Developing a simple computational model of muscle to predict human wrist torques and stiffness from experimental EMGs and joint angles
-
Workshop proposed by Dr. Arnault Caillet, Imperial College London Workshop 5 - SSNR2023, Baiona

Aim of the workshop

In this section of the workshop, you will develop a computational pipeline to **predict the wrist torques** (in N.m) that you physiologically produced during the flexion/deviation tasks on Day 2. To do so, you will develop a mathematical pipeline (Figure 1) to estimate the forces (in N) developed by the 43 muscles of the hand & forearm. In this pre-coded pipeline, you will especially help in developing the model of muscle (blue box in Figure 1).

In brief, the experimental data you recorded on Day 2 (orange boxes in Figure 1) is processed offline (green boxes in Figure 1) into new variables ($l^{MT}, emg, L, F_0^M, l_0^M, l_s^T$). You will use and transform these 6 input quantities to **simulate the 43 muscle contracting and predict their muscle forces $\boldsymbol{F^M}$**, by **adapting the Matlab code that describes the generic model of muscle (blue box in Figure 1)**. These approximated muscle forces F^M, once multiplied by the moment arms L the muscles make with the wrist, will produce the wrist torques T we aim for:

$$T(t) = \sum_{i=1}^{43} L_i(t) \cdot F_i^M(t)$$

Importantly, the model of muscle, that estimates the F^M values, is **generic** (blue box in Figure 1). This means that you only need to code it once. Then, you can reuse it to simulate the 43 muscles by simply scaling it with muscle-specific properties (F_0^M, l_0^M, l_s^T), that are provided.

Main Remarks about the Matlab code

- You will peruse the parent code EMG_MSK_driven_modelling.m that reproduces the pipeline in Figure 1. You will find, commented in the parent code, that 4 functions must be edited for the parent code to run.
- The Matlab functions you will edit are stored in the Matlab_functions_to_edit folder. The solutions, if required, can be copy-pasted from the Matlab_functions_solutions folder.
- Once the functions are edited, you can run the parent code to predict the wrist torques $T(t)$ as described in Figure 1. **Please add the other folders to Matlab's path.**

Other remarks

- The input variables ($l^{MT}, emg, L, F_0^M, l_0^M, l_s^T$) necessary to make the predictions are stored into .mat files, that are saved in the Input_data folder.
- The Matlab code necessary to run the green boxes in Figure 1 and produce that Input Data is provided for information in the folders MSK_modelling and Other_Matlab_functions. You will not have to run those codes.

Important: as the parent code manipulates matrices and not scalars, you must use matrix operations in the Matlab functions you edit, i.e., '.*' or './' rather than '*' and '/'.

