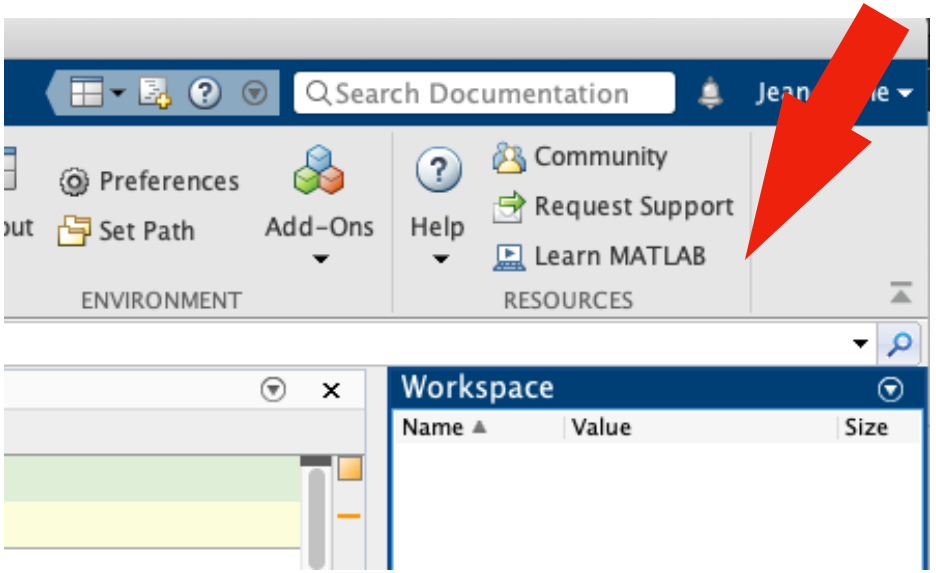
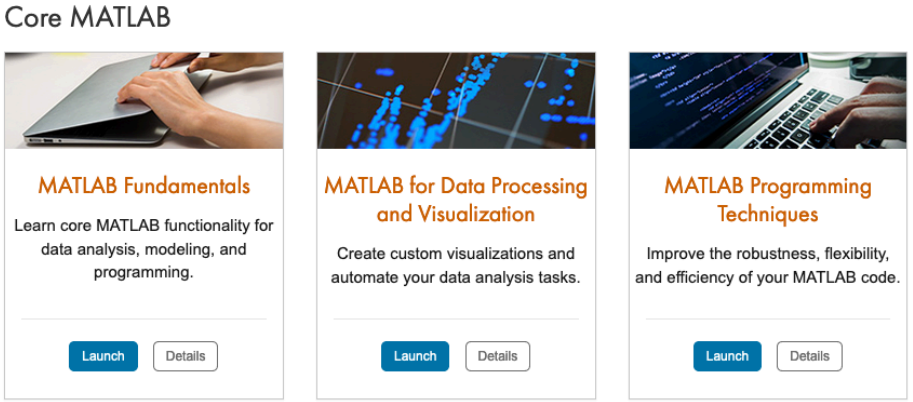
A short introduction to MATLAB

jean-marie.furbringer@epfl.ch

Crash course

- A base to be developed afterwards
- Ask questions!
- Try things
- Read the help notices





Modules - program

- **Matlab on ramp**: if you never had practice in programming or do not feel confortable with new softwares → today
- **Matlab fundamentals**: 18 chapters → **till Monday**
- **Matlab for Data Processing and visualization:** 11 chapters →Not necessary for the DOE course, if you had to import data from files in your work
- **Matlab programming technique**: 10 chapters→Not necessary for the DOE course, interesting if you will have code frequently in Matlab
- **Introduction to symbolic math with Matlab**: cool, powerful,13 chapters → Nice to have for DOE (and as a basic competence) → **till Wednesday**
- **Introduction to statistical method with Matlab**: a short module, 5 chapters → interesting if you have time

Reference

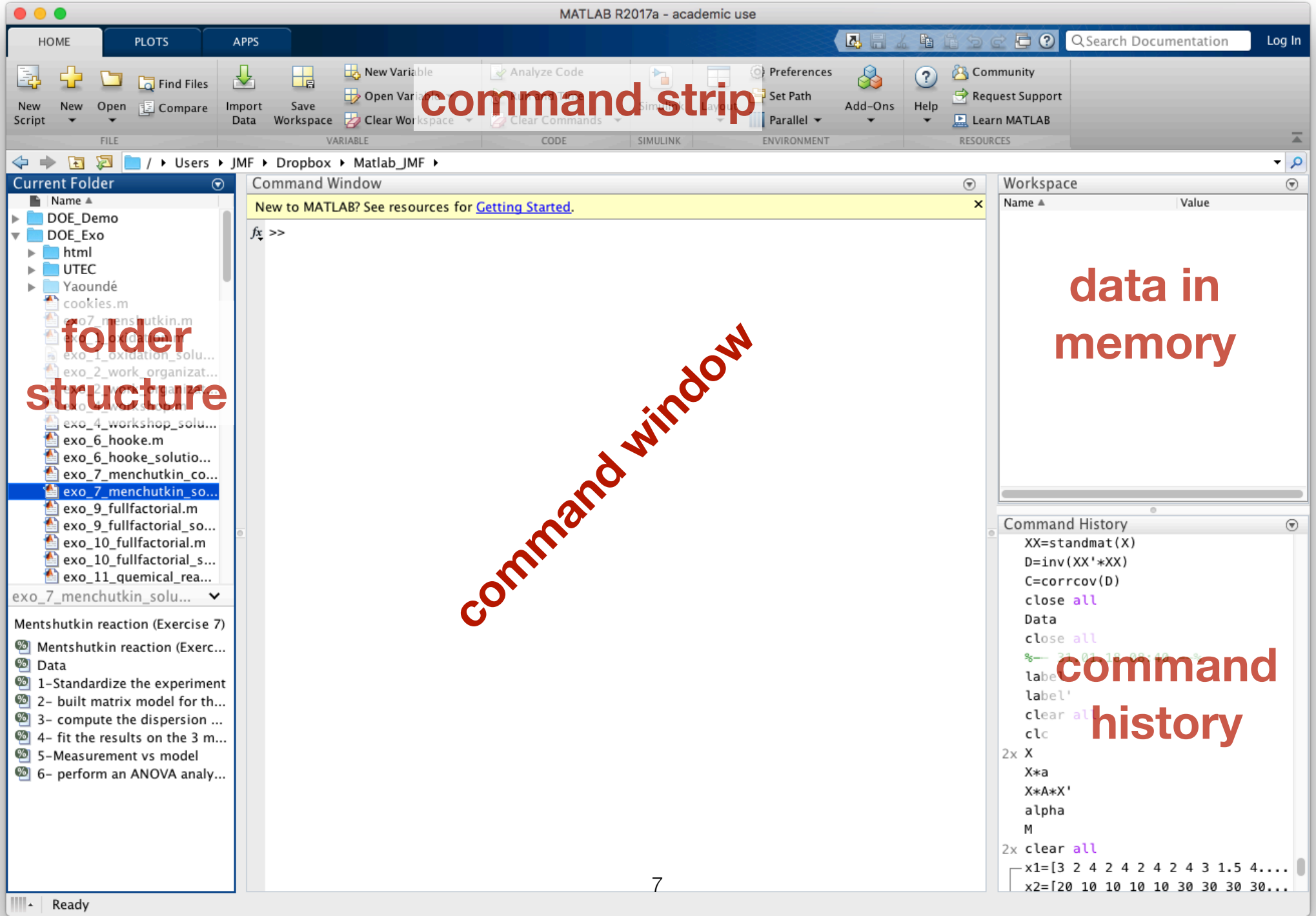
- **Introduction to MATLAB for engineers and scientists**, Sandeep Nagar, Apress, 2017, isbn 978-1-4842-3188-3
- **Matlab for Engineers**, Holly Moore, Pearson, 2015, isbn 978-0-133-48597-4

- Who are you?
- What is your previous experience with Matlab?

