

# OIST mini course: MATLAB

## Basics of programming and data analysis

Alexey Martyshev

# Materials for the class

---

- Go to TIDA in your web-browser
- Find information about today's class in the announcements
- Click on it
- In the description, there should be a dropbox link

# Materials for the class

---

- Go to TIDA in your web-browser
  - Find information about the class in the announcements
  - Click on it
  - In the description, there should be a dropbox link
- 
- Has everyone installed MATLAB to your laptop?

# The content of the class



# The content of the class

---

- MATLAB GUI (Graphical User Interface)
- Basics of programming in MATLAB
- Matrix computations
- Upload data to your computer
- Fitting of the model parameters to the data
- Plotting the results

# The content of the class

---

- **MATLAB GUI (Graphical User Interface)**
- Basics of programming in MATLAB
- Matrix computations
- Upload data to your computer
- Fitting of the model parameters to the data
- Plotting the results



# The content of the class



# MATLAB GUI (Graphical User Interface)

The image displays the MATLAB R2018b academic use interface. The top menu bar includes 'MATLAB', 'Window', and 'Help'. The top toolbar contains icons for 'HOME', 'PLOTS', 'APPS', 'VARIABLE', and 'VIEW'. Below the toolbar is a 'Current Folder' pane showing a list of files and folders. The main editor window displays a script named 'plot\_pf\_pointA\_27052021.m' with a table of data. The 'Command Window' at the bottom shows the execution of a script. The 'Workspace' pane on the right lists variables.

**Current Folder**

Name	Date
long_run_simspikes_model_...	2021/0...
long_run_simspikes_model_...	2021/0...
long_run_pointDRS_Ca05.jpg	2021/0...
long_run_pointDRS_Ca05.fig	2021/0...
long_run_pointF_Ca05.jpg	2021/0...
long_run_pointF_Ca05.fig	2021/0...
long_run_pointE_Ca05.jpg	2021/0...
long_run_pointE_Ca05.fig	2021/0...
long_run_pointD_Ca05.jpg	2021/0...
long_run_pointD_Ca05.fig	2021/0...
long_run_pointC_Ca05.jpg	2021/0...
long_run_pointC_Ca05.fig	2021/0...
long_run_pointB_Ca05.jpg	2021/0...
long_run_pointB_Ca05.fig	2021/0...
long_run_pointA_Ca05.jpg	2021/0...
long_run_pointA_Ca05.fig	2021/0...
long_run_soma_Ca05.jpg	2021/0...
long_run_soma_Ca05.fig	2021/0...
long_run_pointDRS_v05.jpg	2021/0...
long_run_pointDRS_v05.fig	2021/0...
long_run_pointAIS2_v05.jpg	2021/0...
long_run_pointAIS2_v05.fig	2021/0...
long_run_pointAIS_v05.jpg	2021/0...
long_run_pointAIS_v05.fig	2021/0...
long_run_pointF_v05.jpg	2021/0...
long_run_pointF_v05.fig	2021/0...
long_run_pointE_v05.jpg	2021/0...
long_run_pointE_v05.fig	2021/0...
long_run_pointD_v05.jpg	2021/0...
long_run_pointD_v05.fig	2021/0...
long_run_pointC_v05.jpg	2021/0...
long_run_pointC_v05.fig	2021/0...
long_run_simspikes_model_...	2021/0...
long_run_pointB_v05.jpg	2021/0...
long_run_pointB_v05.fig	2021/0...
long_run_simspikes_model_4runs_3...	2021/0...

**Editor - plot\_pf\_pointA\_27052021.m**

Variables - N\_soma\_BK\_260

	1	2	3	4	5	6	7	8	9	10
1	0	1.7156e-...								
2	1.0000e-...	1.7156e-...								
3	0.0020	1.7161e-...								
4	0.0030	1.7166e-...								
5	0.0040	1.7171e-...								
6	0.0050	1.7176e-...								
7	0.0060	1.7181e-...								
8	0.0070	1.7186e-...								
9	0.0080	1.7191e-...								
10	0.0090	1.7196e-...								
11	0.0100	1.7201e-...								
12	0.0110	1.7206e-...								
13	0.0120	1.7211e-...								
14	0.0130	1.7215e-...								

**Command Window**

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_sp
>> 50/1000

ans =

0.0500
```

**Workspace**

Name
lr_sim_spikes3_pointC
lr_sim_spikes3_pointD
lr_sim_spikes3_pointDRS
lr_sim_spikes3_pointE
lr_sim_spikes3_pointF
lr_sim_spikes3_soma
lr_sim_spikes4_pointA
lr_sim_spikes4_pointAIS
lr_sim_spikes4_pointAIS2
lr_sim_spikes4_pointB
lr_sim_spikes4_pointC
lr_sim_spikes4_pointD
lr_sim_spikes4_pointDRS
lr_sim_spikes4_pointE
lr_sim_spikes4_pointF
lr_sim_spikes4_soma
N_pointA_BK_260
N_pointA_SK_260
N_pointB_BK_260
N_pointB_SK_260
N_pointC_BK_260
N_pointC_SK_260
N_pointD_BK_260
N_pointD_SK_260
N_pointE_BK_260
N_pointE_SK_260
N_pointF_BK_260
N_pointF_SK_260
N_soma_260
N_soma_BK_260
N_soma_SK_260
unnamed
unnamedCopy
unnamedCopy7
unnamedConv8





# MATLAB GUI (Graphical User Interface)

Work folder

The image shows the MATLAB R2018b academic use interface. The top menu bar includes Apple logo, MATLAB, Window, and Help. The top toolbar has icons for Home, Plots, Apps, Variable, and View. Below the toolbar is a navigation pane showing the current folder path: / > Users > alexmart > Documents > STEPSmodel1 >. The main area is divided into three panes: Current Folder, Editor, and Workspace.

**Current Folder:** A list of files and folders in the current directory. A red arrow points to the 'long\_run\_simspikes\_model' folder.

**Editor:** Displays the code for 'plot\_pointA\_27052021.m'. The code includes comments and MATLAB commands for plotting and semilogging data. The output of the last command is shown in the Command Window.

**Command Window:** Shows the execution of the MATLAB code. The output is a 14x2 double matrix, displayed as a table.

	1	2	3	4	5	6	7	8	9	10
1	0	1.7156e-...								
2	1.0000e-...	1.7156e-...								
3	0.0020	1.7161e-...								
4	0.0030	1.7166e-...								
5	0.0040	1.7171e-...								
6	0.0050	1.7176e-...								
7	0.0060	1.7181e-...								
8	0.0070	1.7186e-...								
9	0.0080	1.7191e-...								
10	0.0090	1.7196e-...								
11	0.0100	1.7201e-...								
12	0.0110	1.7206e-...								
13	0.0120	1.7211e-...								
14	0.0130	1.7215e-...								

**Workspace:** A list of variables in the workspace. The variable 'N\_pointA\_BK\_260' is highlighted.



# MATLAB GUI (Graphical User Interface)

Work folder

Folder  
content

The image shows the MATLAB R2018b academic use interface. The top menu bar includes 'MATLAB', 'Window', and 'Help'. The top toolbar has tabs for 'HOME', 'PLOTS', 'APPS', 'VARIABLE', and 'VIEW'. Below the toolbar is a 'Current Folder' pane on the left, a 'Variables' pane in the center, and a 'Workspace' pane on the right. The 'Current Folder' pane shows a directory structure with files like 'long\_run\_simspikes\_model', 'long\_run\_simspikes\_m...', 'long\_run\_pointDRS...', 'long\_run\_pointE\_Ca05.fig', 'long\_run\_pointF\_Ca05.fig', 'long\_run\_pointE\_Ca05.jpg', 'long\_run\_pointD\_Ca05.jpg', 'long\_run\_pointC\_Ca05.jpg', 'long\_run\_pointB\_Ca05.jpg', 'long\_run\_pointA\_Ca05.jpg', 'long\_run\_soma\_Ca05.jpg', 'long\_run\_soma\_Ca05.fig', 'long\_run\_pointDRS\_v05.jpg', 'long\_run\_pointAIS2\_v05.jpg', 'long\_run\_pointAIS\_v05.jpg', 'long\_run\_pointAIS\_v05.fig', 'long\_run\_pointF\_v05.jpg', 'long\_run\_pointF\_v05.fig', 'long\_run\_pointE\_v05.jpg', 'long\_run\_pointD\_v05.jpg', 'long\_run\_pointD\_v05.fig', 'long\_run\_pointC\_v05.jpg', 'long\_run\_pointC\_v05.fig', 'long\_run\_simspikes\_model...', 'long\_run\_pointB\_v05.jpg', and 'long\_run\_pointB\_v05.fig'. A red arrow points from the 'Work folder' text to the 'Current Folder' pane. A green box highlights the 'Folder content' text. The 'Variables' pane shows a table with 10 columns and 14 rows of data. The 'Workspace' pane shows a list of variables including 'lr\_sim\_spikes3\_pointC', 'lr\_sim\_spikes3\_pointD', 'lr\_sim\_spikes3\_pointDRS', 'lr\_sim\_spikes3\_pointE', 'lr\_sim\_spikes3\_pointF', 'lr\_sim\_spikes3\_soma', 'lr\_sim\_spikes4\_pointA', 'lr\_sim\_spikes4\_pointAIS', 'lr\_sim\_spikes4\_pointAIS2', 'lr\_sim\_spikes4\_pointB', 'lr\_sim\_spikes4\_pointC', 'lr\_sim\_spikes4\_pointD', 'lr\_sim\_spikes4\_pointDRS', 'lr\_sim\_spikes4\_pointE', 'lr\_sim\_spikes4\_pointF', 'lr\_sim\_spikes4\_soma', 'N\_pointA\_BK\_260', 'N\_pointA\_SK\_260', 'N\_pointB\_BK\_260', 'N\_pointB\_SK\_260', 'N\_pointC\_BK\_260', 'N\_pointC\_SK\_260', 'N\_pointD\_BK\_260', 'N\_pointD\_SK\_260', 'N\_pointE\_BK\_260', 'N\_pointE\_SK\_260', 'N\_pointF\_BK\_260', 'N\_pointF\_SK\_260', 'N\_soma\_260', 'N\_soma\_BK\_260', 'N\_soma\_SK\_260', 'unnamed', 'unnamedCopy', 'unnamedCopy7', and 'unnamedConv8'. The 'Command Window' at the bottom shows the following code:

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_sp
>> 50/1000

ans =

0.0500
```

