Student names: Lena Bruhin, Ludovica Romanin, Hugues Vinzant

Instructions: Update this file (or recreate a similar one, e.g. in Word) to prepare your answers to the questions. Feel free to add text, equations and figures as needed. Hand-written notes, e.g. for the development of equations, can also be included e.g. as pictures (from your cell phone or from a scanner). ***This lab is graded.*** *and must be submitted before the* ***Deadline : 11-04-2018 Midnight****.*
Please submit both the source file (.doc/*.tex) and a pdf of your document, as well as all the used and updated Python functions in a single zipped file called lab6_name1_name2_name3.zip where name# are the team member's last names. Please submit only one report per team!*

The file `lab#.py` is provided to run all exercises in Python. The list of exercises and their dependencies are shown in Figure 1. When a file is run, message logs will be printed to indicate information such as what is currently being run and and what is left to be implemented. All warning messages are only present to guide you in the implementation, and can be deleted whenever the corresponding code has been implemented correctly.

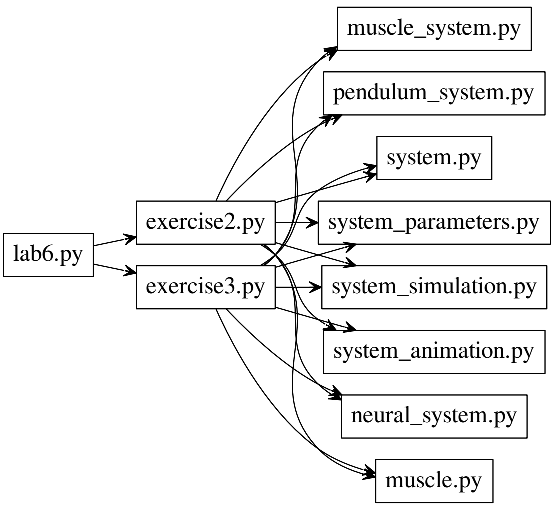


Figure 1: Exercise files dependencies. In this lab, you will be modifying `exercise1.py` and `pendulum_system.py`

Files to complete the exercises

- `lab6.py` : Main file
- `exercise2.py` : Main file to complete exercise 2
- `exercise3.py` : Main file to complete exercise 3
- `system_parameters.py` : Parameter class for Pendulum, Muscles and Neural Network (Create an instance and change properties using the instance. You do not have to modify the file)
- `muscle.py` : Muscle class (You do not have to modify the file)
- `system.py` : System class to combine different models like Pendulum, Muscles, Neural Network (You do not have to modify the file)
- `pendulum_system.py` : Contains the description of pendulum equation and Pendulum class. You can use the file to define perturbations in the pendulum.

- `muscle_system.py` : Class to combine two muscles (You do not have to modify the file)
- `neural_system.py` : Class to describe the neural network (You do not have to modify the file)
- `system_simulation.py` : Class to initialize all the systems, validate and to perform integration (You do not have to modify the file)
- `system_animation.py` : Class to produce animation of the systems after integration (You do not have to modify the file)

NOTE : '*You do not have to modify*' does not mean you should not, it means it is not necessary to complete the exercises. But, you are expected to look into each of these files and understand how everything works. You are free to explore and change any file if you feel so.

Exercise 2 : Pendulum model with Muscles

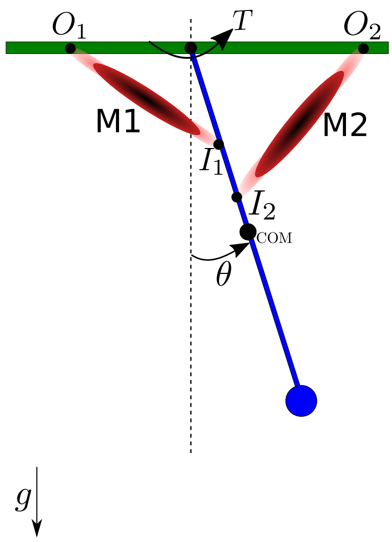


Figure 2: Pendulum with Antagonist Hill Muscles

The system is comprised of a physical pendulum described by equation 1 and a pair of antagonist muscles **M1** and **M2**. Muscle **M1** extends the pendulum (θ increases) and Muscle **M2** flexes the muscle (θ decreases).

