# MatLab Programming Fundamentals

**guarantor:** Maroš Tunák
**tel.:** 3465
**e-mail:** *maros.tunak@tul.cz*

# Course objectives

The aim of the course is to acquire basics knowledge and skills of students the MatLab program. At the end of the course students will be able to use MatLab for their own work and will be ready to deepen their programming skills in MatLab.

## MatLab Programming Fundamentals

| | |
|---|---|
| time requirements: | 0p+2c |
| credits: | 4 |
| exercises: | Monday 10:40-12:15; 12:30-14:05 (B-PC2, Tunák M.)<br>Tuesday 08:50-10:25; 10:40-12:15 (B-PC2, Tunák M.) |
| consultation: | Wednesday 10:40-12:15 (E-KHT) |

## Requirements on student/graded credit

1. participation in exercises (max. 3 absences)
2. elaboration of semester work (after approval of the semester work, you can attend a practical demonstration)
3. practical demonstration of acquired skills (there will be 1-2 examples to solve; elaboration time 1 hour; you can use any materials ...)

# Content

## IS/STAG Syllabus

| | |
|---|---|
| 1. | Getting started with Matlab. Working environment, windows, paths, basic commands, variables. Loading, saving and information about variables. Help. |
| 2. | Mathematics with vectors and matrices. Creating vectors and matrices. Indexing. Special matrices. Matrix operations. Element by element operations. Relational operations, logical operations, examples and tricks. |
| 3. | Control flow. Loops, conditional statements, examples. |
| 4. | Script m-files, Function m-files. |
| 5. | Visualisation. Two-dimensional graphics. Three-dimensional graphics. |
| 6. | Graphical user interface. |
| 7.-10. | Statistics and Machine Learning Toolbox. Basics of statistical data processing, exploratory data analysis, descriptive statistics, data visualisation, hypothesis testing, confidence intervals, regression analysis, control charts. |
| 11.-13. | Solution of practical problems in textile and industrial engineering. |

# Literature

## Recommended

MathWorks. *Getting Started with MATLAB.* [Online]. Dostupné z: https://www.mathworks.com/help/matlab/getting-started-with-matlab.html

## Study materials

http://elearning.tul.cz

## Installation

http://liane.tul.cz/cz/software/MATLAB

**Getting started with Matlab. Working environment, windows, paths, basic commands, variables. Loading, saving and information about variables. Help.**

# Introduction

Matrix Laboratory - is a matrix-oriented high-performance environment for technical and engineering computing. MatLab environment is user friendly and is suitable for calculations, visualization and programming. Typical applications include mathematics and calculations; algorithm development; data acquisition; modelling and simulation; data analysis, exploring and visualization, scientific and engineering graphics; developing applications and creating graphical user interfaces.

MatLab is an interactive system where the basic data element is an array that it does not require dimensioning. This makes it possible to solve many technical computing problems, especially in matrix and vector expression. In addition to basic operations from the field provides also programming options similar to other programming languages.

In addition, MatLab provides additional extensive function files (m-files) that extend its functionality options and are included in specifically oriented libraries (Toolboxes). For statistical For example, the statistical library (Statistics and Machine Learning Toolbox) is used for data analysis.

More information about MatLab can be found in Matlab Help, or at web `http://www.mathworks.com`.

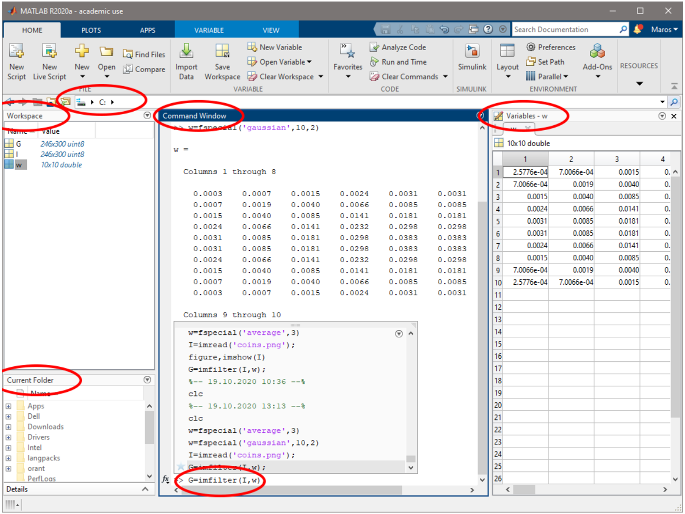
# Working Environment

Windows (MatLab R2020a)