# MATLAB GUI (Graphical User Interface)

Control  
tabs

Buttons

Work folder

Folder  
content

The image shows the MATLAB R2018b interface. The top bar includes the MATLAB logo, window controls, and the title 'MATLAB R2018b - academic use'. Below this is a navigation bar with tabs: HOME, PLOTS, APPS, VARIABLE, and VIEW. The main workspace is divided into four panes:

- Current Folder:** A file explorer showing a directory structure with files like 'long\_run\_simspikes\_model', 'long\_run\_simspikes\_m...', 'long\_run\_pointDRS...', 'long\_run\_pointE\_Ca05.jpg', 'long\_run\_pointF\_Ca05.jpg', 'long\_run\_pointD\_Ca05.jpg', 'long\_run\_pointC\_Ca05.jpg', 'long\_run\_pointB\_Ca05.jpg', 'long\_run\_pointA\_Ca05.jpg', 'long\_run\_soma\_Ca05.jpg', 'long\_run\_soma\_Ca05.fig', 'long\_run\_pointDRS\_v05.jpg', 'long\_run\_pointAIS2\_v05.jpg', 'long\_run\_pointAIS\_v05.jpg', 'long\_run\_pointAIS\_v05.fig', 'long\_run\_pointF\_v05.jpg', 'long\_run\_pointE\_v05.jpg', 'long\_run\_pointD\_v05.jpg', 'long\_run\_pointC\_v05.jpg', 'long\_run\_pointB\_v05.jpg', 'long\_run\_pointA\_v05.jpg', 'long\_run\_simspikes\_model...', 'long\_run\_pointB\_v05.jpg', and 'long\_run\_simspikes\_model\_4runs\_3...'. A red arrow points from the 'Work folder' label to this pane.
- Editor:** Displays a script file 'plot\_pi\_pointA\_27052021.m'. The script contains the following code:

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_spikes2_pointA(:,7),lr_sim_spikes3_pointA(:,1),lr_sim_spikes3_pointA(:,7),lr_sim_spikes4_pointA(:,1),lr_sim_spikes4_pointA(:,7),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),lr_sim_spikes3_soma(:,1),lr_sim_spikes3_soma(:,7),lr_sim_spikes4_soma(:,1),lr_sim_spikes4_soma(:,7))
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_spikes2_pointA(:,7),lr_sim_spikes3_pointA(:,1),lr_sim_spikes3_pointA(:,7),lr_sim_spikes4_pointA(:,1),lr_sim_spikes4_pointA(:,7),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),lr_sim_spikes3_soma(:,1),lr_sim_spikes3_soma(:,7),lr_sim_spikes4_soma(:,1),lr_sim_spikes4_soma(:,7))
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),lr_sim_spikes3_soma(:,1),lr_sim_spikes3_soma(:,7),lr_sim_spikes4_soma(:,1),lr_sim_spikes4_soma(:,7))
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),lr_sim_spikes3_soma(:,1),lr_sim_spikes3_soma(:,7),lr_sim_spikes4_soma(:,1),lr_sim_spikes4_soma(:,7))
>> 50/1000
```

The output of the script is displayed in the Command Window:

```
ans =
0.0500
```
- Command Window:** Shows the execution of the script and the resulting output 'ans = 0.0500'.
- Workspace:** Lists the variables in the workspace, including 'lr\_sim\_spikes3\_pointC', 'lr\_sim\_spikes3\_pointD', 'lr\_sim\_spikes3\_pointDRS', 'lr\_sim\_spikes3\_pointE', 'lr\_sim\_spikes3\_pointF', 'lr\_sim\_spikes3\_soma', 'lr\_sim\_spikes4\_pointA', 'lr\_sim\_spikes4\_pointAIS', 'lr\_sim\_spikes4\_pointAIS2', 'lr\_sim\_spikes4\_pointB', 'lr\_sim\_spikes4\_pointC', 'lr\_sim\_spikes4\_pointD', 'lr\_sim\_spikes4\_pointDRS', 'lr\_sim\_spikes4\_pointE', 'lr\_sim\_spikes4\_pointF', 'lr\_sim\_spikes4\_soma', 'N\_pointA\_BK\_260', 'N\_pointA\_SK\_260', 'N\_pointB\_BK\_260', 'N\_pointB\_SK\_260', 'N\_pointC\_BK\_260', 'N\_pointC\_SK\_260', 'N\_pointD\_BK\_260', 'N\_pointD\_SK\_260', 'N\_pointE\_BK\_260', 'N\_pointE\_SK\_260', 'N\_pointF\_BK\_260', 'N\_pointF\_SK\_260', 'N\_soma\_260', 'N\_soma\_BK\_260', 'N\_soma\_SK\_260', 'unnamed', 'unnamedCopy', 'unnamedCopy7', and 'unnamedConv8'.



# MATLAB GUI (Graphical User Interface)

The image shows the MATLAB R2018b academic use interface. The top bar includes the MATLAB logo, 'Window', 'Help', and a search bar labeled 'Search Documentation' with a magnifying glass icon. Below the top bar is a toolbar with icons for 'New from Selection', 'Open', 'Print', 'Rows', 'Columns', 'Insert', 'Delete', 'Sort', and 'Transpose'. The main workspace is divided into three panes: 'Current Folder', 'Editor', and 'Workspace'. The 'Current Folder' pane shows a list of files and folders, including 'long\_run\_simspikes\_model', 'long\_run\_simspikes\_m...', 'long\_run\_pointDRS...', 'long\_run\_pointA\_05.fig', 'long\_run\_pointE\_Ca05.jpg', 'long\_run\_pointF\_Ca05.jpg', 'long\_run\_pointD\_Ca05.jpg', 'long\_run\_pointC\_Ca05.jpg', 'long\_run\_pointB\_Ca05.jpg', 'long\_run\_pointA\_Ca05.jpg', 'long\_run\_pointA\_Ca05.fig', 'long\_run\_soma\_Ca05.jpg', 'long\_run\_soma\_Ca05.fig', 'long\_run\_pointDRS\_v05.jpg', 'long\_run\_pointAIS2\_v05.jpg', 'long\_run\_pointAIS2\_v05.fig', 'long\_run\_pointAIS\_v05.jpg', 'long\_run\_pointAIS\_v05.fig', 'long\_run\_pointF\_v05.jpg', 'long\_run\_pointF\_v05.fig', 'long\_run\_pointE\_v05.jpg', 'long\_run\_pointE\_v05.fig', 'long\_run\_pointD\_v05.jpg', 'long\_run\_pointD\_v05.fig', 'long\_run\_pointC\_v05.jpg', 'long\_run\_pointC\_v05.fig', 'long\_run\_simspikes\_model...', 'long\_run\_pointB\_v05.jpg', and 'long\_run\_pointB\_v05.fig'. A red arrow points to the 'Current Folder' pane, labeled 'Work folder'. The 'Editor' pane shows a script file 'plot\_pi\_pointA\_27052021.m' with a table of data. The 'Workspace' pane shows a list of variables, including 'lr\_sim\_spikes3\_pointC', 'lr\_sim\_spikes3\_pointD', 'lr\_sim\_spikes3\_pointDRS', 'lr\_sim\_spikes3\_pointE', 'lr\_sim\_spikes3\_soma', 'lr\_sim\_spikes4\_pointA', 'lr\_sim\_spikes4\_pointAIS', 'lr\_sim\_spikes4\_pointAIS2', 'lr\_sim\_spikes4\_pointB', 'lr\_sim\_spikes4\_pointC', 'lr\_sim\_spikes4\_pointD', 'lr\_sim\_spikes4\_pointDRS', 'lr\_sim\_spikes4\_pointE', 'lr\_sim\_spikes4\_pointF', 'lr\_sim\_spikes4\_soma', 'N\_pointA\_BK\_260', 'N\_pointA\_SK\_260', 'N\_pointB\_BK\_260', 'N\_pointB\_SK\_260', 'N\_pointC\_BK\_260', 'N\_pointC\_SK\_260', 'N\_pointD\_BK\_260', 'N\_pointD\_SK\_260', 'N\_pointE\_BK\_260', 'N\_pointE\_SK\_260', 'N\_pointF\_BK\_260', 'N\_pointF\_SK\_260', 'N\_soma\_260', 'N\_soma\_BK\_260', 'N\_soma\_SK\_260', 'unnamed', 'unnamedCopy', 'unnamedCopy7', and 'unnamedConv8'. A yellow arrow points to the 'Search Documentation' button, labeled 'The most important button'. The 'Command Window' at the bottom shows the following code:

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7),
>> 50/1000
```

The output is:

```
ans =
0.0500
```

Work folder

Folder  
content

The most  
important  
button



The screenshot displays the MATLAB R2018b environment. The top toolbar includes icons for file operations, editing, and viewing. The main window is divided into three panes:

