MATLAB
-
Your First Project

Table of Contents

1. The only three rules

1.1 Rule 01

Any questions ?

Ask your neighbour on the right ...

Doesn't he have the answer ?

Try your neighbour on the left ...

Oh no, he doesn't have the answer either?

Don't worry, google has your answer.

1.2 Rule 02

A good programmer is a programmer who masters copy and paste. So don't hesitate to copy and paste code you find on the internet. Try to understand them and improve them to match your problem.

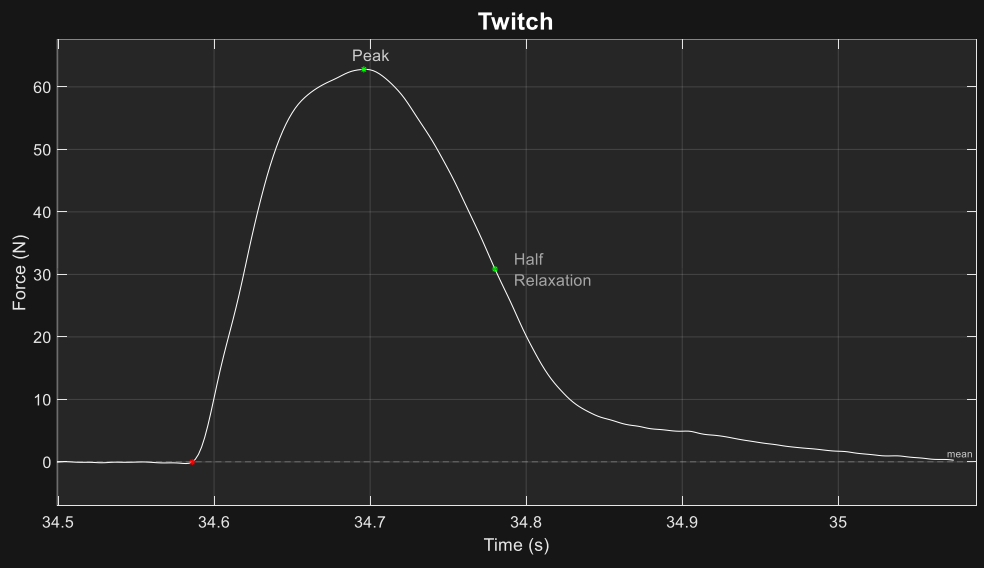
1.3 Rule 03

If google does not have your answer, refer to rule 01.

2. Introduction

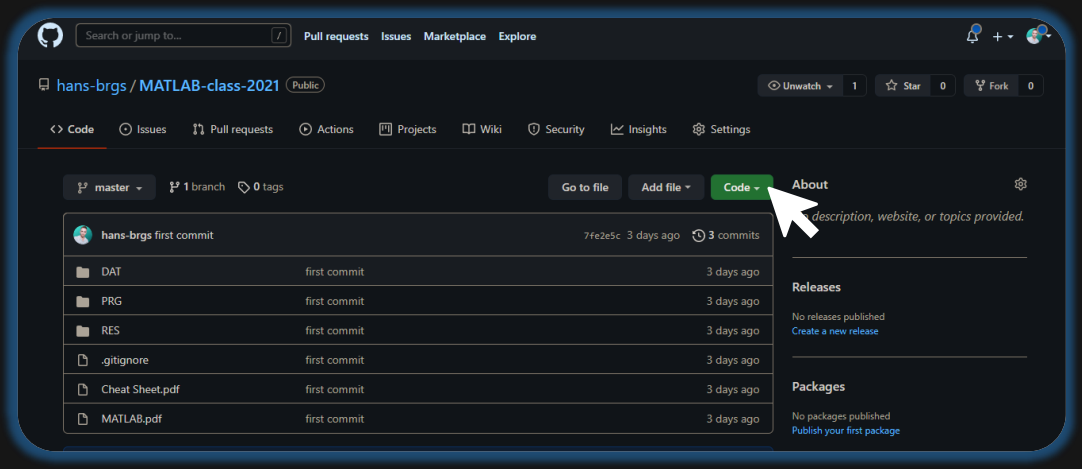
For your first project, you will analyse the mechanical data of three simple M-max stimulations. You will have three variables to analyse for each stimulation:

- The maximum force produced (PtP).
- The time of contraction (TC).
- The time of half relaxation (HRT).

