Influence of ENG electrode characteristics on signal quality - User Instructions

By Varvara Tebieva

Supervisor: Anne Vanhoestenberghe

Github link:
https://github.com/Zavarochka/The-influence-of-ENG-electrode-characteristics-on-signal-quality

About this Guide

- This guide provides a step-by-step walkthrough of the installation, troubleshooting, and analysis procedures used in the dissertation. Its aim is to help future users reproduce the workflow and extend it.

- The project applied ASCENT, which combines finite element modelling (COMSOL) with axon simulations (NEURON), to explore whether physiologically plausible spontaneous neural activity can be simulated in silico.

Covered in this guide:

- ASCENT setup
- Reproduced figure from Peña et al.
- Explored research question with further analyses
- Main debugging steps applied

Contents page

Configurations Used:

Name	Version Number
macOS (8-core GPU | 8GB RAM)	15.4.1
Conda	25.5.1
Python	3.11.13
NEURON	8.2.4
COMSOL Multiphysics	6.2
Java SE Developer Kit	11.0.28

Table 1: Configurations used in the project

Tutorial notes

- ASCENT tutorial can be accessed through this link: https://wmglab-duke-ascent.readthedocs.io/en/latest/Getting_Started.html

- On **KCL Windows PCs**: KCL no longer provides a Java license. Alternatives are available via IT services.
- Refer to the following link for more details: JAVA license KCL update
- These should work in theory, but some function calls may behave differently.
- Windows setup was not tested in this project.

Getting started with shell commands

- In this tutorial you need to use shell commands.
- The shell is the program that interprets text commands for your computer.

- On **macOS** this is accessed through **Terminal**, while on **Windows** you can use **PowerShell** or **Git Bash**.

- Via these tools you can navigate directories, manage files, check software versions, and run scripts.

Shell commands used in project

Command	**Purpose**
pwd	Show current directory
ls	List files/folders
cd <folder>	Change directory
cd ..	Move up one directory
mkdir <name>	Create a new folder
rm <file>	Delete a file
python --version / python3 --version	Check Python version
conda --version	Check Conda version
java --version	Check Java version
neuron --version	Check NEURON version
git clone <repo-url>	Clone a GitHub repository

Table 2: Frequently used shell commands

Warp

I highly recommend using Warp (available for macOS and Windows) in this project:

- Gives better control over shell
- Has an inbuild AI tool that can help with debugging (came in **very** useful in the debugging stages, especially in areas with low expertise)

Link to read/download Warp: https://www.warp.dev/

Debugging steps for macOS ASCENT pt1:

1) Choose your download location and run the command:

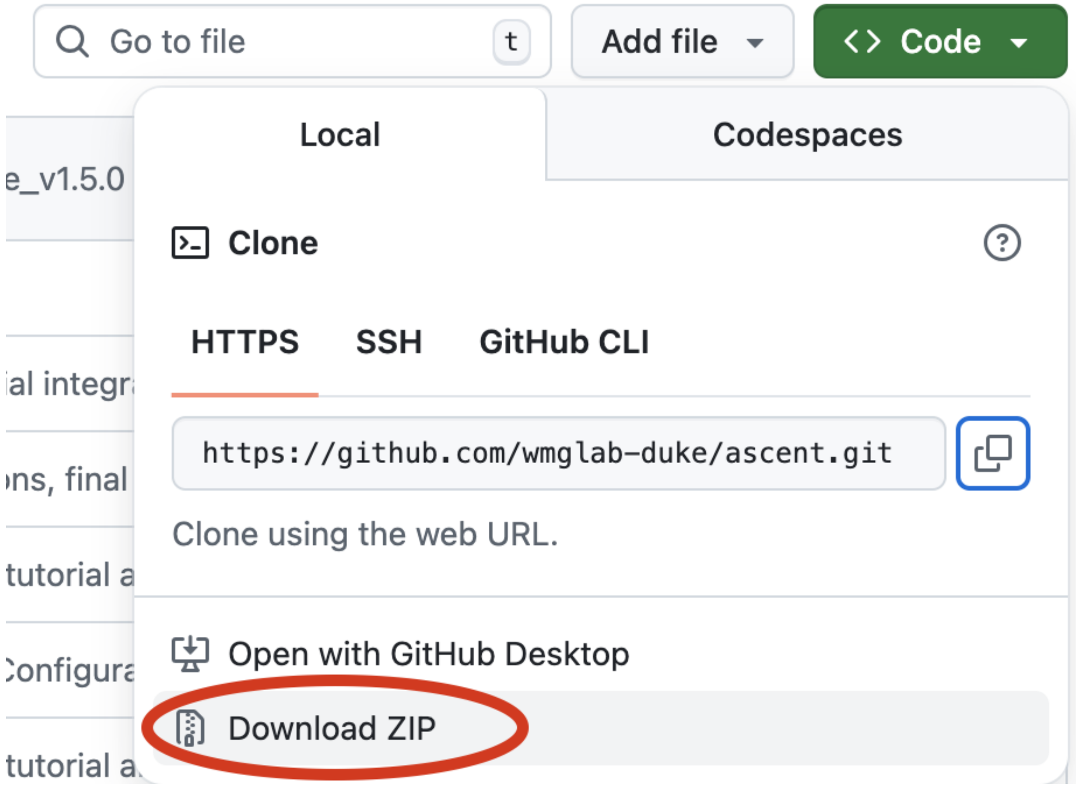
```
cd /path/to/where/you/want/ascent
```

(use **`pwd`** command to confirm where you are)

2) For quick download from the terminal use

```
git clone https://github.com/wmglab-duke/ascent.git
```

3) If this step fails (can't login/denied access), download zip file into chosen directory (see image to the right)



https://github.com/wmglab-duke/ascent.git

Clone using the web URL.

Figure 1: Github download guide

Debugging steps for macOS ASCENT pt2:

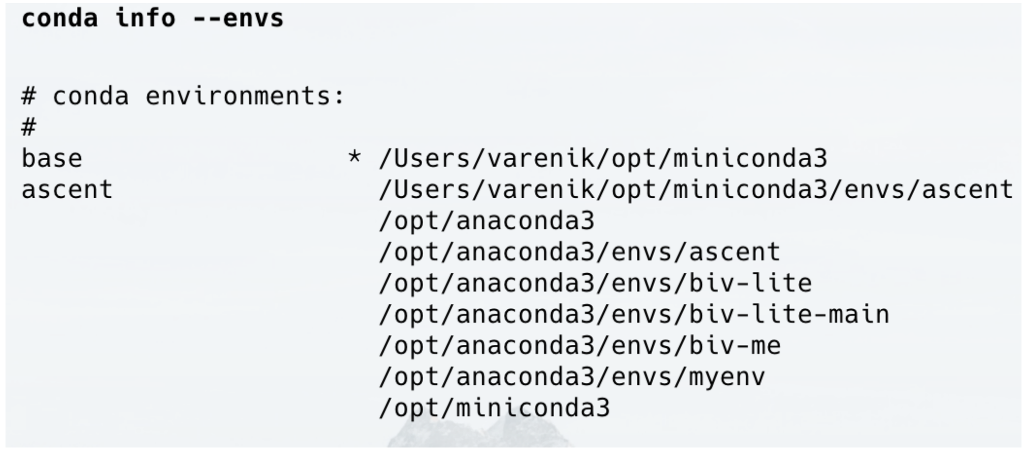
3) running `python run install` will create a new conda environment

4) check that you are located in that conda environment by running this command (marked by *****), before installing python

```
conda info –envs
```

If located in the wrong one, run:

```
conda activate <env-name>
```



```
conda info --envs

# conda environments:
#
base                  * /Users/varenik/opt/miniconda3
ascent                  /Users/varenik/opt/miniconda3/envs/ascent
                        /opt/anaconda3
                        /opt/anaconda3/envs/ascent
                        /opt/anaconda3/envs/biv-lite
                        /opt/anaconda3/envs/biv-lite-main
                        /opt/anaconda3/envs/biv-me
                        /opt/anaconda3/envs/myenv
                        /opt/miniconda3
```

Figure 2: Environment list example

Debugging steps for macOS ASCENT pt3:

- Failure during the **`python run submit.py 0`**

- When running the ASCENT tutorial on newer macOS systems, several problems can appear. Most of these come from how NEURON is installed and how the compilers are set up.
- The errors show up as missing outputs, failed MOD file compilations, or missing library files. The steps below explain each issue, why it happens, and the quick fix with the command you need to run.