- Editor:** Shows a script named `plot_pf_pointA_27052021.m`. The script contains the following commands:
 

```
semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_spikes2_pointA(:,7))
plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_spikes2_pointA(:,7))
plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7))
semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7))
plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7))
semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes2_soma(:,7))
50/1000
```
- Command Window:** Shows the output of the script, including the value `0.0500` and the prompt `fx >>`.
- Workspace:** Lists variables in the workspace, including `lr_sim_spikes3_pointC`, `lr_sim_spikes3_pointD`, `lr_sim_spikes3_pointE`, `lr_sim_spikes3_pointF`, `lr_sim_spikes3_soma`, `lr_sim_spikes4_pointA`, `lr_sim_spikes4_pointAIS`, `lr_sim_spikes4_pointAIS2`, `lr_sim_spikes4_pointB`, `lr_sim_spikes4_pointC`, `lr_sim_spikes4_pointD`, `lr_sim_spikes4_pointDRS`, `lr_sim_spikes4_pointE`, `lr_sim_spikes4_pointF`, `lr_sim_spikes4_soma`, `N_pointA_BK_260`, `N_pointA_SK_260`, `N_pointB_BK_260`, `N_pointB_SK_260`, `N_pointC_BK_260`, `N_pointC_SK_260`, `N_pointD_BK_260`, `N_pointD_SK_260`, `N_pointE_BK_260`, `N_pointE_SK_260`, `N_pointF_BK_260`, `N_pointF_SK_260`, `N_soma_260`, `N_soma_BK_260`, `N_soma_SK_260`, `unnamed`, `unnamedCopy`, `unnamedCopy7`, and `unnamedConv8`.

☐ N\_sim\_spikes3\_pointA  
☐ lr\_sim\_spikes3\_pointE  
☐ lr\_sim\_spikes3\_pointF  
☐ lr\_sim\_spikes3\_soma  
☐ lr\_sim\_spikes4\_pointA  
☐ lr\_sim\_spikes4\_pointAIS  
☐ lr\_sim\_spikes4\_pointAIS2  
☐ lr\_sim\_spikes4\_pointB  
☐ lr\_sim\_spikes4\_pointC  
☐ lr\_sim\_spikes4\_pointD  
☐ lr\_sim\_spikes4\_pointDRS  
☐ lr\_sim\_spikes4\_pointE  
☐ lr\_sim\_spikes4\_pointF  
☐ lr\_sim\_spikes4\_soma  
☒ N\_pointA\_BK\_260  
☐ N\_pointA\_SK\_260  
☐ N\_pointB\_BK\_260  
☐ N\_pointB\_SK\_260  
☐ N\_pointC\_BK\_260  
☐ N\_pointC\_SK\_260  
☐ N\_pointD\_BK\_260  
☐ N\_pointD\_SK\_260  
☐ N\_pointE\_BK\_260  
☐ N\_pointE\_SK\_260  
☐ N\_pointF\_BK\_260  
☐ N\_pointF\_SK\_260  
☐ N\_soma\_260  
☐ N\_soma\_BK\_260  
☐ N\_soma\_SK\_260  
☐ unnamed  
☐ unnamedCopy  
☐ unnamedCopy7  
☐ unnamedConv8

```

>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_sp
>> 50/1000

ans =

    0.0500

fx >>

```



# MATLAB GUI (Graphical User Interface)

**Data and script content**

The image displays the MATLAB R2018b graphical user interface. The main window is titled 'MATLAB R2018b - academic use'. The top menu bar includes 'HOME', 'PLOTS', 'APPS', 'VARIABLE', and 'VIEW'. Below the menu bar, there are tabs for 'Current Folder', 'Editor - plot\_pf\_pointA\_27052021.m', 'Variables - N\_soma\_BK\_260', and 'Workspace'. The 'Current Folder' tab shows a list of files and folders. The 'Editor' tab shows a script with a table of data. The 'Variables' tab shows a table of data. The 'Workspace' tab shows a list of variables. The 'Command Window' at the bottom shows the execution of a script.

	1	2	3	4	5	6	7	8	9	10
1	0	1.7156e-...								
2	1.0000e-...	1.7156e-...								
3	0.0020	1.7161e-...								
4	0.0030	1.7166e-...								
5	0.0040	1.7171e-...								
6	0.0050	1.7176e-...								
7	0.0060	1.7181e-...								
8	0.0070	1.7186e-...								
9	0.0080	1.7191e-...								
10	0.0090	1.7196e-...								
11	0.0100	1.7201e-...								
12	0.0110	1.7206e-...								
13	0.0120	1.7211e-...								
	0.0130	1.7215e-...								

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_sp
>> 50/1000

ans =

0.0500
```



# MATLAB GUI (Graphical User Interface)

The image displays the MATLAB R2018b academic use interface. The top menu bar includes 'MATLAB', 'Window', and 'Help'. The main toolbar contains tabs for 'HOME', 'PLOTS', 'APPS', 'VARIABLE', and 'VIEW'. Below these are various tool icons for file operations, variable manipulation, and editing.

The 'Current Folder' pane on the left shows a directory structure with files like 'long\_run\_simspikes\_model\_...', 'long\_run\_pointDRS\_Ca05.jpg', 'long\_run\_pointF\_Ca05.jpg', 'long\_run\_pointE\_Ca05.jpg', 'long\_run\_pointD\_Ca05.jpg', 'long\_run\_pointC\_Ca05.jpg', 'long\_run\_pointB\_Ca05.jpg', 'long\_run\_pointA\_Ca05.jpg', 'long\_run\_soma\_Ca05.jpg', 'long\_run\_pointDRS\_v05.jpg', 'long\_run\_pointAIS2\_v05.jpg', 'long\_run\_pointAIS\_v05.jpg', 'long\_run\_pointF\_v05.jpg', 'long\_run\_pointE\_v05.jpg', 'long\_run\_pointD\_v05.jpg', 'long\_run\_pointC\_v05.jpg', 'long\_run\_pointB\_v05.jpg', and 'long\_run\_simspikes\_model\_4runs\_3...'. The 'Editor' pane in the center shows a script named 'plot\_pf\_pointA\_27052021.m' with a table of data for 'N\_soma\_BK\_260'.

	1	2	3	4	5	6	7	8	9	10
1	0	1.7156e-...								
2	1.0000e-...	1.7156e-...								
3	0.0020	1.7161e-...								
4	0.0030	1.7166e-...								
5	0.0040	1.7171e-...								
6	0.0050	1.7176e-...								
7	0.0060	1.7181e-...								
8	0.0070	1.7186e-...								
9	0.0080	1.7191e-...								
10	0.0090	1.7196e-...								
11	0.0100	1.7201e-...								
12	0.0110	1.7206e-...								
13	0.0120	1.7211e-...								
14	0.0130	1.7215e-...								

The 'Command Window' at the bottom shows the following commands and output:

```
>> semilogy(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_
>> plot(lr_sim_spikes1_pointA(:,1),lr_sim_spikes1_pointA(:,7),lr_sim_spikes2_pointA(:,1),lr_sim_
>> plot(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),
>> semilogy(N_soma_BK_260(:,1),N_soma_BK_260(:,2),lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,
>> plot(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_spikes
>> semilogy(lr_sim_spikes1_soma(:,1),lr_sim_spikes1_soma(:,7),lr_sim_spikes2_soma(:,1),lr_sim_sp
>> 50/1000

ans =

0.0500
```

The 'Workspace' pane on the right lists variables: 'lr\_sim\_spikes3\_pointC', 'lr\_sim\_spikes3\_pointD', 'lr\_sim\_spikes3\_pointDRS', 'lr\_sim\_spikes3\_pointE', 'lr\_sim\_spikes3\_pointF', 'lr\_sim\_spikes3\_soma', 'lr\_sim\_spikes4\_pointA', 'lr\_sim\_spikes4\_pointAIS', 'lr\_sim\_spikes4\_pointAIS2', 'lr\_sim\_spikes4\_pointB', 'lr\_sim\_spikes4\_pointC', 'lr\_sim\_spikes4\_pointD', 'lr\_sim\_spikes4\_pointDRS', 'lr\_sim\_spikes4\_pointE', 'lr\_sim\_spikes4\_pointF', 'lr\_sim\_spikes4\_soma', 'N\_pointA\_BK\_260', 'N\_pointA\_SK\_260', 'N\_pointB\_BK\_260', 'N\_pointB\_SK\_260', 'N\_pointC\_BK\_260', 'N\_pointC\_SK\_260', 'N\_pointD\_BK\_260', 'N\_pointD\_SK\_260', 'N\_pointE\_BK\_260', 'N\_pointE\_SK\_260', 'N\_pointF\_BK\_260', 'N\_pointF\_SK\_260', 'N\_soma\_260', 'N\_soma\_BK\_260', 'N\_soma\_SK\_260', 'unnamed', 'unnamedCopy', 'unnamedCopy7', and 'unnamedConv8'.



# The content of the class

---

- MATLAB GUI (Graphical User Interface)
- **Basics of programming in MATLAB**
- Matrix computations
- Upload data to your computer
- Fitting of the model parameters to the data
- Plotting the results

## **ADVANTAGES (list of some, not all):**

- GUI: easy access to data, plot editor, and other functions
- Toolboxes (see APPS tab in your MATLAB): more than 20 are available for various scientific, engineering and data analysis needs (that requires you to do less coding)
- Easy debugging tools

## **DISADVANTAGES (list of some, not all):**

- Organization of user-defined functions
- Some argue that it is slow, maybe...maybe...

## BASICS OF CREATING YOUR MATLAB CODE:

- First, you need to create (or open an existing) script (function) file
- Data structures (VARIABLES) keep necessary information in your code such as single variables, arrays, strings.
- After you assigned some value to a variable(s), you provided an input to your code.
- If you wish the values of the variables been changed, as a result of some computations, you should state this in your code.



# Basics of programming in MATLAB

## DEFINING VARIABLES:

Not required, but may be necessary for some datatypes, like arrays, cells and other complex data structures

## NAMES OF VARIABLES:

Can't begin with a number or other special characters (!@#\$%^&\*()\_+=)

EXAMPLES (you may also type in the Command window):

