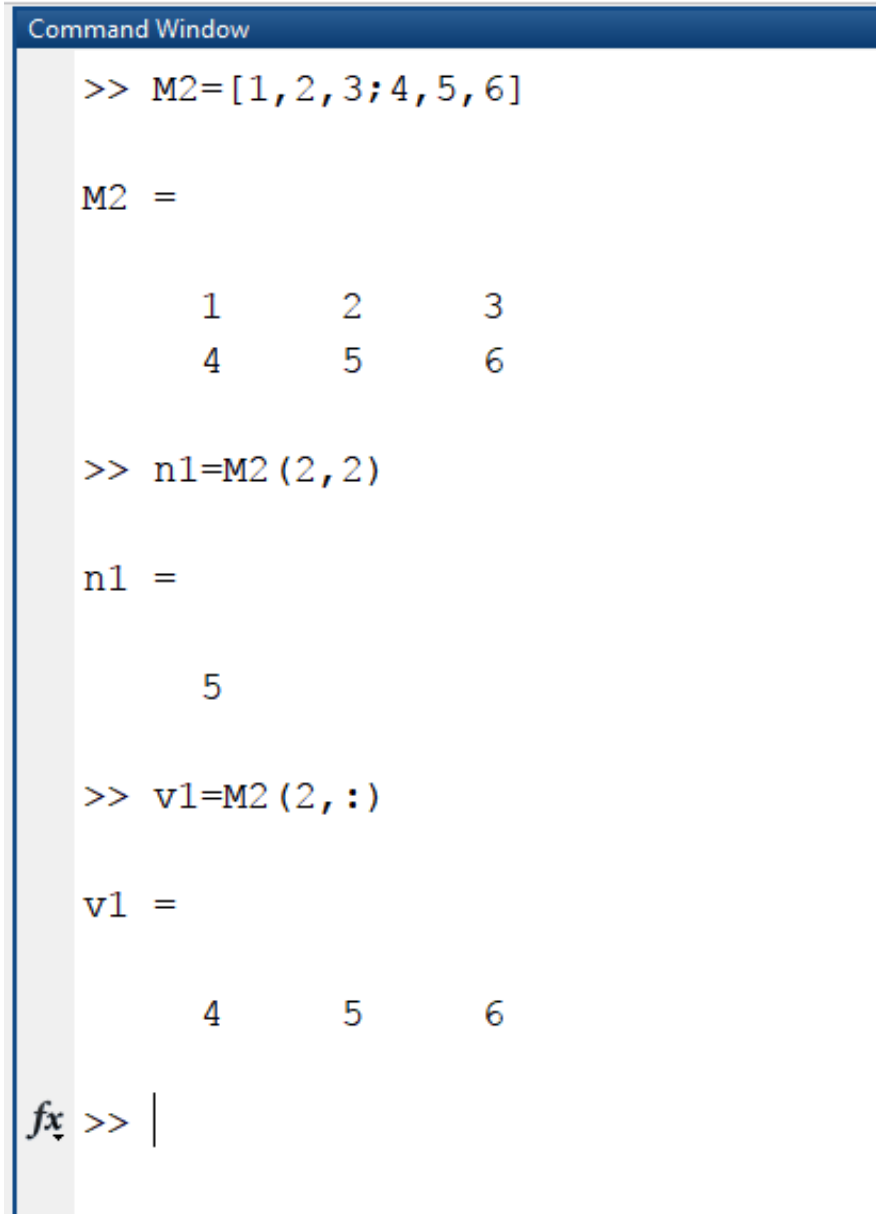
- Arrays are defined between square brackets.
- In vectors and matrices, columns are separated with space or comma, and rows are separated by semicolons
- The maximum number of dimensions is not fixed

Command Window

```
>> M1=[1 2 3; 4 5 6];
>> M2=[1,2,3;4,5,6];
>> M1-M2

ans =

     0     0     0
     0     0     0

fx >>
```

Addressing array elements

- You can address individual elements using round brackets
- You can address chunks of arrays using square brackets and Excel-like ranges



