Neo: representing and manipulating electrophysiology data in Python

Andrew Davison[1], Thierry Brizzi[1], Luc Estabanez[1], Florent Jaillet[2], Yann Mahnoun[3], Philipp Rautenberg[4], Andrey Sobolev[4], Thomas Wachtler[4], Pierre Yger[1] and Samuel Garcia[5]

[1]Unité de Neuroscience, Information et Complexité, CNRS UPR 3293, Gif-sur-Yvette, France
[2]Institut de Neurosciences Cognitives de la Méditerranée, CNRS UMR 6193 - Université de la Méditerranée, Marseille, France
[3]Laboratoire de Neurosciences Intégratives et Adaptatives, CNRS UMR 6149 - Université de Provence, Marseille, France
[4]G-Node, Ludwig-Maximilians-Universität, Munich, Germany
[5]Laboratoire Neurosciences Sensorielles, Comportement, et Cognition, CNRS UMR 5020 - Université Claude Bernard, Lyon, France

In neuroscience, electrophysiological signals, whether from electroencephalography, intracellular and extracellular electrophysiology, or simulation, are acquired, analysed and visualized using a wide variety of software, much of it proprietary, developed by recording hardware manufacturers, very often written in Matlab, and increasingly written in Python.

To improve interoperability of different tools and to share data between different projects, a common representation of the core data is needed. A number of efforts have been made in this direction, including the Neuroshare project (http://neuroshare.sourceforge.net) for proprietary formats, and the FIND (http://find.bccn.uni-freiburg.de/) and sigTOOL (http://sigtool.sourceforge.net/) toolboxes for Matlab. For Python, however, such a common representation has so far been missing.

We have developed a Python package, Neo, for representing electrophysiology data in memory, and for reading/writing the data to/from a variety of commonly-used file formats. Neo is deliberately limited to represention of data, with no functions for data analysis or visualization, in order to be as lightweight a dependency as possible. A common package or packages for neurophysiology data analysis and visualization, such as the NeuroTools package (http://neuralensemble.org/NeuroTools), can then build on top of Neo. Currently Neo is used as the core representation of data in OpenElectrophy (http://neuralensemble.org/trac/OpenElectrophy) and it is planned to also use it in NeuroTools and in the G-Node portal (http://www.g-node.org/) in the near future.

Neo implements a hierarchical data model well adapted to intracellular and extracellular electrophysiology and EEG data with support for multi-electrodes (for example tetrodes), including classes such as SpikeTrain and AnalogSignal. The Neo package also provides a set of input/output (IO) modules for various neurophysiology file formats (Plexon, Spike2, NeuroExplorer, Axon, AlphaOmega, Micromed, EEGLab, WinWcp, Elan, Elphy, PyNN, Neuroshare (Win32 only), TDT, ASCII and raw binary data). Neo builds on the quantities package (http://pypi.python.org/pypi/quantities), which in turn builds on NumPy.

Neo is distributed under a BSD licence, and is available through the Python Package Index (http://pypi.python.org/pypi/neo/), with documentation at http://packages.python.org/neo/ and a development website at http://neuralensemble.org/trac/neo. New contributions are welcome.