- Command Window - typing commands and answers (error messages) appear here
- Workspace - list of defined variables (double click - Array Editor, Variables)
- Current Folder - content of Current Directory (path to the current directory in the top bar)
- Command History - past commands history (keyboard up and down arrows)

Note: windows can be docked or undocked (separate windows)

# Working Environment



| Name | Value |
|---|---|
| G | 246x300 uint8 |
| I | 246x300 uint8 |
| w | 10x10 double |

```
>> w=fspecial('gaussian',10,2)

w =

  Columns 1 through 8

    0.0003    0.0007    0.0015    0.0024    0.0031    0.0031
    0.0007    0.0019    0.0040    0.0066    0.0085    0.0085
    0.0015    0.0040    0.0085    0.0141    0.0181    0.0181
    0.0024    0.0066    0.0141    0.0232    0.0298    0.0298
    0.0031    0.0085    0.0181    0.0298    0.0383    0.0383
    0.0031    0.0085    0.0181    0.0298    0.0383    0.0383
    0.0024    0.0066    0.0141    0.0232    0.0298    0.0298
    0.0015    0.0040    0.0085    0.0141    0.0181    0.0181
    0.0007    0.0019    0.0040    0.0066    0.0085    0.0085
    0.0003    0.0007    0.0015    0.0024    0.0031    0.0031

  Columns 9 through 10
```

```
w=fspecial('average',3)
I=imread('coins.png');
figure,imshow(I)
G=imfilter(I,w);
%-- 19.10.2020 10:36 --%
clc
%-- 19.10.2020 13:13 --%
clc
w=fspecial('average',3)
w=fspecial('gaussian',10,2)
I=imread('coins.png');
G=imfilter(I,w);
```

```
>> G=imfilter(I,w)
```

10x10 double

|  | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 2.5776e-04 | 7.0066e-04 | 0.0015 | 0. |
| 2 | 7.0066e-04 | 0.0019 | 0.0040 | 0. |
| 3 | 0.0015 | 0.0040 | 0.0085 | 0. |
| 4 | 0.0024 | 0.0066 | 0.0141 | 0. |
| 5 | 0.0031 | 0.0085 | 0.0181 | 0. |
| 6 | 0.0031 | 0.0085 | 0.0181 | 0. |
| 7 | 0.0024 | 0.0066 | 0.0141 | 0. |
| 8 | 0.0015 | 0.0040 | 0.0085 | 0. |
| 9 | 7.0066e-04 | 0.0019 | 0.0040 | 0. |
| 10 | 2.5776e-04 | 7.0066e-04 | 0.0015 | 0. |

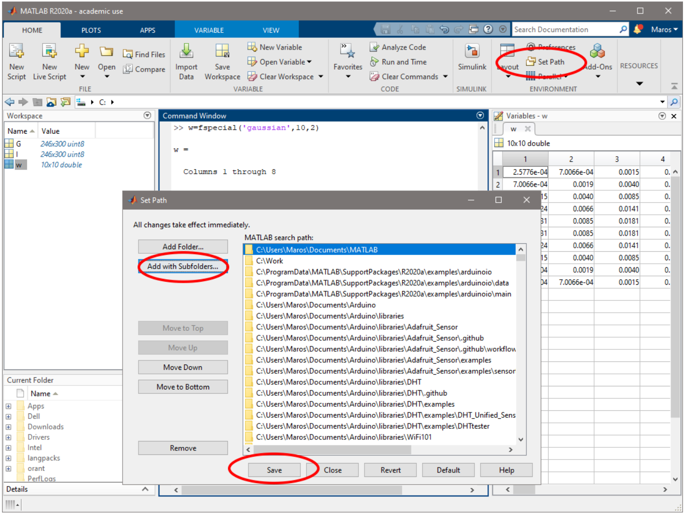
**Figure:** Matlab Windows.