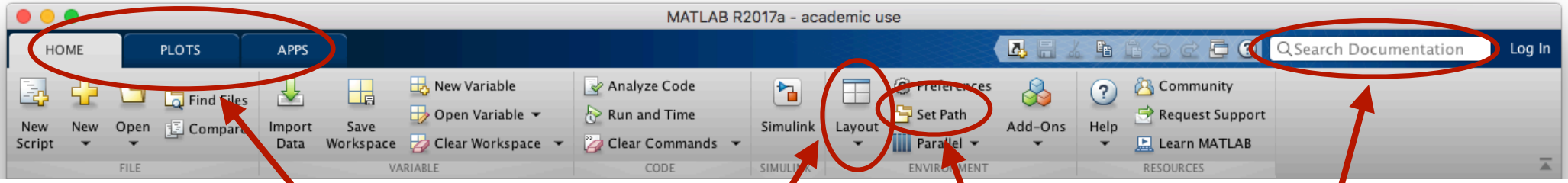
Matlab is a “REPL”

- Reads
- Evaluates
- Prints
- Loops back

User interface



Elements of the interface



Tabs

to access different tools/command

layout menu

path= list of folders where Matlab goes to find files

help

- Menus
- Tabs
- Folders and path
- Help
- Layout

So, it is time to jump in!

Matlab as a calculator

- 3+5
- 6-10
- 3.5*4
- 2/3
- 2+5*2
- (2+5)*2
- sqrt(25)
- exp(1)

- sin(pi/6)
- exp(3)
- atan(10)/pi
- log10(10)
- log(10)

When MATLAB has no variable where to put the results of a calculation, the result is put in the variable 'ans' (answer)

Matlab as a calculator

Modifying Variables

You can use the **Up** (↑) and **Down** (↓) arrows on the keyboard to scroll through previous commands.

Command: Format style

Possible styles

- short / long : fix-decimal format 4 or 15 digits
- shortE / longE : scientific notation 4 or 15 digits
- shortEng / longEng : engineering notation 4 or 15 digits
- shortG / longG: fixed-decimal format or scientific notation, whichever is more compact
- bank : two digits after the decimal point
- + : positive/negative format
- hex : hexadecimal
- rat : rational
- format alone set back the standard which is the short, fixed-decimal format for floating-point notation and loose line spacing for all output lines

> Test the different formats for A=[1, 4.56565656, 1/3, 1.56e-5]

Self learning and getting help

- On line help
  - `>> help bbdesign`
  - `>> lookfor 'design of experiment'`
- "help/documentation" in the menu
- "Learn Matlab" in the command strip

Variables

- **Data types:** logical, char, int8, int16, int32, int64, single, double,…

- **Naming**
  - Names should not start with a number; however, numbers can be used anywhere afterward.
  - Variable names are case sensitive.
  - *Keywords* cannot be used as names.
  - Names can include underscores (_).

Variables

- **Data types:** logical, char, int8, int16, int32, int64, single, double,…

- **Try**

  `a=123456789012345` with short and long format

  ```
  b=int8(123456789012345)
  ```

  ```
  c=int16(123456789012345)
  ```

  ```
  d=int32(123456789012345)
  ```

  ```
  f=int16(32768)
  ```

- check attentively what you get back

Clear - Save

- `clear variable_name`
- `clear all`
- `clc`

- `save(file_name var_name)`
- `save(file_name var_name, fmt)`
- `save(file_name)`

Format:

- `'-mat'`: binary
- `'-acsii','-tabs'`: Tab-delimited text format with 8 digits of precision

Array based computing

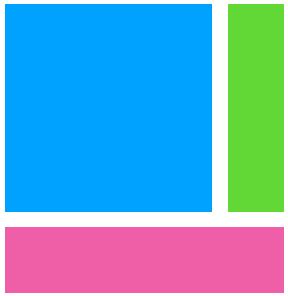
- An horizontal vector: `A=[ 123 45.5 301 ]` or equivalently `A=[ 123, 45.5, 301 ]`
- A vertical vector: `B=[10;15;20]` equivalent to `B=[10,15,20]'`
- The prime sign « ‘ » means « transpose »
- Regular array `C=0:0.25:5`
- 2D arrays `D=[1 5;5 -2]`
- Useful fonctions `ones(Ni,Nj)`, `zeros(Ni,Nj)`, `repmat(D,2,3)`, `eye(N)`

Array based computing

- Linear combinations of arrays of the same size and same class `C=A+B`
- Matrix multiplication `D=A*B`
- Inversion `D=A/B` (eq. to `D=A*inv(B)`) or `D=A\B` (eq. to `D=inv(A) * B`)
- Operations element by element: .*, ./ ; .^

Try these manipulations

- `A=[ones(3),zeros(3,1);ones(1,4)]`
- `B=rand(4), B2=rand(4,2)`
- `C=A.*B`
- `D=eye(4)/C`
- `E=D*C`
- `E=int8(E)`
- `S=size(E)`



Built-in functions

- find()
- sort(A, ‘descent’)
- sortrows()
- trace(A)
- reshape()

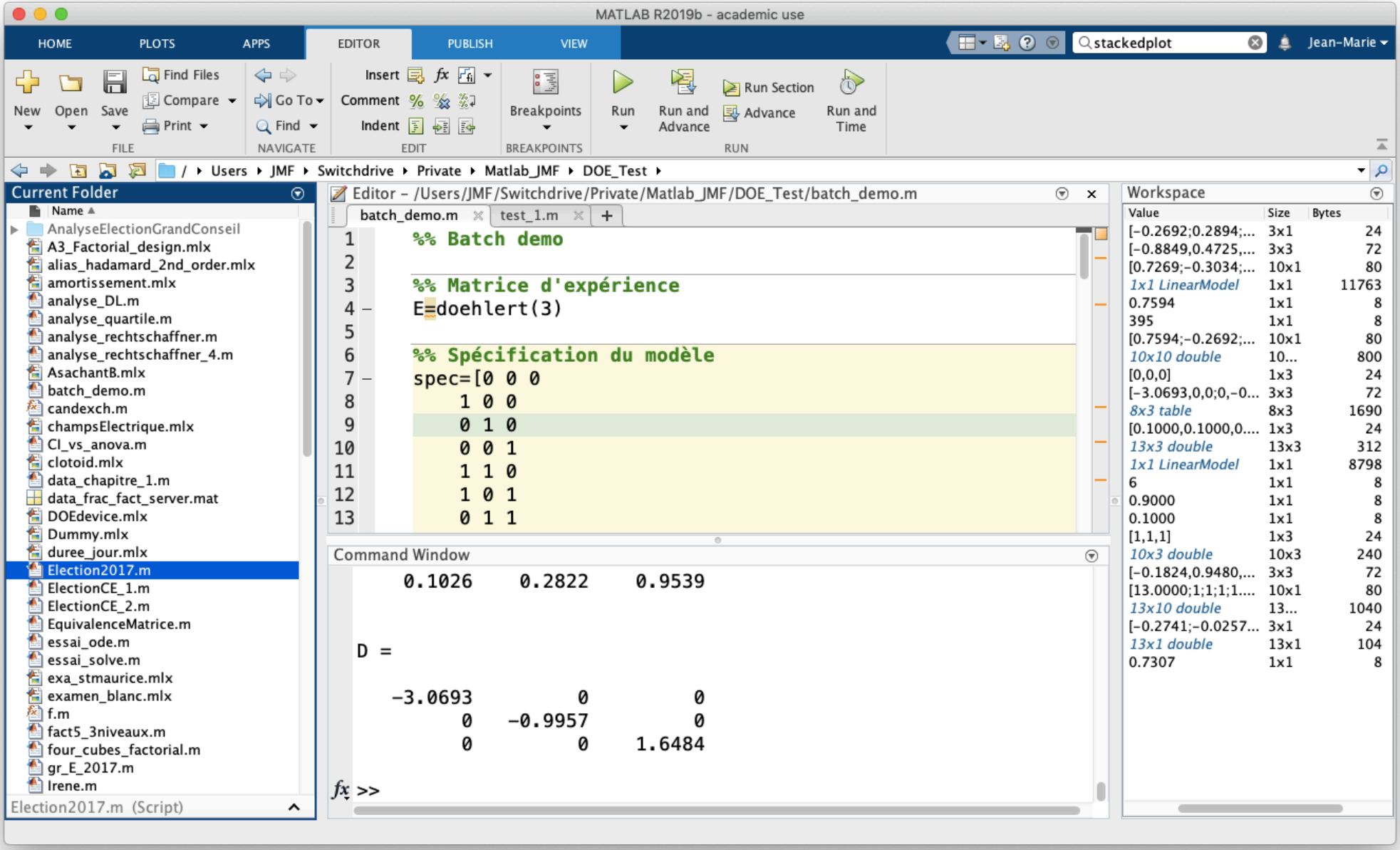
Using indices

- A=ones(4)
- A(1,1)=0
- A(1:2,1:2)=zeros(2)

