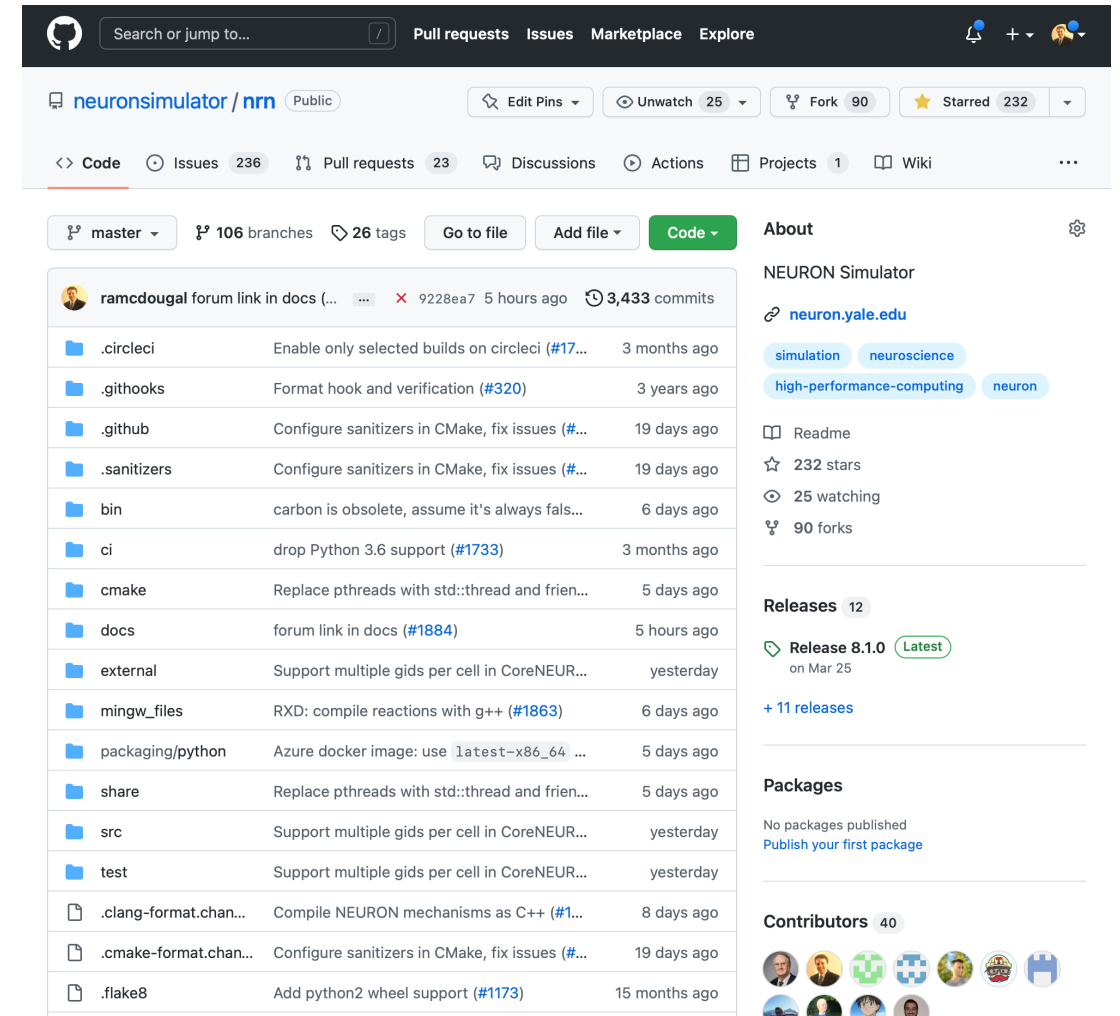


Other resources

git + GitHub

- Setup
 - git init
- Stage new/modified files for next commit:
 - git add FILENAME
- See what has changed
 - git diff
- Commit a version (so you can return to it later); you will be prompted to enter a commit message:
 - git commit
- Push to an external repository:
 - git push
- Checkout a version of a file from 2 commits ago:
 - git checkout HEAD~2 FILENAME



DO YOU NEED...