Elementary operations with numbers

- addition +
- subtraction -
- multiplication *
- division /
- power ^

- Parentheses and operation precedence are observed

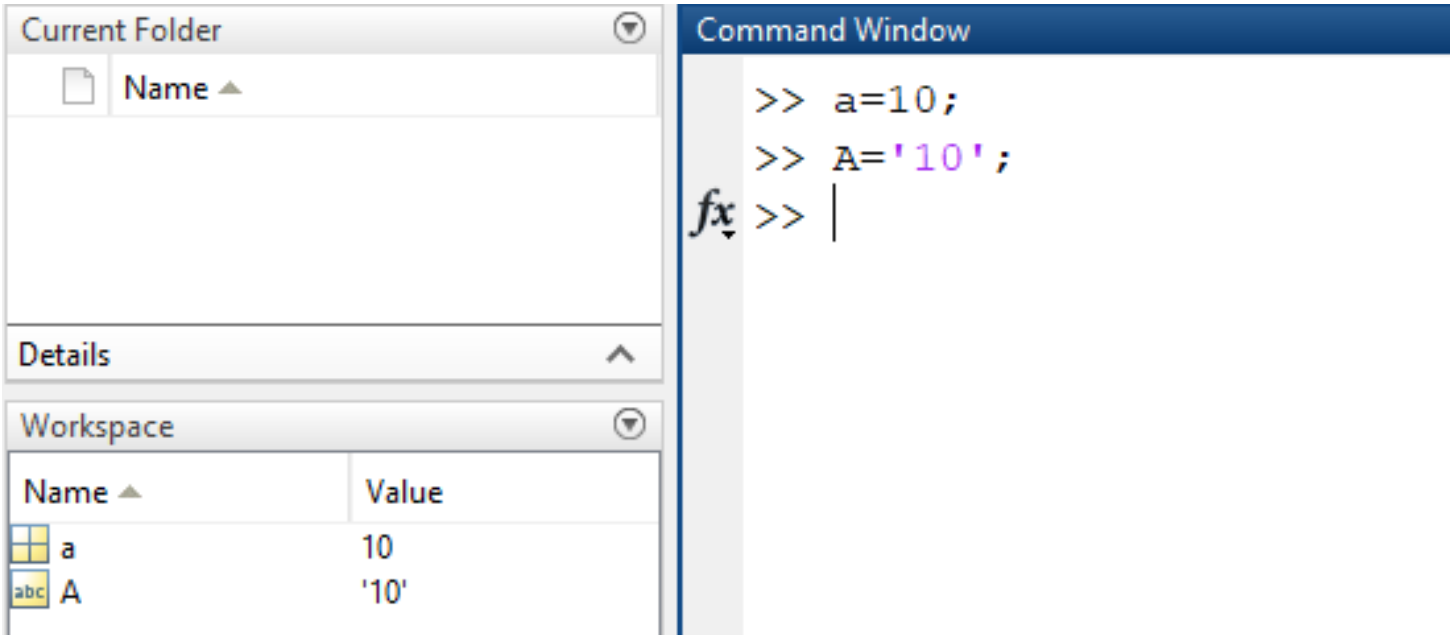
Elementary operations with arrays

- all of the above
- MATLAB will treat any variables as matrices and it will multiply them accordingly.
- If you want to multiply or divide matrices or vectors element by element, you should use .* or ./

Command Window

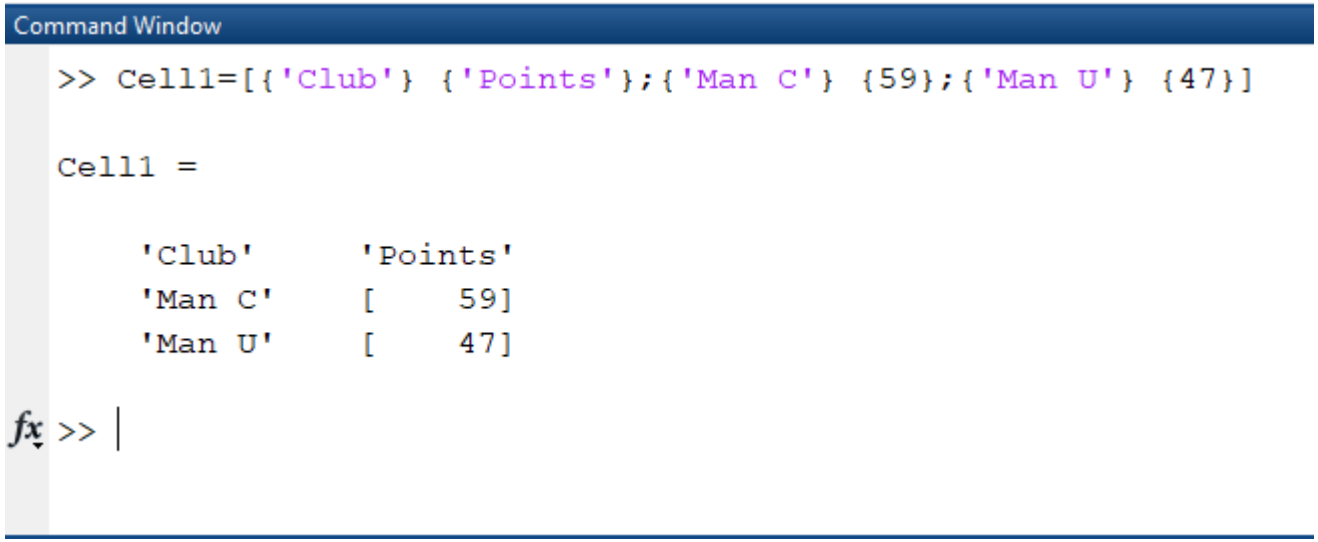
```
>> M1

M1 =

     1     2     3
     4     5     6

>> M2

M2 =

     1     2     3
     4     5     6

>> M1*M2
Error using  *
Inner matrix dimensions must agree.

>> M1.*M2

ans =

     1     4     9
    16    25    36
```