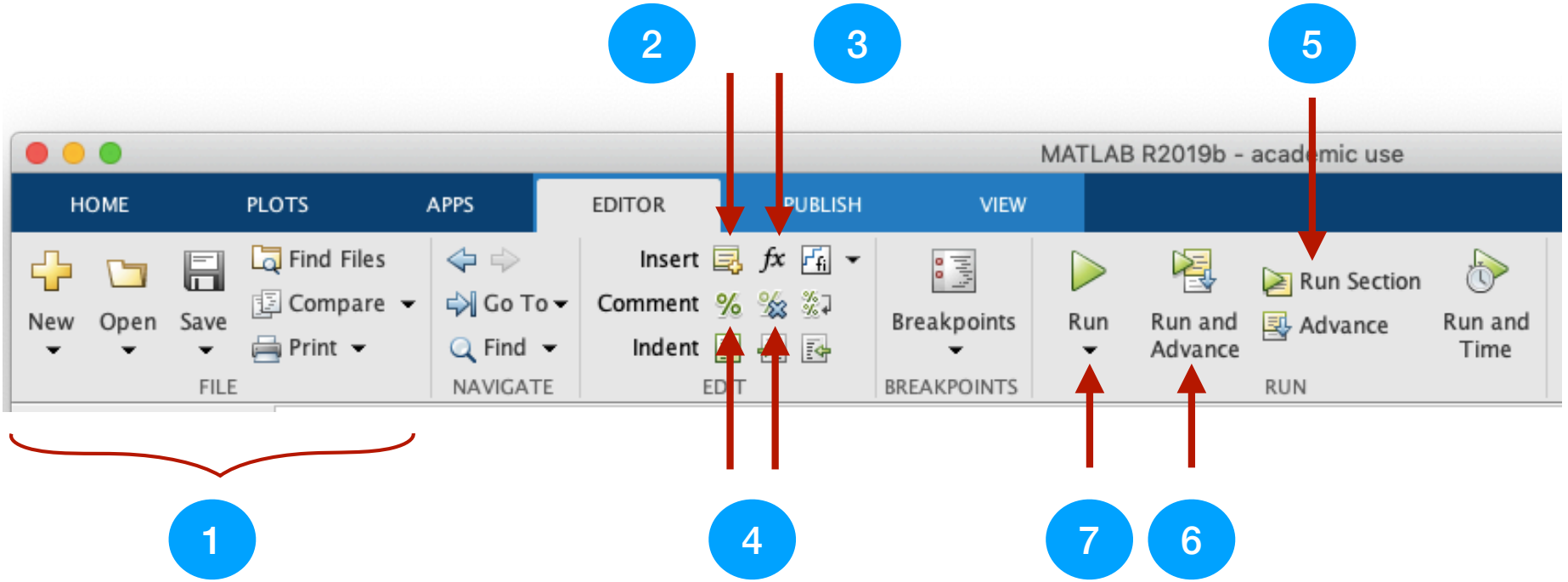
Script

- A script file is a text file where commands are listed, read to be executed ‘as in the command window’
- call m-file because the extension is ‘.m’
- can be separated in cells, with ‘%%’ and that can be executed individually
- it is comparable to a ‘program’, but no variable definition is necessary (excepted for *global variables*, and preallocation)