Debugging steps for macOS ASCENT pt4:

- MOD files not compiling (clang: No such file or directory)
- **Error message:** …/_build_env/bin/clang: No such file or directory
- **Why**: NEURON was built against clang, but the required compiler wasn’t available.
- **Step**: Install missing compilers (system + conda).
- **Commands**:
  - `xcode-select --install`
  - `conda install -c conda-forge clang llvm-openmp -y`

Debugging steps for macOS ASCENT pt5:

- NEURON compiled with wrong/default compiler
- **Error message:** … **Error in compiling of NEURON files. Exiting...**
- **Why**: nrnivmodl defaults to system compilers, which mismatched Conda's NEURON build.
- **Step**: Force compilation with explicit clang paths.
- **Command:**
  - `cd MOD_Files && CC=clang CXX=clang++ nrnivmodl`

Debugging steps for macOS ASCENT pt6:

- libnrnmech.dylib not found during simulation
- Error message (from the log files): dyld[36737]: Library not loaded: @rpath/libnrnmech.dylib
- **Why:** ASCENT expected compiled NEURON libraries (special, libnrnmech.dylib) in each sim folder.
- **Step:** Manually copy compiled files into the simulation folder.
- **Commands:**
  - `cp MOD_Files/x86_64/special out/n_sims/0_0_0_0/`
  - `cp MOD_Files/x86_64/libnrnmech.dylib out/n_sims/0_0_0_0/`

Tutorial results

- During the tutorial, an activation threshold heatmap is generated, shown in Figure 3 (different from the tutorial website's image)
- The first threshold value is: -0.027281 mA (if using COMSOL 6.2)

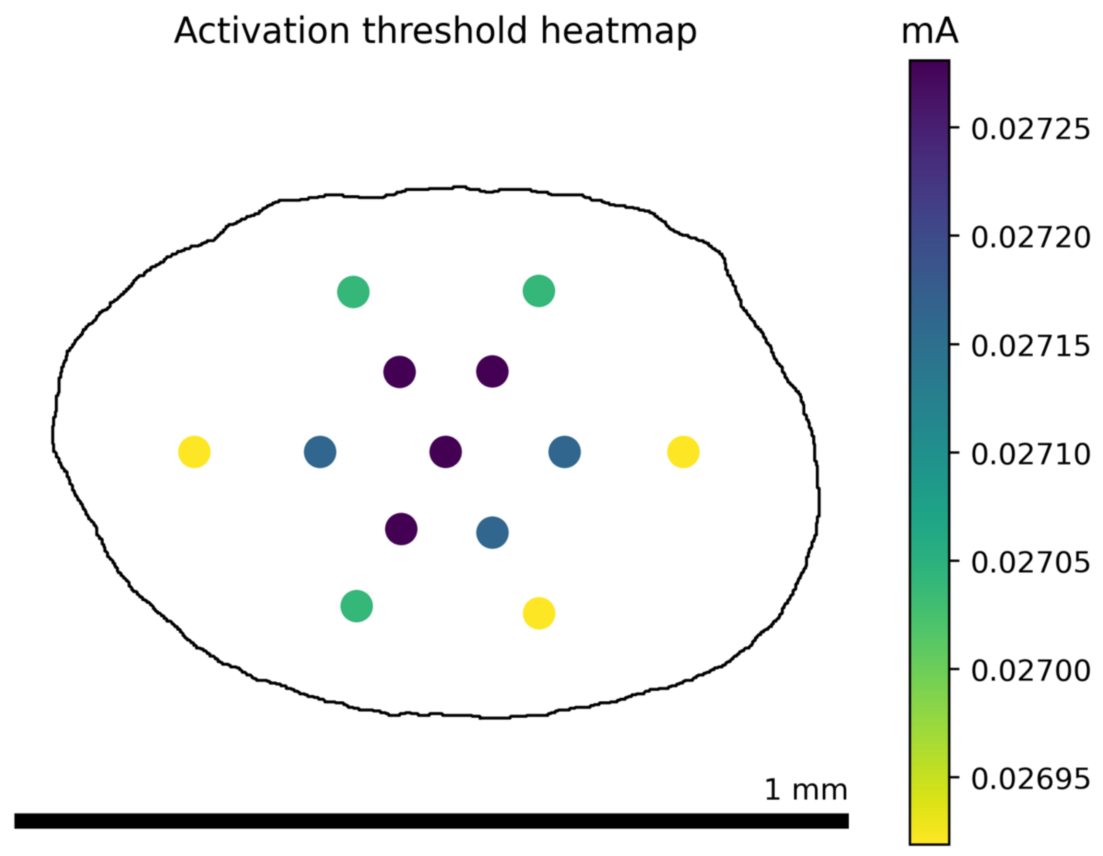


Figure 3: Activation threshold heatmap generated by following the tutorial

Replicating Peña’s Figure

What the Figure Shows:
Figure 4 from Peña et al. (2024) comparing two methods for modelling CNAPs in the rat cervical vagus nerve:

- Brute force: simulates each fibre individually (accurate but very slow).
- Interpolated templates: uses 193 pre-computed fibre templates to represent all 1,676 fibres (much faster).

