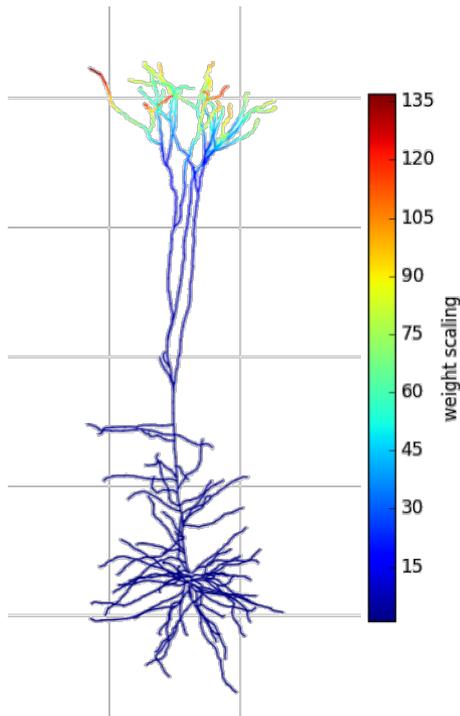


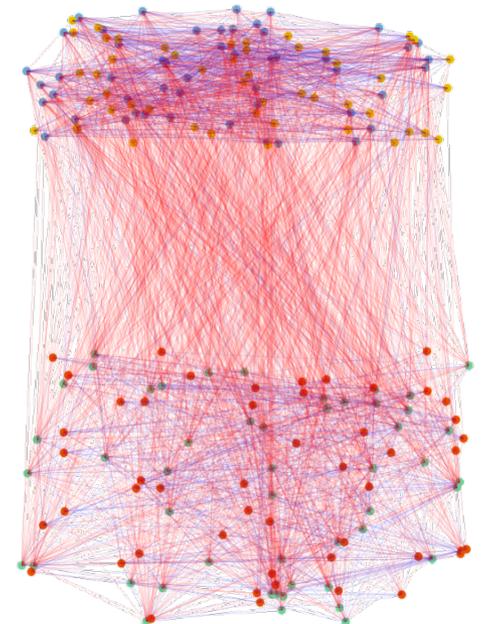
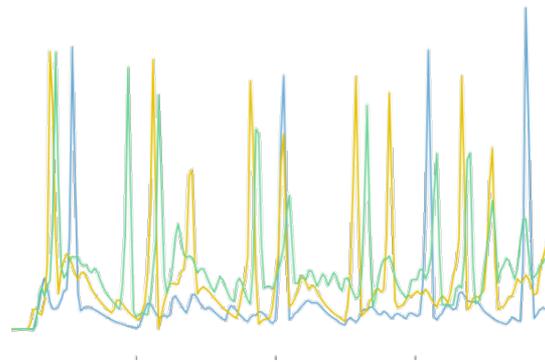


(**Networks** in **Python** and **NEURON**)

A Python package to facilitate the development, simulation and analysis of biological neuronal networks in NEURON