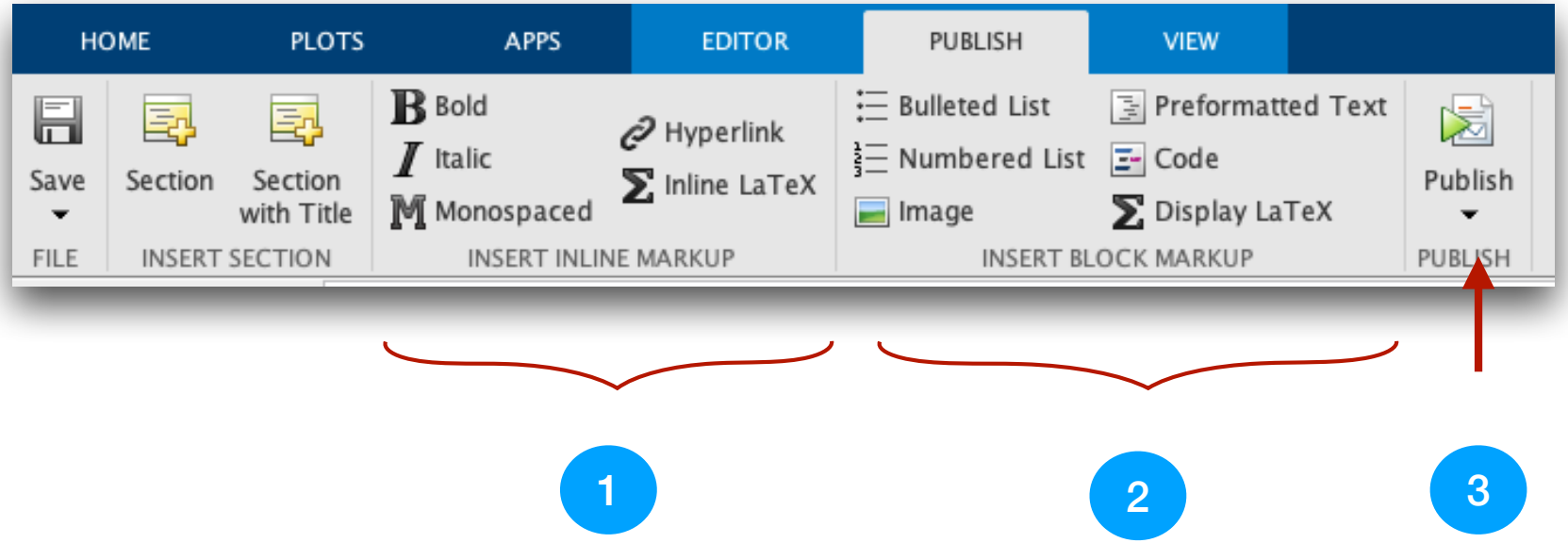
Script

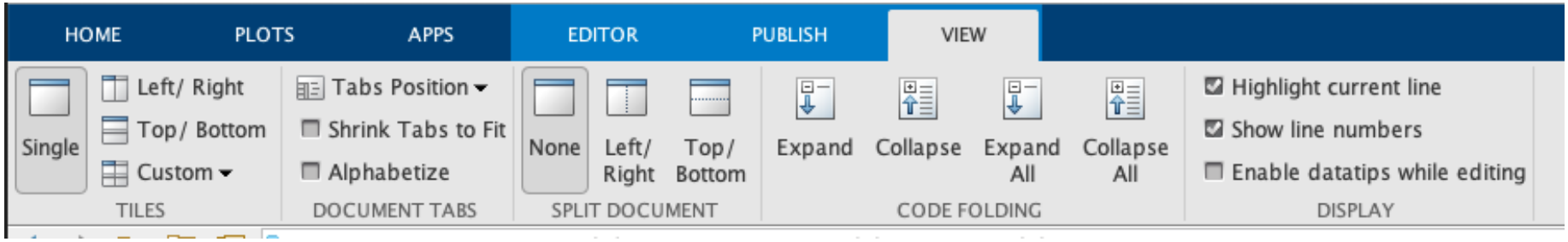

MATLAB R2019b - academic use
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
stackedplot
Jean-Marie
New
Open
Save
Find Files
Compare
Print
FILE
Go To
Find
NAVIGATE
Insert
Comment
Indent
EDIT
Breakpoints
BREAKPOINTS
Run
Run and Advance
Run Section
Advance
Run and Time
RUN
/ ▸ Users ▸ JMF ▸ Switchdrive ▸ Private ▸ Matlab_JMF ▸ DOE_Test ▸
Current Folder
Name
AnalyseElectionGrandConseil
A3_Factorial_design.mlx
alias_hadamard_2nd_order.mlx
amortissement.mlx
analyse_DL.m
analyse_quartile.m
analyse_rechtschaffner.m
analyse_rechtschaffner_4.m
AsachantB.mlx
batch_demo.m
candexch.m
champsElectrique.mlx
CI_vs_anova.m
clotoid.mlx
data_chapitre_1.m
data_frac_fact_server.mat
DOEdevice.mlx
Dummy.mlx
duree_jour.mlx
Election2017.m
ElectionCE_1.m
ElectionCE_2.m
EquivalenceMatrice.m
essai_ode.m
essai_solve.m
exa_stmaurice.mlx
examen_blanc.mlx
f.m
fact5_3niveaux.m
four_cubes_factorial.m
gr_E_2017.m
Irene.m
Election2017.m (Script)
Editor - /Users/JMF/Switchdrive/Private/Matlab_JMF/DOE_Test/batch_demo.m
batch_demo.m
test_1.m
1 %% Batch demo
2
3 %% Matrice d'expérience
4 E=doehlert(3)
5
6 %% Spécification du modèle
7 spec=[0 0 0
8 1 0 0
9 0 1 0
10 0 0 1
11 1 1 0
12 1 0 1
13 0 1 1
Command Window
0.1026 0.2822 0.9539
D =
-3.0693 0 0
0 -0.9957 0
0 0 1.6484
fx >>
Workspace
Value Size Bytes
[-0.2692;0.2894;... 3x1 24
[-0.8849,0.4725,... 3x3 72
[0.7269;-0.3034;... 10x1 80
1x1 LinearModel 1x1 11763
0.7594 1x1 8
395 1x1 8
[0.7594;-0.2692;... 10x1 80
10x10 double 10... 800
[0,0,0] 1x3 24
[-3.0693,0,0;0,-0... 3x3 72
8x3 table 8x3 1690
[0.1000,0.1000,0.... 1x3 24
13x3 double 13x3 312
1x1 LinearModel 1x1 8798
6 1x1 8
0.9000 1x1 8
0.1000 1x1 8
[1,1,1] 1x3 24
10x3 double 10x3 240
[-0.1824,0.9480,... 3x3 72
[13.0000;1;1;1;1.... 10x1 80
13x10 double 13... 1040
[-0.2741;-0.0257... 3x1 24
13x1 double 13x1 104
0.7307 1x1 8

Script


2
3
5
MATLAB R2019b - academic use
HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
New
Open
Save
Find Files
Compare
Print
FILE
Go To
Find
NAVIGATE
Insert
Comment
Indent
EDIT
Breakpoints
BREAKPOINTS
Run
Run and Advance
Run Section
Advance
RUN
Run and Time
1
4
7
6

Script


HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Save
Section
Section with Title
Bold
Italic
Monospaced
Hyperlink
Inline LaTeX
Bulleted List
Numbered List
Image
Preformatted Text
Code
Display LaTeX
Publish
FILE
INSERT SECTION
INSERT INLINE MARKUP
INSERT BLOCK MARKUP
PUBLISH
1
2
3


HOME
PLOTS
APPS
EDITOR
PUBLISH
VIEW
Single
Left/ Right
Top/ Bottom
Custom
Tabs Position
Shrink Tabs to Fit
Alphabetize
None
Left/ Right
Top/ Bottom
Expand
Collapse
Expand All
Collapse All
Highlight current line
Show line numbers
Enable datatips while editing
TILES
DOCUMENT TABS
SPLIT DOCUMENT
CODE FOLDING
DISPLAY

Try this

- Open en new scrip file
- Write as comment “ this is a test file”
- Write the command “ $A = \cos \frac{\pi}{6}$ ”
- Save the file in the current directory as “test1.m”
- Execute the file test1.m in the command window