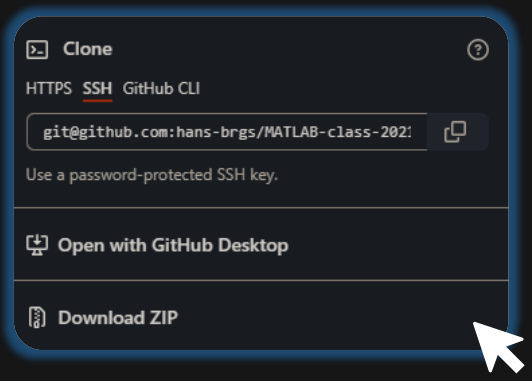


3. Preamble

Click on this link to download the folders and files necessary for the successful completion of the project.

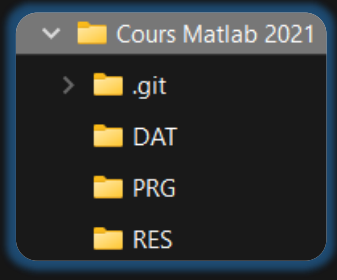


Select Code, and click on “Download Zip”.



Save the .zip file in your working folder, and extract it.

Perfect, everything is ready! Now take a look at the working architecture.



In your DAT directory you have all the Spike2 data and a copy of each of its files in .mat format.
In the PRG directory, you will save your script and your MATLAB functions required to analyse the data contained in the DAT folder.
In your RES directory you will save the excel file containing the final analysis of your data.

4. Step 1 – Create and Save a MATLAB Script

4.1 Objective

Create and save a MATLAB script in the newly created "PRG" folder.

Script Name	`main_stim_analyse.mat`
Put in folder	`/PRG`
External Functions to use	`None`
Key word for google	`"MATLAB Script" "Create" "Save"`

5. Step 2 – Load Spike Data in MATLAB environment

5.1 Objective

Load the mechanical data contained in the ".mat" file extension into the MATLAB workspace.

5.2 Description

Write in line of code in the Script "stim_analyse.mat" a way to load the file "S1_STIM.mat" in a variable called "DAT".

Write in file	`stim_analyse.mat`
Variable to create	`DAT`
External Functions to use	`load()`
Key word for google	`"load MAT file in matlab" "help load matlab"`

5.3 Bonus part I

Now, try to load the data starting from your current directory.

For this you need to :

- Store in a variable named "`PRG_path`", the path of your current directory (.../PRG).
- Store in a variable named "`parent_folder_path`", the path of your parent directory.
- Store in a variable named "`DAT_path`", the path of your folder .../DAT.
- Store in a variable named "`DAT_file`", the path of your file "S1_STIM.mat".

Write in file	`stim_analyse.mat`
Variable to create	`PRG_path, parent_folder_path, DAT_path, DAT_file`
External Functions to use	`pwd, fileparts(), fullfile()`
Key word for google	`"load MAT file in matlab" "help load matlab" "current directory" "parent directory"`

This implementation allows your script to be used by your lab partners using the same folder architecture.

5.4 Bonus part II

Try to create a function that encapsulates the feature of the previous script.
Your function will have the following prototyping:

```
%% Load Data
DAT = load_data(file_name);
```

Your function will be saved in your current directory.

Function name	`load_data`
Prototype	`DAT = load_data(file_name)`
External Functions to use	`pwd, fileparts(), fullfile(), load();`
Return value	`DAT. A structure containing data from the file "file_name".`
Key word for google	`"create function in MATLAB"`

6. STEP 3 – Pre-processing : Create sample time.

6.1 Objective

Create a field called "time" in our structure field "`DAT.force`". This field contains the time data of the force channel.

6.2 Description

First, find on google “create time vector from sampling rate/frequency”. (By the way, If you don't know what a sample frequency/rate is, this is also the time to google it.)

Secondly, calculates the sampling frequency from the value of the interval variable “`DAT.Force.interval`”. This value corresponds to the time interval between two samples. Store this value in a variable named “`fe`”.

Thirdly, implement your script to create a field called “time” in our structure field “DAT.force”.