Consider the system only for the pendulum range $\theta = [-\pi/2, \pi/2]$

$$I\ddot{\theta} = -0.5 \cdot m \cdot g \cdot L \cdot sin(\theta) \tag{1}$$

Where,

- I - Pendulum inertia about the pendulum pivot joint $[kg \cdot m^2]$
- θ - Pendulum angular position with the vertical $[rad]$
- $\ddot{\theta}$ - Pendulum angular acceleration $[rad \cdot s^{-2}]$
- m - Pendulum mass $[kg]$
- g - System gravity $[m \cdot s^{-2}]$
- L - Length of the pendulum $[m]$

Each muscle is modelled using the Hill-type equations that you are now familiar with. Muscles have two attachment points, one at the origin and the other at the insertion point. The origin points are denoted by $O_{1,2}$ and the insertion points by $I_{1,2}$. The two points of attachment dictate how the length of the muscle changes with respect to the change in position of the pendulum.

The active and passive forces produced by the muscle are transmitted to the pendulum via the tendons. In order to apply this force on to the pendulum, we need to compute the moment based on the attachments of the muscle.

Using the laws of sines and cosines, we can derive the length of muscle and moment arm as below. The reference to the paper can be found here Reference,

$$L_1 = \sqrt[2]{a_1^2 + a_2^2 + 2 \cdot a_1 \cdot a_2 \cdot \sin(\theta)} \tag{2}$$

$$h_1 = \frac{a_1 \cdot a_2 \cdot \cos(\theta)}{L_1} \tag{3}$$

Where,

- L_1 : Length of muscle 1
- a_1 : Distance between muscle 1 origin and pendulum origin ($|O_1C|$)
- a_2 : Distance between muscle 1 insertion and pendulum origin ($|I_1C|$)
- h_1 : Moment arm of the muscle

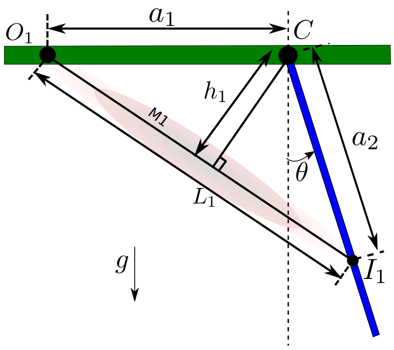


Figure 3: Computation of muscle length and moment arm

Equation 2 can be extended to the Muscle 2 in similar way. Thus, the final torque applied by the muscle on to the pendulum is given by,

$$\tau = F \cdot h \tag{4}$$

Where,

- τ : Torque $[N \cdot m]$
- F : Muscle Tendon Force $[N]$
- h : Muscle Moment Arm $[m]$

In this exercise, the following states of the system are integrated over time,

$$X = \begin{bmatrix} \theta & \dot{\theta} & A_1 & l_{CE1} & A_2 & l_{CE2} \end{bmatrix} \tag{5}$$

Where,

- θ : Angular position of the pendulum [rad]
- $\dot{\theta}$: Angular velocity of the pendulum [rad/s]
- A_1 : Activation of muscle 1 with a range between [0, 1]. 0 corresponds to no stimulation and 1 corresponds to maximal stimulation.
- l_{CE1} : Length of contracticle element of muscle 1
- A_2 : Activation of muscle 2 with a range between [0, 1]. 0 corresponds to no stimulation and 1 corresponds to maximal stimulation.
- l_{CE2} : Length of contracticle element of muscle 2

To complete this exercise you will make use of the following files, `exercise2.py`, `system_parameters.py`, `muscle.py`, `system.py`, `pendulum_system.py`, `muscle_system.py`, `system_simulation.py`

2a. For a given set of attachment points, compute and plot the muscle length and moment arm as a function of θ between $[-\pi/4, \pi/4]$ using equations in eqn:2 and discuss how it influences the pendulum resting position and the torques muscles can apply at different joint angles. You are free to implement this code by yourself as it does not have any other dependencies.