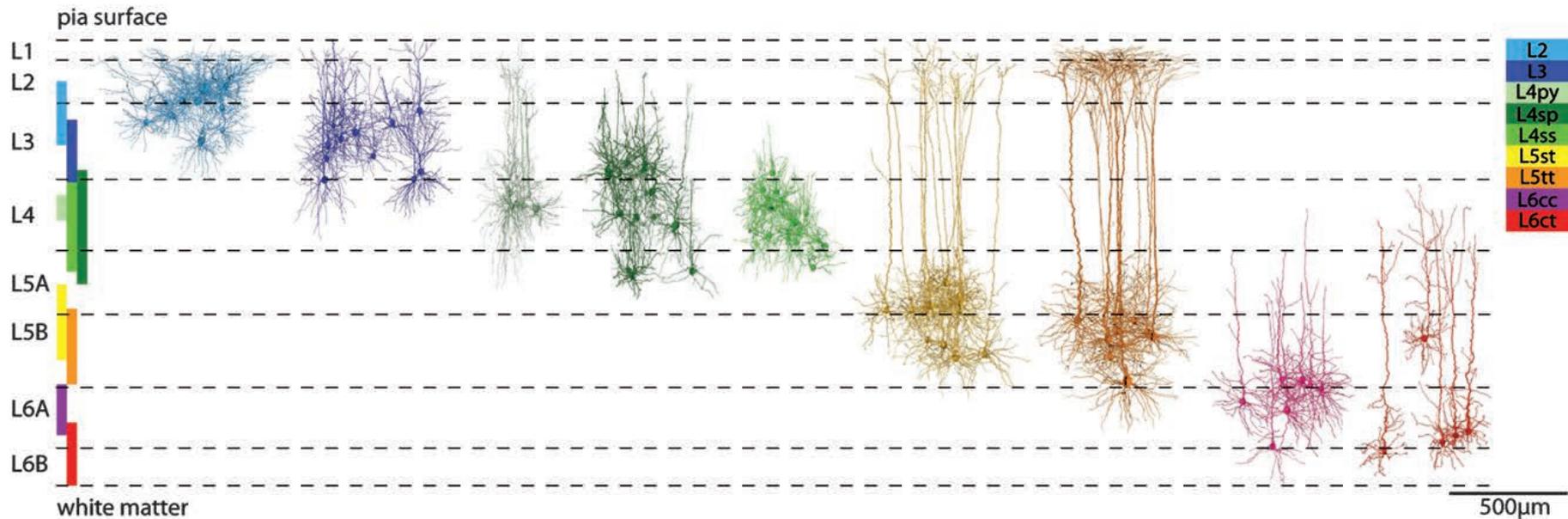


www.netpyne.org



Motivation

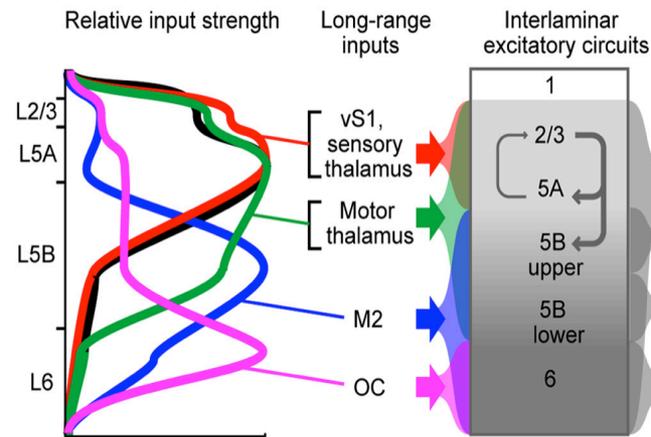
- Facilitate incorporation of experimental data at multiple scales