Link to the paper: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011833

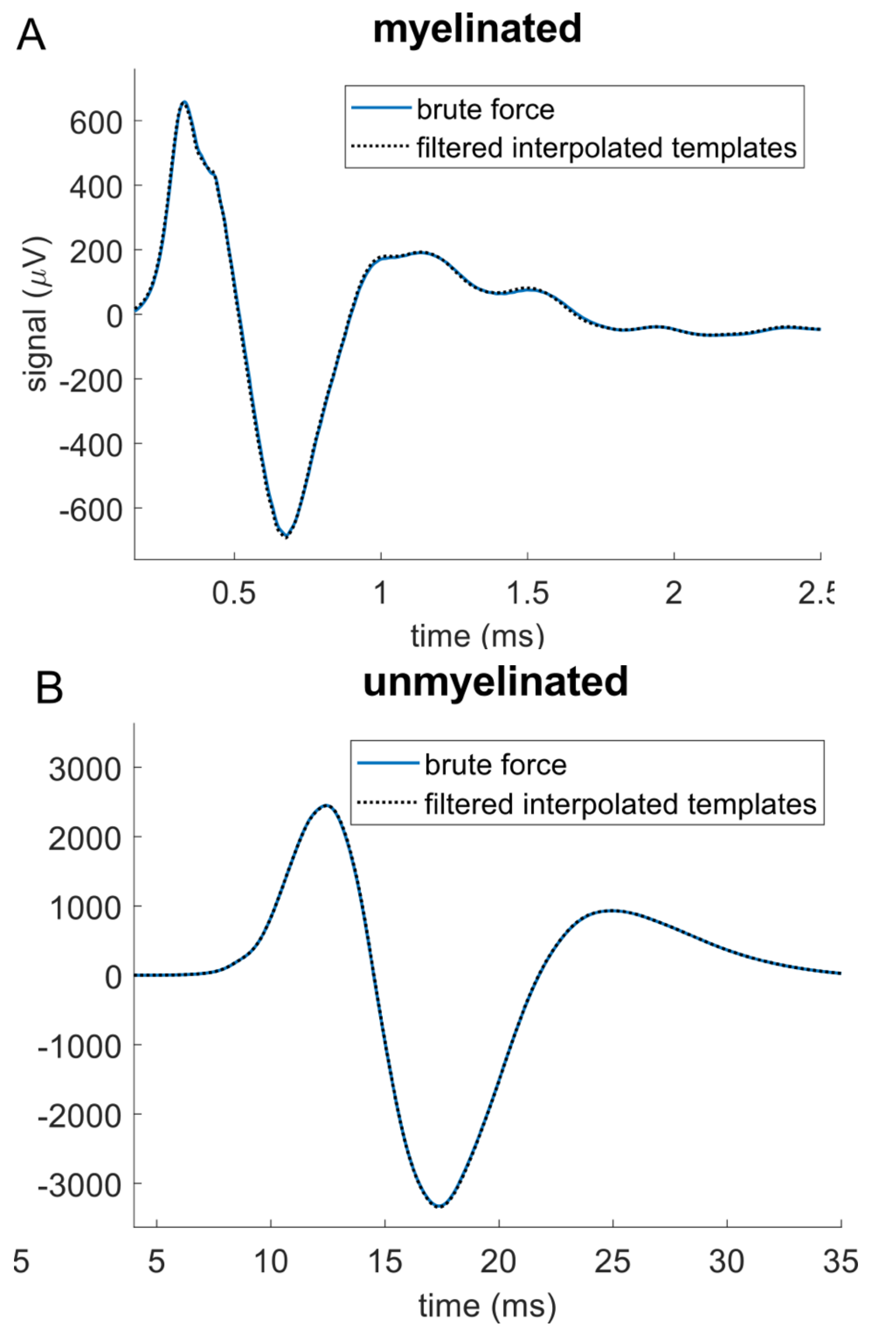
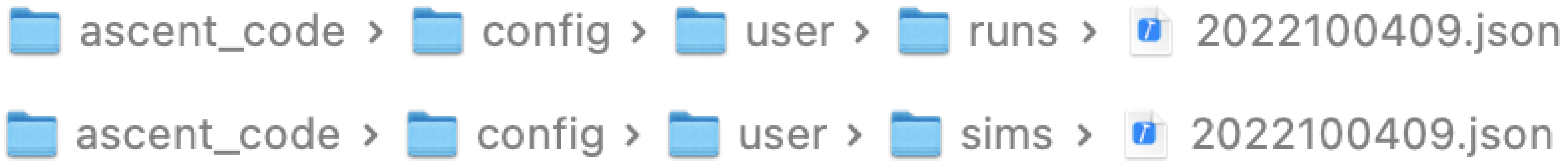


Figure 4: Original figure from Peña’s paper

Steps to Replicate Myelinated fibre graph

1) Download the dataset: https://research.repository.duke.edu/concern/datasets/qb98mg49w?locale=en
2) The dataset will include all the needed json files in the ASCENT folder, but double check all the parameters were set correctly. The next couple of slides will guide through where each value is located.

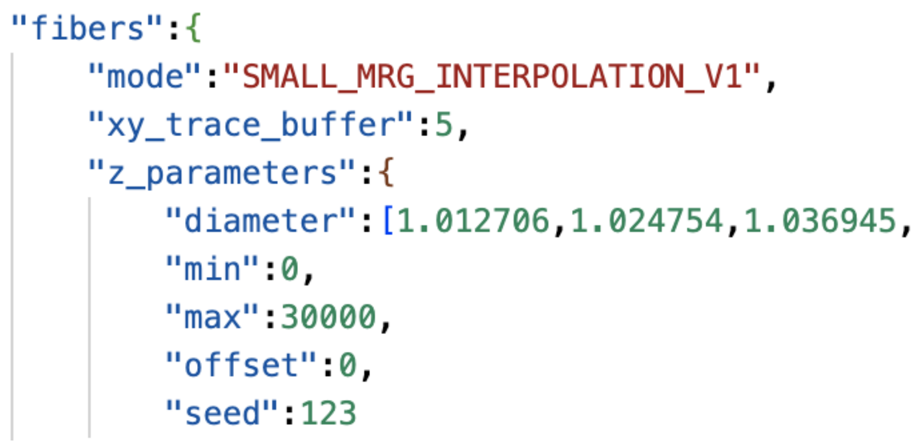
Choose the correct run/sim file to simulate



ascent_code › config › user › runs › 2022100409.json

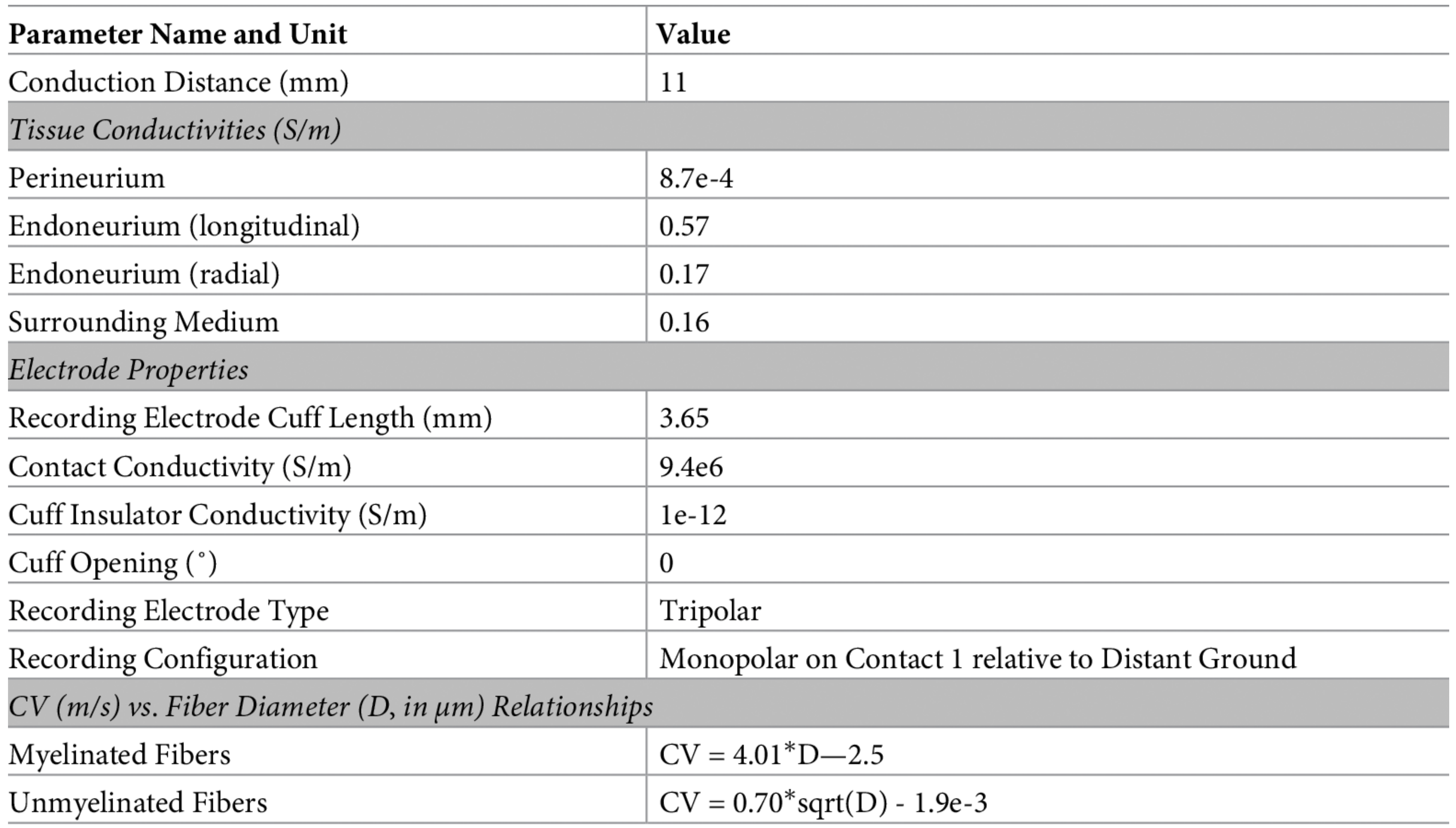
ascent_code › config › user › sims › 2022100409.json