## Working Environment

Paths - commands or programs are contained in *m-files* (plain text files with extension *.m). m-files must be located in one of the directories which MatLab automatically searches.

- current directory (» cd)
- predefines MatLab directories (» path)
- modifying the path (Home - Set Path), (» addpath('directory'))

# Working Environment



Workspace:

| Name | Value |
|---|---|
| G | 246x300 uint8 |
| I | 246x300 uint8 |
| w | 10x10 double |

Command Window:

```
>> w=fspecial('gaussian',10,2)

w =

  Columns 1 through 8
```

Variables - w: 10x10 double

|  | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 2.5776e-04 | 7.0066e-04 | 0.0015 | 0. |
| 2 | 7.0066e-04 | 0.0019 | 0.0040 | 0. |
|  | 15 | 0.0040 | 0.0085 | 0. |
|  | 24 | 0.0066 | 0.0141 | 0. |
|  | 31 | 0.0085 | 0.0181 | 0. |
|  | 31 | 0.0085 | 0.0181 | 0. |
|  | 24 | 0.0066 | 0.0141 | 0. |
|  | 15 | 0.0040 | 0.0085 | 0. |
|  | 04 | 0.0019 | 0.0040 | 0. |
|  | 04 | 7.0066e-04 | 0.0015 | 0. |

Set Path:

All changes take effect immediately.

MATLAB search path:

```
C:\Users\Maros\Documents\MATLAB
C:\Work
C:\ProgramData\MATLAB\SupportPackages\R2020a\examples\arduinoio
C:\ProgramData\MATLAB\SupportPackages\R2020a\examples\arduinoio\data
C:\ProgramData\MATLAB\SupportPackages\R2020a\examples\arduinoio\main
C:\Users\Maros\Documents\Arduino
C:\Users\Maros\Documents\Arduino\libraries
C:\Users\Maros\Documents\Arduino\libraries\Adafruit_Sensor
C:\Users\Maros\Documents\Arduino\libraries\Adafruit_Sensor\.github
C:\Users\Maros\Documents\Arduino\libraries\Adafruit_Sensor\.github\workflow
C:\Users\Maros\Documents\Arduino\libraries\Adafruit_Sensor\examples
C:\Users\Maros\Documents\Arduino\libraries\Adafruit_Sensor\examples\sensor
C:\Users\Maros\Documents\Arduino\libraries\DHT
C:\Users\Maros\Documents\Arduino\libraries\DHT\.github
C:\Users\Maros\Documents\Arduino\libraries\DHT\examples
C:\Users\Maros\Documents\Arduino\libraries\DHT\examples\DHT_Unified_Sens
C:\Users\Maros\Documents\Arduino\libraries\DHT\examples\DHTtester
C:\Users\Maros\Documents\Arduino\libraries\WiFi101
```

**Figure:** Set Path.

# Interrupting a Command or Program

Sometimes might occur an error in command or program - command or program does not stop.

- press Ctrl-C (or Ctrl-Break) for stopping the process
- sometimes press few times
- after this MatLab prompt re-appear

# Command Window

MatLab prompt is » - cursor is flickering and MatLab is waiting for further instructions, commands followed by Enter are executed immediately, the response (if desired) is displayed on screen

| Arithmetic Operators | Operation |
|---|---|
| + | addition |
| - | subtraction |
| * | multiplication |
| / | division |
| \ | left division |
| ^ | power |
| sqrt() | square root |
| ' | transposition |

If variable is not defined, answer is stored in variable ans (answer).