The difference between numbers and strings



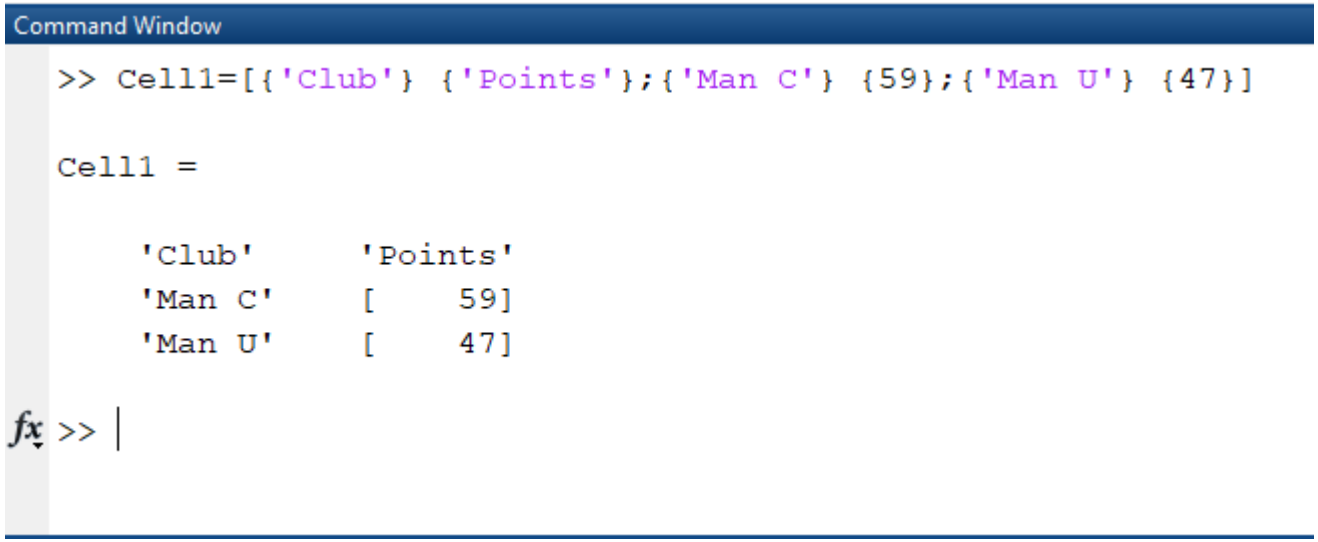
- Notice the different symbol from the Workspace
  - conversion from one to another: num2str or str2num
- MATLAB is case sensitive, variables a and A are distinct

Defining cell variables and arrays



- You define cell arrays as you define matrices, but putting individual elements between curly brackets
- Cells can store any type of element
- Excel or database imports with more than one type of values will be automatically stored by MATLAB into cell arrays

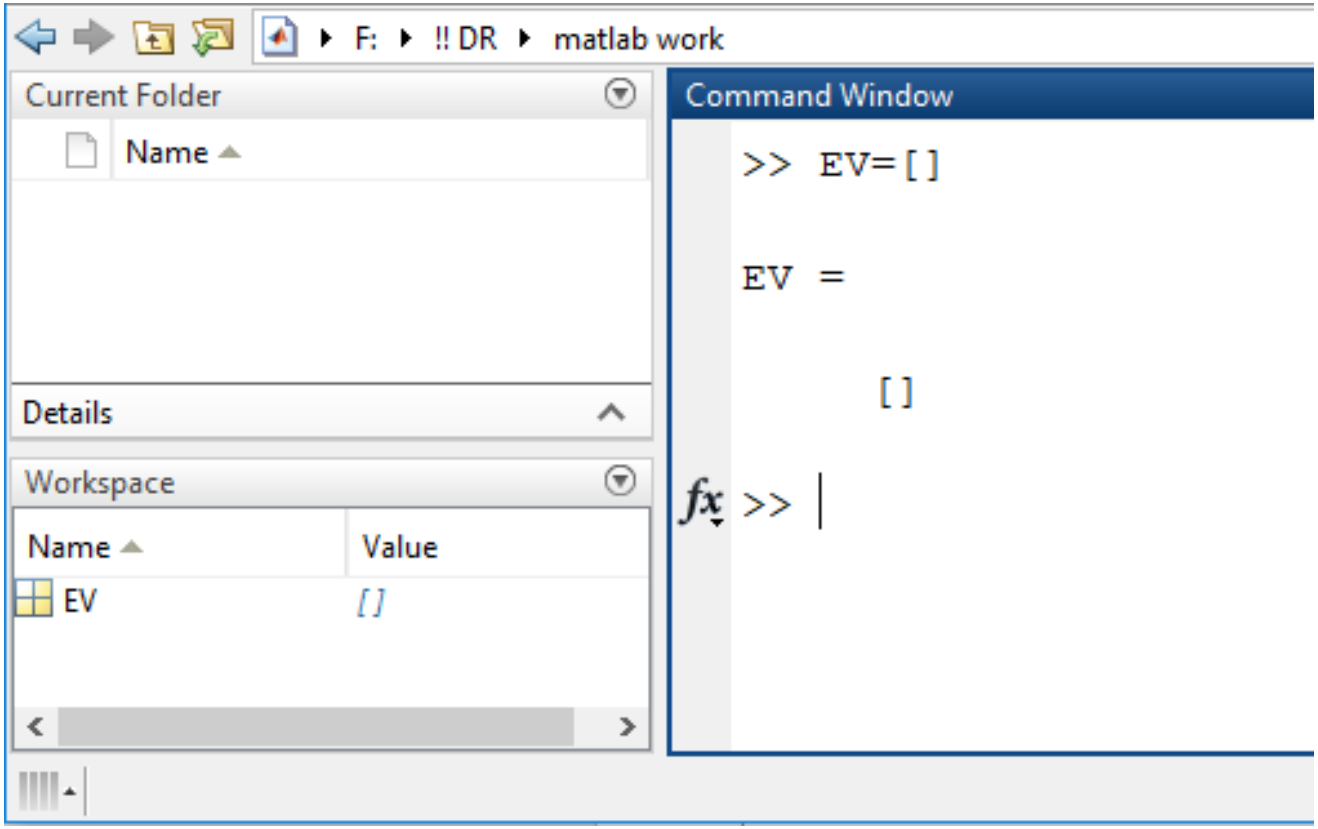
Converting between cells and cell arrays and numbers or strings