- ✓ Free and open access to Supercomputers, High-Performance, High-Throughput & cloud computing for research?
- ✓ Access to popular computational neuroscience tools and data processing software?
- ✓ Web portal and programmatic access to supercomputing resources?
- ✓ A place to disseminate your research software and tools to the neuroscience community?



NSG is an easy to use interface for neuroscience researchers to access HPC, HTC and cloud resources for simulation and data processing.

Contact us at nsghelp@sdsc.edu

NSG facilitates access and use of High Performance Computing, High Throughput Computing, and academic cloud computing resources by neuroscientists for applications that include computational modeling, data processing, and development and dissemination of neuroscience software. Software currently available via NSG includes NEURON, NetPyNE, BMTK, NEST, BRIAN 2, BluePyOpt, MATLAB, EEGLAB, FREESURFER, PyTorch, TensorFlow, DEAP; this list is updated based on user needs.

LFPy

Module LFPy

Cell classes

Point processes

Networks

Forward models

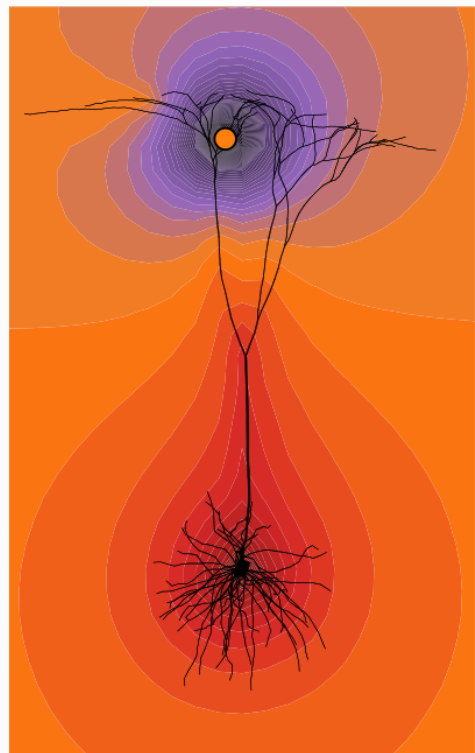
Current Dipole Moment forward models

Current Source Density (CSD)

Misc.



Love Documentation? Write the Docs is for people like you! Join our virtual conferences or Slack.



LFPy

Welcome to LFPy's documentation!

(Looking for the old LFPy v1.* documentation? Follow [link](#))

Contents



BluePyOpt: Leveraging Open Source Software and Cloud Infrastructure to Optimise Model Parameters in Neuroscience

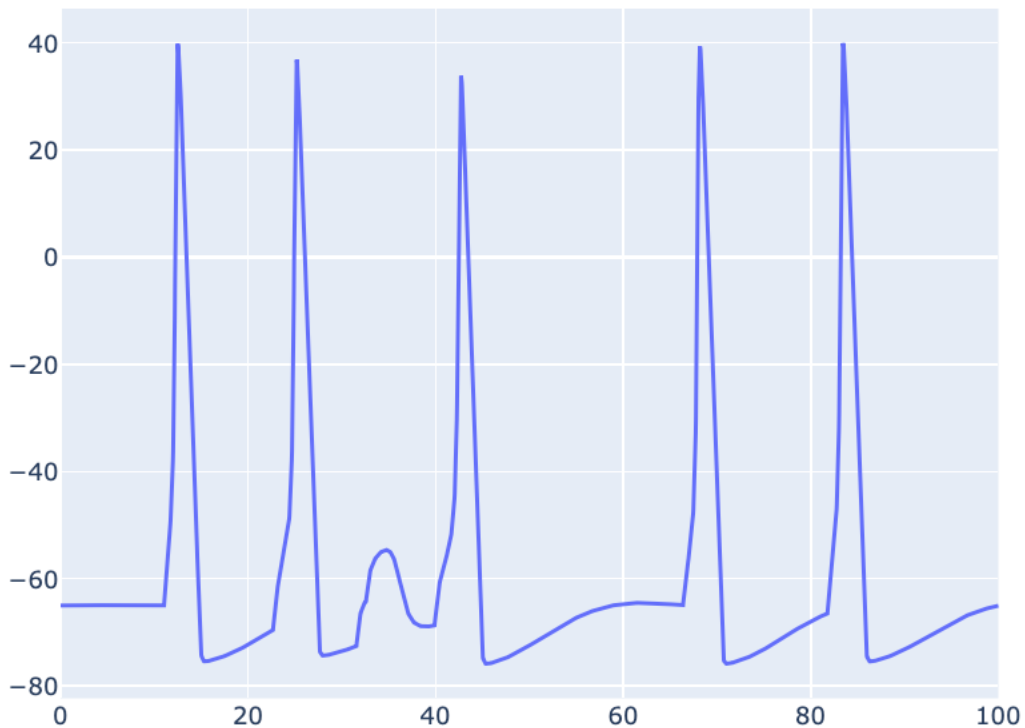
Werner Van Geit^{1}, Michael Gevaert¹, Giuseppe Chindemi¹, Christian Rössert¹, Jean-Denis Courcol¹, Eilif B. Muller¹, Felix Schürmann¹, Idan Segev^{2,3} and Henry Markram^{1,4*}*