## Command Window

- **Example:** Try arithmetic operators, notice the created variable ans:

```
>> 5+17
ans =  22

>> 178-259
ans = -81

>> 128*2
ans = 256

>> ans/4
ans =  64

>> 9^2
ans =  81

>> sqrt(ans)
ans =  9
```

# Variables

- if variable name is not defined, answer is stored in variable ans (answer).
- variable names must starting with a letter and combination of letters and digits and underscores
- letters are case sensitive
- values are assigned to variables (assignment statements)
- created variables are stored in the Workspace
- variables can be called in the Command Window by typing the variable name
- you can edit variables by double-clicking on the variable name, an Array Editor will open in a separate window

## Variables

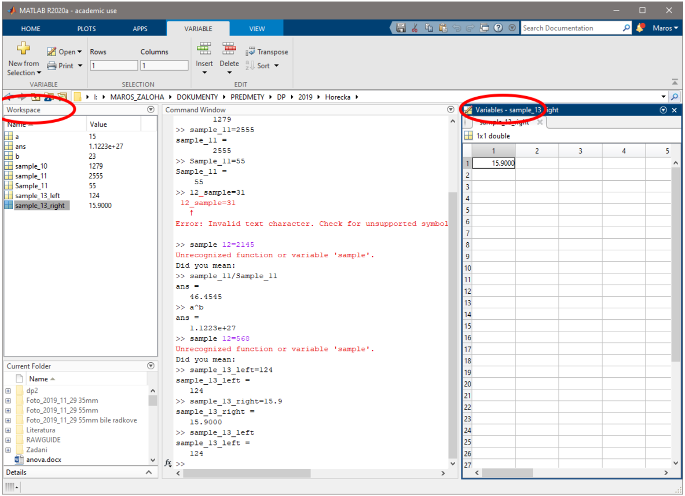
- **Example:** Create a few variables:

```
>> a=15
a = 15

>> b=23
b = 23

>> a^b
ans = 1.1223e+27

>> sample_10=1279
sample_10 = 1279

>> sample_11=2555
sample_11 = 2555

>> Sample_11=55
Sample_11 = 55

>> sample_11/Sample_11
ans = 46.4545
```

# Variables

```
>> 12_sample=31
Error:  Invalid text character.  Check for unsupported symbol,
invisible character, or pasting of non-ASCII characters.

>> sample 12=2145
Unrecognized function or variable 'sample'.

>> sample_13_left=124
sample_13_left = 124

>> sample_13_right=15.9
sample_13_right = 15.9000

>> sample_13_left
sample_13_left = 124
```

# Variables



Rows: 1 Columns: 1

| Name | Value |
|---|---|
| a | 15 |
| ans | 1.1223e+27 |
| b | 23 |
| sample_10 | 1279 |
| sample_11 | 2555 |
| Sample_11 | 55 |
| sample_13_left | 124 |
| sample_13_right | 15.9000 |

```
         1279
>> sample_11=2555
sample_11 =
         2555
>> Sample_11=55
Sample_11 =
    55
>> 12_sample=31
 12_sample=31
   ↑
Error: Invalid text character. Check for unsupported symbol

>> sample 12=2145
Unrecognized function or variable 'sample'.
Did you mean:
>> sample_11/Sample_11
ans =
   46.4545
>> a^b
ans =
   1.1223e+27
>> sample 12=568
Unrecognized function or variable 'sample'.
Did you mean:
>> sample_13_left=124
sample_13_left =
   124
>> sample_13_right=15.9
sample_13_right =
   15.9000
>> sample_13_left
sample_13_left =
   124
fx >>
```

Current Folder — Name
- dp2
- Foto_2019_11_29 35mm
- Foto_2019_11_29 55mm
- Foto_2019_11_29 55mm bile radkove
- Literatura
- RAWGUIDE
- Zadani
- anova.docx

1x1 double

|  | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 15.9000 |  |  |  |  |

**Figure:** Variables.

# Variables

- some built-in variable names

| Variable Name | Description |
|---|---|
| » ans | default variable name using for storing the last result |
| » pi | 3.1415926 . . . |
| » eps | the smallest positive number that added to 1 creates a result larger than 1 |
| » inf | representation for positive infinity |
| » NaN or » nan | not-a-number |
| » i nebo » j | imaginary unit, $\sqrt{-1}$ |

built-in variable names can be overwritten, eg:

```
>> pi = 78;
>> pi
>> clear pi
>> pi
```

