Project presentation (part 1)

SSY081

Transforms, signals and systems

Silvia Muceli
muceli@chalmers.se

SSY081 Python project

- You work in groups of 4 and write a common report
- The report should be written in English (use the template)
- Each group has to submit report and code through Canvas
- Report is due on the 18th of October (Please indicate group ID)
- Oral presentation is scheduled in the period 21st to 30th of October (time slots are available on the Canvas calendar. Please book one)
- During the project discussion, we will discuss about what you have done in the project and how it relates to what we have studied in the course
- You do not need to prepare any slide for the oral presentation. We will ask questions depending on your report/code
- The grade (UG) is individual
- It may happen that there is something to be changed in the report/code. You will be granted 10 days to implement the changes

Introduction

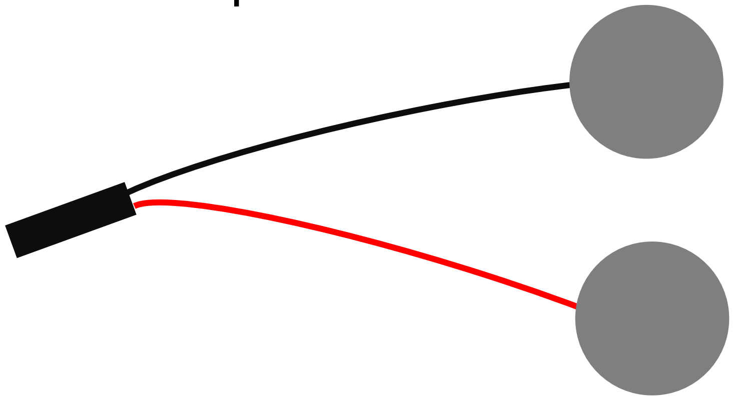
- You have learned in the course that systems can be used to model biological systems.
- In this project, you will learn how to model an electromyographic (EMG) signal recorded with intramuscular electrodes (electrodes placed within a muscle).