- Cell to numbers: cell2mat
- Numbers to cells: num2cell or mat2cell
- Cell to string: char
- Strings to cell: cellstr

The empty variable

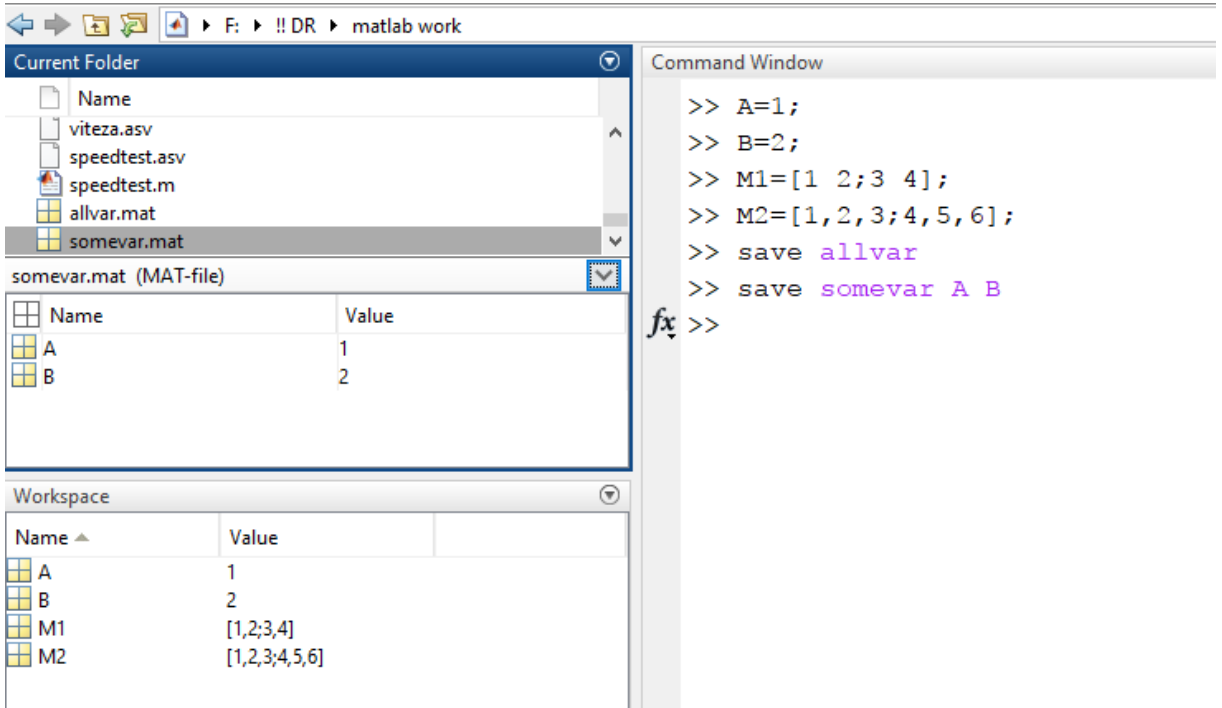
- You can define an empty variable to which you can assign values later.