```
>> n_v = 123; %integer variable
```

```
>> Duration_spikes = 0.785; %double variable
```

```
>> t_spikes = []; %empty array
```

VARIABLES CAN BE CONVERTED FROM ONE TYPE TO ANOTHER (with some conditions)



# Basics of programming in MATLAB

If you want to print out a variable value, the simplest way is to type the name in you SCRIPT or in the COMMAND WINDOW:

For example:

```
>> n_v = 123
```

```
>> 123
```

or

```
>> n_v
```

```
>> 123
```



# Basics of programming in MATLAB

For example:

```
>> s_v = 'SOME TEXT' %this is a string variable, and the  
text after % is just a comment stat will not be interpreted by  
MATLAB
```

```
>> 'SOME TEXT'
```

# Basics of programming in MATLAB

If you want to display a text message to the command window, use this:

```
>> disp('SOME TEXT');
```

or if you want to display the content of a variable:

```
>> disp(s_v);
```

Using disp for displaying a variable means it must be some text, if a variable value is a number, you should convert it to string first:

```
>> n_v_text = num2str(n_v); disp(n_v_text);
```

# Basics of programming in MATLAB

More scientific ways of displaying a variables involve using:

```
>> fprintf('n_v is equal to %i.\n', n_v);
```

Or more complex:

```
>> float_point_variable = 123456789.123456789
```

```
>> 1.2346e+08
```

```
>> fprintf('ls equal to %6.2f.\n', float_point_variable);
```

```
>> 123456789.12
```



# Basics of programming in MATLAB

Briefly about arrays:

```
array1 = [1, 3, 4, 5, 7]; %is a row
```

```
array2 = [1; 2; 7; 8; 9]; %is a column
```

```
array3 = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]; %is a 3x4 matrix
```

Access to data in arrays:

IMPOTANT: indexes start from 1 ~~not from 0~~.

```
>> array1(1)
```

```
>> 1
```



# Basics of programming in MATLAB

Briefly about arrays:

array1 = [1, 3, 4, 5, 7]; %is a row

array2 = [1; 2; 7; 8; 9]; %is a column

array3= [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]; %is a 3x4 matrix

NOTE: using of **commas** (,) for inputting column elements in a row is NOT necessary, but **semicolon** is required to separate rows

array1 = [1 3 4 5 7]; %is a row

array2 = [1; 2; 7; 8; 9]; %is a column

array3= [1 2 3 4; 5 6 7 8; 9 10 11 12]; %is a 3x4 matrix



# Basics of programming in MATLAB

```
array1 = [1, 3, 4, 5, 7]; %is a row
```

```
array2 = [1; 2; 7; 8; 9]; %is a column
```

```
array3 = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]; %is a 3x4 matrix
```

```
>> array1(2)
```

```
>> 3
```

```
>> array2(3)
```

```
>> 7
```

array3(row number, column number) to access an element in array

```
>> array3(3,2)
```

```
>> 10
```



# Basics of programming in MATLAB

```
array1 = [1, 3, 4, 5, 7]; %is a row
```

```
array2 = [1; 2; 7; 8; 9]; %is a column
```

```
array3 = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]; %is a 3x4 matrix
```

If you want to change an element in the arrays:

```
>> array1(2) = 123;
```

```
>> array1(2)
```

```
>> 123
```

```
>> array3(3,2) = 111;
```

```
>> array3(3,2)
```

```
>> 111
```



## MATRIX OR VECTOR TRANSPOSITION:

use “ ‘ ” apostrophe at the end of an array name

```
array1 = [1, 3, 4, 5, 7]; %is a row
```

```
array2 = [1; 2; 7; 8; 9]; %is a column
```

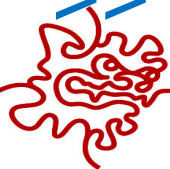
```
array3= [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]; %is a 3x4 matrix
```

For example, try to apply transposition to the above arrays:

```
>> array1'
```

```
>> array2'
```

```
>> array3'
```





# Basics of programming in MATLAB

If you need to test some condition in your code, for example,  $a > b$ ,  $a < b$ , and  $a == b$ , then use the if-elseif-else statements.

## IF, ELSEIF, ELSE STATEMENTS:

```
if (expression)
    statements;
elseif (expression)
    statements;
else
    statements;
end
```

### %AN EXAMPLE:

```
a = 7;
b = 1;
if (a > b)
    disp('a is greater than b');
elseif (a == b)
    disp('a is equal to b');
else
    disp('a is less than b');
end
```



# Basics of programming in MATLAB

If you need to test some condition in your code, for example,  $a > b$ ,  $a < b$ , and  $a == b$ , then use the if-elseif-else statements.

## IF, ELSEIF, ELSE STATEMENTS:

```
if (expression)
    statements;
elseif (expression)
    statements;
else
    statements;
end
```

**IMPORTANT:** the block must start with `if` and end up with `end`.

The key commands, as `if`, `else`, `end` are case-sensitive, don't use CAPITAL letter in them.

A statement should end up with semicolon “;” if you don't want the result been printed to the screen



# Basics of programming in MATLAB

If you need to repeat, iterate or repeat doing something until a condition is true, then use loop statements.

## LOOPS:

For, while, repeat until

## AN EXAMPLE:

**for** *index = values*  
*statements;*

**end**

### %AN EXAMPLE:

```
a = 7;
```

```
n = 10;
```

```
for i = 1:10
```

```
    if (i > a)
```

```
        fprintf('i = %i is greater than a = %i \n', i, a);
```

```
    elseif (i == a)
```

```
        fprintf('i = %i is equal to a = %i \n', i, a);
```

```
    else
```

```
        fprintf('i = %i is less than a = %i \n', i, a);
```

```
    end
```

```
end
```



# Basics of programming in MATLAB

## EXERCISE #1:

Let's imagine a situation you need to count the number of users in the age group between 18 and 24 (including person that are 18 and 24), and printout to the screen: 'The number of users of age between 18 and 24 is = the number'.

The input data comes as an array:

```
users_age = [18, 65, 34, 40, 25, 24, 45, 21, 34, 27, 22, 16, 21];
```

You can count directly for a small dataset, but if you have thousands of entries that is not easy.

## EXERCISE #1

You have 10 minutes to do this task.

### HINT:

Try to access the array elements in a “for” loop, use if-elseif-else to filter them, and “disp” or “fprintf” function to display the result.

The number of elements in `users_age` can be provided with “length”, “size” functions, see manual

## EXERCISE #2:

(For students who quickly completed Exercise #1, or for homework)

Try to pick first three elements from `users_age` that satisfy the age criteria from EXERCISE #1. Then, save to another array information about position of the picked elements in `users_age`. At the end, print to screen, the picked elements and their positions.



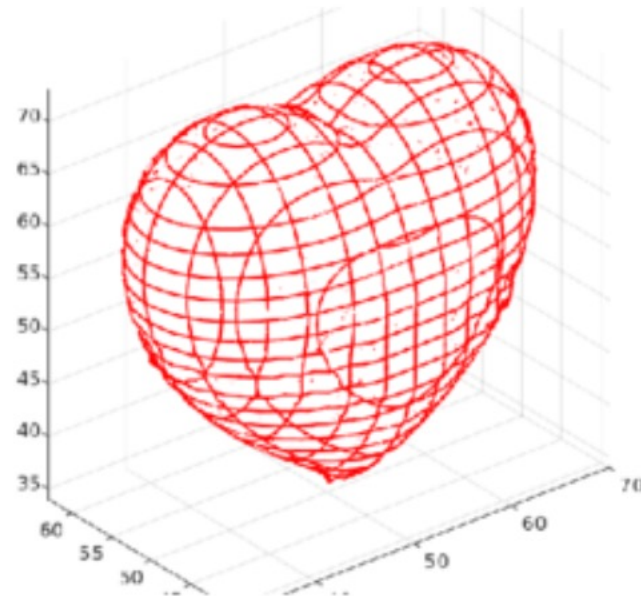
## EXERCISE #2

You have 5 minutes to do this task.

### HINT:

Try to use “while” or “repeat until” loops, and index variables to determine the position of picked elements.

# Basics of programming in MATLAB



# Basics of programming in MATLAB

MY ANSWER (Exercise #1):

```
users_age = [18, 65, 34, 40, 25, 24, 45, 21, 34, 27, 22, 16, 21];
```

```
n = size(users_age,2); %or length(users_age)
```

```
count = 0;
```

```
for i=1:n
```

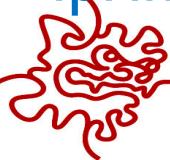
```
    if((users_age(i) >= 18) && (users_age(i) <= 24))
```

```
        count = count + 1;
```

```
    end
```

```
end
```

```
fprintf('The number of users between 18 and 24 years old is = %i \n', count);
```



# Basics of programming in MATLAB

MY ANSWER (Exercise #2):