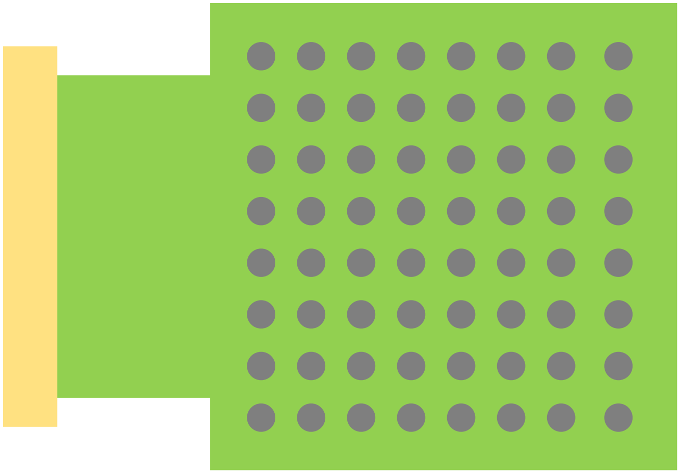
EMG recordings

Electrode grid

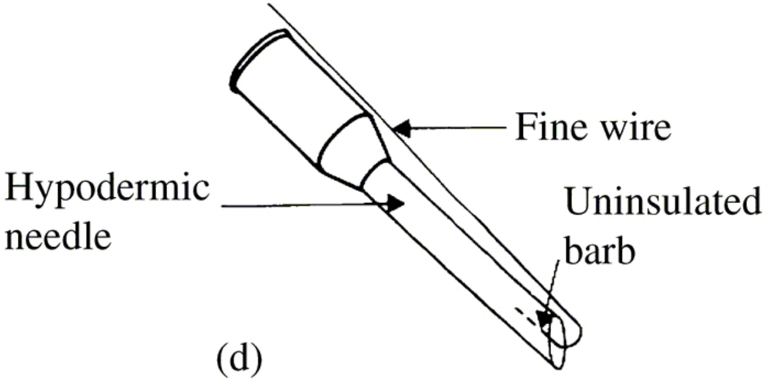
Bipolar electrodes

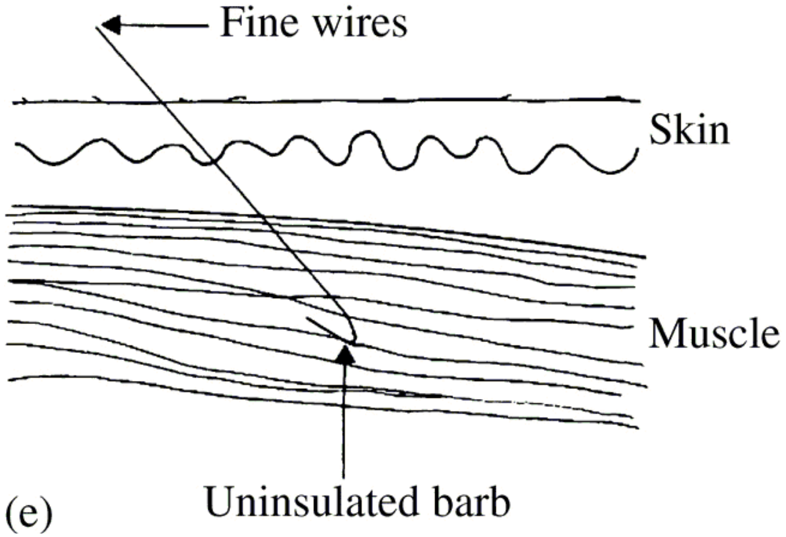
Surface EMG



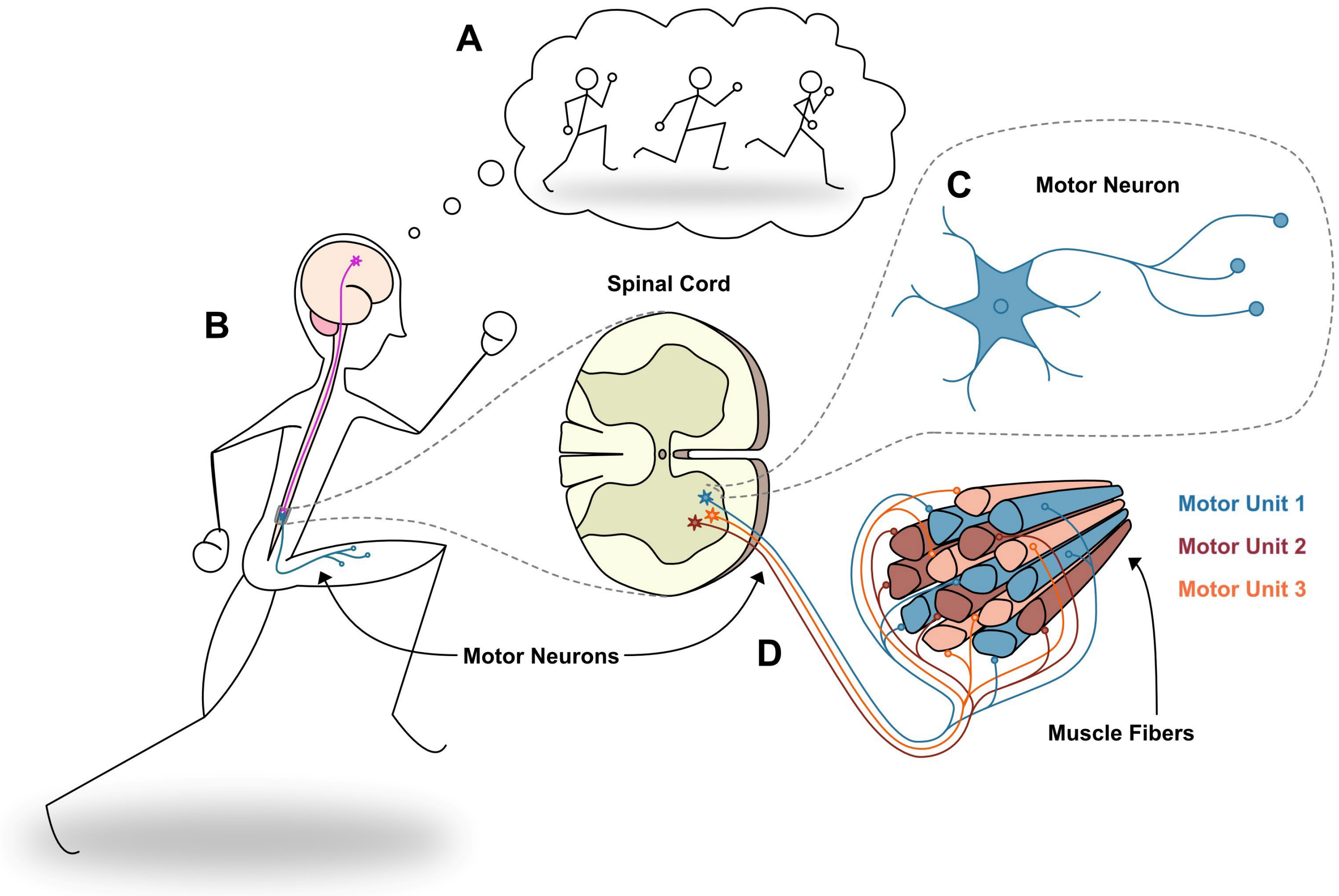


Intramuscular EMG





Our muscles are controlled by spinal motoneurons



Hamstreet & Muceli, Frontiers Young Minds, 2022

Introduction

- A muscle is composed by several functional units called motor units. An EMG signal is the sum of the electric potential generated by motor units when they are activated. When a motor unit is activated, it produces a spike known as action potential.
- Graphically, we can represent the EMG as the sum of trains of action potentials (Figure 1).