The pendulum angular position is representative of the muscle contraction. The maximum and minimum contractions are respectively obtained for the smallest (negative) and largest (positive) angle values. In a physiological representation, this muscle would be a flexor muscle. The more the muscle is contracted the higher the momentum and the lower the length. These conditions produce the maximum torque for a given force, as defined in the equation 4.

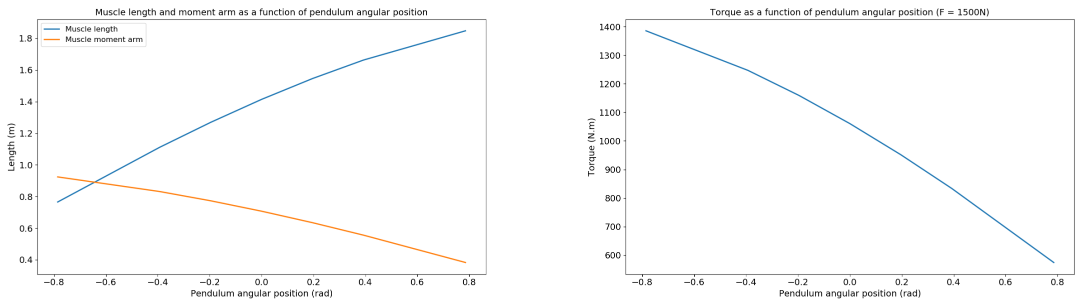
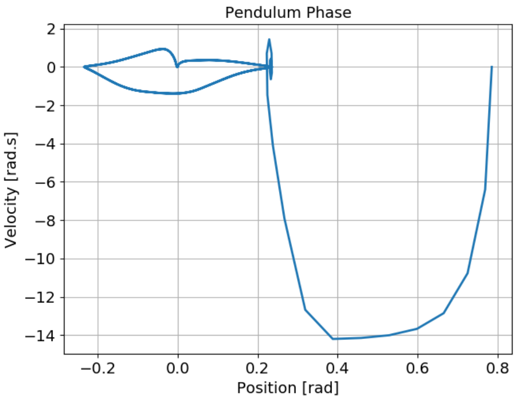


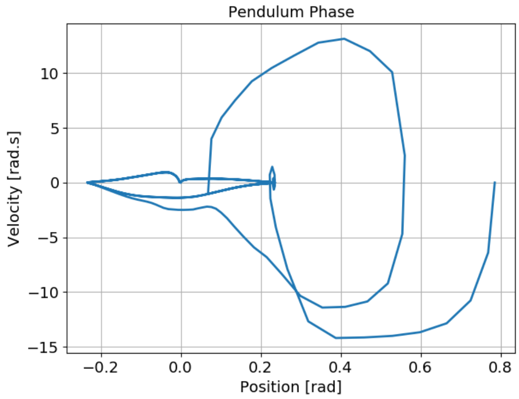
Figure 4: Muscle length and moment arm as a function of the angular position

2b. Using simple activation wave forms (example : sine or square waves) applied to muscles (use `system_simulation.py::add_muscle_activations` method in `exercise2.py`), try to obtain a limit cycle behavior for the pendulum. Use relevant plots to prove the limit cycle behavior. Explain and show the activation wave forms you used. Use `pendulum_system.py::PendulumSystem::pendulum_system` function to perturb the model.

