A Short Introduction to PyPN, the Peripheral Nerve Simulator

Carl H. Lubba*[1], Yann Le Guenn[1], Sarah Jarvis[1], and Simon Schultz[1]

[1] *Department of Bioengineering, Imperial College London*

Submitted: March 16, 2016

1 Get PyPN

This guide is for Linux. As a necessary prerequisite of PyPN, you need to have NEURON [1] with Python support[1] and LFPy [2]. Then, to install PyPN, download it from Git, navigate to its location and enter `python setup.py` in the console. This acts only local right now and the PyPN package does not get copied to the Python lib directory. You can only use it from the directory `setup.py` is in! To import and use it in a project type `import PyPN`.

The following description is best read with the `example1.py`-script.

2 Get started with PyPN

PyPN's main object is a `Bundle`. A `Bundle` consists of single `Axons`, mechanisms to excite `Axons` as well as mechanisms to record from `Axons`. The defining characteristics of a `Bundle` need to be provided in form of a dictionary and handed to the `Bundle`-constructor `bundle = PyPN.Bundle(**Parameters)`. See 1.

Code 1: Bundle dictionary

```
bundleParameters = {
'radiusBundle': 150,
'lengthOfBundle': 2000,
'numberOfAxons': 30,
'p_A': .1,
'p_C': .9,
'myelinated_A': myelinatedParametersA,
'unmyelinated': unmyelinatedParameters,
'bundleGuide' : bundleGuide
'recordingElecPos': [2000, 2050],
'numberElecs': 1}
```

The bundle can be stimulated through `ExcitationMechanisms`. Right now, PyPN provides two of those mechanisms, `Stimulus` for intra- and extracellular electrical stimulation with various waveforms and `UpstreamSpiking` for continuous spiking. Both need to be defined with a corresponding dictionary and then added to the `Bundle` via `bundle.add_excitation_mechanism()`.

Code 2: Stimulus dictionary

```
upstreamSpikingDict = {
'lambd' : 500,
'tStop' : tStop,
'nAxons' : numberOfAxons}

bundle.add_excitation_mechanism(PyPN.
UpstreamSpiking(**upstreamSpikingDict))
```

To simulate the bundle the user needs to call `bundle.simulate()`. As much data is generated it is automatically saved to disk. Once the simulation is done, the plotting functions can be called to see the result.

Code 3: Plotting the compound action potential

```
PyPN.plot.CAP1D(bundle)
```

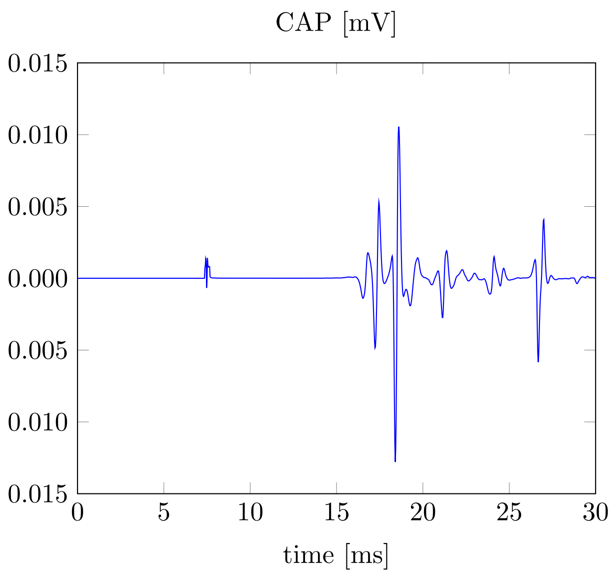

Figure 1: Recorded compound action potential of bundle composed of 30 axons, 10% of which myelinated, stimulated at once with a monophasic pulse.

References

[1] M. L. Hines, A. P. Davison, and E. Muller, Neuron and python., *Frontiers in neuroinformatics* **3**, 1 (2009).

[2] H. Lindén, E. Hagen, S. Leski, E. S. Norheim, K. H. Pettersen, and G. T. Einevoll, Lfpy: a tool for biophysical simulation of extracellular potentials generated by detailed model neurons, *Frontiers in Neuroinformatics* **7**, 1 (2014).

*Corresponding author: c.lubba15@imperial.ac.uk

[1] See http://www.tc.umn.edu/ haszx010/files/vpl_dbs_docs/Installation.html http://www.davison.webfactional.com/notes/installation-neuron-python/