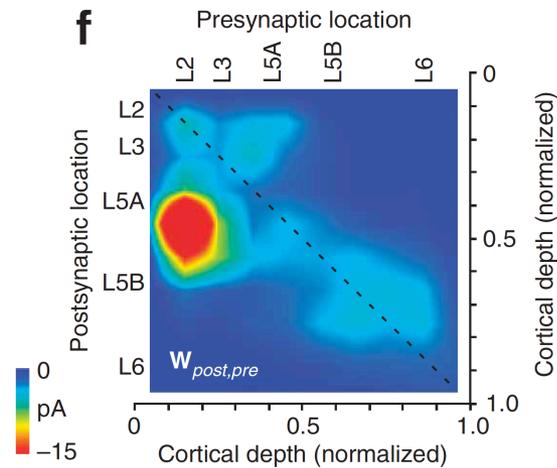
Motivation

- Facilitate incorporation of experimental data at multiple scales

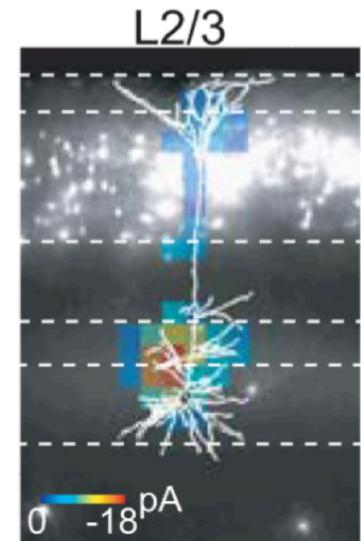
Long-range inputs



Local microcircuits



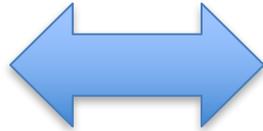
Dendritic inputs



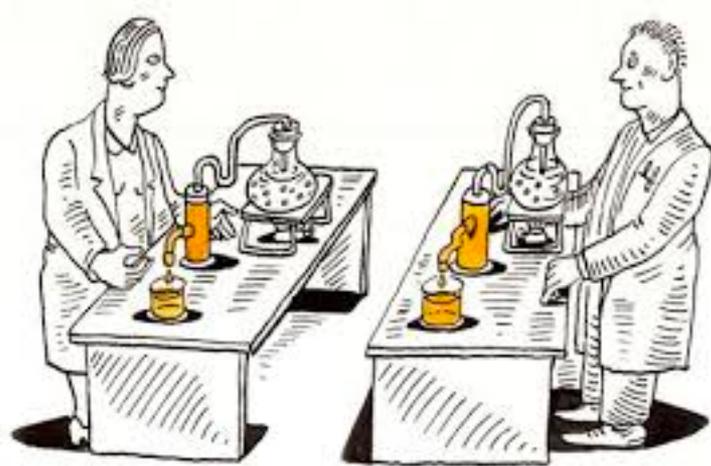
Motivation

- Separate model parameters from implementation
- Standardize format – easy to read, interpret, edit, share etc

```
popParams['EXC_L2'] = {  
  'cellType': 'PYR',  
  'yRange':   [100, 400],  
  'numCells': 50}
```



```
for cellParams in range(pop['numCells']):  
  cell = sim.Cell(cellParams)  
  cell.tags['y'] = numpy.random(100,400)  
  cell.tags['cellType'] = 'PYR'
```

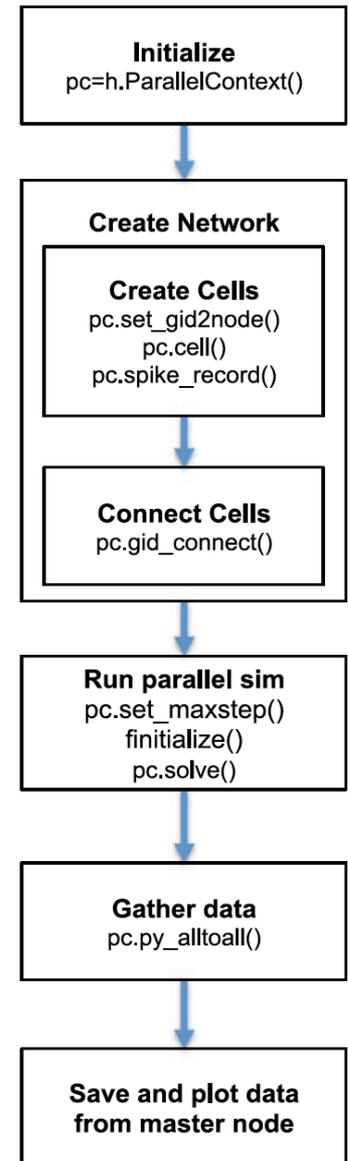
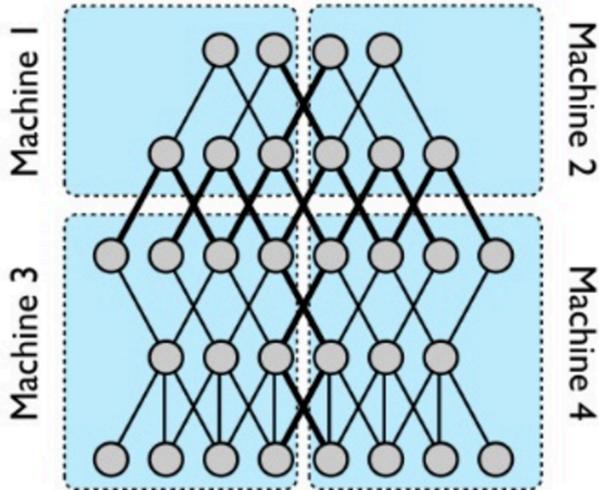


Replicate: get same thing to run again

Reproduce: make it yourself

Motivation

- Facilitate model parallelization (HPCs)
- Batch parameter exploration/optimization



NetPyNE

High level specifications

Network Parameters

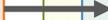
- Cell properties
- Connectivity
- ...

Simulation config

- Duration
- Saving options
- ...

NEURON
cell models

NeuroML cell
and network
models



NetPyNE

High level specifications

Network Parameters

- Cell properties
- Connectivity
- ...