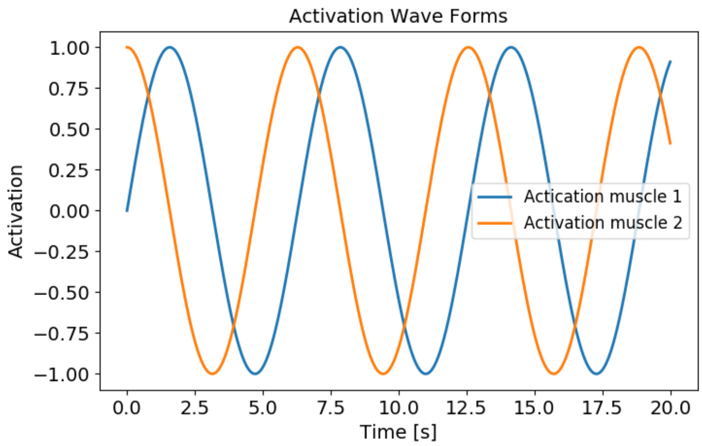
By using activation wave forms, such as sine and cosine waves, we can obtain a limit cycle behaviour for the pendulum; the activation of the two muscles will have a phase shift and after a certain time interval will get back to their initial state. The phase shift is important so that the two muscles do not get activated at the same time and work against each other, while still getting back to the initial state to repeat the cycle again. The form of the limit cycle is not a cycle or an oval shape, as can be seen in Figure 5a, but more an irregular shape. It is nevertheless a limit cycle because the pendulum will



(a) Phase plane for simple activation wave form



(b) Phase plane for simple activation wave form with perturbation



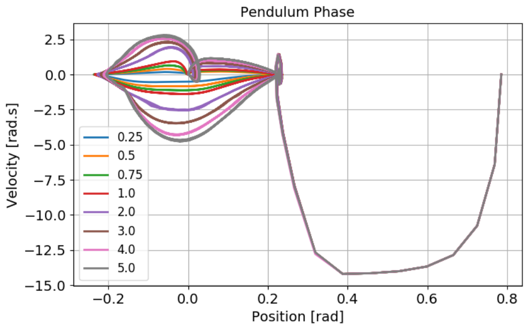
(c) Activation wave forms

Figure 5: Behaviour of the pendulum model with muscles with simple activation wave forms

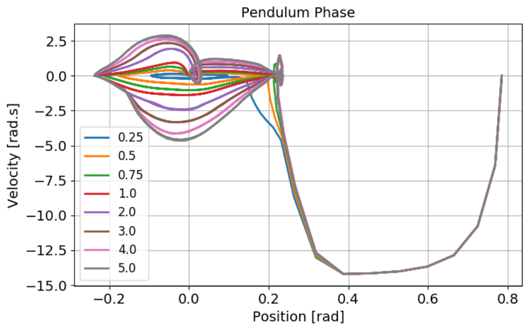
remain on this trajectory after an initial period of stabilization. When the same pendulum receives a perturbation, the trajectory will momentarily deviate from the limit cycle but eventually also return on it (Fig. 5b). The used activation wave forms can be seen in 5c. One could also use sine and -sine or other wave forms combinations, as long as these activation waves result in different times of activation of the muscles, so that the pendulum can move.

2c. Show the relationship between stimulation frequency and amplitude with the resulting pendulum's behavior.

The activation wave form describes the stimulation of the muscles. When changing the frequencies of the stimulation, the extremum position of the trajectories will remain the same for all frequencies, but the speed with which the pendulum gets to these extrema changes. A low frequency results in low velocities, while a high frequency will result in higher velocities (Fig. 6a). When keeping the frequency constant but varying the amplitude of the stimulation, you influence both the extrema of the trajectories as well as the velocities. A small amplitude will result in small changes of position with small speeds while a large amplitude will result in larger trajectories and higher velocities.



(a) Phase Plane with different stimulation frequencies



(b) Phase Plane with different stimulation amplitudes

Figure 6: Phase plane of the pendulum model with muscles with different stimulations

Exercise 3 : Neural network driven pendulum model with muscles

In this exercise, the goal is to drive the above system 2 with a symmetric four-neuron oscillator network. The network is based on Brown's half-center model with fatigue mechanism. Here we use the leaky-integrate and fire neurons for modelling the network. Figure 7 shows the network structure and the complete system.