Figure 1

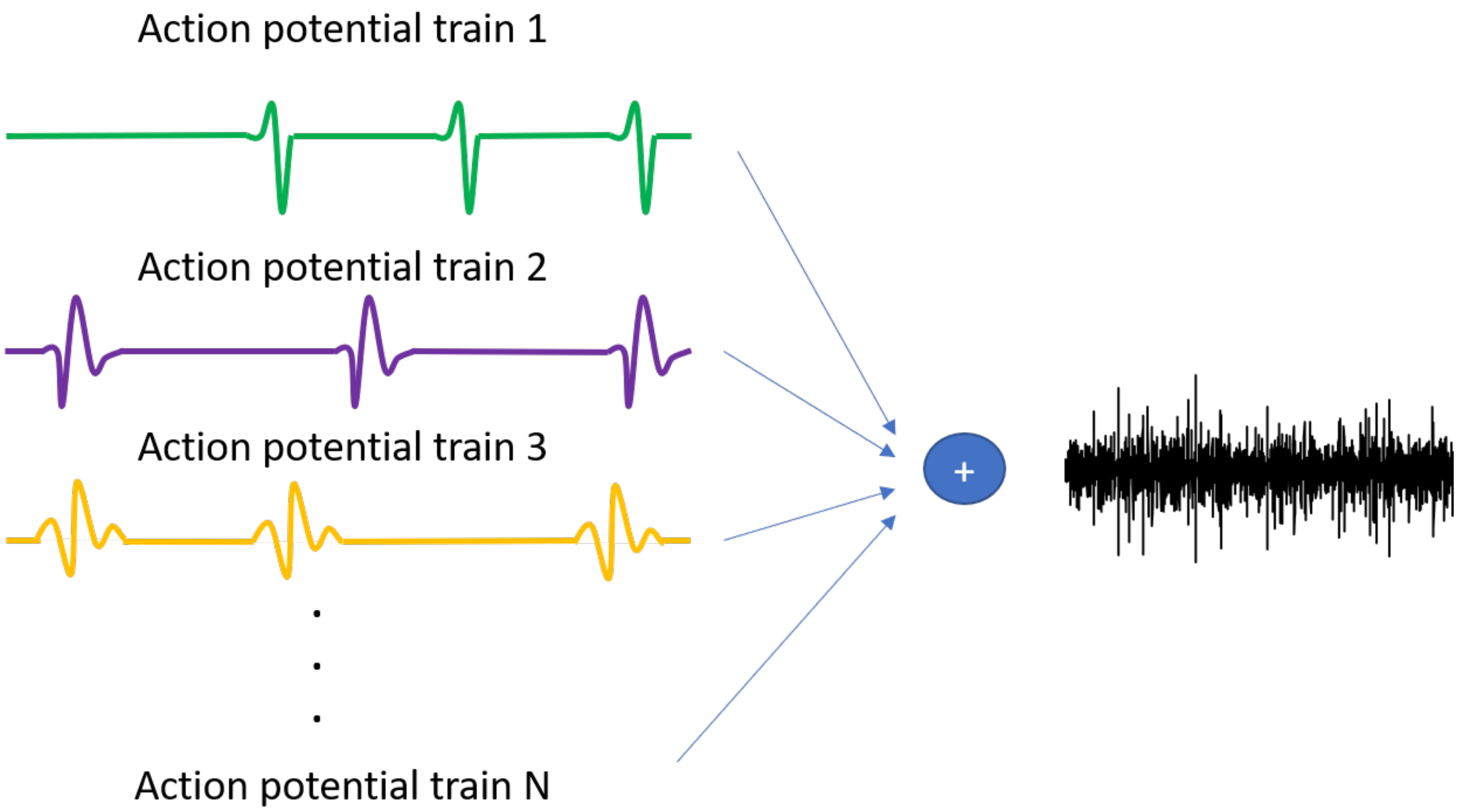
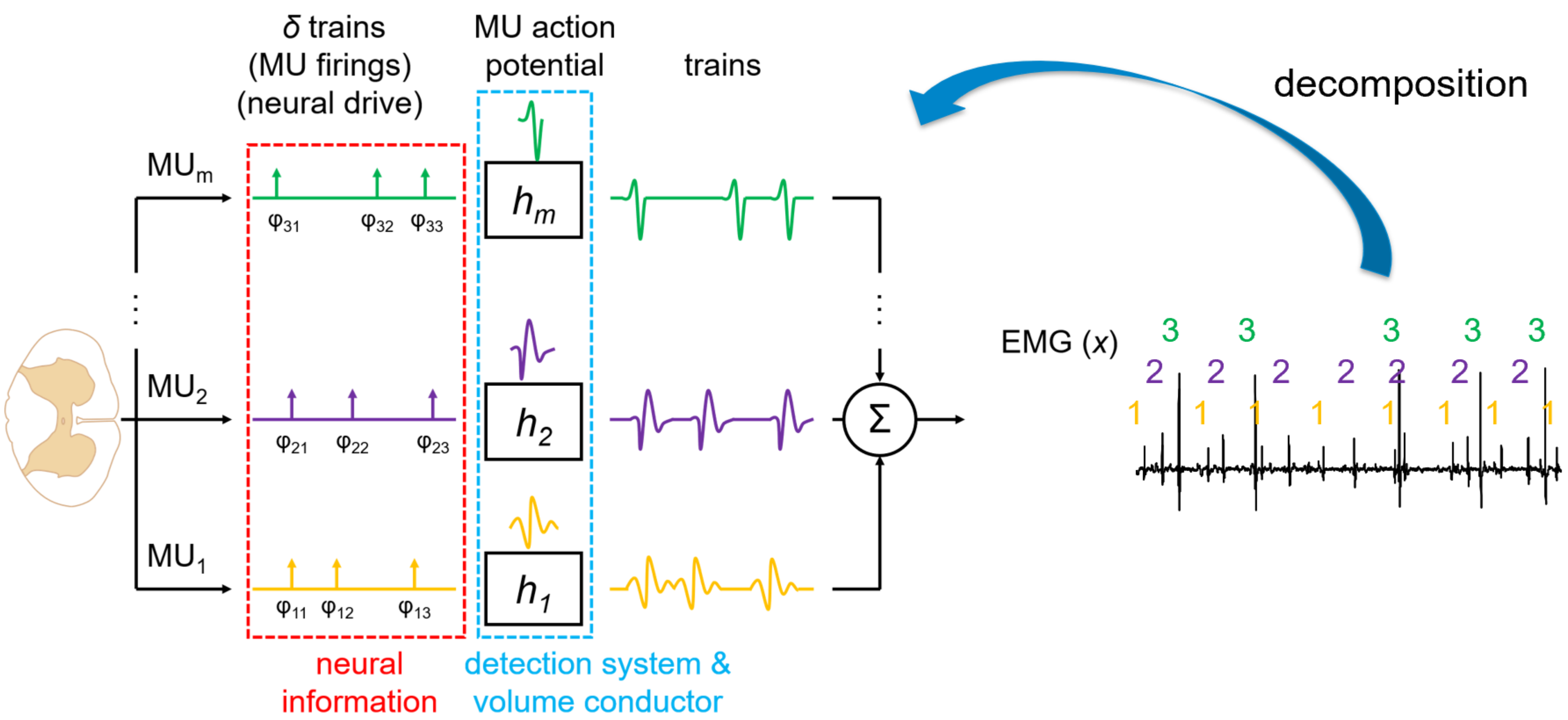