¹ Blue Brain Project, École Polytechnique Fédérale de Lausanne, Geneva, Switzerland, ² Department of Neurobiology, Alexander Silberman Institute of Life Sciences, The Hebrew University of Jerusalem, Jerusalem, Israel, ³ The Edmond and Lily Safra Centre for Brain Sciences, The Hebrew University of Jerusalem, Jerusalem, Israel, ⁴ Laboratory of Neural Microcircuitry, Brain Mind Institute, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

OPEN ACCESS

At many scales in neuroscience, appropriate mathematical models take the form of complex dynamical systems. Parameterizing such models to conform to the multitude of available experimental constraints is a global non-linear optimisation problem with a

`pip install bluepyopt`



features:

```
[{'AP_amplitude': array([103.29412261, 105.09884958, 105.28924314, 103.58659989,
                        105.59554847]),
  'AP_begin_time': array([11.1, 22.7, 31.6, 66.4, 81.8]),
  'ISIs': array([12.7, 17.5, 25.5, 15.2]),
  'spike_half_width': array([1.52675742, 1.41003978, 1.40225347, 1.48675265,
                            1.64899456])}]
```

eFEL:

Electrophys Feature Extraction Library

```
import efel
```

```
trace = {"T": t, "V": v,
         "stim_start": [5 * ms],
         "stim_end": [100 * ms]}
```

```
traces = [trace]
```

```
features = efel.getFeatureValues(
    traces,
    ["AP_amplitude", "ISIs",
     "spike_half_width", "AP_begin_time"])
```




Elephant

Navigation

[Installation](#)

[Tutorials](#)

[Function Reference by Module](#)

[Contributing to Elephant](#)

[Release Notes](#)

[Acknowledgments](#)

[Authors and contributors](#)

[Citing Elephant](#)

Quick search

LaunchDarkly ➔

LaunchDarkly: Change how you deliver software with the #1 platform for managing feature flags

Elephant - Electrophysiology Analysis Toolkit

Elephant (Electrophysiology Analysis Toolkit) is an emerging open-source, community centered library for the analysis of electrophysiological data in the Python programming language.

The focus of Elephant is on generic analysis functions for spike train data and time series recordings from electrodes, such as the local field potentials (LFP) or intracellular voltages. In addition to providing a common platform for analysis codes from different laboratories, the Elephant project aims to provide a consistent and homogeneous analysis framework that is built on a modular foundation. Elephant is the direct successor to [Neurotools](#) and maintains ties to complementary projects such as [ephyviewer](#) and [neurotic](#) for raw data visualization.

The input-output data format is either [Neo](#), [Quantity](#) or [Numpy](#) array. Quantity is a Numpy-wrapper package for handling physical quantities like seconds, milliseconds, Hz, volts, etc. Quantity is used in both Neo and Elephant.

Visualization of Elephant analysis objects

[Viziphant](#) package is developed by Elephant team and provides a high-level API to easily generate plots and interactive visualizations of neuroscientific data and analysis results. The API uses and extends the same structure as in Elephant to ensure intuitive usage for scientists that are used to Elephant.