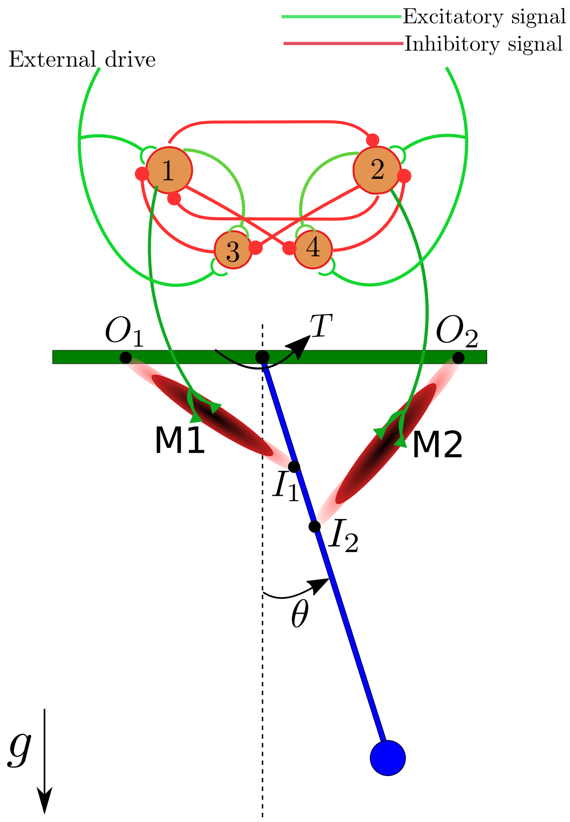


Figure 7: Pendulum with Antagonist Hill Muscles Driven Half Center Neural Network.

Since each leaky-integrate and fire neuron comprises of one first order differential equation, the states to be integrated now increases by four(one state per neuron). The states are,

$$X = \begin{bmatrix} \theta & \dot{\theta} & A_1 & l_{CE1} & A_2 & l_{CE2} & m_1 & m_2 & m_3 & m_4 \end{bmatrix} \tag{6}$$

Where,

- m_1 : Membrane potential of neuron 1
- m_2 : Membrane potential of neuron 2
- m_3 : Membrane potential of neuron 3
- m_4 : Membrane potential of neuron 4

To complete this exercise, additionally you will have to use `neural_system.py` and `exercise3.py`

3a. Find a set of weights for the neural network that produces oscillations to drive the pendulum into a limit cycle behavior. Plot the output of the network and the phase plot of the pendulum

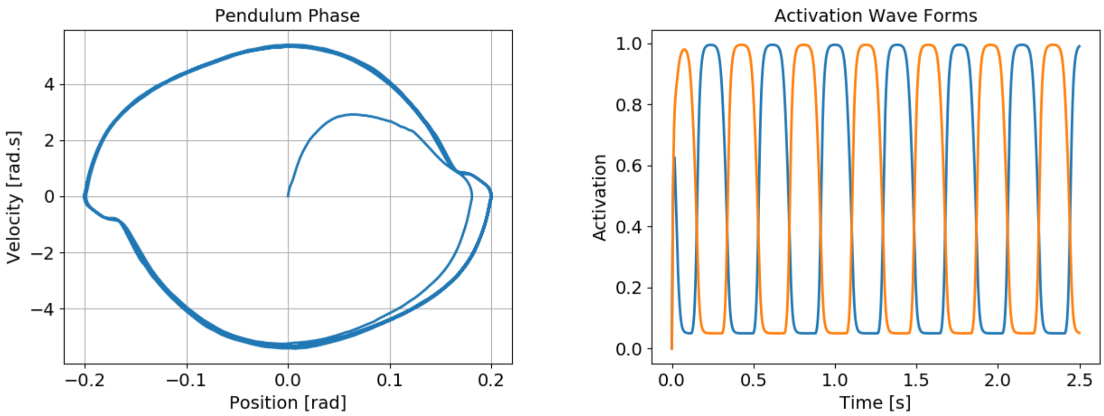


Figure 8: Phase diagram of the pendulum system during a limit cycle behavior (left) and the corresponding activation wave forms for muscles 1 and 2 over time (right).