```
users_age = [18, 65, 34, 40, 25, 24, 45, 21, 34, 27, 22, 16, 21];  
n = size(users_age,2); %or length(users_age)  
i = 0;  
count = 0;  
age_array = zeros(1,3);  
pos_array = zeros(1,3);  
while (count<3)  
    i = i + 1;  
    if((users_age(i) >= 18) && (users_age(i) <= 24))  
        count = count + 1;  
        age_array(i) = users_age(i);  
        pos_array(i) = i;  
        fprintf('User of age %i in position of %i of the array is found!\n', age_array(i), pos_array(i));  
    end  
end
```



# The content of the class



# The content of the class

---

- MATLAB GUI (Graphical User Interface)
- Basics of programming in MATLAB
- **Matrix computations**
- Upload data to your computer
- Fitting of the model parameters to the data
- Plotting the results



# Matrix computations in MATLAB

## SIMPLEST OPERATORS:

### EXAMPLE:

$A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12];$

$b = [1; 2; 3; 4];$

$\gg A*b$

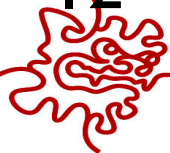
$\gg ans =$

30

70

110

$A*b = [1*1 + 2*2 + 3*3 + 4*4; 5*1 + 6*2 + 7*3 + 8*4; 9*1 + 10*2 + 11*3 + 12*4];$



## SIMPLEST OPERATORS:

### EXMAPLE:

```
A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12];
```

```
b = [1; 2; 3; 4];
```

```
>> A + b
```

```
>> Matrix dimensions must agree.
```

## SIMPLEST OPERATORS:

### EXAMPLE:

```
A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12];
```

```
b = [1; 2; 3; 4];
```

```
>> A + b
```

```
>> Matrix dimensions must agree. Why?
```



## SIMPLEST OPERATORS:

### EXMAPLE:

```
A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12];
```

```
b = [1; 2; 3; 4];
```

```
>> A + b
```

```
>> Matrix dimensions must agree. Why?
```

Because “+” operator sums single elements in A and b. If the number of single elements is different between A and b, the error message pops up



## SIMPLEST OPERATORS:

A correct method of using “+” operator will be for:

```
c = [5, 6, 7, 8];
```

```
d = [9, 10, 11, 12];
```

```
>> c + d
```

```
>> ans =
```

```
14    16    18    20
```



## SIMPLEST OPERATORS:

Or for:

```
A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12];
```

```
E = [10, 11, 16, 12; 3, 8, 6, 9; 2, 1, 3, 0];
```

```
>> A + E
```



## SIMPLEST OPERATORS:

```
a = [1:5]; %[1, 2, 3, 4, 5]
```

```
a>3; %[0, 0, 0, 1, 1] %could also be <, >=, <=, ==
```

```
a(a>3); %[4, 5]
```


```
a.*a; %[1, 4, 9, 16, 25]
```

## Excercise #3:

You buy some quantity of apples, bananas, mango, and pineapples three months in a row. The price of these fruits was constant during these three months. Could we calculate the amount of money you spent on fruits during that period?

5 minutes to do

## HINT:

Use matrix computations and “sum” function (check in the manual  )



## Excercise #4 (optional):

Using matrix (or) vector operators, perform the task of Exercise #1 and #2 (optional).

10 minutes to do

HINT: no hints to experts!

# Basics of programming in MATLAB

MY ANSWER (Exercise #3):

```
cost = [3.5 1.5 5 4.5];
```

```
quacity_mon1 = [10 12 5 2];
```

```
quacity_mon2 = [12 8 4 1];
```

```
quacity_mon3 = [11 7 3 5];
```

```
Q_array = [quacity_mon1; quacity_mon2; quacity_mon3];
```

```
Spent_money = sum(Q_array*(cost'));
```

```
fprintf('The amount of money spent on fruits for the last three months  
is %.2f\n', Spent_money);
```



# Basics of programming in MATLAB

## MY ANSWER (Exercise #4a):

In principle, with using MATLAB in a better way the whole code for the task could be looking like this:

```
users_age = [18, 65, 34, 40, 25, 24, 45, 21, 34, 27, 22, 16, 21];  
count = sum((users_age >= 18).*(users_age <= 24));  
fprintf('The number of users between 18 and 24 years old is = %i  
\n', count);
```



# Basics of programming in MATLAB

MY ANSWER (Exercise #4b):

```
users_age = [18, 65, 34, 40, 25, 24, 45, 21, 34, 27, 22, 16, 21];
```

```
n = length(users_age);
```

```
pos = [1:n];
```

```
indexes = ((users_age >= 18).*(users_age <= 24));
```

```
picked_users = users_age(indexes>0);
```

```
postions = pos(indexes>0);
```

```
picked_users = picked_users([1:3]);
```

```
postions = postions([1:3]);
```

```
disp('Picked users:');
```

```
disp(picked_users);
```

```
disp('Their indexes:');
```

```
disp(postions);
```





# The content of the class

---

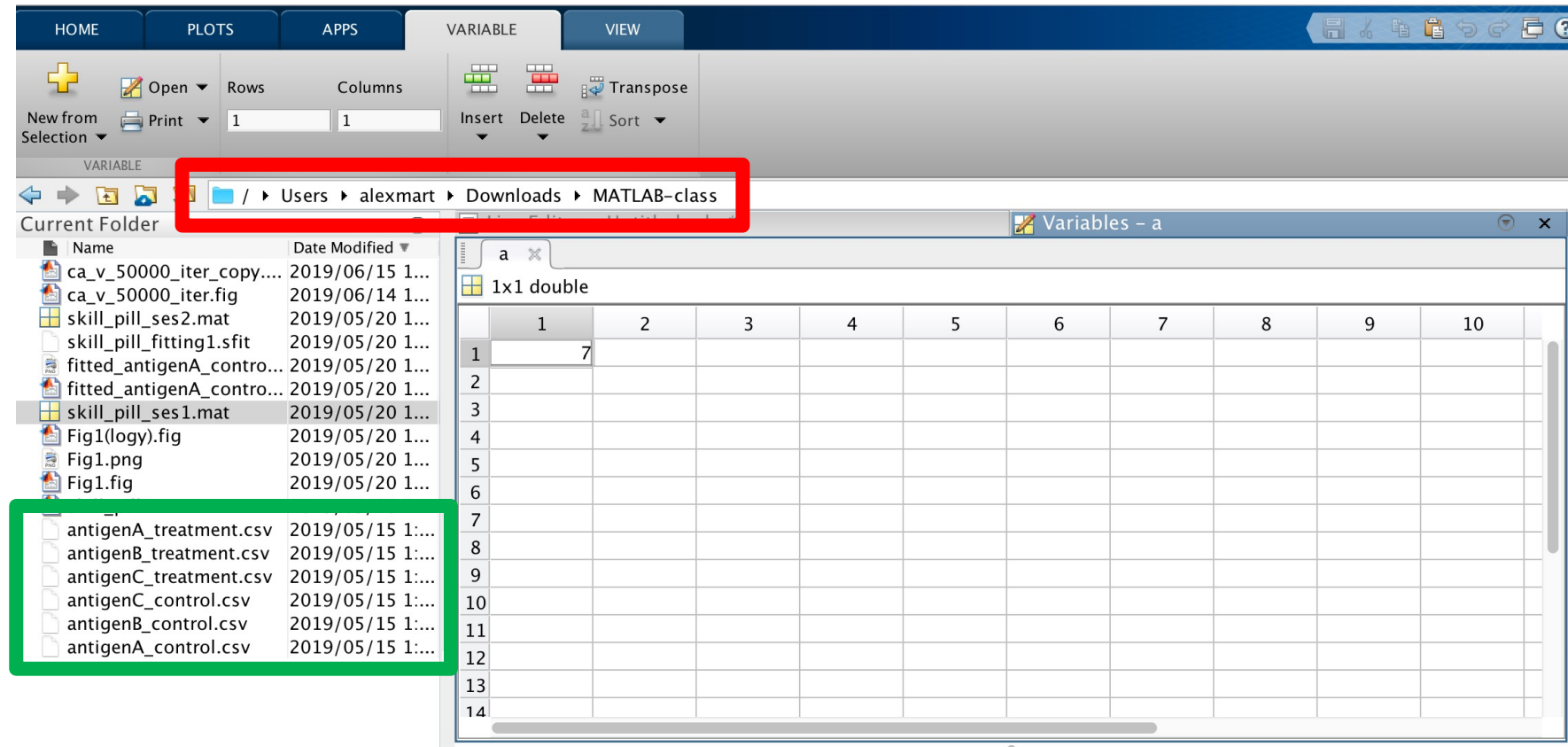
- MATLAB GUI (Graphical User Interface)
- Basics of programming in MATLAB
- Matrix computations
- **Data analysis: upload data to your computer**
- Fitting of the model parameters to the data
- Plotting the results

# Upload data to your computer

- Please access .csv data files through the DropBox
- After download, you should have six files on your computer in a folder:
  - 1) antigenA\_control.csv
  - 2) antigenB\_control.csv
  - 3) antigenC\_control.csv
  - 4) antigenA\_treatment.csv
  - 5) antigenB\_treatment.csv
  - 6) antigenC\_treatment.csv

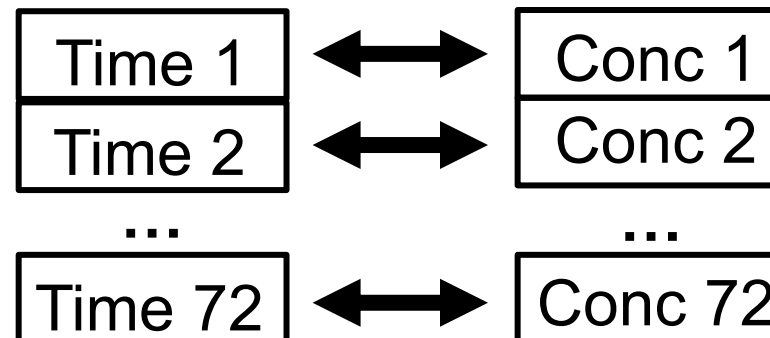
# Upload data to your computer

- Go to **the folder with the downloaded files** in the MATLAB interface, until you get **the files listed in the left panel**:



# Data interpretation

- Each of the six files is a time series of a specific antigen (A, B or C) measured in hosts of a group (control or treatment)
- In other words, each csv file can be interpreted as a table of two columns, the first column is the time when an antigen was measured and the second column is the concentration of the antigen)



*We have 72 elements  
in each row of the data*



# Uploading the data to the workspace

- Upload data to MATLAB (type or copy-paste to the Command Window):