Table of Contents

- [Installation](#)
- [Tutorials](#)
- [Function Reference by Module](#)
- [Contributing to Elephant](#)
- [Release Notes](#)
- [Acknowledgments](#)
- [Authors and contributors](#)
- [Citing Elephant](#)

elephant.readthedocs.io



NeuroML Documentation

🔍 Search this book...

USER DOCUMENTATION

Mission and Aims

How to use this documentation

Getting started with NeuroML ▼

Finding and sharing NeuroML models

Creating NeuroML models

Validating NeuroML Models

Visualising NeuroML Models ▼

Simulating NeuroML Models

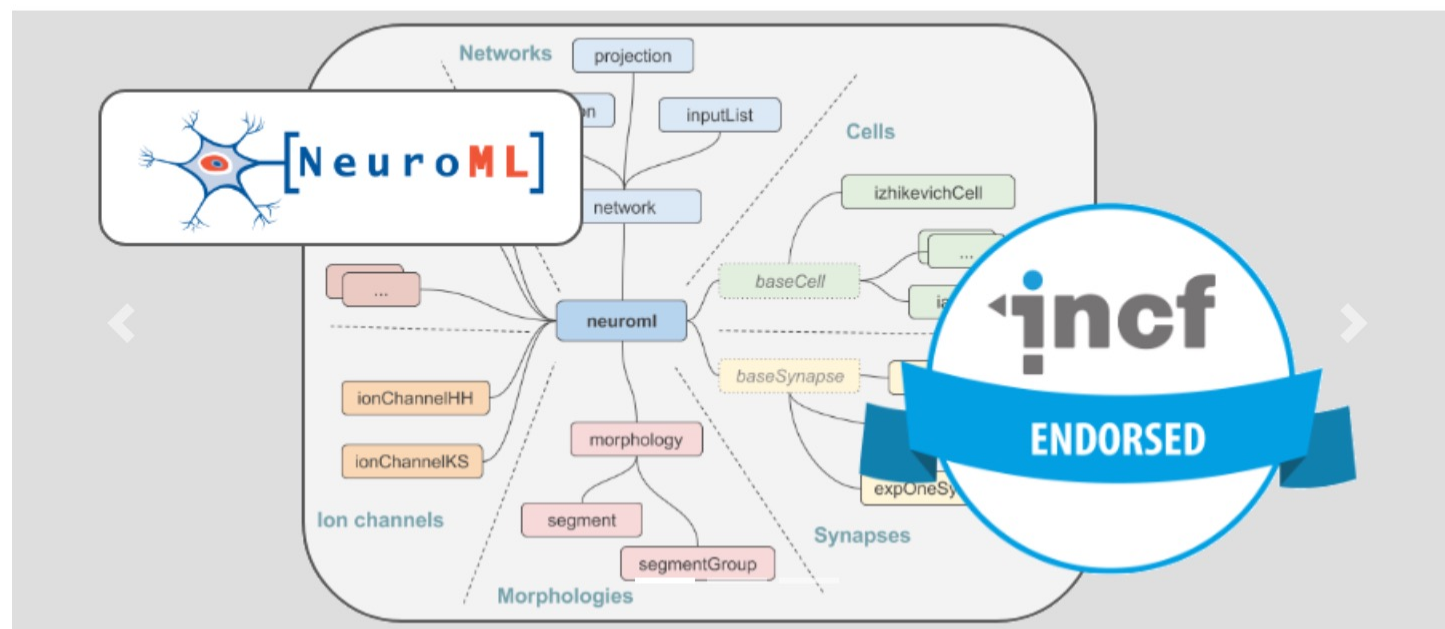
Optimising/fitting NeuroML Models

Schema/Specification ▼



NeuroML

A model description language for computational neuroscience.



NeuroML is an [international, collaborative initiative](#) to develop a language for describing detailed models of neural systems, which will serve as a standard data format for defining and exchanging descriptions of neuronal cell and network models. NeuroML is:

To convert a NeuroML model to NEURON, use e.g.
`pynml model.nml -neuron`

docs.neuroml.org

NeuroElectro: organizing information on cellular neurophysiology.