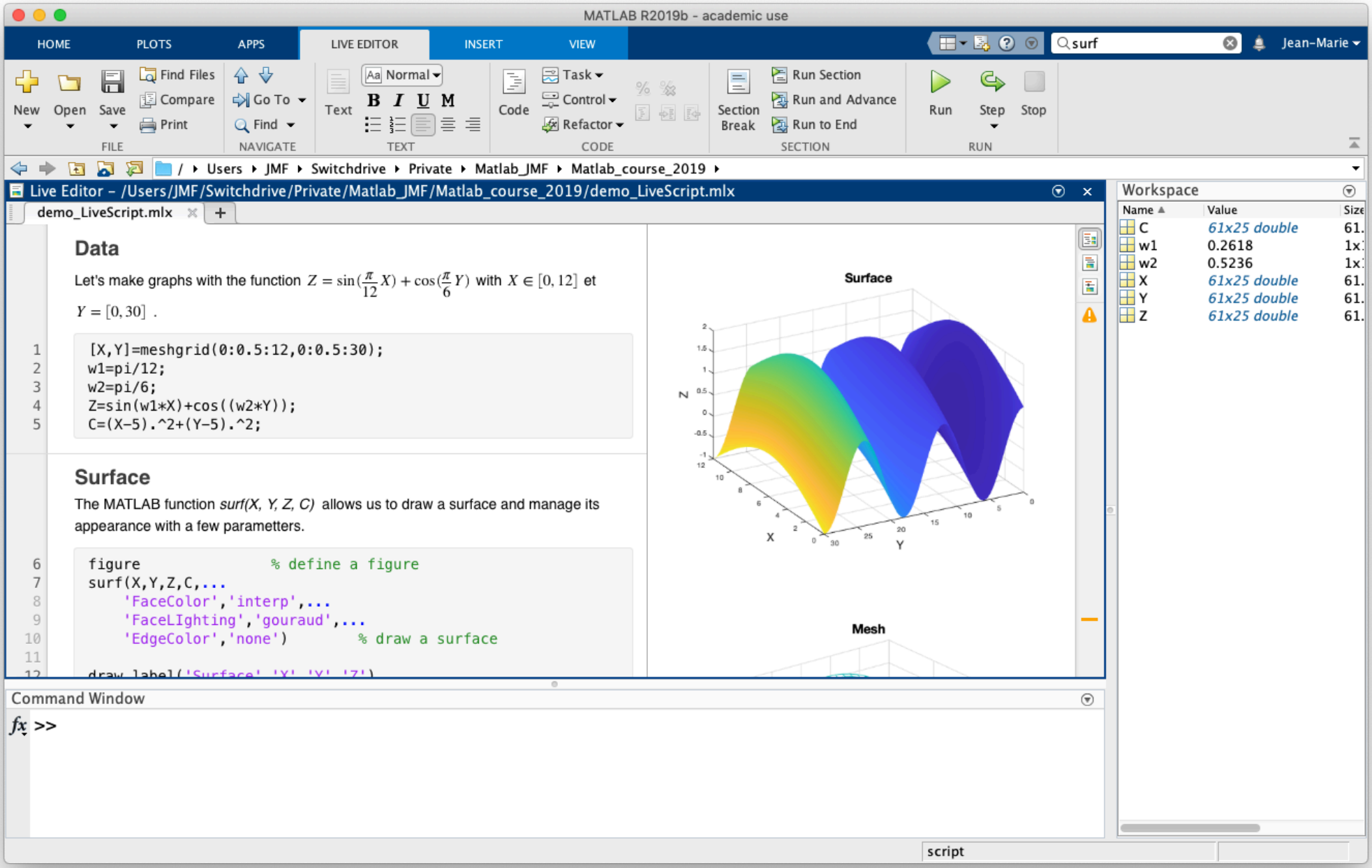
Publish as latex, html or pdf

- Menu ‘Publish’
- Editing publishing options

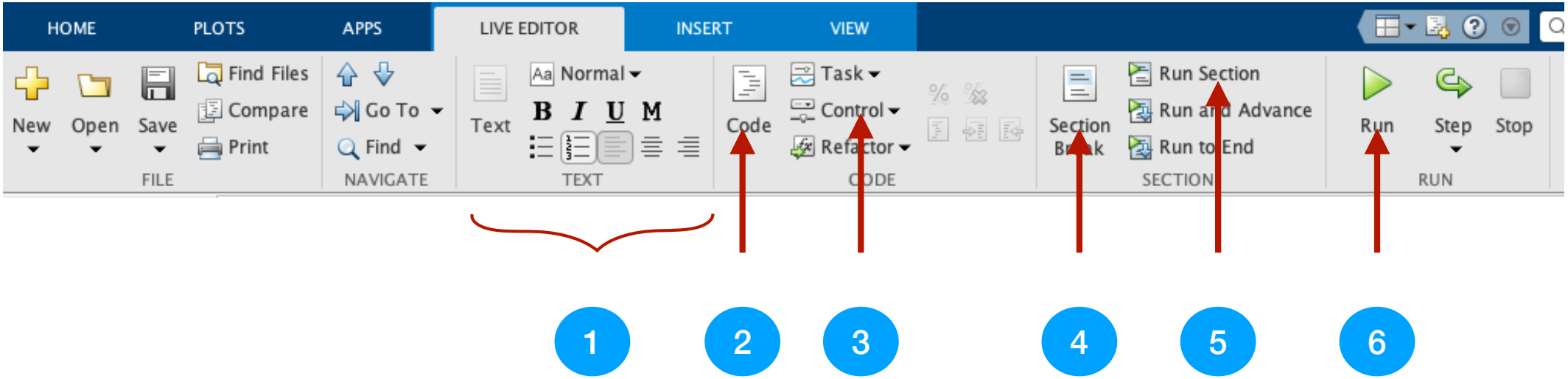
Program

1. **Base**: day 1
2. **Live editor, batches and functions**: day 1
3. **Data class**: day 1
4. **Graphics**: day 2
5. **Symbolic calculation**: day 2

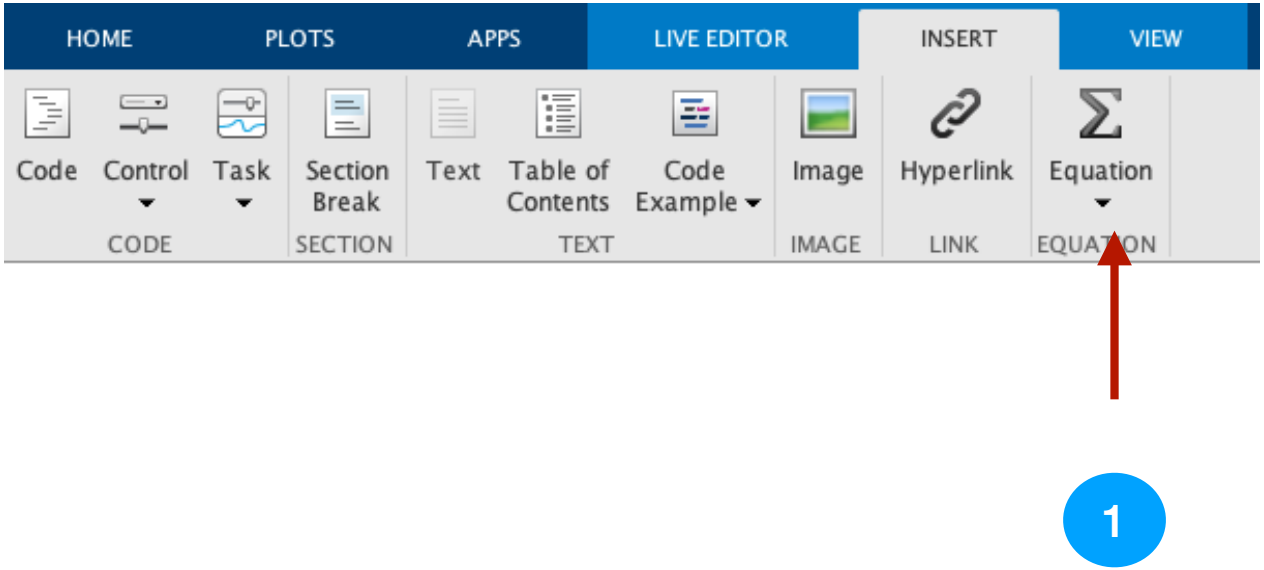
2.1 Live script



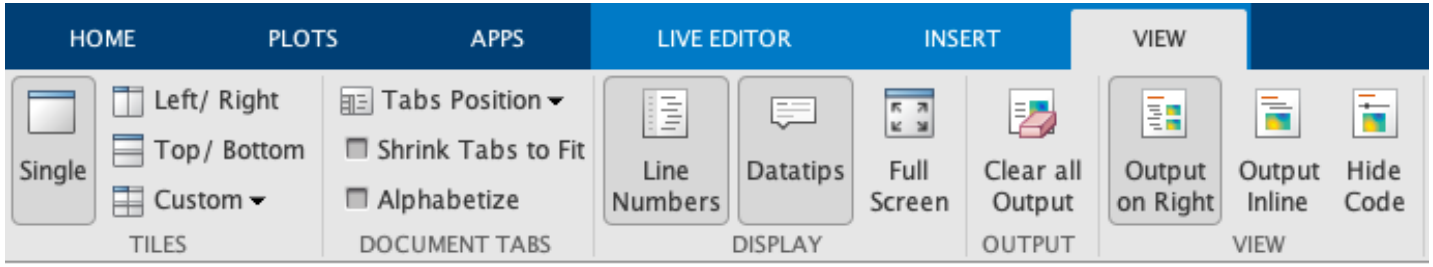
2.1 Live script - main tab



2.1 Live script - insert tab



2.1 Live script - view tab



2.2 User built functions

```
function  [out1,out2]=func_name(in1,in2, …)
    commands
end
```

- To manage the number of input and output
  - `nargin`: number of input arguments
  - `nargout`: number of output arguments
- On PC, there is an incompatibility with the declation of a function within a script
- By default, variables within the function are local variables and then not known outside of the function
- `global x` : define x as a global variable

If several functions all declare a particular variable name as `global`, then they all share a single copy of that variable.

2.3 Standard loops

- if condition, execution, end
- if condition, execution, else, execution end
- if condition, execution, else if condition, execution end
- for k=1:10, execution, end
- while k>limit, execution, end
- switch case_list, case 1 execution case 2 execution,… end

Program

1. **Base**: Monday a.m.
2. **Live editor, batches and functions**: Monday p.m.
3. **Data class**: Monday p.m.
4. **Graphics**: Tuesday a.m.
5. **Symbolic calculation**: Tuesday p.m.