Figure 1 Generation of the EMG signal.

EMG recordings

- Usually, when we record the EMG signals, the train of action potentials are mixed together, but we can reverse-engineer the process and extract individual trains from the “mixed signal”
- This process is named decomposition

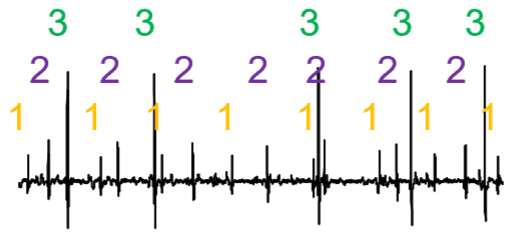
EMG decomposition



Muceli & Farina, Springer Handbook of Neuroengineering, 2021

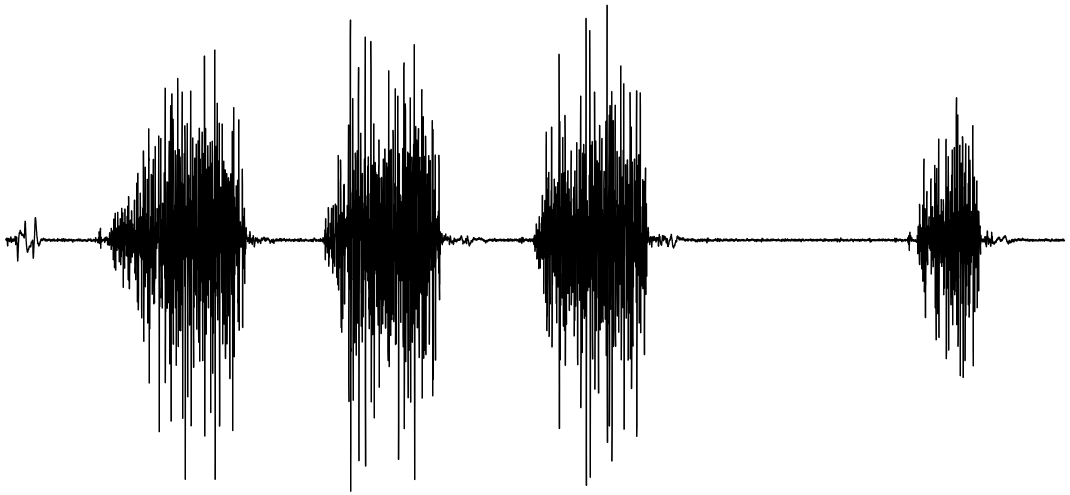
How do you decompose?