The two half-centers 1 and 2 are responsible for the generation of the contraction in muscles 1 and 2 respectively, whereas the neurons 3 and 4 model the inhibitory interneurons, responsible for the reciprocal inhibition between the two muscles.

In order to find a limit cycle behavior or, in other words, the moment when the pendulum oscillates between activation of muscle 1 and activation of muscle 2, we set the following weights:

$$\begin{bmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \\ w_{13} & w_{23} & w_{33} & w_{43} \\ w_{14} & w_{24} & w_{34} & w_{44} \end{bmatrix} = \begin{bmatrix} 0 & -5 & -5 & 0 \\ -5 & 0 & 0 & -5 \\ 5 & -5 & 0 & 0 \\ -5 & 5 & 0 & 0 \end{bmatrix}$$

The absolute value of the weights is the same for all neurons, so that we have a balanced activation for the two muscles. The positive weights represent activation and the negatives ones inhibition. As we can see in Figure 8, for this set of weights the half-center pendulum system exhibits a limit cycle behavior, in which muscles 1 and 2 are alternatively activated.

A physiological interpretation of the system, as detailed in the paper from Markin et al., would separate the limit cycle into a swing phase for negative velocities and a stance phase for positive velocities.

Moreover, we can observe alternating activation of the two antagonists muscles, flexor (muscle 1) and extensor (muscle 2).

3b. As seen in the course, apply an external drive to the individual neurons and explain how the system is affected. Show plots for low [0] and high [1] external drives. To add external drive to the network you can use the method `system_simulation.py::add_external_inputs_to_network`

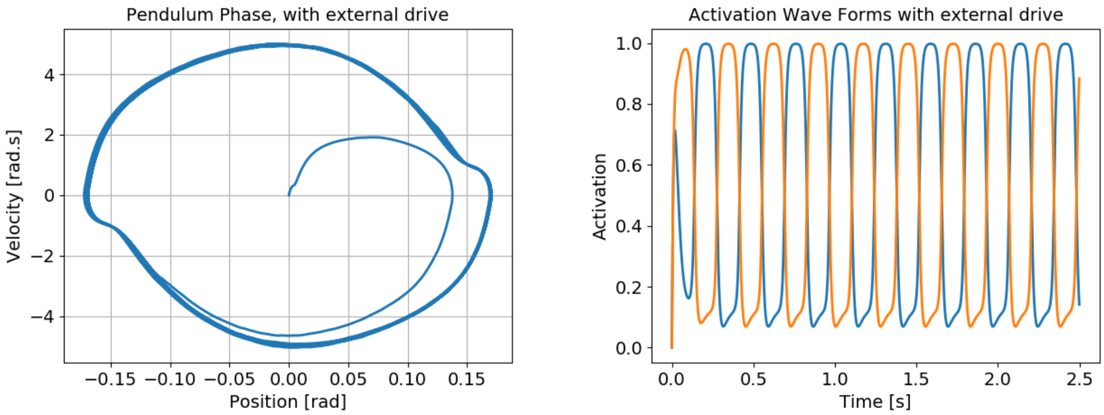


Figure 9: Phase diagram and muscle activation wave forms when applying an external drive.

When applying an external drive (Figure 9), we increase the system's speed and thus the duration of muscles' activation. Nevertheless, the behavior of the system (alternation between contractions of muscle 1 and 2) remains the same.

For low external drive, at zero in Figure 8, the system relies purely on the afferent inputs from the neurons, thus the frequency of activation in the muscles is lower. When increasing the external drive at 1, high level, the rate of oscillation increases and consequently the cycle duration decreases.

3c. [Open Question] What are the limitations of the half center model in producing alternating patterns to control the pendulum? What would be the effect of sensory feedback on this model? (No plots required)

This model does not contain any feedback loop from the muscles to the four neurons. The sensory feedback would provide stabilization to the pendulum movement and adjust the behavior of the system based on the environment characteristics (perturbations).