3.1 Classes of data

a) Array (vectors, matrices)

- array of double `A=[1 3 5 8], B=eye(4)`
- array of integer `Ai=int16([1 3 5 8])`
- array of character `As='abcdef'`

3.1 Classes of data

b) Cell array

- `C=cell(i,j)` for initialization of a cell array with ixj void elements
- `C={data1,data2,data3,…}` build a cell array with data1, data2, …
- `C=cellstr(S)` convert to cell array of character vectors
- `C=cellstr([‘abcdef’]’)`convert the string in a column of letters

3.1 Classes of data

c) Structure

- `S=struct(field,value)` create a structure with given fields and corresponding values
- `S.FieldName=value` add a field to an existing table
- access elements with `S().FieldName()`

3.1 Classes of data

d) **Table**

- `T=table` for initialization of a (void) table
- `T = table(var1,…,varN,Name,Value)` create a table with variable var1, …
- `T.fieldname=value` add a variable to an existing table
- `T = readtable(filename,opts,Name,Value)` read a table in an external file (Excel)
- `T = array2table(A,Name,Value)` for transforma array in table (variable names can be specified)
- `summary(MyTable)` summarize the content of a table
- `T.variable=categorical(T.variable)` transform a variable in categorical
- `T.Properties.PropertyName=value`

Program

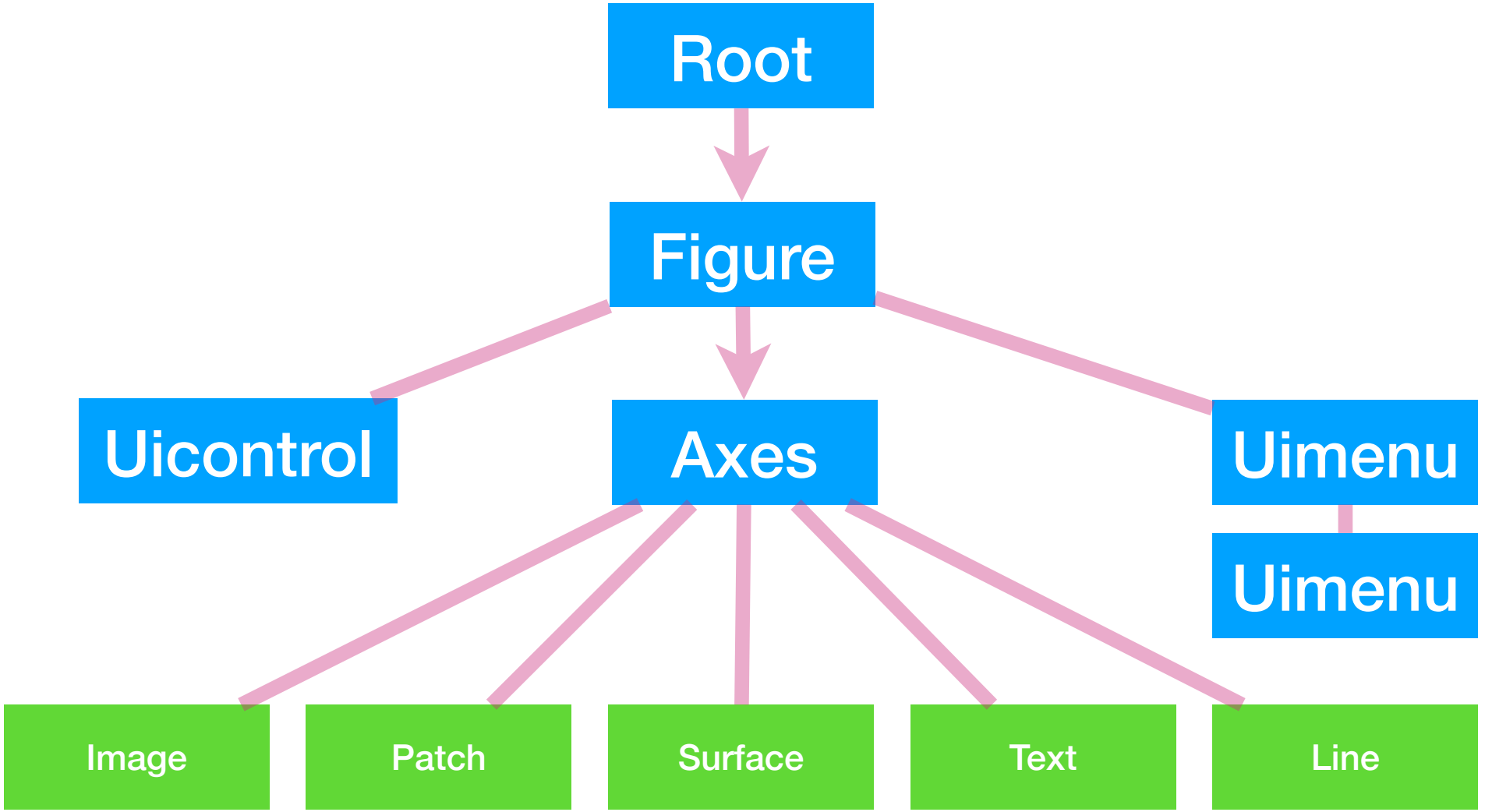
1. **Base**: day 1
2. **Live editor, batches and functions**: day 1
3. **Data class**: day 2
4. **Graphics**: day 2
5. **Symbolic calculation**: day 2

4.1 Figure window

```
figure
figure(1)
figure(Property,value)

Properties
 - Name : figure('Name','Measured Data’)
 - Number
 - Color
 - Position [left bottom width height]
  figure('Position',[400 500 1000 750])
 - Units: pixels, normalized, centimeters,
 points,…
```

4.2 Hierarchical model



4.3 Plot

- plot(x,y,LineSpec)
- plot(x1,y1,…,xn,yn)
- plot(__,Name, Value)

LineSpec

‘-‘ for hard line

‘- -‘ for dashed line

‘:’ for dotted line

o,*,.,+,x,s,d as markers

y, m, c, r, g, b, k as colors

combined: ‘r:0’

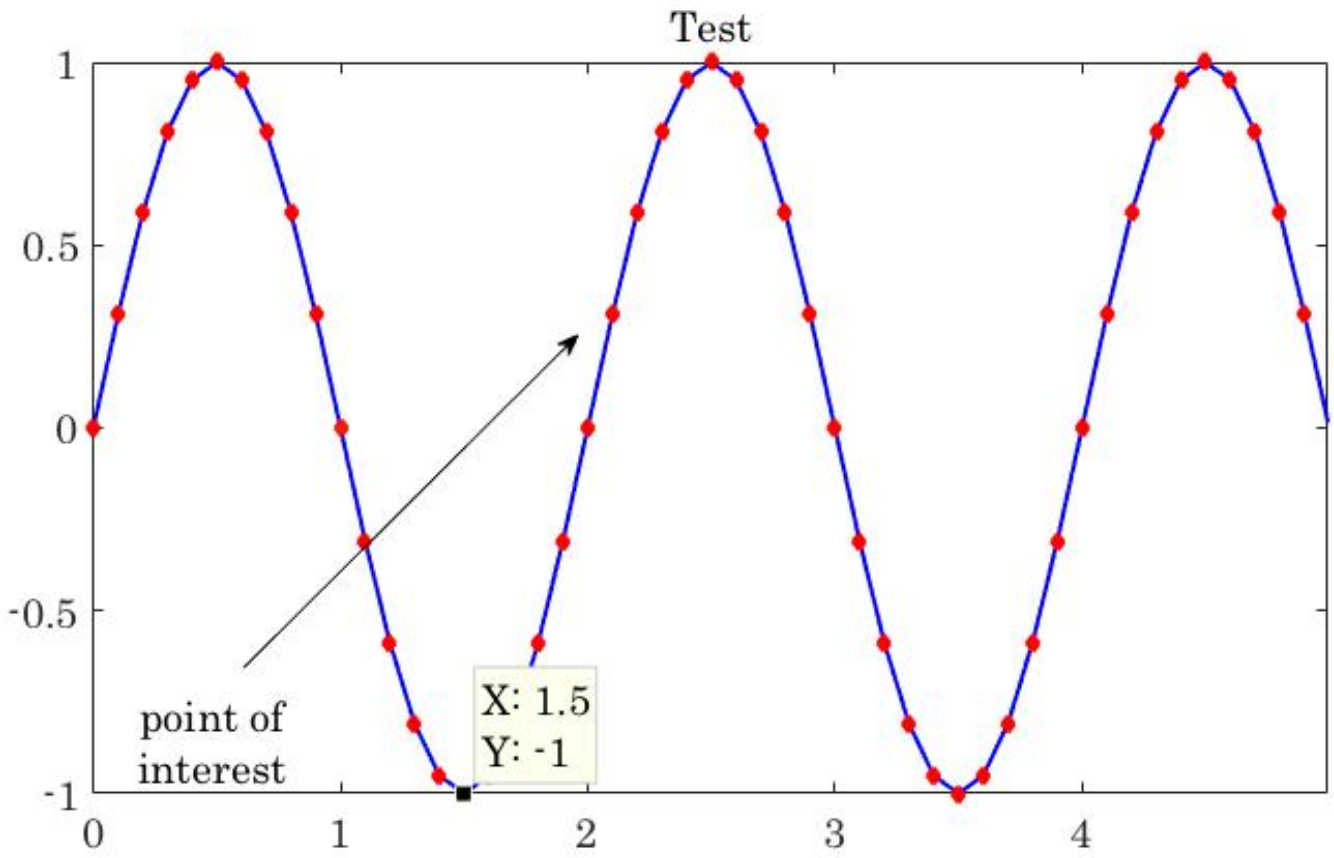
- semilogx(x,y,…)
- semilogy(x,y,…)
- loglog(x,y,…)