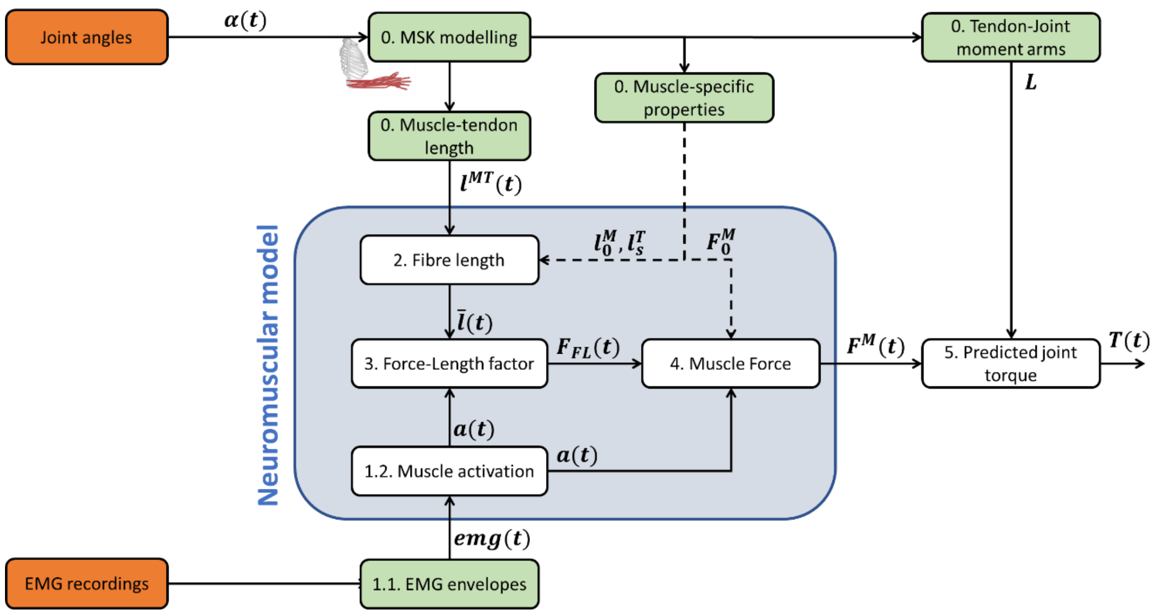

Figure 1: Summary of the simulation pipeline for a single of the 43 investigated muscles. Two experimental inputs: the joint angles and the EMG signals (orange boxes). After offline processing (green boxes), those will control a neuromuscular model (blue box), which predicts the muscle force $F^M(t)$. You will help in coding this box. The model is scaled with muscle-specific properties (F_0^M, l_0^M, l_s^T) obtained from the MSK model. Finally, the predicted muscle forces are multiplied to the tendon-joint moment arms L to estimate the joint torque $T(t)$. The numbering corresponds to the sections of the ***EMG_MSK_driven_modelling.m*** *parent code.*

The dynamics of contracting muscles – a few things

Muscles are force-generating structures that transform some neural message from the spinal cord into voluntary muscle forces. This physiological process relies on a multiscale summation of molecular forces into the whole muscle force, following neuromechanical events that we will model phenomenologically in this workshop (blue box in Figure 1). The whole muscle forces are transmitted to the skeleton via tendons, that make moment arms with the joint they cross. Consequently, by contracting and pulling on their tendons, muscles can produce joint torques, which allow motion, for example. This is summarized in Figure 2 for 3 muscles.

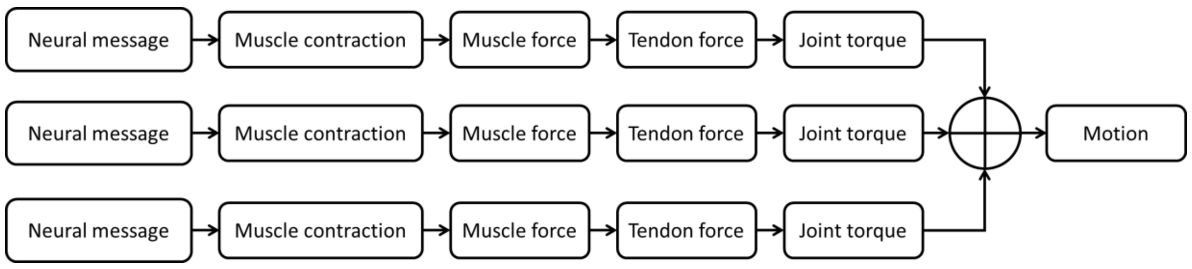

Figure 2: From neural messages to limb motion: how muscles and tendons create joint torques

What are computational models of muscle?

It is impossible to directly measure *in vivo* the forces developed by the muscles of the human body. To approximate those muscle forces F^M, computational models of muscles (like in the blue box in Figure 1) are commonly used. They aim to replicate the dynamics of muscle contraction, with various levels of modelling precision, to predict some of the quantities that are impossible to measure experimentally, like the muscle forces F^M.

What about the model of muscle you will develop? (blue box in Figure 1)

In the model developed in this workshop, the muscle force is approximated with the multiplication of 3 key factors:

$$\boldsymbol{F^M(t) = F_0^M \cdot a(t, emg) \cdot F_{FL}(t, a, \bar{l})}$$

- $\boldsymbol{F_0^M}$: the muscle maximum isometric force (N), i.e., the maximum force the muscle can theoretically generate.
- $\boldsymbol{a}$: the muscle active state (in [0-100%]), i.e., the level of muscle activation according to the 'strength' of the neural message.
- $\boldsymbol{F_{FL}}$: the Force-Length scaling factor (in [0-1]), i.e., how much the force-generating capacities of the muscle are penalized because the muscle is contracting at short or stretched lengths.

These factors rely on the following variables:

- $\boldsymbol{t}$: time. In the parent code, a time step consistent with the experimental sampling frequency (2048 Hz) is used.
- $\boldsymbol{emg}$: normalized EMG envelopes (in [0-1]). They are obtained by filtering and normalizing the experimental EMG data.
- $\boldsymbol{\bar{l}}$: normalized average length of the muscle fibres (in [0.5-1.5]). Let's discuss that parameter a bit later.