It has the important parameter set up for generating myelinated fibres specifically (set in `mode`)
Executing the 2022100409.json run file will ultimately lead to executing the sims file with the same name



```
"fibers":{
    "mode":"SMALL_MRG_INTERPOLATION_V1",
    "xy_trace_buffer":5,
    "z_parameters":{
        "diameter":[1.012706,1.024754,1.036945,
        "min":0,
        "max":30000,
        "offset":0,
        "seed":123
```

Parameters to check

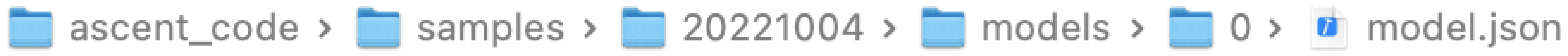


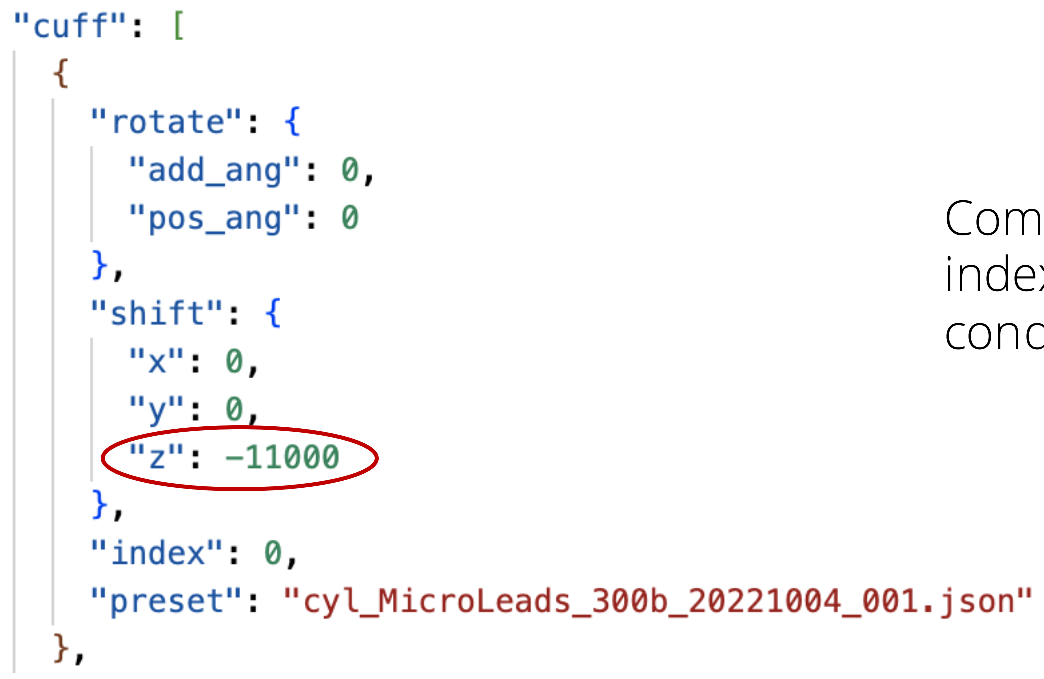
Parameter Name and Unit	Value
Conduction Distance (mm)	11
Tissue Conductivities (S/m)	
Perineurium	8.7e-4
Endoneurium (longitudinal)	0.57
Endoneurium (radial)	0.17
Surrounding Medium	0.16
Electrode Properties	
Recording Electrode Cuff Length (mm)	3.65
Contact Conductivity (S/m)	9.4e6
Cuff Insulator Conductivity (S/m)	1e-12
Cuff Opening (°)	0
Recording Electrode Type	Tripolar
Recording Configuration	Monopolar on Contact 1 relative to Distant Ground
CV (m/s) vs. Fiber Diameter (D, in μm) Relationships	
Myelinated Fibers	CV = 4.01*D—2.5
Unmyelinated Fibers	CV = 0.70*sqrt(D) - 1.9e-3

https://doi.org/10.1371/journal.pcbi.1011833.t003

Table 3: parameters set up in Pena's paper

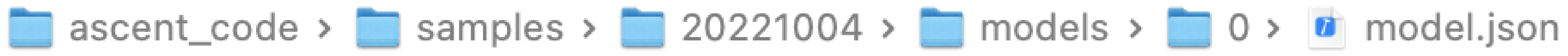
Conduction distance variable location

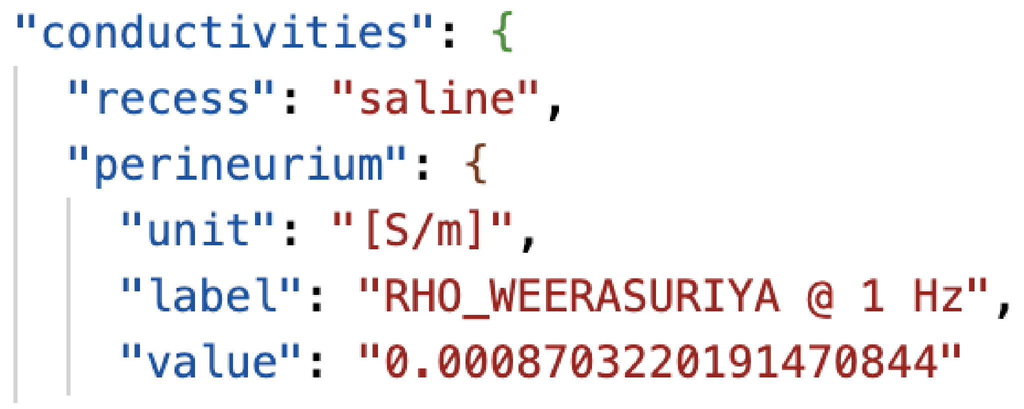

ascent_code › samples › 20221004 › models › 0 › model.json


```
"cuff": [
  {
    "rotate": {
      "add_ang": 0,
      "pos_ang": 0
    },
    "shift": {
      "x": 0,
      "y": 0,
      "z": -11000
    },
    "index": 0,
    "preset": "cyl_MicroLeads_300b_20221004_001.json"
  },
```

Compare the z location of the cuff with index 0, to the cuff with index 1 to get conduction distance in um

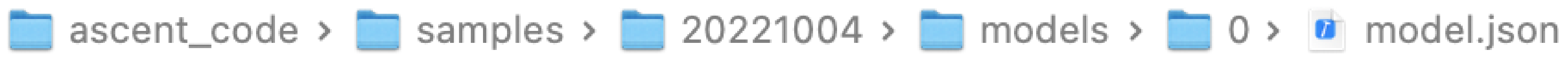
Perineurium variable location

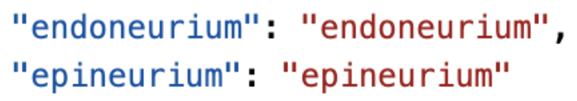

ascent_code › samples › 20221004 › models › 0 › model.json


```
"conductivities": {
  "recess": "saline",
  "perineurium": {
    "unit": "[S/m]",
    "label": "RHO_WEERASURIYA @ 1 Hz",
    "value": "0.0008703220191470844"
```