Saving variables for later use – the save statement

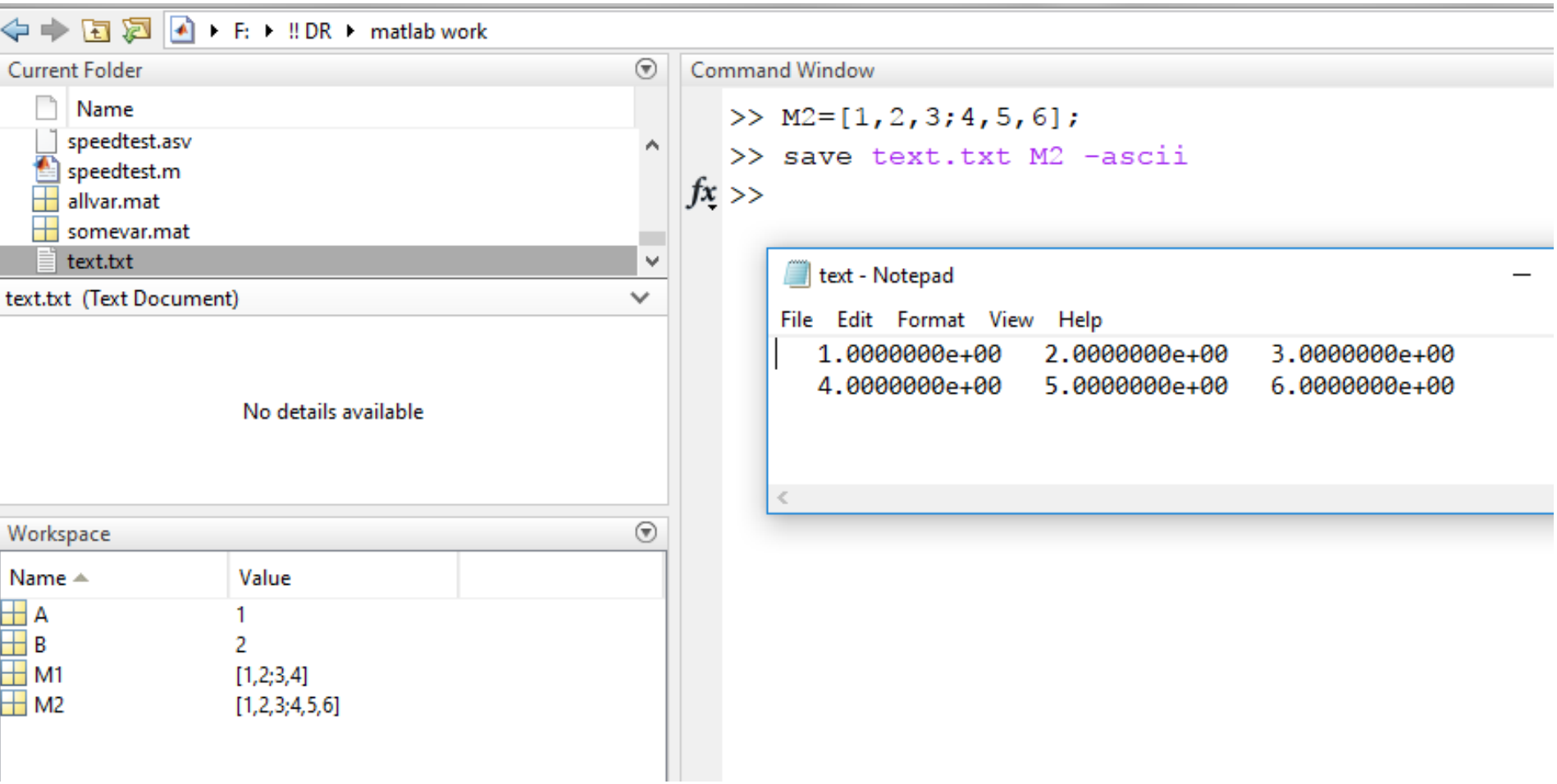
- MATLAB can save variables into .mat files
- You can save all the variables from the workspace or just some of them

```
save filename
save filename var1 var 2 var 3
```