```
antigenA_control = csvread('antigenA_control.csv');
```

```
antigenB_control = csvread('antigenB_control.csv');
```

```
antigenC_control = csvread('antigenC_control.csv');
```

```
antigenA_treatment = csvread('antigenA_treatment.csv');
```

```
antigenB_treatment = csvread('antigenB_treatment.csv');
```

```
antigenC_treatment = csvread('antigenC_treatment.csv');
```

# Organizing the data

- If you want to merge all data to one big table (use console):

```
resTable = [antigenA_control(:,1), antigenA_control(:,2), Time column,  
antigenB_control(:,2), antigenC_control(:,2), antigenA_treatment(:,2),  
antigenB_treatment(:,2), antigenC_treatment(:,2)];
```

*\*IMPORTANT: because column 2 in our csv data corresponds to the time column  
column 1, therefore, the merging as shown above is correct only when the first  
column is the same across all tables to be merged (fortunately, this is our  
case)*

- Use “,” for merging columns or “ ”
- Use “;” for merging rows into one row

# Organizing data in MATLAB

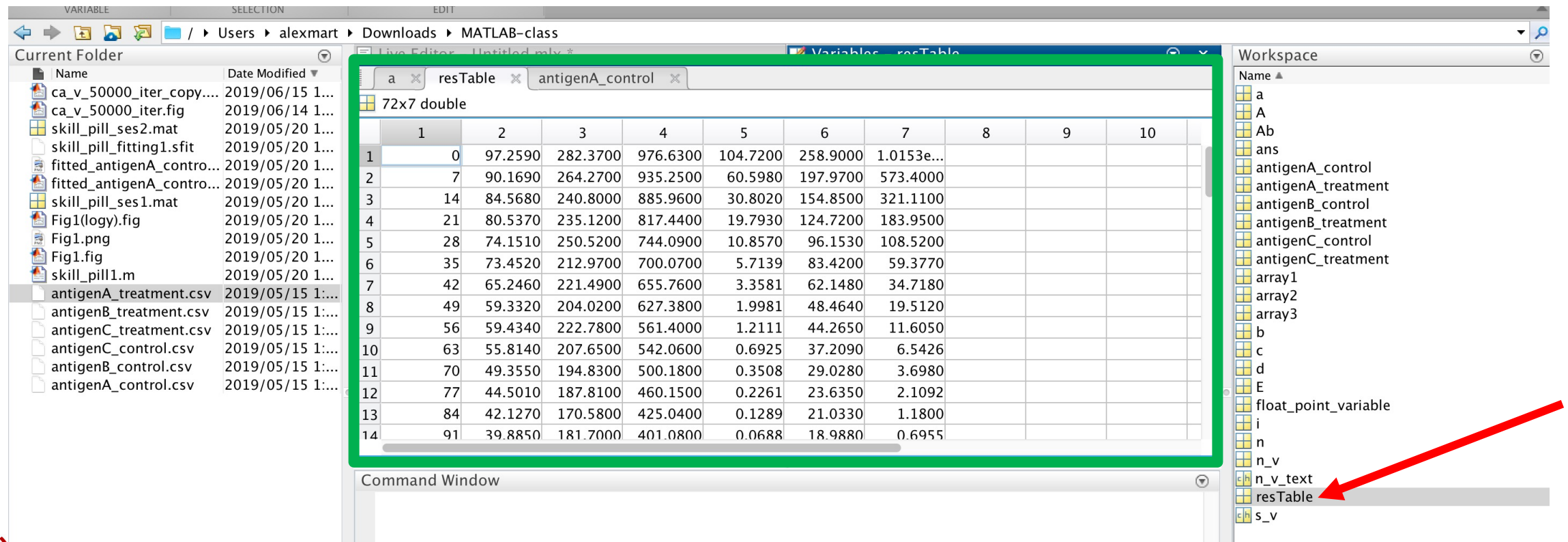
In the resulting table we will have the first column for the measurement times, and other six columns for the antigen concentrations:

Time	A_ctrl	B_ctrl	C_ctrl	A_treat	B_treat	C_treat
Day t1	#	#	#	#	#	#
Day t2	#	#	#	#	#	#
...						

Double click on the **resTable** in the **Workspace** to open ~~(and edit)~~ the data.

# Organizing data in MATLAB

**Double click** on the **resTable** in the **Workspace** to open (and edit) **the data**.



The image shows the MATLAB interface with the following components:

- Current Folder:** A list of files including `antigenA_treatment.csv`, `antigenB_treatment.csv`, `antigenC_treatment.csv`, `antigenB_control.csv`, and `antigenA_control.csv`.
- Variable Viewer:** A window titled `resTable` showing a `72x7 double` matrix. The matrix has columns numbered 1 to 10. The data is as follows:

	1	2	3	4	5	6	7	8	9	10
1	0	97.2590	282.3700	976.6300	104.7200	258.9000	1.0153e...			
2	7	90.1690	264.2700	935.2500	60.5980	197.9700	573.4000			
3	14	84.5680	240.8000	885.9600	30.8020	154.8500	321.1100			
4	21	80.5370	235.1200	817.4400	19.7930	124.7200	183.9500			
5	28	74.1510	250.5200	744.0900	10.8570	96.1530	108.5200			
6	35	73.4520	212.9700	700.0700	5.7139	83.4200	59.3770			
7	42	65.2460	221.4900	655.7600	3.3581	62.1480	34.7180			
8	49	59.3320	204.0200	627.3800	1.9981	48.4640	19.5120			
9	56	59.4340	222.7800	561.4000	1.2111	44.2650	11.6050			
10	63	55.8140	207.6500	542.0600	0.6925	37.2090	6.5426			
11	70	49.3550	194.8300	500.1800	0.3508	29.0280	3.6980			
12	77	44.5010	187.8100	460.1500	0.2261	23.6350	2.1092			
13	84	42.1270	170.5800	425.0400	0.1289	21.0330	1.1800			
14	91	39.8850	181.7000	401.0800	0.0688	18.9880	0.6955			

- Workspace:** A list of variables including `resTable`, which is highlighted with a red arrow.





# A correct mathematical model of the data

- We can plot our data...
- In the Command Window, type this:

```
plot(resTable(:,1), resTable(:,2)); %plot the data
```

Or:

```
plot(resTable(:,1), resTable(:,2), resTable(:,1), resTable(:,3));
```

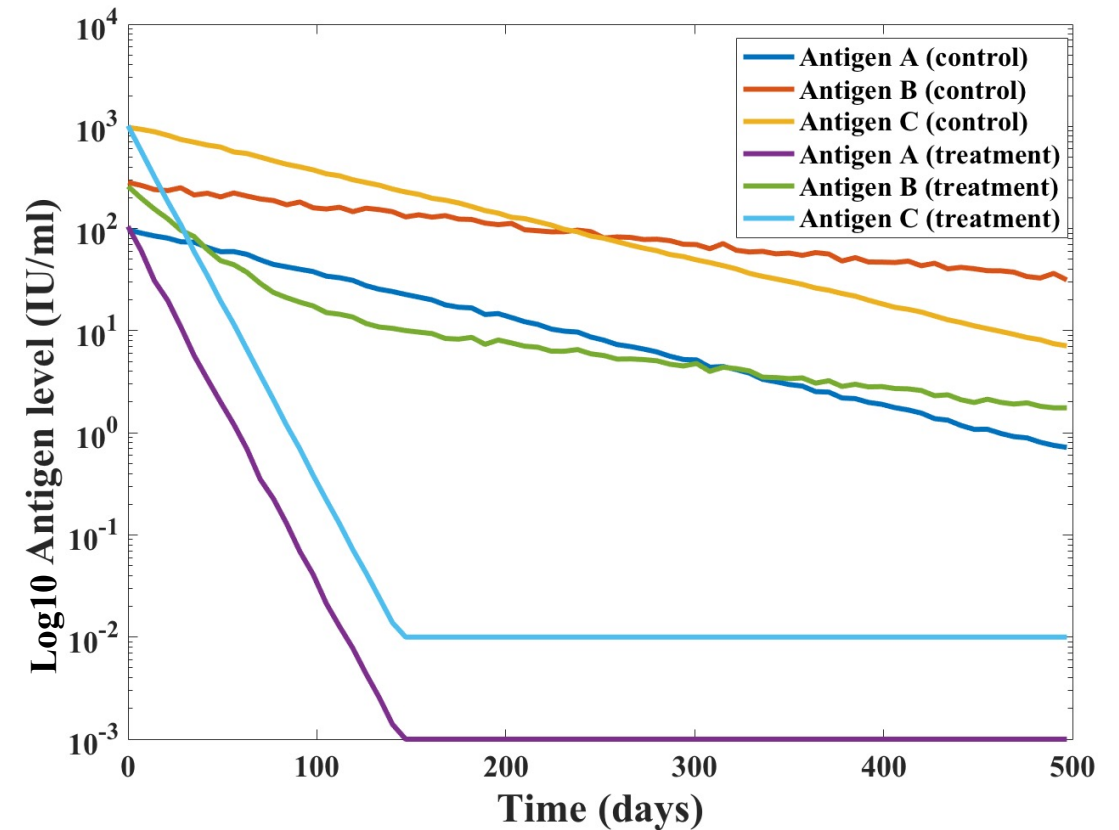
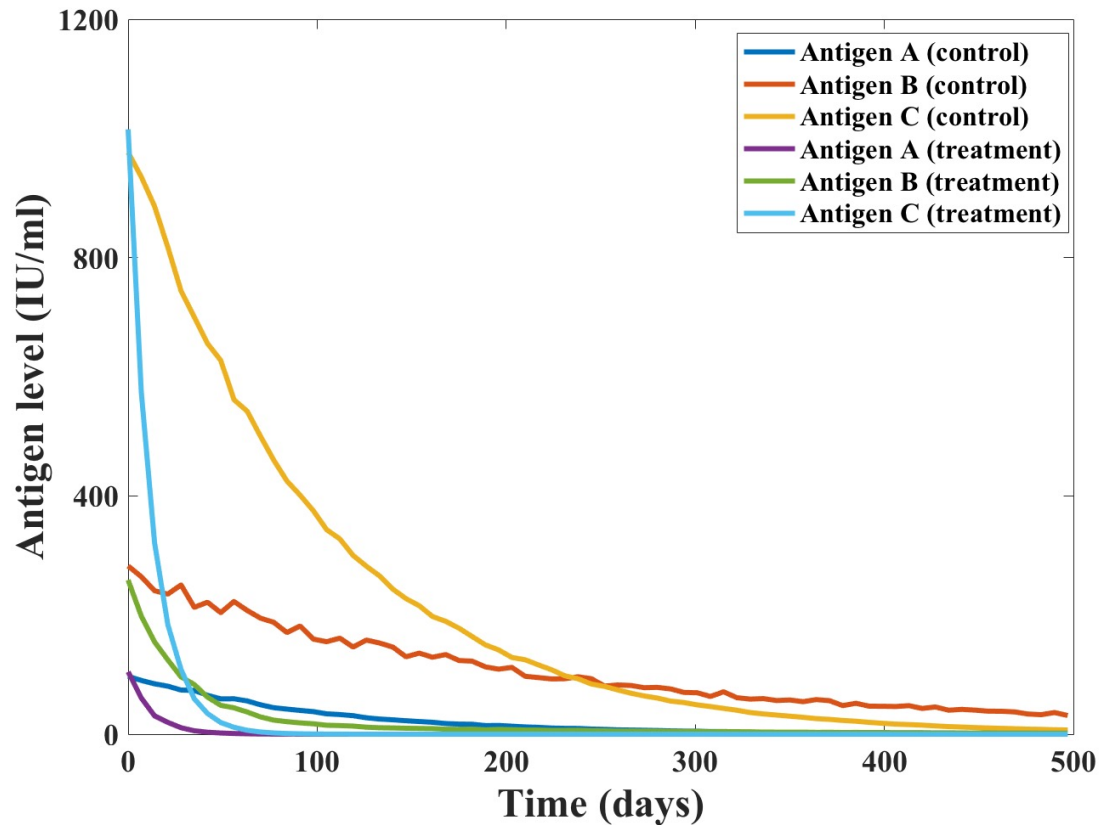
- Also, you can use different plotting axis scales instead of linear  
“plot” you can use “semilogy” (linear x-axis and logarithmic y-axis),  
“semilogx” (logarithmic x-axis and linear y-axis),  
loglog(both axis are logarithmic)

# A correct mathematical model of the data

- Plot all together by using `plot(t, y1, t, y2, t, y3, t, y4, t, y5, t, y6);`
- Note: you must use a time column (t) every time before the data column in the input arguments of the plot functions
- Or, `plot(resTable(:,1), resTable(:,[2:7]));`

# A correct mathematical model of the data

- `plot(resTable(:,1), resTable(:,[2:7]))`



# Exponential decay models

- Try to search in “Exponential models” in MATLAB help
- Let's consider two types of the model, single-phase:

$$\text{Antigen concentration } (t) = A * e^{k1 * t} + \text{plateau1}$$

- and double-phase decay:

$$\text{Antigen concentration } (t) = B * e^{k2 * t} + C * e^{k3 * t} + \text{plateau2}$$

*Note: in the models  $A > 0$ ,  $B > 0$ ,  $C > 0$  and  $k1 < 0$ ,  $k2 < 0$ ,  $k3 < 0$  (because the antigen concentration decays, for a growth  $k$  should be positive). Also,  $\text{plateau1} > 0$  and  $\text{plateau2} > 0$ . The plateaus represent detection limits of the assays.*

# The content of the class

---

- MATLAB GUI (Graphical User Interface)
- Basics of programming in MATLAB
- Matrix computations
- Upload data to your computer
- **Fitting of the model parameters to the data**
- Plotting the results

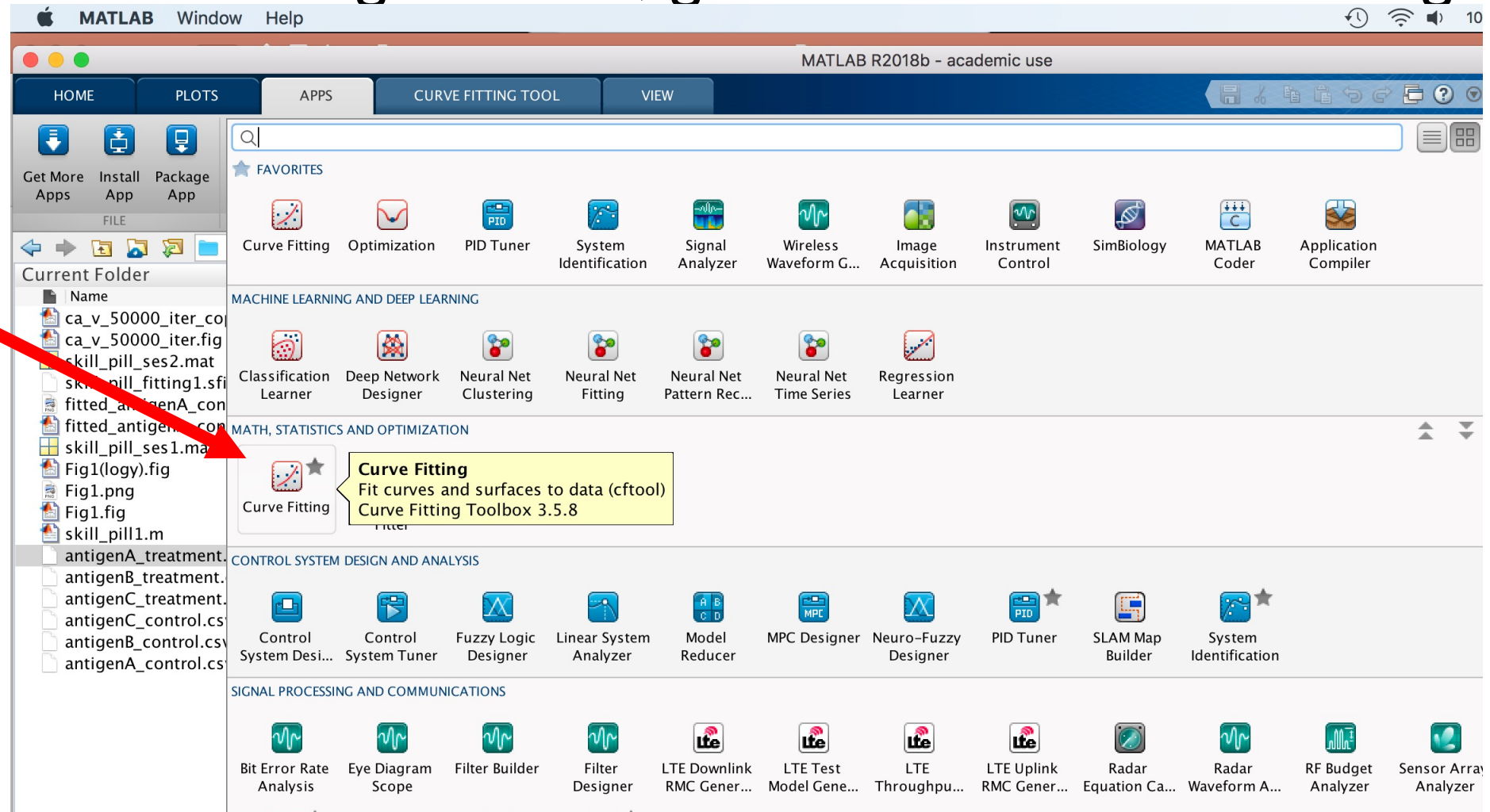
# Fitting of the model parameters

---

- Use Curve Fitting Toolbox, go to APPS - > Curve Fitting
- Setting up the toolbox
- Results

# Fitting of the model parameters

- Use Curve Fitting Toolbox, go to APPS - > Curve Fitting

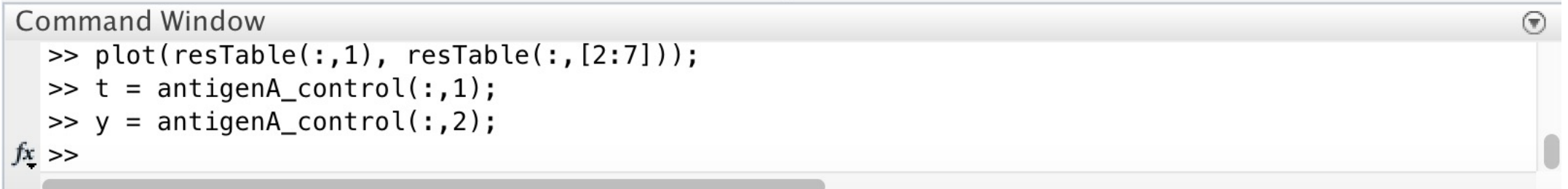


# Fitting of the model parameters

- Use Curve Fitting Toolbox, go to APPS - > Curve Fitting
- Setting up the toolbox