LET'S START CODING THE MODEL OF MUSCLE

1. Let's compute the muscle active state $a(t, emg)$

Function to edit: **activation_function.m,** called in **Section 1.2.** of the parent code

The muscle active state $a(t, emg)$ gives an indication of the level of muscle activation in [0-100%], based on the 'strength' of the neural message, that is given by the amplitude of the EMG envelope $emg(t)$. It relies on very complex physiological mechanisms, but can be approximated phenomenologically with a simple differential equation:

$$\frac{da}{dt} = \frac{1}{\tau(emg, a)}(emg - a)$$

$$\begin{cases} \tau(emg, a) = 0.01 \cdot (0.5 + 1.5a) & ; \quad emg > a \\ \tau(emg, a) = \dfrac{0.04}{(0.5 + 1.5a)} & ; \quad emg \leq a \end{cases}$$

Because we are solving the ODE time step-by time -step (dt: step size) with the Euler method, **you should code in activation_function.m**:

$$\boxed{\begin{gathered} \boldsymbol{da = \frac{1}{\tau(emg, a)} \cdot (emg - a) \cdot dt} \\ \begin{cases} \boldsymbol{\tau(emg, a) = 0.01 \cdot (0.5 + 1.5a)} & ; \quad \boldsymbol{emg > a} \\ \boldsymbol{\tau(emg, a) = \dfrac{0.04}{(0.5 + 1.5a)}} & ; \quad \boldsymbol{emg \leq a} \end{cases} \end{gathered}}$$

2. Let's compute the normalized average length of the muscle fibres $\bar{l}$

Function to edit: **get_muscle_length_function.m,** called in **Section 2.** of the parent code

Depending on the average length of its fibres l^M, i.e., depending on whether the muscle is short or stretched when it contracts, a muscle can physiologically produce more or less force. It is therefore important to estimate l^M. How can we do that?

On Day 2, you tracked wrist angles $\alpha(t)$ (orange box in Figure 1). Before this workshop on Day 3, we used those angles $\alpha(t)$ to move the limbs of the musculoskeletal (MSK model) in Figure 3 to reproduce the wrist motion you tracked on Day 2.
In this MSK model, the generic geometry of the 43 muscle-tendon systems (muscle + tendon) we investigate is reproduced and fixed to the bone segments (red lines in Figure 3). The platform Opensim (Delp et al., 2007) computes, using this geometrical reproduction of the human forearm, the time-histories of the muscle-tendon lengths $l^{MT}(t)$ of the 43 muscles during the wrist motions. The $l^{MT}(t)$ values are already computed for you (green box in Figure 1) and stored into **./Input_data/lMT.mat**, which is loaded in the parent code.

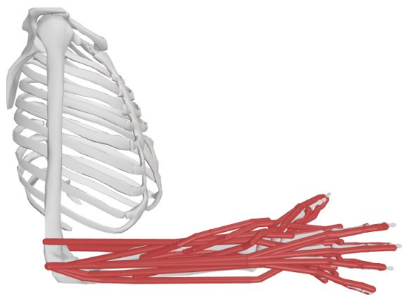
Figure 3: MSK model of the hand and forearm, from the OpenSim platform (Delp et al., 2007), used in this workshop

Now, how to go from muscle-tendon lengths l^{MT} to normalized muscle fibre lengths $\bar{l}$?

First, let's simplify the muscle-tendon system (muscle + its tendon) as a single muscle fibre (of length l^M) connected to a tendon (of length l^T). It yields for each muscle: $l^{MT} = l^M + l^T$.
Although the physiological tendon can stretch during contractions (and this can be accounted for with more complex modelling approaches), let's further assume the tendons are here rigid, with constant length $l^T = l_s^T$.
Therefore, $l^{MT} = l^M + l_s^T$. Or, $l^M = l^{MT} - l_s^T$.

Finally, depending on the muscle, the fibres generate their peak force at different lengths (in m), that are called optimal fibre lengths l_0^M. It is therefore common to normalize $l^M(t)$ by l_0^M, to make the muscle model generic.