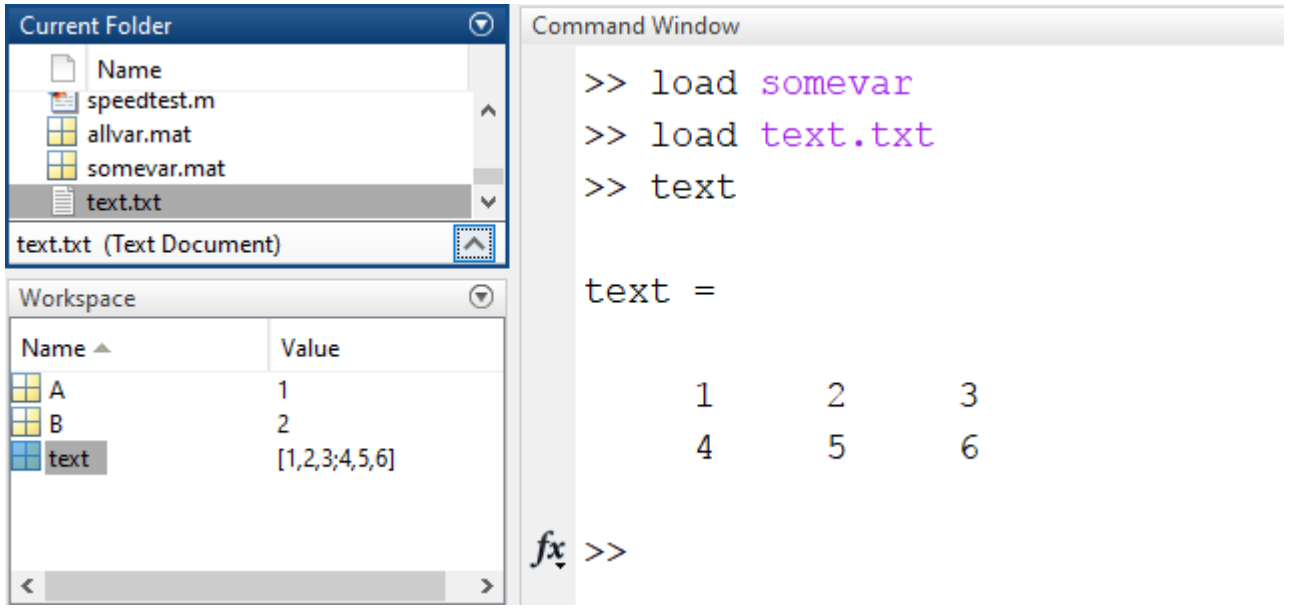


Saving variables as text files

- You can save only one variable from the Workspace as a text file

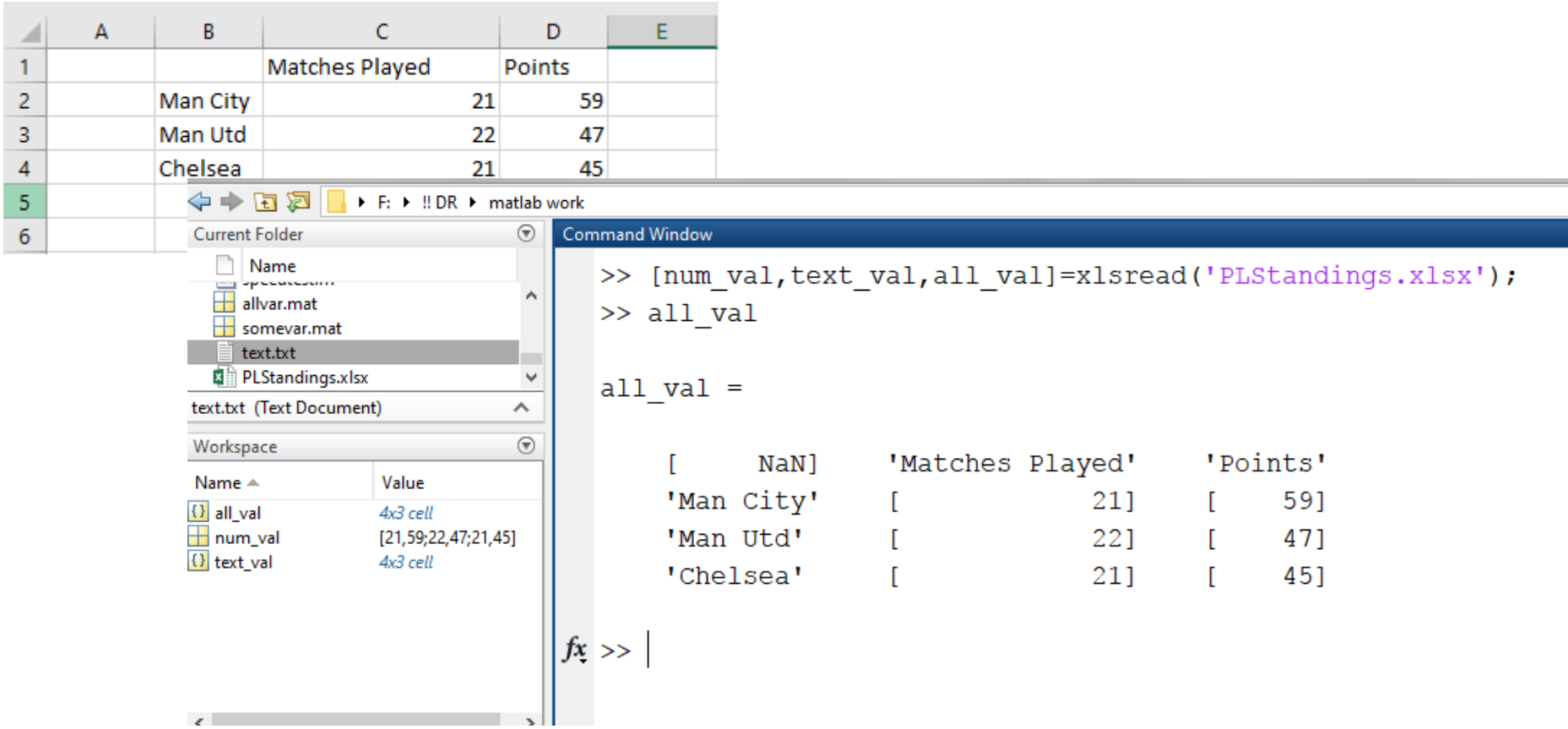


Loading variables from .mat and .txt files – the load statement



- MATLAB can read tab delimited and space delimited text files and import them into matrices
- For .mat file, you can skip the extension. For others, you cannot.