Write in file	`stim_analyse.mat`
Variable to create	`fe, DAT.Force.time`
External Functions to use	`length(),`
Key word for google	`“time vector” “sampling rate” “create sequence of number in matlab”`

Here are some tips to help you :

You have to find the number of samples that corresponds to the size of your vector "DAT.Force.values".

6.3 Bonus

Try to create a function that encapsulates the feature of the previous script.
Your function will have the following prototyping:

```
%% Load Data
time = time_vector(data, fe);
```

Your function will be saved in your current directory.

Function name	`time_vector`
Prototype	`time = time_vector(data, fe)`
External Functions to use	`length()`
Return value	`time. A vector containing time data.`
Key word for google	`“create function in MATLAB”`

7. STEP 4 – Pre-processing : Cut and isolate each twitch in a vector with a specific window time.

First, without coding, try to think of a way to cut and isolate the force and time data from each contraction (Twitch) according to a time window:

- starting 0.5 seconds before the stimulation event
- ending 0.5 seconds after the stimulation event.

Here are some tips to help you :

Look in "DAT.Stim" at the field "times". The values in this field correspond to the time when the stimulations are delivered.

Learn about “array indexing” in matlab.

7.1 Objectif

Create a new structure called “`Twitch`” with two field “`Twitch.time`” and “`Twitch.force`”.
The "`Twitch.force`" field contains several matrices. Each matrix contains the force data of each contraction, according to a time window starting 0.5 seconds before the stimulation event and ending 0.5 seconds after this stimulation.
The "`Twitch.time`" field follows the same logic as "`Twitch.force`" but with temporal data.

7.2 Description

Firstly, using array indexing, create a function to find the corresponding index of “`DAT.Stim.times`” in “`DAT.Force.times`” and then in “`DAT.Force.values`”.

Here are some tips to help you :

The time resolution of the Stim channel is 40000hz and 500hz for the Force channel. Thus, the time value of "`DAT.Stim.times`" does not match the time values contained in "`DAT.Force.times`".
So we have to find the closest time value.

Function name	`Get_index`
Prototype	`idx = get_index(val, array)`
External Functions to use	`Abs()`
Return value	`idx. the position of the closest value to the "val" argument in the value array “array”.`
Key word for google	`“Find closest value in array MATLAB”`

Second, from the index value of each stimulus, select 250 samples (the equivalent of 0.5 seconds) before the position (index) of the stimulus and 250 samples after that index, for the force and time data.

Write in file	`stim_analyse.mat`
Variable to create	`Twitch(n).force, Twitch(n).time`
External Functions to use	`none`
Key word for google	`"Convert duration to a number of a samples" "array-indexing"`

8. STEP 5 - Pre-processing : Remove baseline offset

8.1 Objective

Modify "`Twitch.force`" set of data, to remove baseline offset of each set of data.

8.2 Description

First, we need to calculate the average value of the baseline before the twitch. The baseline values will be averaged over a window of 250 samples or 0.5 seconds.

Second, subtract the calculated average from the twitch data.

Write in file	`stim_analyse.mat`
Variable to modify	`Twitch(n).force`, `Twitch(n).baseline.mean`
External Functions to use	`mean()`
Key word for google	`"help mean() MATLAB"`

9. STEP 6 - Pre-processing : Filter

9.1 Objective

Learn how to use the "Signal Analyzer App", an interactive tool for smoothing and filtering data.

9.2 Description

First, open “Signal Analyzer App” from MATLAB.
Second to import the force data, You need to create a temporary matrix containing the force data of the first twitch, “`Twitch(1).force`”.

Use your “Command Window” to do this.

Write in	`command window`
Variable to create	`tmp_twitch1`
External Functions to use	`none`
Key word for google	

Third, use “Signal Analyzer App” to filter the noise or smoothing data. You can search the web for the most appropriate filter or smoothing for mechanical data.

Fourth, generate a function from the "Signal Analyzer App" with your own smoothing and filtering pre-processing. Apply this function to all data sets in "`Twitch(n).force`".

Function name	`preprocess_filt`
Prototype	`[y,ty] = preprocess_filt(x,tx)`
External Functions to use	`Depends on your filter and smoothing choice`
Return value	`y. filtering data.` `ty. Time data.`
Key word for google	`“generate function from Signal Analyzer App MATLAB”`

10. STEP 7 - Data Analysis : Detect the starting point of the contraction.

10.1 Objective

To detect the starting point of the twitch, we need to find the inflection point of the force data. Find a mathematical way to do this, write it in code.