Intramuscular EMG



Template matching

Surface EMG



blind source separation
(cocktail-party problem)

Cerone et al, ISEK-JEK tutorial

No worries!

Decomposition is already done!

Introduction

- In Figure 1, we have considered 3 action potentials, represented in Figure 2.

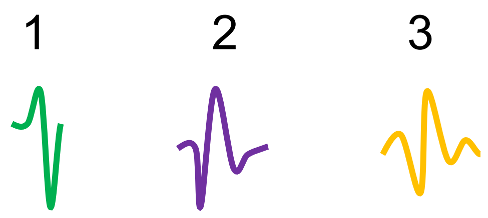


Figure 2 Action potentials.

- In each train of action potentials, the same action potential is repeated several times (only three times in Figure 1 for simplicity).
- We can indicate with t11, t12, t13, … the time instants at which action potential 1 is discharged. The corresponding samples are s11, s12, s13, … This is represented in Figure 3.

Figure 3

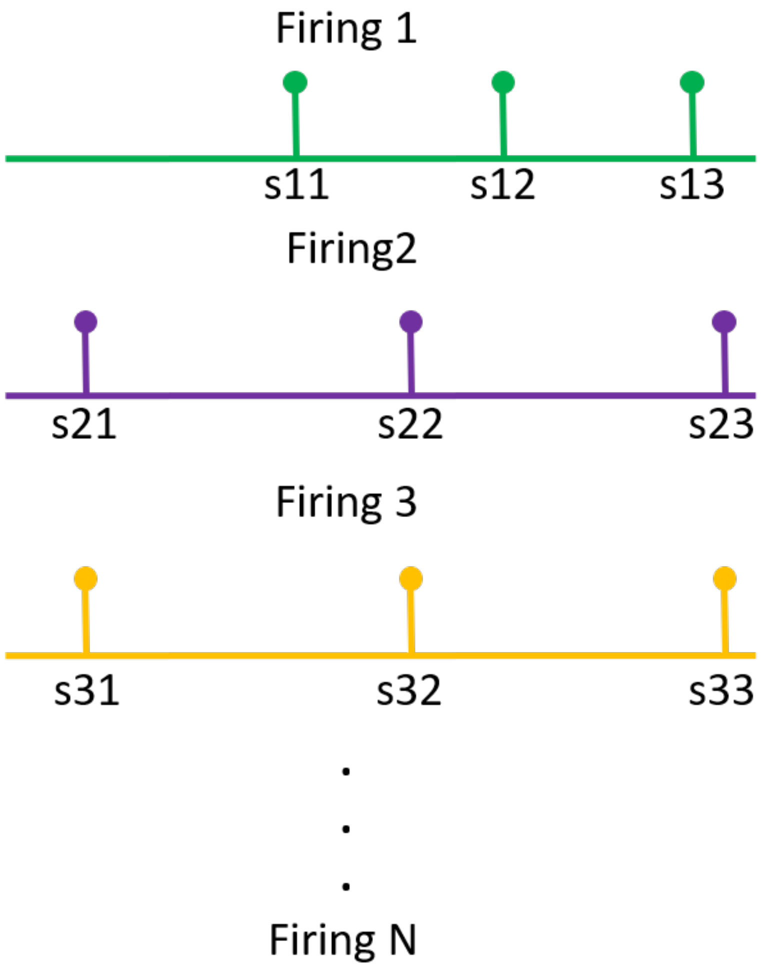


Figure 3 Firings of the action potentials represented in Figure 2.

Assignment

- You are supposed to create an EMG signal composed by 8 action potential trains.
- The signal duration should be 20 s, the sampling frequency 10,000 Hz.
- We provide you two NumPy array files:
- action_potentials.npy

It is a 8x100 matrix that contains the action potentials of 8 motor units.