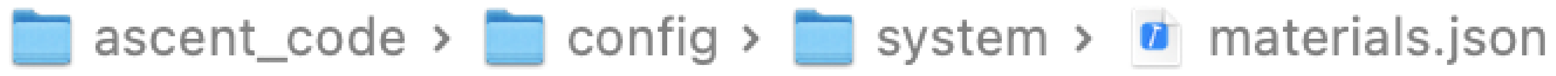
Variable stored in `value`

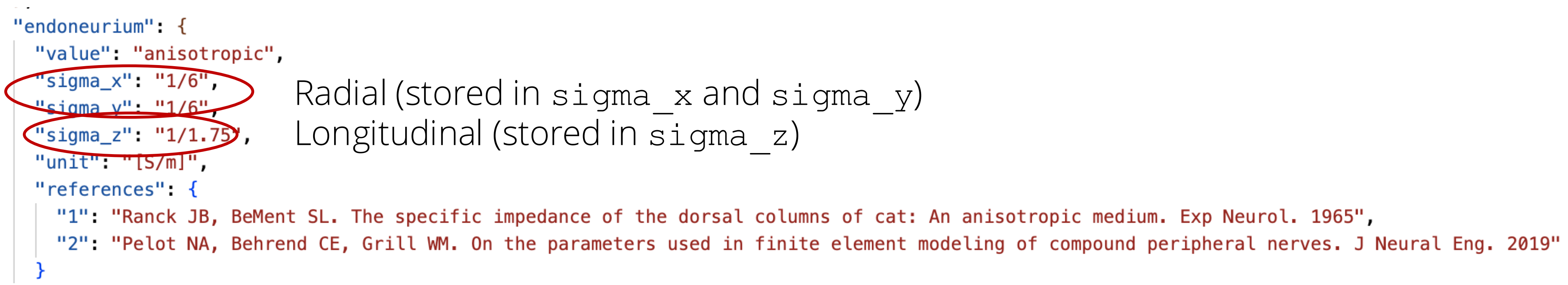
Endoneurium variable location


ascent_code › samples › 20221004 › models › 0 › model.json


```
"endoneurium": "endoneurium",
"epineurium": "epineurium"
```

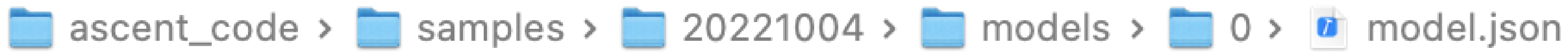
Variables set in `conductivities`, with actual values set in material.json file

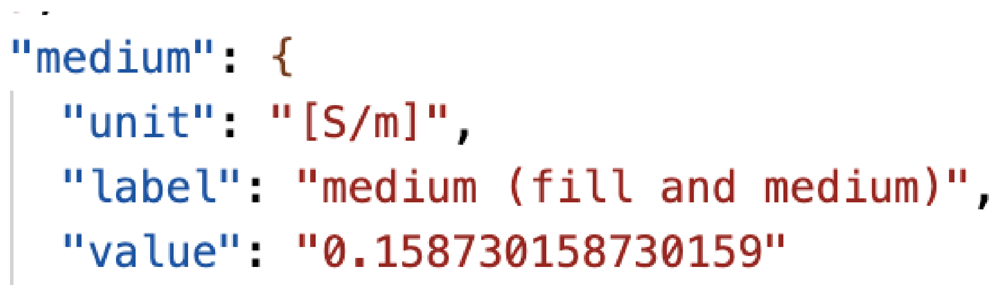

ascent_code › config › system › materials.json


```
"endoneurium": {
  "value": "anisotropic",
  "sigma_x": "1/6",
  "sigma_y": "1/6",
  "sigma_z": "1/1.75",
  "unit": "[S/m]",
  "references": {
    "1": "Ranck JB, BeMent SL. The specific impedance of the dorsal columns of cat: An anisotropic medium. Exp Neurol. 1965",
    "2": "Pelot NA, Behrend CE, Grill WM. On the parameters used in finite element modeling of compound peripheral nerves. J Neural Eng. 2019"
  }
```

Radial (stored in `sigma_x` and `sigma_y`)
Longitudinal (stored in `sigma_z`)

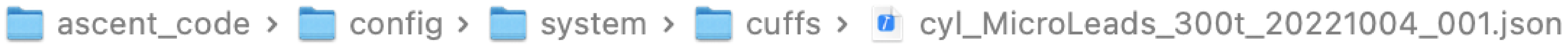
Surrounding medium variable location

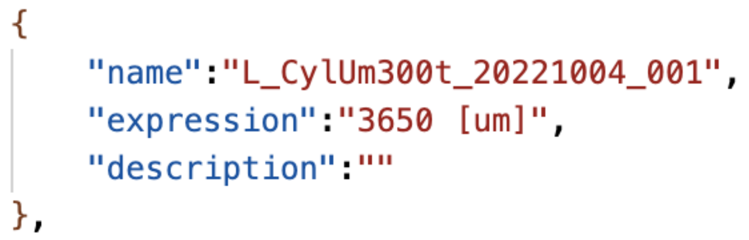

ascent_code › samples › 20221004 › models › 0 › model.json


```
"medium": {
  "unit": "[S/m]",
  "label": "medium (fill and medium)",
  "value": "0.158730158730159"
```

Stored in `value`

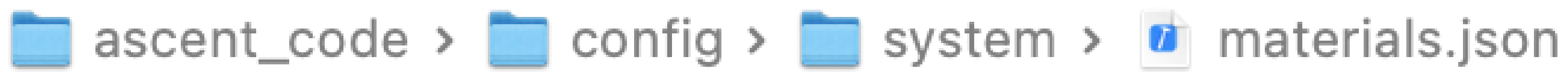
Recording electrode cuff length

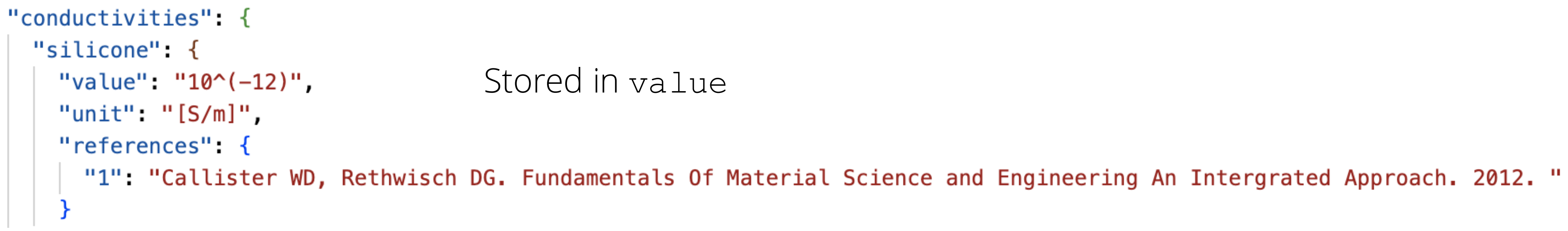

ascent_code › config › system › cuffs › cyl_MicroLeads_300t_20221004_001.json


```
{
    "name":"L_CylUm300t_20221004_001",
    "expression":"3650 [um]",
    "description":""
},
```