Simulation config

- Duration
- Saving options
- ...

Network instantiation

Representation of **all cells, connections, etc**

Parallel Simulation

Distribution and gathering across MPI nodes

Simulation results

Spikes, voltage traces, ...

NEURON cell models

NeuroML cell and network models

NEURON simulator

NetPyNE

NEURON
cell models

NeuroML cell
and network
models

High level specifications

Network Parameters

- Cell properties
- Connectivity
- ...

Simulation config

- Duration
- Saving options
- ...

Network instantiation

Representation of **all**
cells, connections, etc

Parallel Simulation

Distribution and
gathering across
MPI nodes

NEURON
simulator

Simulation results

Spikes, voltage traces, ...

Analysis and saving

Analysis and Visualization

Connectivity matrix, raster plot, ...

Save to pickle, json,
mat, hdf5, ...

Export to NeuroML
format

Matlab,
Scipy,
Pandas,
Excel, ...

NeuroML

Brian, NEST,
MOOSE,
PyNN

NetPyNE

Batch simulation module (parameter exploration, MPI/HPC job submission, etc)

High level specifications

Network Parameters

- Cell properties
- Connectivity
- ...

Simulation config

- Duration
- Saving options
- ...

Network instantiation

Representation of **all cells, connections, etc**

Parallel Simulation

Distribution and gathering across MPI nodes

Simulation results

Spikes, voltage traces, ...

Analysis and saving

Analysis and Visualization

Connectivity matrix, raster plot, ...

Save to pickle, json, mat, hdf5, ...

Export to NeuroML format

NEURON cell models

NeuroML cell and network models

NEURON simulator

Matlab, Scipy, Pandas, Excel, ...

NeuroML

Brian, NEST, MOOSE, PyNN

High level specifications

NEURON

```
# add exc connection
postSyn1 = h.ExpSyn(postCell.dend(0.5))
postSyn1.tau = 2
postSyn1.e = -90

pre1Con = h.NetCon(preCell1.soma(0.5)._ref_v,
                  postSyn1,
                  sec=preCell1.soma)
pre1Con.delay = 1
pre1Con.weight[0] = 0.001
pre1Con.threshold = 0
```

High level specifications

NetPyNE

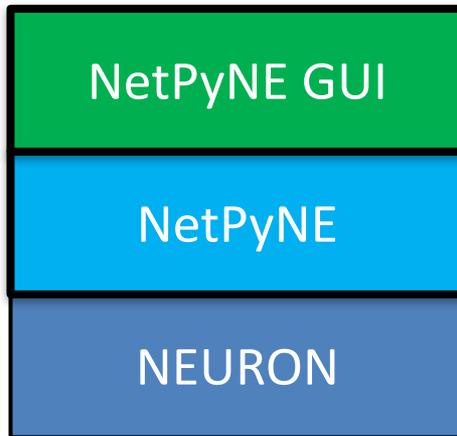
NEURON

```
## Cell connectivity rules
netParams.connParams['S->M'] = {
    'preConds': {'pop': 'S'},
    'postConds': {'pop': 'M'},
    'probability': 0.5,
    'weight': 0.01,
    'delay': 5,
    'synMech': 'exc'}
```

```
# add exc connection
postSyn1 = h.ExpSyn(postCell.dend(0.5))
postSyn1.tau = 2
postSyn1.e = -90

pre1Con = h.NetCon(preCell1.soma(0.5)._ref_v,
                  postSyn1,
                  sec=preCell1.soma)
pre1Con.delay = 1
pre1Con.weight[0] = 0.001
pre1Con.threshold = 0
```

High level specifications



Connectivity rules

Define here the rules to generate the connections in your network

The name of the connectivity rule

S->M

Add new Postsynaptic neuron section

dend

Add new Postsynaptic neuron location (0-1)