The goal of the NeuroElectro Project is to extract information about the electrophysiological properties (e.g. [resting membrane potentials](#) and [membrane time constants](#)) of diverse neuron types from the existing literature and place it into a centralized database.

Published literature

Novel subcellular distribution pattern of A-type K⁺ channels on neuronal surface.
Unique clustering of A-type potassium channels on different cell types of the main olfactory bulb.
Kollo M, Holderith N, Antal M, Nusser Z.
Theoretical and functional studies predicted a highly non-uniform distribution of voltage-gated ion channels on the neuronal surface. This was confirmed by recent immunolocalization experiments for Na⁺, Ca²⁺, and hyperpolarization activated mixed cation and K⁺ channels. These experiments also indicated that some K⁺ channels were clustered in synaptic or non-synaptic membrane specializations. Here we analysed the subcellular distribution of Kv4.2 and Kv4.3 subunits in the rat main olfactory bulb at high resolution to address whether clustering characterizes their distribution, and whether they are concentrated in synaptic or non-synaptic junctions. The cell surface distribution of the Kv4.2 and Kv4.3 subunits is highly non-uniform. Strong Kv4.2 subunit-immunopositive clusters were detected in intercellular junctions made by mitral, external puffed and granule cells (GFCs). We also found Kv4.3 subunit-immunopositive clusters in periglomerular (PGC), deep short-axon and GFCs. In the juxtglomerular region some calretinin-immunopositive glial cells enwrap neighboring PGC somata in a cap-like manner. Kv4.3 subunit clusters are present in the cap membrane that directly contacts the PGC, but not the one that faces the neuropil. In membrane specializations established by members of the same cell type, K⁺ channels are enriched in both membranes, whereas specializations between different cell types contain a high density of channels asymmetrically. None of the K⁺ channel-rich junctions showed any of the ultrastructural features of known chemical synapses. Our study provides evidence for highly non-uniform subcellular distributions of A-type K⁺ channels and predicts their involvements in novel

Physiology database

Olfactory Bulb Mitral Cell

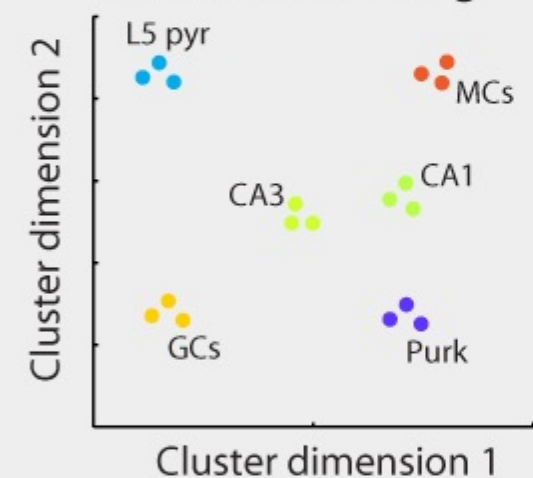
Input resistance	200 MΩ
V _{rest}	-65 mV
Spike width	1 ms
...	

CA1 Pyramidal Cell

Input resistance	400 MΩ
V _{rest}	-70 mV
Spike width	.5 ms
...	

Extracted from Literature

Neuron clustering



Our goal is to facilitate the discovery of [neuron-to-neuron relationships](#) and better understand the role of functional diversity across neuron types.

Pandas:

File storage, data frames, and databases

- Saving as CSV with pandas:

```
import pandas as pd
pd.DataFrame({
    "t": t, "v": v}
).to_csv("data.csv", index=False)
```

- Loading from CSV with pandas:

```
import pandas as pd
data = pd.read_csv("data.csv")
t = h.Vector(data["t"])
v = h.Vector(data["v"])
```