In params section the length is stored in `expression`

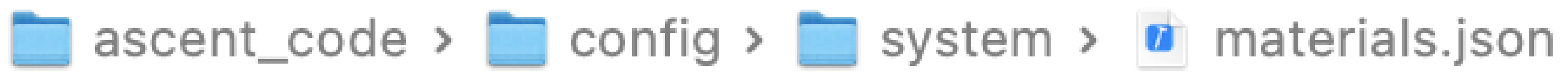
Contact insulator conductivity variable location


ascent_code › config › system › materials.json

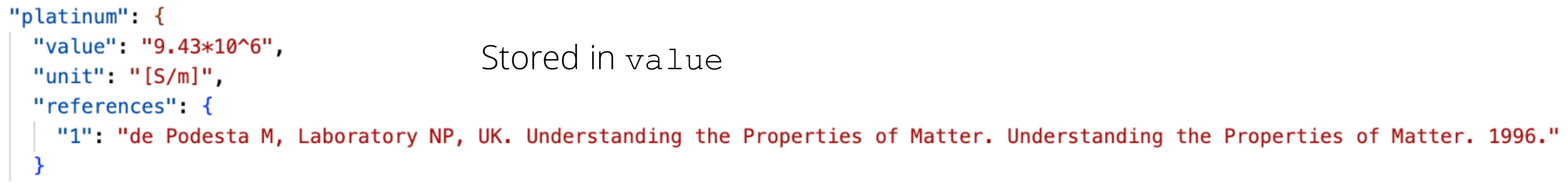

```
"conductivities": {
  "silicone": {
    "value": "10^(-12)",
    "unit": "[S/m]",
    "references": {
      "1": "Callister WD, Rethwisch DG. Fundamentals Of Material Science and Engineering An Intergrated Approach. 2012. "
    }
```

Stored in `value`

Contact conductivity variable location



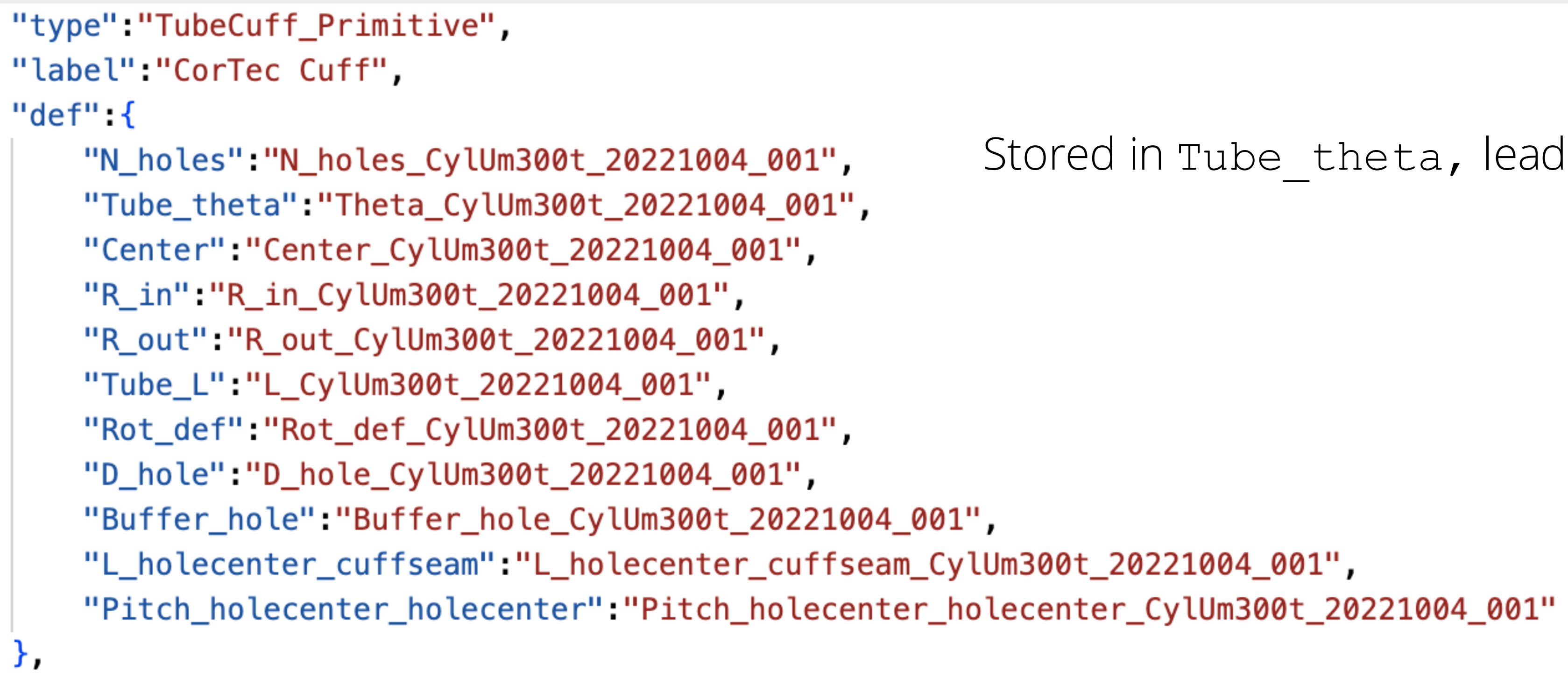
ascent_code › config › system › materials.json



```
"platinum": {
  "value": "9.43*10^6",
  "unit": "[S/m]",
  "references": {
    "1": "de Podesta M, Laboratory NP, UK. Understanding the Properties of Matter. Understanding the Properties of Matter. 1996."
  }
```

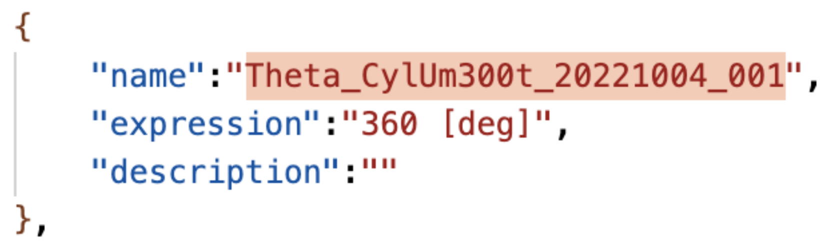
Stored in `value`

Cuff opening (in degrees) variable location



```
"type":"TubeCuff_Primitive",
"label":"CorTec Cuff",
"def":{
    "N_holes":"N_holes_CylUm300t_20221004_001",
    "Tube_theta":"Theta_CylUm300t_20221004_001",
    "Center":"Center_CylUm300t_20221004_001",
    "R_in":"R_in_CylUm300t_20221004_001",
    "R_out":"R_out_CylUm300t_20221004_001",
    "Tube_L":"L_CylUm300t_20221004_001",
    "Rot_def":"Rot_def_CylUm300t_20221004_001",
    "D_hole":"D_hole_CylUm300t_20221004_001",
    "Buffer_hole":"Buffer_hole_CylUm300t_20221004_001",
    "L_holecenter_cuffseam":"L_holecenter_cuffseam_CylUm300t_20221004_001",
    "Pitch_holecenter_holecenter":"Pitch_holecenter_holecenter_CylUm300t_20221004_001"
},
```