- stackedplot(x,Y)
- stackedplot(tbl)

4.4 A few graphical functions and key words

- `axis([x1,x2,y1,y2,z1,z2])`
- `subplot(a,b,c)`
- `title(str)`
- `xlabel(str)`
- `hold on ….hold off`
- `box on`
- `grid on`
- `get(handle[,property])`
- `set(handle,property,value)`
- `text(x,y,str)`
- `annotation(LineType,x,y)`

4.5 Editing a figure



4.6 Editing the properties of a figure

- `figure`
- `t=0:5/100:5`
- `plot(t,sin(pi*t))`
- `get(gca)`
- `set(gca, property, value)`
- `set(gca,'XTick',1:5,'XTickLabel',{'A' 'B' 'C' 'D' 'E'})`
- `title('Distribution','FontName','Time new roman', 'FontSize', 16)`

4.7 Plot 3D

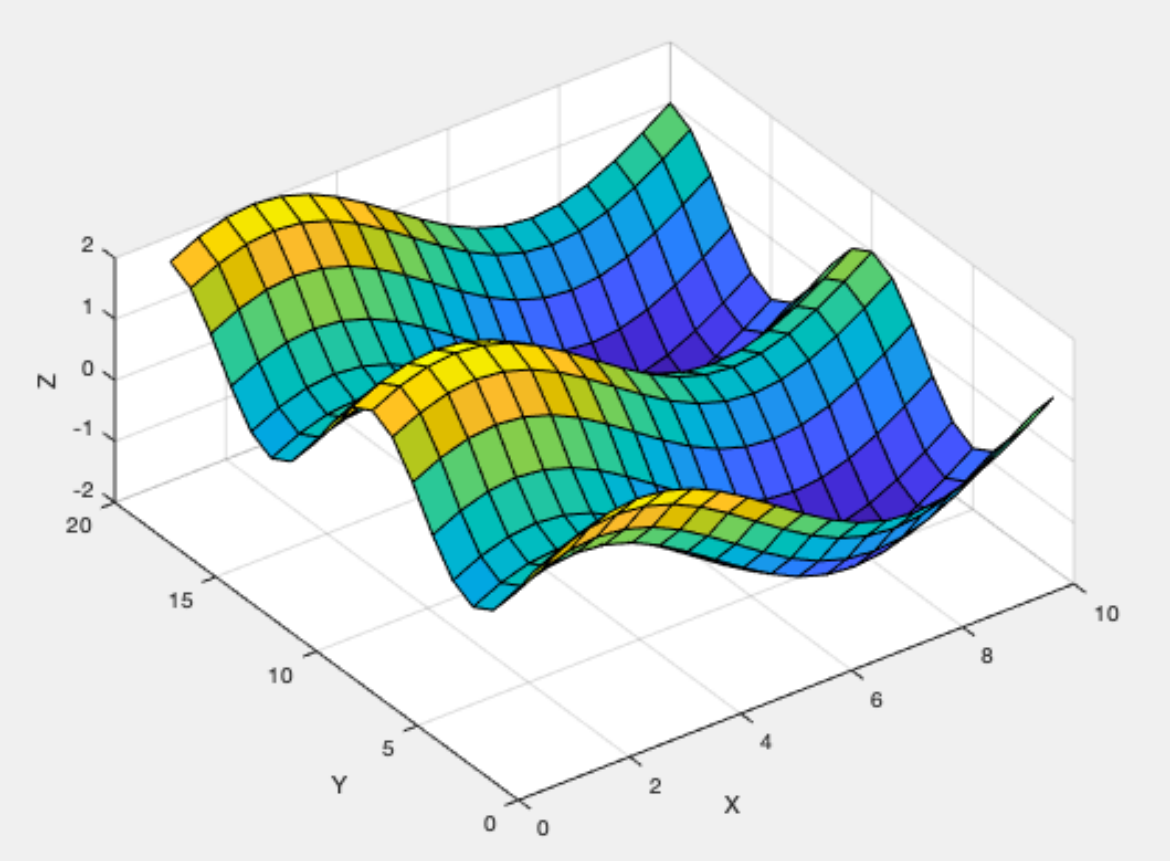
- plot3(X,Y, Z, LineSpec,…)

4.8 Surface

- surf(X,Y, Z)
- surf(X,Y, Z,C)
- mesh(X,Y, Z)
- surfc(X,Y, Z)
- contour(X,Y, Z)
- [X,Y]=meshgrid(x,y)
- colorbar

4.8 Surface

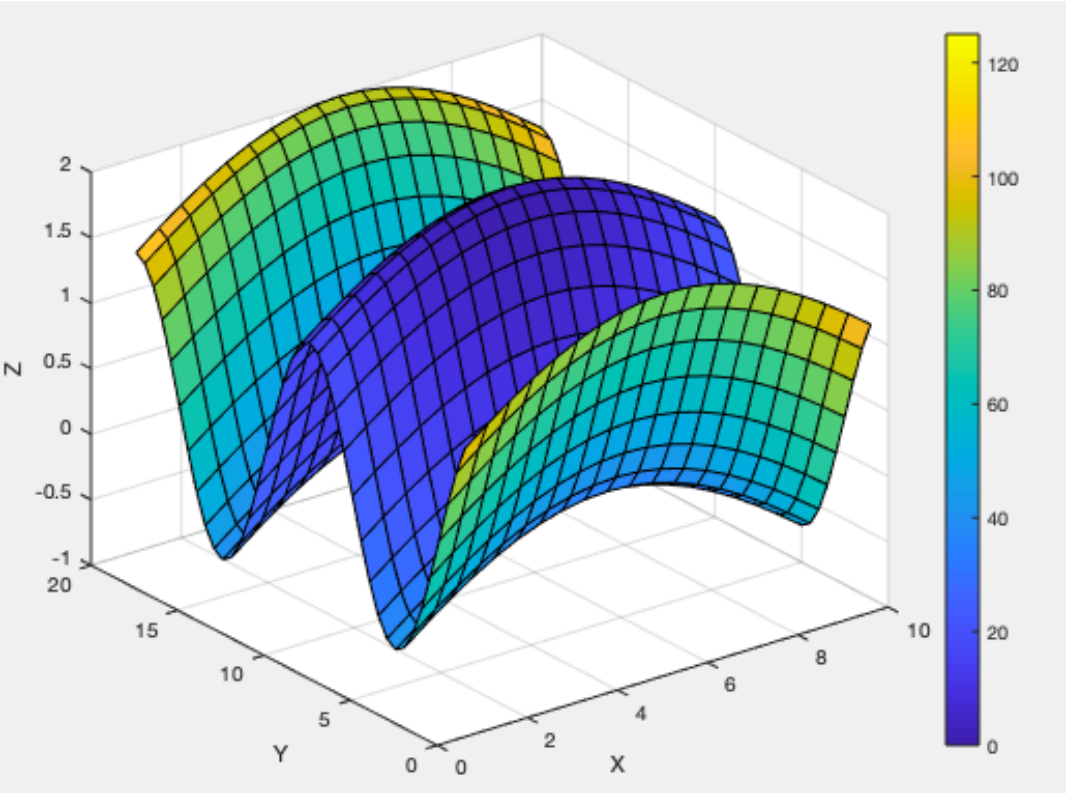
- surf(X,Y, Z)
- surf(X,Y, Z,C)
- mesh(X,Y, Z)
- surfc(X,Y, Z)




```
[X,Y]=meshgrid(1:0.5:10,1:20);
Z=sin(pi/5*X)+cos((pi/5*Y));
figure
surf(X,Y,Z)
```