# Variables

| Command | Operation |
|---|---|
| » exist | check existence of variable name, function, directory |
| » namelengthmax | maximum length of variable name |
| » who | list o variables in workspace |
| » whos | detailed list |
| » clear variable | removing variables |
| » clear all | removing all variables |
| » clc | clear command window |

# Variables

```
>> exist pi
ans = 5

>> namelengthmax
ans = 63

>> who
Your variables are:
Sample_11        b                sample_13_left
a                sample_10        sample_13_right
ans              sample_11

>> whos
  Name                   Size               Bytes  Class     Attributes
  Sample_11              1x1                    8  double
  a                      1x1                    8  double
  ans                    1x1                    8  double
  b                      1x1                    8  double
  sample_10              1x1                    8  double
  sample_11              1x1                    8  double
  sample_13_left         1x1                    8  double
  sample_13_right        1x1                    8  double
```

# Saving and Loading Data

| Command | Operation |
|---|---|
| » save filename | saving workspace to binary file (extension *.mat) |
| | Menu: File - Save Workspace as |
| » cd | current directory path |
| » cd path | sets the current directory |
| » dir | list of current directory content |
| » what | list of files with extension *.mat, *.m |
| » load filename | load variables from file |
| | Menu: File - Open |
| » delete filename | deletes the file |
| » edit filename | opens the file in the Editor |
| » type filename | the contents of a file |
| » exit | terminate MatLab |

## Saving and Loading Data

```
>> save example_1

>> clear sample_10

>> clear all

>> clc

>> who

>> load example_1

>> who

Your variables are:
Sample_11       ans             sample_10       sample_13_left
a               b               sample_11       sample_13_right
```

## Saving and Loading Data

- MatLab remembers the commands that were last used and are stored in Command History. You can use the up and down arrow keys to scroll through the command history
- after a command or operation is written in the command window, it is executed and the response is displayed in the Command Window
- ; a semicolon is used to suppress the statement in the Command Window

| Command | Operation |
|---|---|
| ; | suppressing output in Command Window |
| % | indicates a note, and commands are not executed |
| ... | continuation of expression |

# Saving and Loading Data

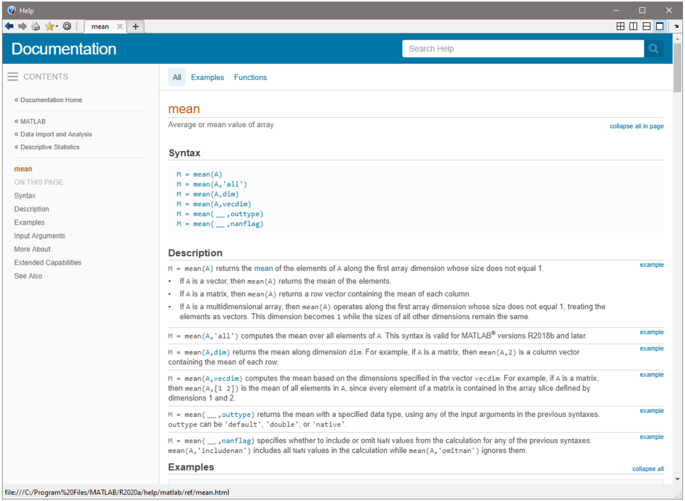
```
>> c=15
c = 15

>> d=28;

>> % commands after are not executed

>> e=127/15+ ...
22

e = 30.4667
```

# Help

There are more possibilities to get information about functions in MatLab. Help does not provide only information about functions and commands, it contains examples, but also refers to other related help features.

| Command | Operation |
|---|---|
| » help | display the list of possible topics |
| » help functionname | description and the syntax for functionname |
| » which functionname | cpath for the functionname |
| » doc | separated Help Window<br>Home - Help |
| » doc functionname | help function in a separate window |
| » function( | whisperer |
| » lookfor keyword | a list of the commands they contain keyword in a brief description of function |

# Working Environment



## mean

Average or mean value of array

### Syntax

```
M = mean(A)
M = mean(A,'all')
M = mean(A,dim)
M = mean(A,vecdim)
M = mean( __ ,outtype)
M = mean( __ ,nanflag)
```

### Description

`M = mean(A)` returns the mean of the elements of `A` along the first array dimension whose size does not equal 1.