- Saving data to a database:

```
import pandas as pd
import sqlite3
import time

# do all the following inside a loop

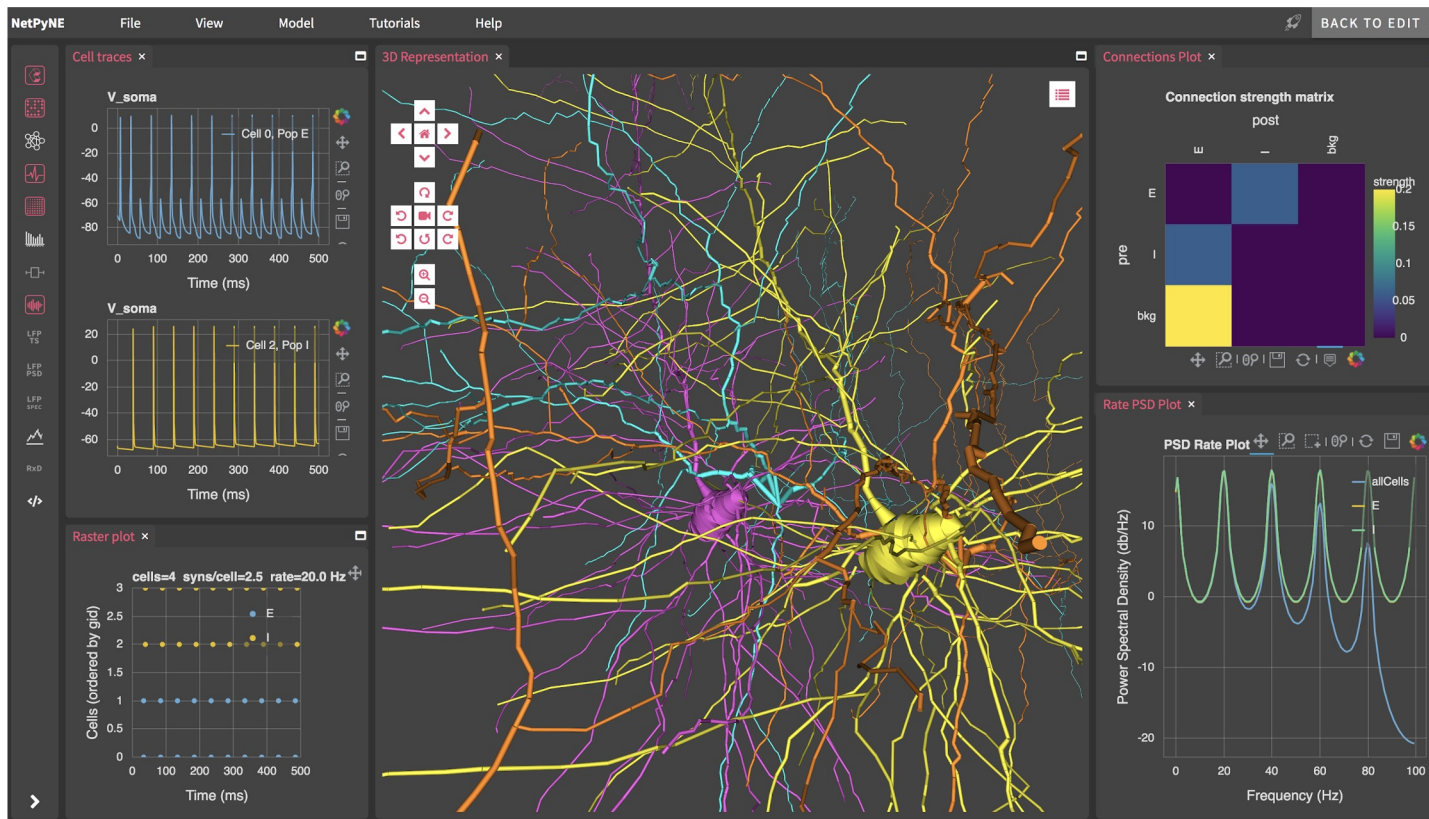
start = time.perf_counter()
# do the simulation here

data = pd.DataFrame(
    {
        "dx": [dx],
        "L": L,
        "diam": diam,
        "alpha": alpha,
        "temperature": h.celsius,
        "spike_half_width": spike_half_width,
        "calculated_value": calculated_value,
        "runtime": time.perf_counter() - start,
    }
)

with sqlite3.connect(DB_FILENAME) as conn:
    data.to_sql("data", conn,
                if_exists="append", index=False)
```



A python package to facilitate the development, parallel simulation, optimization and analysis of biological neuronal networks using the NEURON simulator.



Funded by:



National Institutes
of Health