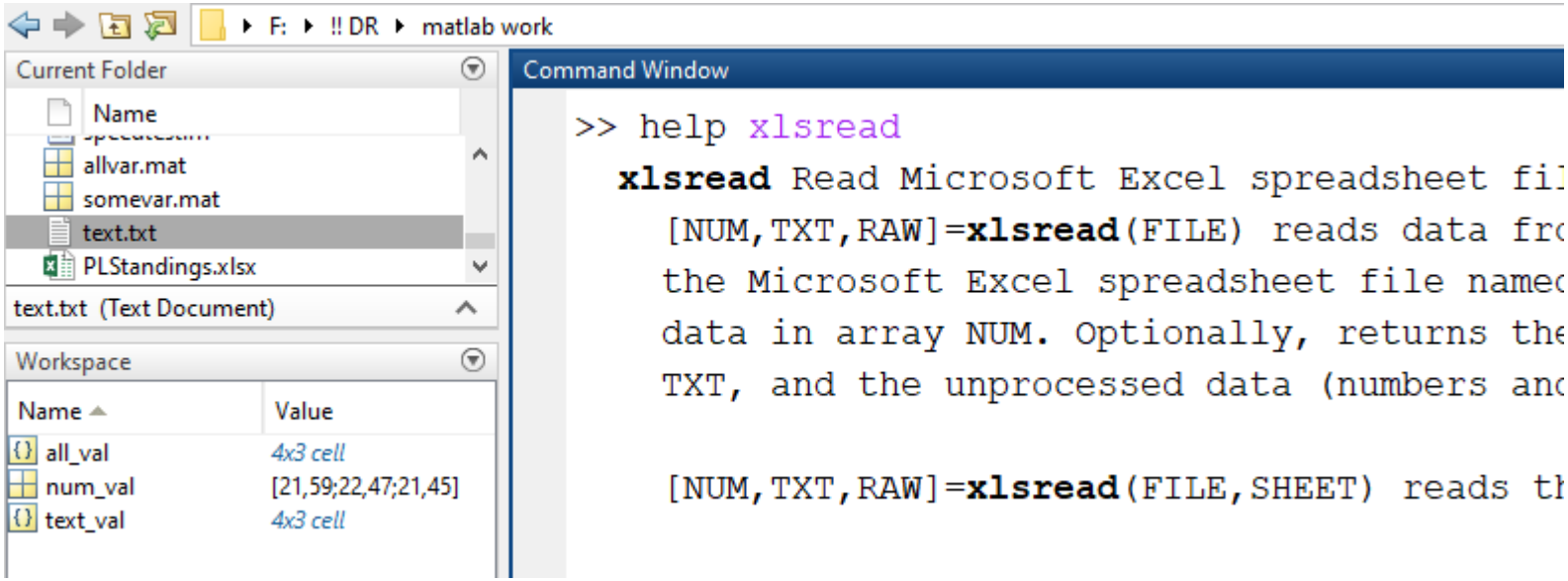
Reading from Excel files – the xlsread statement



	A	B	C	D	E
1			Matches Played	Points	
2		Man City	21	59	
3		Man Utd	22	47	
4		Chelsea	21	45	
5					
6					

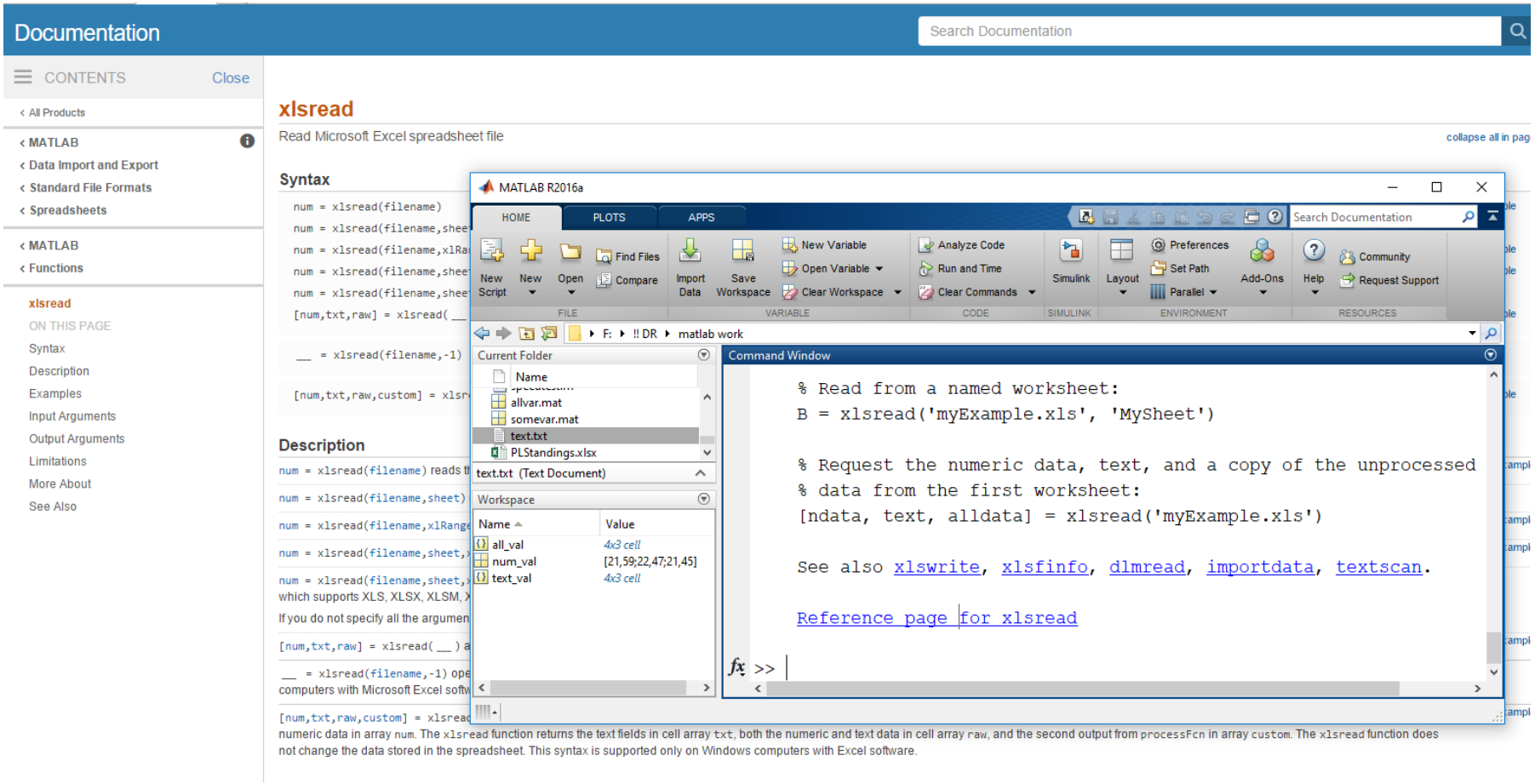
- MATLAB can also write into Excel files, with **xlswrite**