- If `A` is a vector, then `mean(A)` returns the mean of the elements.
- If `A` is a matrix, then `mean(A)` returns a row vector containing the mean of each column.
- If `A` is a multidimensional array, then `mean(A)` operates along the first array dimension whose size does not equal 1, treating the elements as vectors. This dimension becomes `1` while the sizes of all other dimensions remain the same.

`M = mean(A,'all')` computes the mean over all elements of `A`. This syntax is valid for MATLAB® versions R2018b and later.

`M = mean(A,dim)` returns the mean along dimension `dim`. For example, if `A` is a matrix, then `mean(A,2)` is a column vector containing the mean of each row.

`M = mean(A,vecdim)` computes the mean based on the dimensions specified in the vector `vecdim`. For example, if `A` is a matrix, then `mean(A,[1 2])` is the mean of all elements in `A`, since every element of a matrix is contained in the array slice defined by dimensions 1 and 2.

`M = mean( __ ,outtype)` returns the mean with a specified data type, using any of the input arguments in the previous syntaxes. `outtype` can be `'default'`, `'double'`, or `'native'`.

`M = mean( __ ,nanflag)` specifies whether to include or omit `NaN` values from the calculation for any of the previous syntaxes. `mean(A,'includenan')` includes all `NaN` values in the calculation while `mean(A,'omitnan')` ignores them.

### Examples

**Figure:** Matlab Help.

## Display format

By default, MatLab displays numbers with four decimal places (format short):

| Command | $\pi$ |
|---|---|
| » format short | 3.1416 |
| » format long | 3.141592653589793 |
| » format short e | 3.1416e+000 |
| » format long e | 3.141592653589793e+000 |
| » format short g | 3.1416 |
| » format long g | 3.14159265358979 |
| » format short eng | 3.1416e+000 |
| » format long eng | 3.14159265358979e+000 |
| » format bank | 3.14 |
| » format rat | 355/113 |

# Some Mathematical Built-in Functions

| Command | Operation |
|---|---|
| » sin | sine |
| » cos | cosine |
| » tan | tangent |
| » asin | inverse sine |
| » acos | inverse cosine |
| » atan | inverse tangent |
| » exp | exponential |
| » log | natural logarithm |
| » log10 | common (base 10) logarithm |
| » abs | absolute value |
| » round | round to nearest integer |
| » fix | round toward zero |
| » floor | round toward $-\infty$ |
| » ceil | round toward $\infty$ |
| » sign | function signum |
| ⋮ | ⋮ |

## Examples for practice

## Examples for practice

1. Evaluate following expressions, where $a$ = -2, $b = 1$ and $c = 1.5$.

$$A = a + \frac{3b^2}{-a^3} + 2c - 1$$

$$B = \frac{(a + 3b)^2}{(-a^3 + 2)c}$$

2. Let $m = 2.05$ g be a weight of yarn, $l = 100$ m be a length of yarn. Find fineness of yarn $T$ in [tex]
3. Let $\rho_{SS} = 7500$ kg/m$^3$ be a density of stainless steel and $\rho_{PP} = 910$ kg/m$^3$ be a density of polypropylene circular cross-sectional shape fiber. Find diameter $d$ [$\mu$m] of these fibers having fineness $t = 2$ dtex
4. Find the name of the function for calculating the sample variance
5. Find the name of the function for calculating the correlation coefficient
6. Find the name of the function for converting angles from radians to degrees

## Solution