The number of rows (8) is the number of motor units.

The number of columns (100) is the number of samples.

- firing_samples.npy

It is composed by 8 cells, one for each motor unit. Each cell is a vector that contains the index of the samples at which the discharges of action potentials occur. Note that the number of discharges is different for the different units.

Libraries

- Below is a basic list of Python libraries than can be used for the project. We assume you use the latest version of each package. Note that you are allowed to use other libraries if you want to. If using other libraries or older versions of the listed libraries, please clarify this in your code.
  - Math
  - Matplotlib
  - NumPy
  - OS
  - SciPy

Question 1

a) Create the trains of action potentials corresponding to each unit (8 trains in total).

Hint:

First step, for each action potential train, you can create a binary vector with samples equal to 1 in correspondence of the firings and 0 otherwise (you are supposed to simulate 20 s of signal).

Second step, you need to have a replica of the action potential in correspondence of each firing time. How can you obtain that?

Describe the procedure you have followed to create the trains.

b) Explain why you have followed the procedure in question 1a. **We expect a well-reasoned answer based on the theories discussed in class.**

c) How many samples does each action potential train contain? Why? Is the number of samples you obtained compatible with the procedure you have followed to create the trains of action potentials? **We expect a well-reasoned answer based on the theories discussed in class.**

Question 1

d) Plot 1 of the 8 action potential trains as a function of time (that means you should have time, rather than samples in the horizontal axis). In addition, plot the same action potential train in the time interval 10-10.5 s.

Note: To facilitate visualization, we suggest using a line thickness of 0.5. All axes must be labelled. The unit for the time axis should be s (seconds); the unit for the amplitude of the action potentials is not provided and you should indicate A.U. (which stands for arbitrary unit). That means that the label for the horizontal axis should be Time [s]. Use analog notation for the vertical axis and for all figures in the report.

e) Comment on the procedure you have followed to obtain the proper time axis. Which is the distance between two samples? **We expect a well-reasoned answer based on the theories discussed in class.** Note, we are not asking details about the Python implementation, we will look at that in the code. We want to know how you established the correspondence between each sample number and the time instant corresponding to that sample number.

f) Sum the 8 action potential trains to obtain the EMG signal. Plot the EMG signal as function of time (in the time interval 10-10.5 s).

Question 2

a) Filter the 8 binary vectors with samples equal to 1 in correspondence of the firing times. You should use a filter with impulse response equal to a Hanning window of duration 1 s (the Python function for creating the Hanning window with NumPy is "hanning"). Describe the procedure you have followed to filter the binary vectors.

b) Explain why you have followed the procedure in question 2a. **We expect a well-reasoned answer based on the theories discussed in class.**

c) Plot the 8 filtered signals as a function of time, in seconds (all in the same graph).

Question 2

a) Create another figure where you plot the fourth binary vector and the corresponding filtered version obtained in a). Describe the filter characteristics (e.g., low-pass, high-pass). We expect a well-reasoned answer based on the theories discussed in class.

e) Create another figure where you plot the seventh binary vector and the corresponding filtered version obtained in question 2a). Compare this figure to the previous one (question 2d). Which unit fires faster (4 or 7)?

f) Why is this unit firing faster? Did you reach that conclusion looking at the binary vectors or their filtered version? We expect a well-reasoned answer based on the theories discussed in class.

Question 3

- We have to study the Discrete Fourier Transform first

You are supposed to provide

- A report (**pdf** file) that contains the replies to all the above questions (figures and comments on how you achieved the results). The report should not exceed **6 A4** pages.
- The Python code (one file main.py with the solution to questions 1 and 2, one file mainQ3.py with the solution to question 3. No need to upload action_potentials.npy, firing_samples.npy, f.npy). Note: when I run the Python scripts, **we should be able to see all figures you included in the report without modifying the scripts**.
- Filled **checklist** (pdf file). We would suggest to read it first.

Good luck!