Stored in `Tube_theta`, leads to new variable

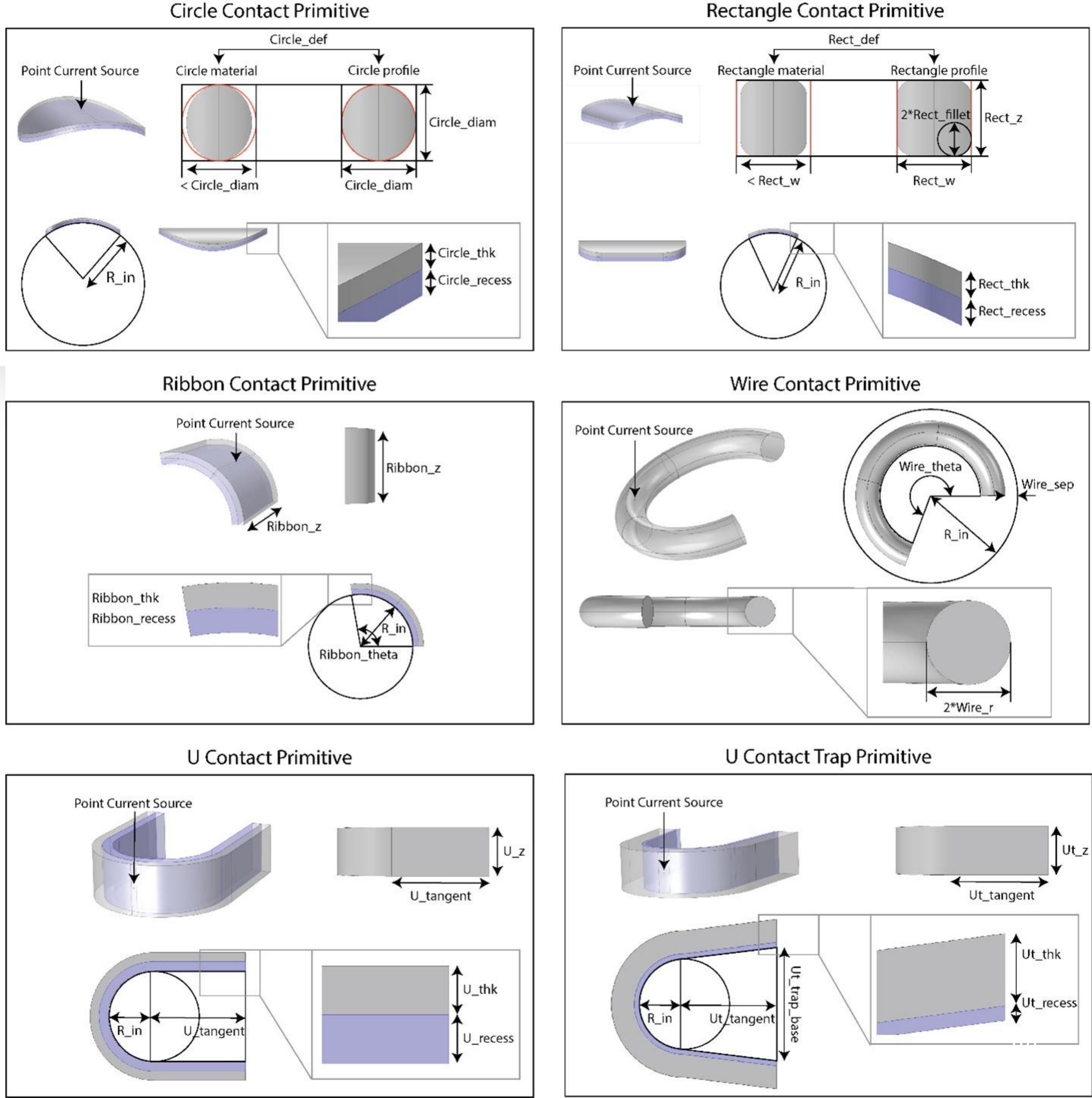


```
{
    "name":"Theta_CylUm300t_20221004_001",
    "expression":"360 [deg]",
    "description":""
},
```

Stored in `expression`, 360 means there is 0 degrees opening, if you want to have 16 degree opening, change to 344

Cuff library variables

- This is a link to the cuff library, it contains all the variable names used in the project.



The next steps for reproducing Pena's graph

Execute the following commands:

- `python run pipeline 2022100409`
- `python submit.py  20221004` (if macOS user, please follow the same debugging structure as described for the tutorial)

Move onto modelling neural recordings (more details here)

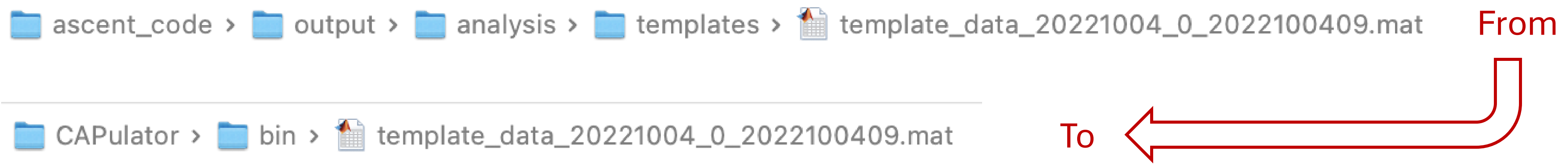
- `python examples/analysis/generate_templates.py`

This will generate the templates to be used in plotting the figure

CAPulator pt1.

- ASCENT leads the user to the following repository to complete neural recording simulation: CAPulator Github

- Follow these steps to generate the figure:
  - Download the CAPulator repository onto the laptop/PC
  - Replace (if exists) or add (if absent), the `template_data_20221004_0_2022100409.mat`, file that was previously generated during the template generation



ascent_code › output › analysis › templates › template_data_20221004_0_2022100409.mat

CAPulator › bin › template_data_20221004_0_2022100409.mat

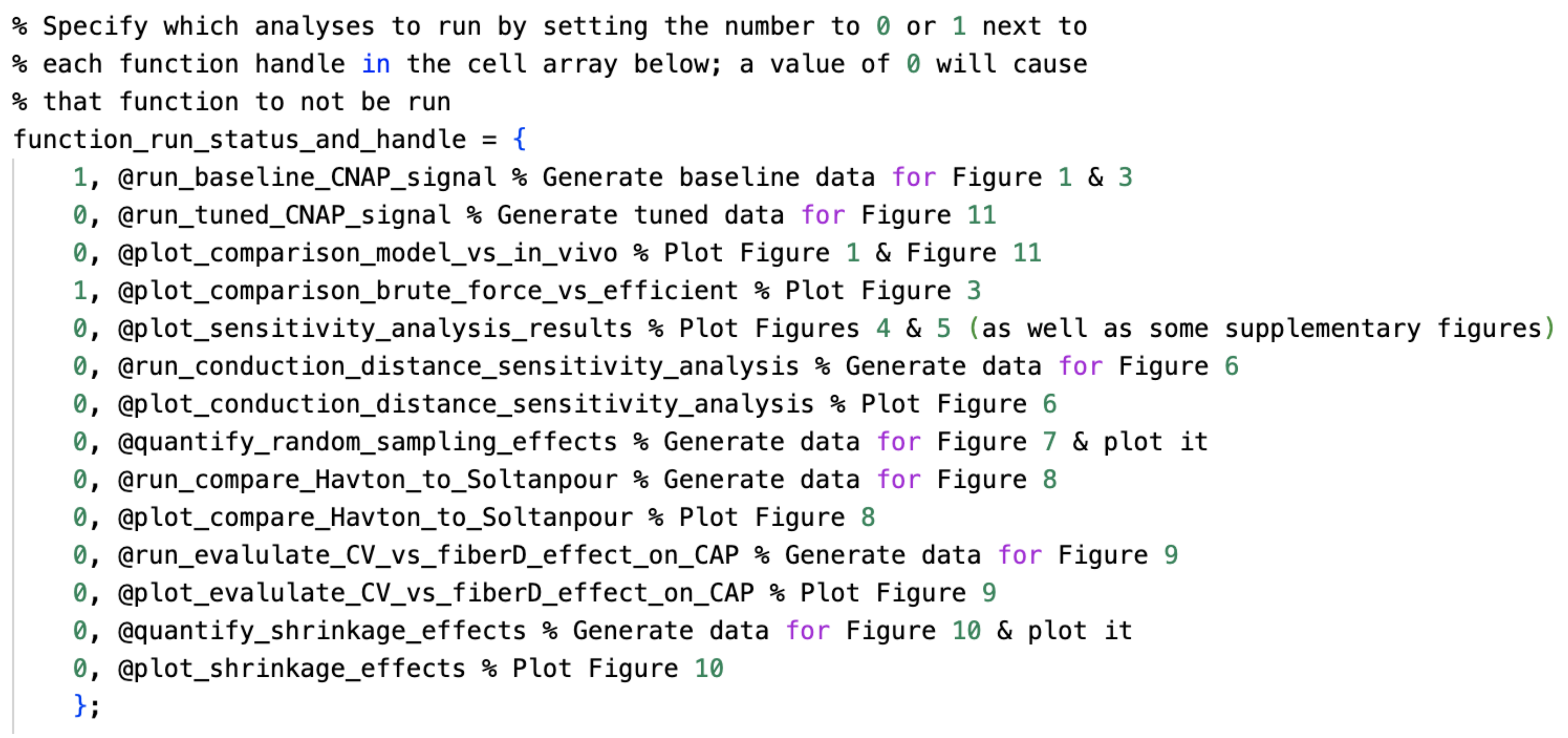
CAPulator pt2.