0.5

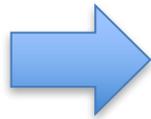
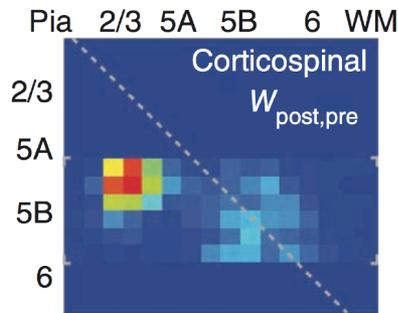
```
## Cell connectivity rules
netParams.connParams['S->M'] = {
    'preConds': {'pop': 'S'},
    'postConds': {'pop': 'M'},
    'probability': 0.5,
    'weight': 0.01,
    'delay': 5,
    'synMech': 'exc'}
```

```
# add exc connection
postSyn1 = h.ExpSyn(postCell.dend(0.5))
postSyn1.tau = 2
postSyn1.e = -90

pre1Con = h.NetCon(preCell1.soma(0.5)._ref_v,
                  postSyn1,
                  sec=preCell1.soma)
pre1Con.delay = 1
pre1Con.weight[0] = 0.001
pre1Con.threshold = 0
```

High level specifications

- ❑ Specifications are provided in a **standardized, declarative** Python format (JSON-like, lists and dicts).
- ❑ Clear **separation** of parameters from implementation code.
- ❑ Error **checking** and **suggestions** to facilitate model definition.



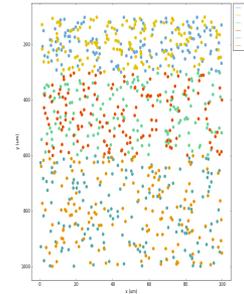
```
connParams['bin3->CSP'] = {  
    'preConds':    {'y': [100, 150]},  
    'postConds':  {'pop': 'CSP'},  
    'probability': 0.15,  
    'weight':     0.4,  
    'delay':      5,  
    'synMech':    'AMPA'}
```

NetPyNE facilitates building models based on experimental data

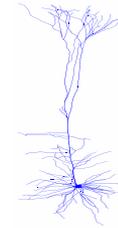
High level specifications

□ User can define:

- **Populations:** cell type, number of neurons or density, spatial extent, ...



- **Cell properties:** Morphology, biophysics, implementation, ...



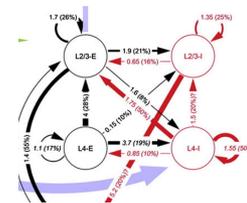
- **Synaptic mechanisms:** Time constants, reversal potential, implementation, ...



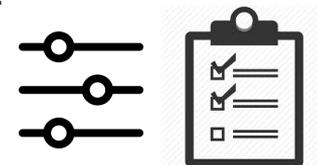
- **Stimulation:** Spike generators, current clamps, spatiotemporal properties, ...



- **Connectivity rules:** conditions of pre- and post-synaptic cells, different functions, ...