10.2 Description

There are several ways to find an abrupt change in data.
The two main ones are the first derivate or an abrupt change in mean.

MATLAB offers predefined functions like "Ischange" or "findchangepoints" but these functions are complex, time consuming for the processor, and do not always work well for our datasets.

So try to create your own function.

10.3 One possible solution

Code review of my own “threshold_detection” function.

11. STEP 8 - Data Analysis : Detect Max peak of the twitch

11.1 Objective

Find peak value of the twitch.

11.2 Description

Find the maximum value of the twitch and the corresponding time.

Write in file	`stim_analyse.mat`
Variable to create	`Twitch(n).peak.v` `Twitch(n).peak.t`
External Functions to use	`max()`
Key word for google	`“help max() MATLAB” “array indexing”`

12. STEP 9 - Data Analysis : Detect time of Half Relaxation value

12.1 Objective

Find the value of Half Relaxation and the corresponding time.

12.2 Description

The value of the half-relaxation is half the value of the peak contraction.

First, calculate this value and store them in a variable called “`Twitch(n).half_relaxation.v`”.

Use our own function "`get_index`" to find the position of this value in the array "`Twitch(n).force`". Store it into the variable "`Twitch(n).half_relaxation.li`". With this index, find the true value the half-relaxation and the corresponding time. Store this two values in "`Twitch(n).half_relaxation.v`" and "`Twitch(n).half_relaxation.t`".

Write in file	`stim_analyse.mat`

Variable to create	`Twitch(n).half_relaxation.v`
	`Twitch(n).half_relaxation.li`
	`Twitch(n).half_relaxation.t`

External Functions to use	`get_index()`
Key word for google	"`array indexing`"

13. STEP 10 - Data Visualisation : Plot data

13.1 Objective

Plot the data in a figure.

13.2 Description

Plot the data in a figure to see if you script work well.
Information needed about your plot :

- A visual mark to see `Twitch(n).peak.v`
- A visual mark to see `Twitch(n).half_relaxation.v`
- A visual mark to see `Twitch(n).thr.start_v`
- `A title`
- `X and Y Label`

Write in file	`stim_analyse.mat`
Variable to create	`none`
External Functions to use	`figure, plot(), xlabel(), ylabel(), title()`
Key word for google	"`help plot() MATLAB`"

14. STEP 11 – SAVE DATA : Create variable of interest

14.1 Objective

Create and store variable of interest.

14.2 Description

Create and store the following variable :

- ct : contraction time, the difference between the time of the peak twitch and the time of the starting point of the contraction phase.
- ptp : peak to peak, the difference between the value of the peak twitch and the value of the starting point of the contraction phase.
- htr : half relaxation time, the difference between the time of the half relaxation value and the time of the starting point of the contraction phase.
- stim : The number of the twitch analysed.

Write in file	`stim_analyse.mat`
Variable to create	`ct(n, :), ptp(n, :), htr(n, :), stim(n, :)`
External Functions to use	`none`
Key word for google	`none`

15. STEP 12 – SAVE DATA : Store variable in table

15.1 Objective

Store previous data in table.

15.2 Descriptive

Table is a data structure under matlab. Google it, to learn about this data structure.
The tables facilitate the export of data to excels.

3x5 table

	1 sujet	2 stim	3 ct	4 ptp	5 htr
1	1	1	0.1120	62.0545	0.1960
2	1	2	0.1100	62.8213	0.1940
3	1	3	0.1100	63.6609	0.1940

Write in file	`stim_analyse.mat`
Variable to create	`RES`
External Functions to use	`table`
Key word for google	`"help table() MATLAB"`

16. STEP 13 – SAVE DATA : Export table to excel

16.1 Objective

Export data table to excel.

16.2 Description

Export the data table to a file called "resum.xlsx" in the /RES folder.
Each subject must automatically fill in the correct cell.

Write in file	`stim_analyse.mat`
Variable to create	`RES_path, RES_file`
External Functions to use	`writetable()`
Key word for google	`"help table() MATLAB"`