- Make sure that the `bin/template_data_20221004_0_2022100410.mat` file that was generated by Pena is also in the bin folder.

- `cd` into the `src` directory and run the `RUN_simulate_CAP.m` script.

CAPulator pt3.

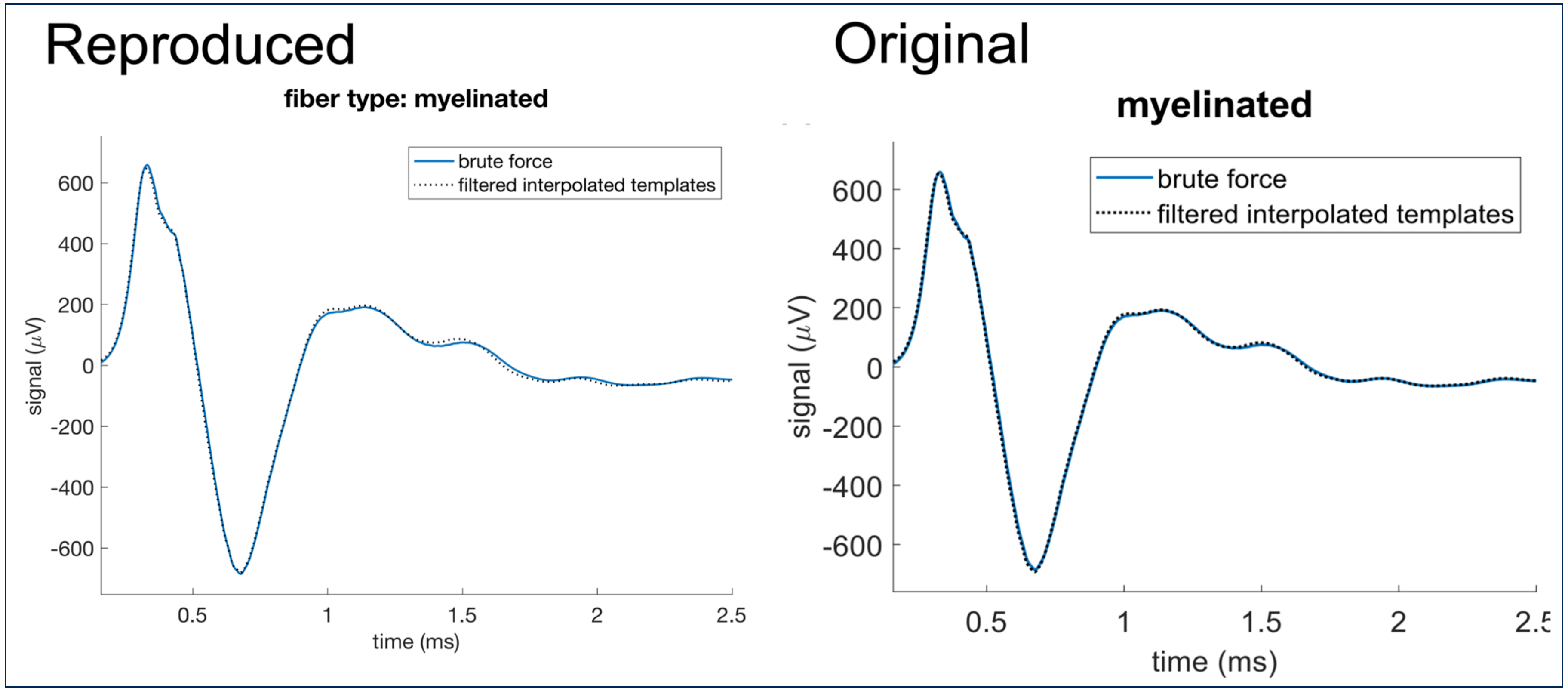
- `RUN_simulate_CAP.m` script will generate all the figures from the paper, but if you wish to generate just the one discussed in these instructions, set the 1st and the 4th functions to 1, 0 the rest in the mat file.



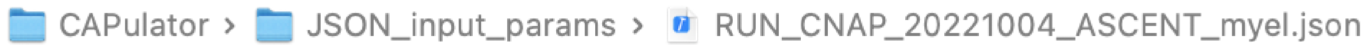
```
% Specify which analyses to run by setting the number to 0 or 1 next to
% each function handle in the cell array below; a value of 0 will cause
% that function to not be run
function_run_status_and_handle = {
    1, @run_baseline_CNAP_signal % Generate baseline data for Figure 1 & 3
    0, @run_tuned_CNAP_signal % Generate tuned data for Figure 11
    0, @plot_comparison_model_vs_in_vivo % Plot Figure 1 & Figure 11
    1, @plot_comparison_brute_force_vs_efficient % Plot Figure 3
    0, @plot_sensitivity_analysis_results % Plot Figures 4 & 5 (as well as some supplementary figures)
    0, @run_conduction_distance_sensitivity_analysis % Generate data for Figure 6
    0, @plot_conduction_distance_sensitivity_analysis % Plot Figure 6
    0, @quantify_random_sampling_effects % Generate data for Figure 7 & plot it
    0, @run_compare_Havton_to_Soltanpour % Generate data for Figure 8
    0, @plot_compare_Havton_to_Soltanpour % Plot Figure 8
    0, @run_evalulate_CV_vs_fiberD_effect_on_CAP % Generate data for Figure 9
    0, @plot_evalulate_CV_vs_fiberD_effect_on_CAP % Plot Figure 9
    0, @quantify_shrinkage_effects % Generate data for Figure 10 & plot it
    0, @plot_shrinkage_effects % Plot Figure 10
    };
```

CAPulator pt4.

- Running the script will produce a figure, it looks slightly different to the one in Pena's paper, but this may be attributed to newer COMSOL and Neuron versions used.



Important note about Recording Sensitivity Function


CAPulator › JSON_input_params › RUN_CNAP_20221004_ASCENT_myel.json

- The file named RUN_CNAP_20221004_ASCENT_myel contains the fibre diameter values used in templates, the fibre diameter values to be simulated in the run and most importantly the `extracellular_recording_model_filename:"../bin/default_volume_conductor_potentials.mat”`

- This `../bin/default_volume_conductor_potentials.mat` contains the recordings from the volume conductor model that will be used to calculate therecording sensitivity function for the final graph.

- In order to explore the way the geometry of the recording electrode affects the simulated neural recording, those volume conductor potentials need to be extracted each time a change is made. Currently, there is no clear way to extract that data, so please come in contact with the Pena and ASCENT team to discuss this question.

Contact information

KCL

- Varvara Tebieva - var.tebieva@gmail.com
- Anne Vanhoestenberghe - a.vanhoest@kcl.ac.uk

Duke University (ASCENT developers)

- Warren M Grill - warren.grill@duke.edu
- Nikki Pelot - nikki.pelot@duke.edu