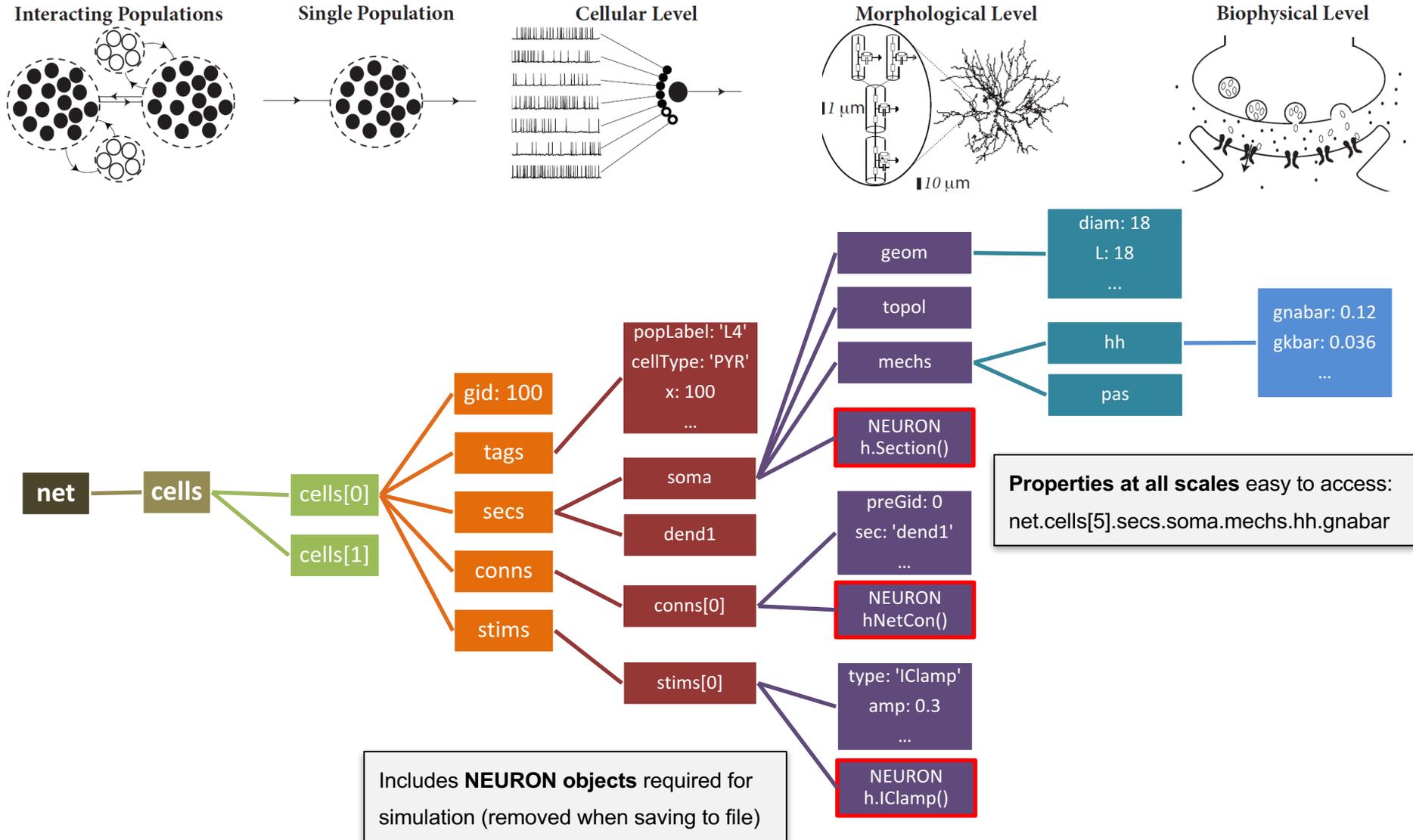


- **Simulation configuration:** duration, saving and analysis, graphical output, ...



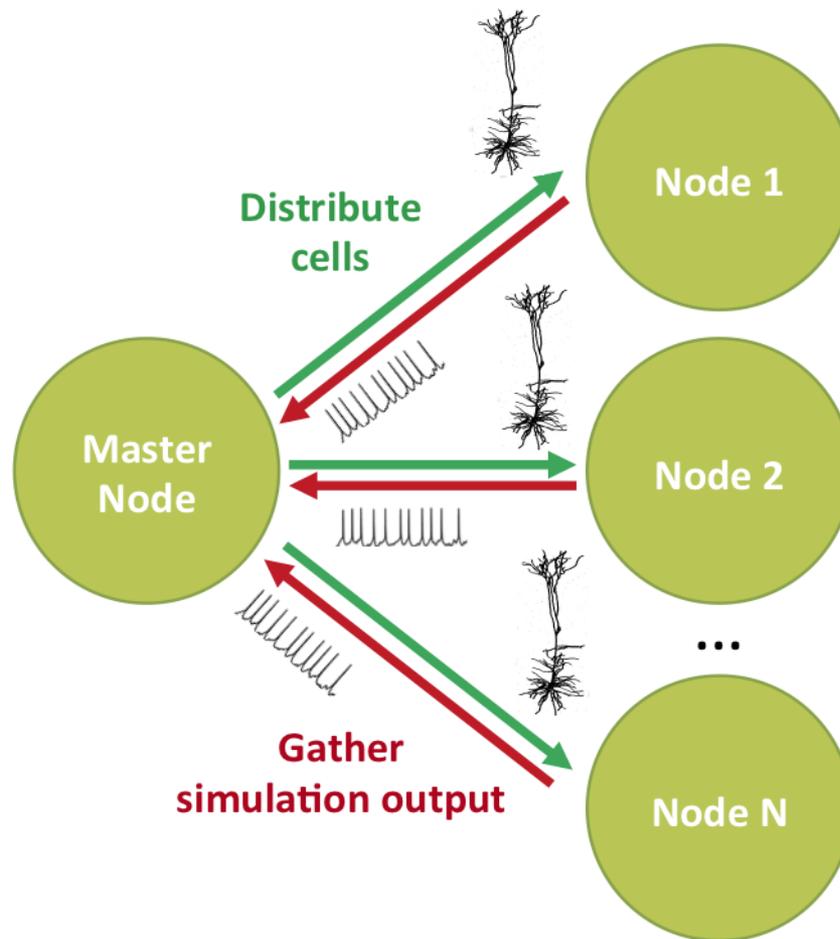
Network instantiation

□ Network instance as **standardized hierarchical** Python structure (JSON-like, lists and dicts)