Getting help



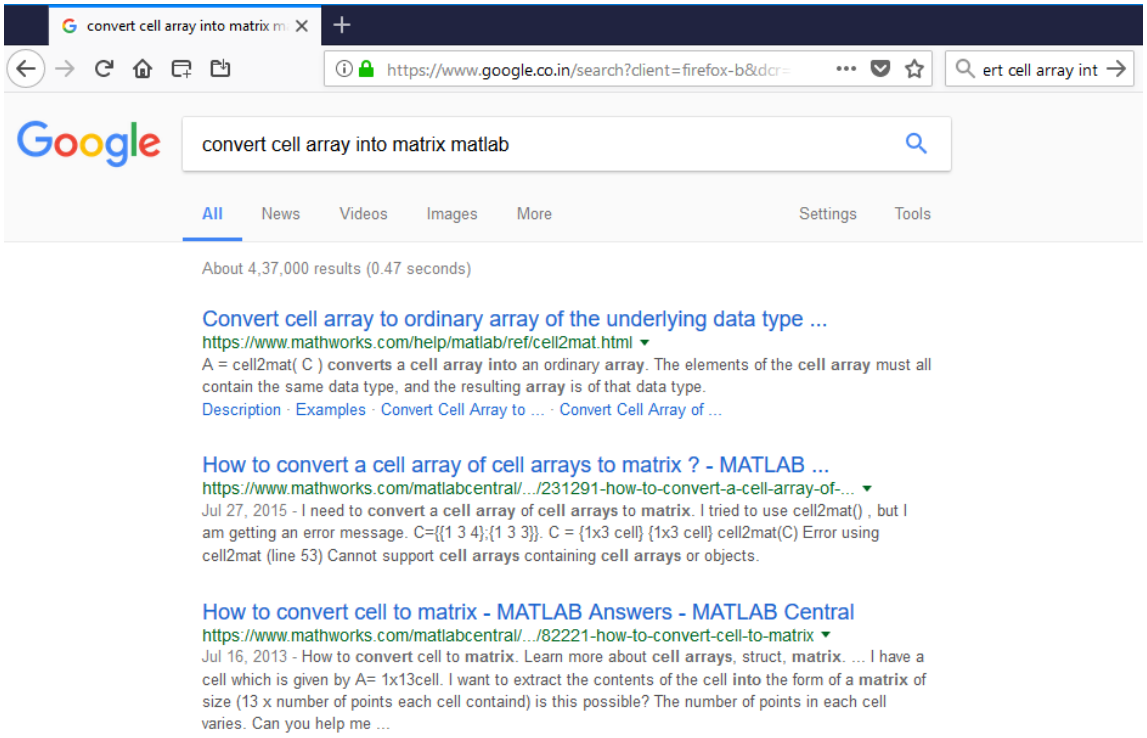
- Write help followed by the function name
- You can make xlsread to read from a specific sheet and cell range

Getting extended help



- Scroll down use the offline documentation reference pages

Getting extended help



- Google and www.mathworks.com are there to help you with code and examples !

Other useful commands

- clc will clear the Command Window screen, but not the memory (Workspace)
- clear will clear the Workspace, but not the screen
- tic and toc will time the execution of your code
- exit will exit MATLAB
- rand will generate a random number in (0,1)
- pause stops execution until a key is pressed
- break exits a loop

Useful keyboard shortcuts

- Ctrl+C breaks the execution of a running program
- F5 runs a program
- F10 runs a program step by step
- Shift+F5 exits debugging mode
- Ctrl+I will align the selected code in the Editor window
- Use the arrow keys to access the history feature of the Command Window

- How to define, save and load data
  - The most used types of variables
  - The load and save commands
  - xlsread / xlswrite
- Some useful functions and statements of general purpose
  - Getting help
  - Clearing your screen and memory

End of Part Two - Review

- **Feel free to contact me with questions !**

Ovidiu Ivanov

Lecturer, PhD

The ‘Gheorghe Asachi’ Technical University

The Electrical Engineering Faculty

www.tuiasi.ro

ovidiuivanov@tuiasi.ro

THANK YOU !