```
>> t = antigenA_control(:,1);
```

```
>> y = antigenA_control(:,2);
```

A screenshot of the MATLAB Command Window. The title bar reads "Command Window". The window contains the following MATLAB code:

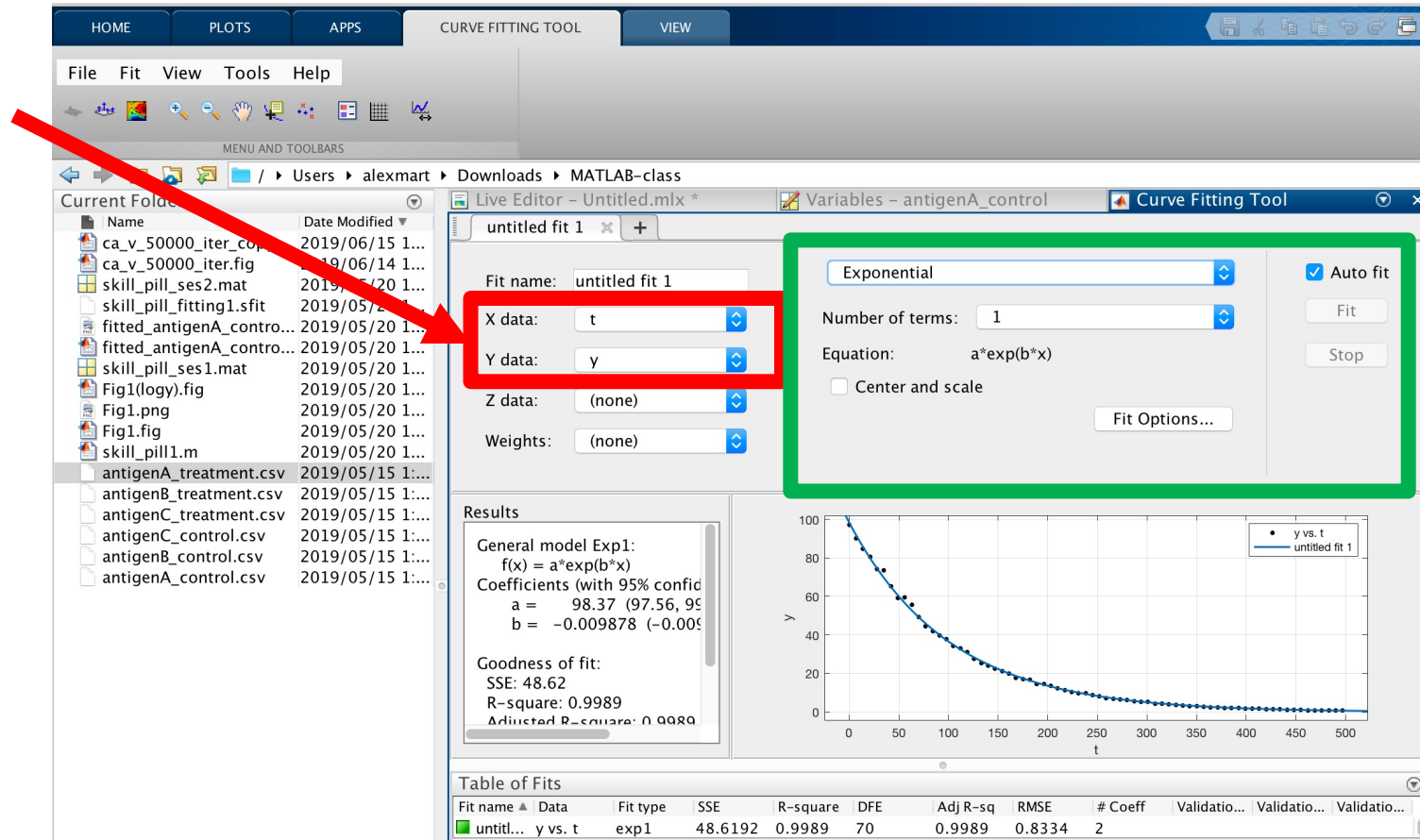
```
>> plot(resTable(:,1), resTable(:,[2:7]));  
>> t = antigenA_control(:,1);  
>> y = antigenA_control(:,2);  
fx >>
```

The cursor is at the end of the last line. The window has a standard macOS-style title bar with a close button on the right.



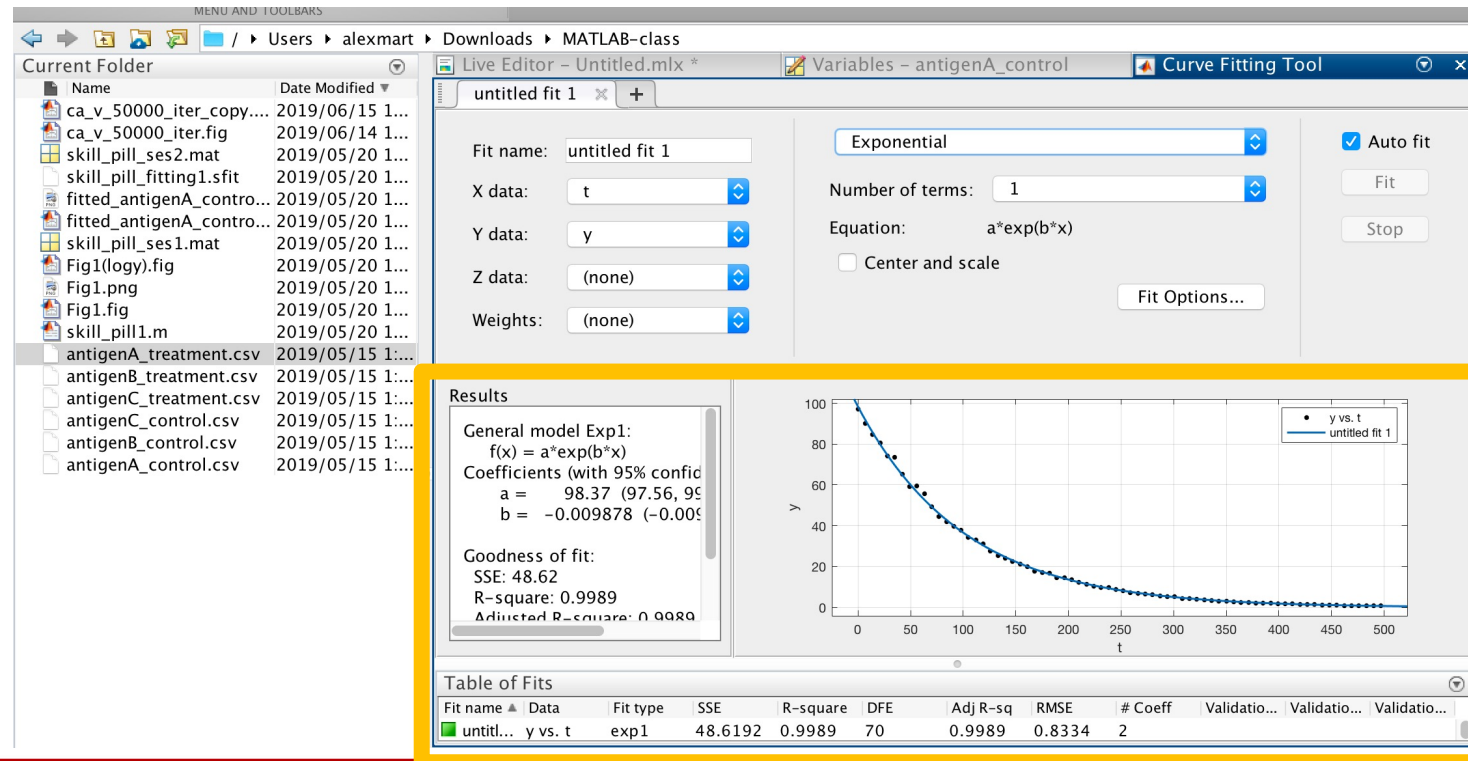
# Fitting of the model parameters

- Use Curve Fitting Toolbox, go to APPS - > Curve Fitting
- Setting up the toolbox



# Fitting of the model parameters

- Use Curve Fitting Toolbox, go to APPS - > Curve Fitting
- Setting up the toolbox
- Results



## Excercise #5:

Do the same as for antigenA\_control, but for:  
antigenB\_control, antigenC\_control, antigenA\_treatment,  
antigenB\_treatment, antigenC\_treatment

5 minutes to do

# Fitting of the model parameters

---

- Group comparison tests to compare decay rates of antigens between the groups (control and treatment)
- Correlation tests (Spearman, Pearson)
- All of them are available as functions in MATLAB, check in the manual

# Thanks

---

**Thank you all for visiting our class today!**