Therefore, to get the normalized lengths $\bar{l}$, **you should code in get_muscle_length_function.m**

$$\boxed{\bar{l} = \frac{l^{MT} - l_s^T}{l_0^M}}$$

l_s^T and l_0^M are muscle-specific properties and a list of those values are provided to you for the 43 muscles with **./Input_data/Muscle_properties.mat** (green box in Figure 1).

3. Let's compute the Force-Length scaling factor $F_{FL}(t, a, \bar{l})$

Function to edit: **F_L_function.m,** called in **Section 3.** of the parent code

At this stage, we know at each time step the normalized length of the muscle fibres $\bar{l}(t)$. Thanks to experimental works, we know that the force-generating capacities of muscles vary with $\bar{l}$ **according to the following equation, that you should code in F_L_function.m**. In summary, if $\bar{l}$ diverges from 1 (optimal length) by muscle shortening or lengthening, the muscle can physiologically produce less force, following a bell-shape function, for a similar 'strength' of the neural message.

$$\boldsymbol{F_{FL}(\bar{l}, a) = \exp\left(-\left(\frac{\bar{l} - l_0(a)}{0.45}\right)^2\right)}$$

$$\boldsymbol{l_0(a) = 0.15 \cdot (1 - a) + 1}$$

4. Time to compute the muscle forces $F^M(t)$!

At this stage, you will have computed all the data necessary to estimate the muscle forces (in N) with the following equation, that is already coded in **Section 4.** of the parent code.

$$\boldsymbol{F^M(t) = F_0^M \cdot a(t, emg) \cdot F_{FL}(t, a, \bar{l})}$$

The muscle-specific F_0^M values are provided to you for the 43 muscles with **./Input_data/Muscle_properties.mat** (green box in Figure 1).

5. Let's wrap up with the joint Torques $T(t)$

Function to edit: **get_Torque_function.m,** called in **Section 5.** of the parent code

Let's wrap up this workshop by computing the wrist torques $T(t)$. When contracting, the muscles pull on the skeleton and generate torques (in N.m) around the joints they cross. Each of the 43 muscles makes with the wrist joint in the flexion/deviation directions a specific moment arm L, that varies with the joint angles $\alpha(t)$, i.e., with time t. Those moments arms $L(t)$ were computed with the MSK model in Figure 3 and are provided to you for the 43 muscles with **./Input_data/L_flex.mat** and **./Input_data/L_dev.mat** (green box in Figure 1).

Therefore, **you should code in get_Torque_function.m**

$$\boldsymbol{T = L \cdot F^M}$$

CONCLUSION

Once the 4 functions are coded, hit F5 on **EMG_MSK_driven_modelling.m**. This will generate the predicted wrist torques $T(t)$ in flexion/abduction.

In a nutshell, the wrist torques were estimated time step-by time -step as followed:

$T(t) = \sum_{i=1}^{43} L_i(t) \cdot F_i^M(t)$	Joint torque = action of the muscle forces through their moment arms over the wrist joint, summed across the 43 muscles	Section 5, L126 and L141
$F^M = F_0^M \cdot a \cdot F_{FL}$	Using a generic model of muscle, the estimated muscle force F^M for each of the 43 muscle is the product of the muscle maximum force F_0^M, its active state a, and its FL factor F_{FL}	Section 4, L112
$F_{FL}(\bar{l}, a) = \exp\left(-\left(\frac{\bar{l} - l_0(a)}{0.45}\right)^2\right)$ $l_0(a) = 0.15 \cdot (1 - a) + 1$	Description of the FL scaling factor F_{FL}	Section 3, L109
$\bar{l} = \frac{l^{MT} - l_s^T}{l_0^M}$	Description of the normalized fibre length $\bar{l}$, estimated from the muscle-tendon length l^{MT}, the optimal fibre length l_0^M, and the tendon slack length l_s^T. l^{MT} was derived before the workshop from the experimental joint angles $\alpha(t)$ and MSK modelling	Section 2, L96
$da = \frac{1}{\tau(emg, a)} \cdot (emg - a) \cdot dt$ $\begin{cases} \tau(emg, a) = 0.01 \cdot (0.5 + 1.5a)\,; emg > a \\ \tau(emg, a) = \frac{0.04}{(0.5 + 1.5a)}\quad ;\ emg \le a \end{cases}$	Description of the muscle active state a, estimated from the EMG envelopes emg. emg was derived before the workshop from the experimental EMG signals	Section 1.2 L74 to 77