4.8 Surface

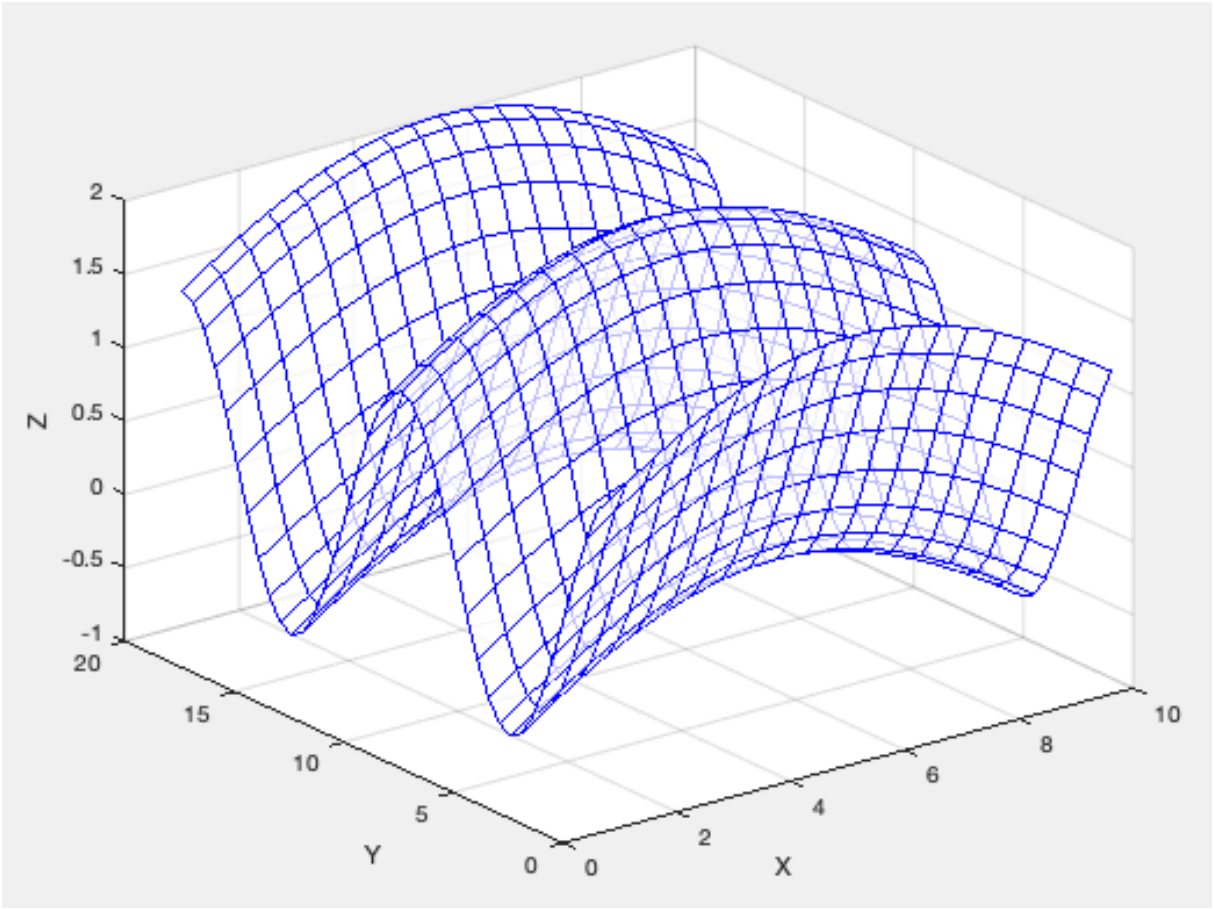
- surf(X,Y,Z)
- surf(X,Y,Z,C)
- mesh(X,Y,Z)
- surfc(X,Y,Z)




```
[X,Y]=meshgrid(1:0.5:10,1:0.5:20);
Z=sin(pi/11*X)+cos((pi/5*Y));
C=(X-5).^2+(Y-10).^2;
figure
surf(X,Y,Z,C)
```

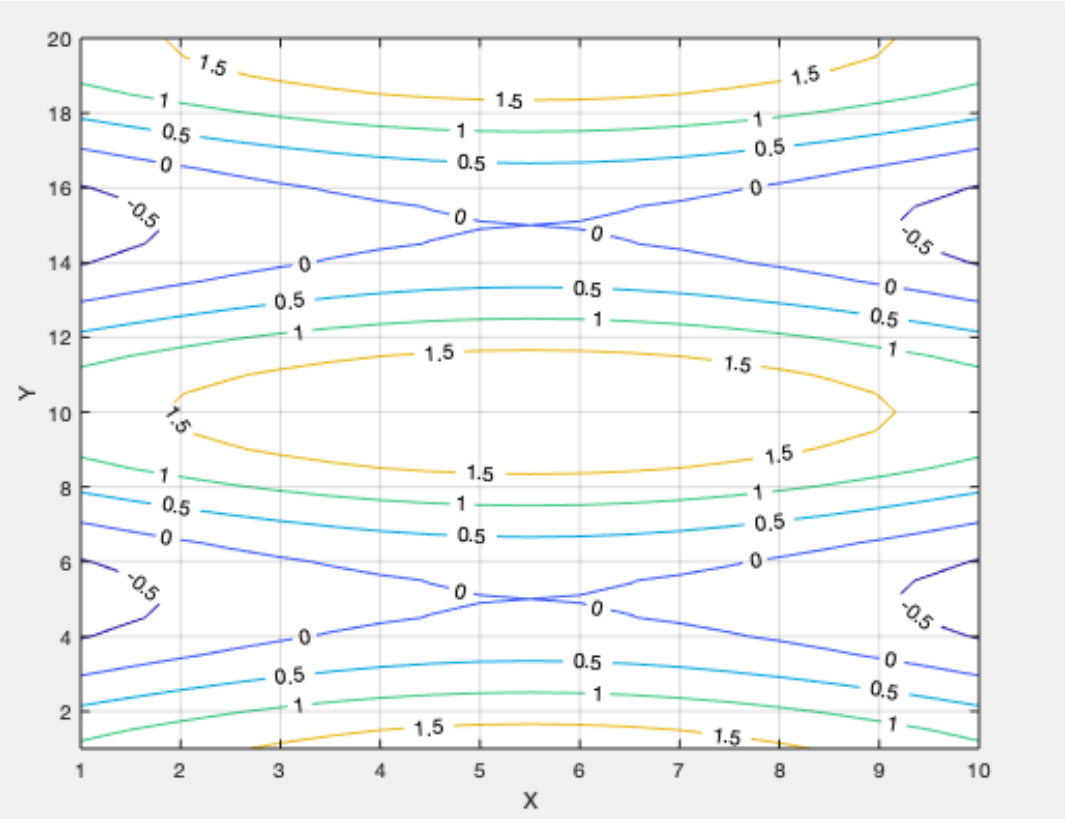
4.8 Surface

- surf(X,Y, Z)
- surf(X,Y, Z,C)
- mesh(X,Y, Z)
- surfc(X,Y, Z)




```
[X,Y]=meshgrid(1:0.5:10,1:20);
Z=sin(pi/5*X)+cos((pi/5*Y));
figure
mesh(X,Y,Z,…
'EdgeColor','b',…
'FaceAlpha',0.75)
```

4.8 Surface



- contour(X,Y,Z)
- [X,Y]=meshgrid(x,y)
- colorbar

```
[X,Y]=meshgrid(1:0.5:10,1:0.5:20);
Z=sin(pi/11*X)+cos((pi/5*Y));
figure
contour(X,Y,Z,C,'ShowText','on')
```

Program

1. **Base**: Day1
2. **Live editor, batches and functions**: Day1
3. **Data class**: Day2
4. **Graphics**: Day2
5. **Symbolic calculation**: Day2