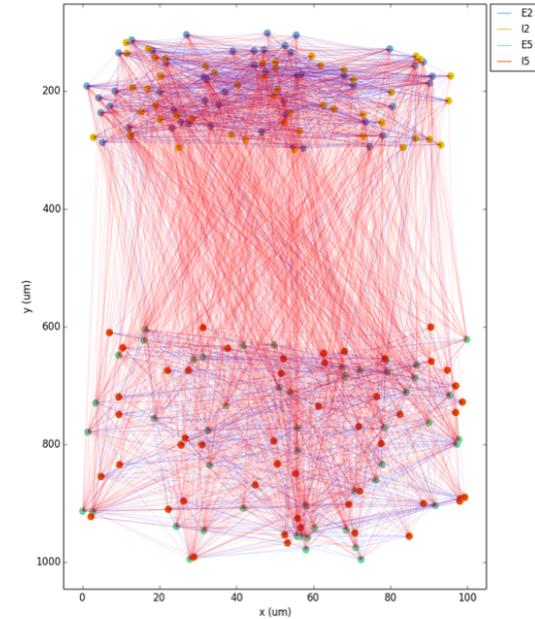
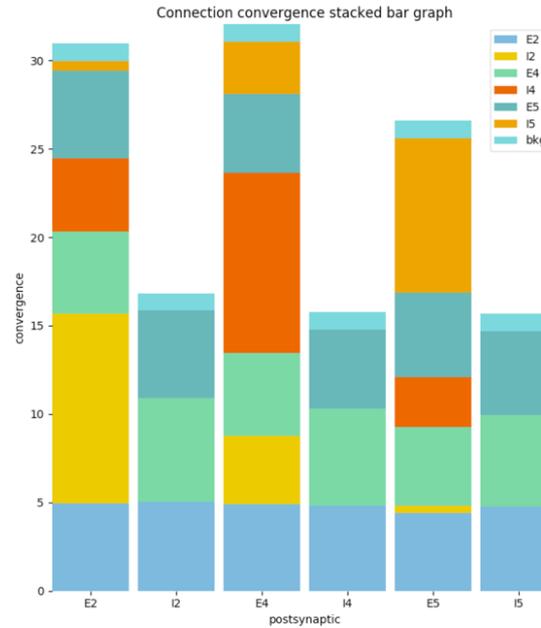
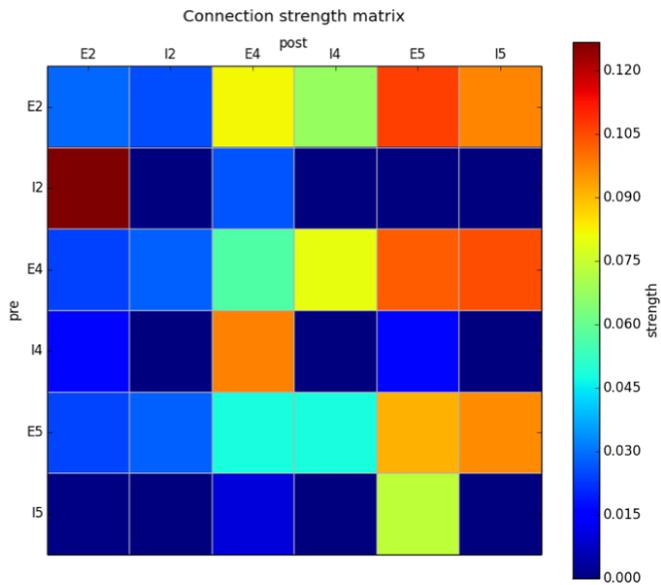
Parallel Simulation

- ❑ Set up for MPI **parallel simulation** across multiple nodes (via NEURON simulator).
- ❑ Takes care of balanced **distribution** of cells and **gathering** of simulation output from nodes.



Analysis

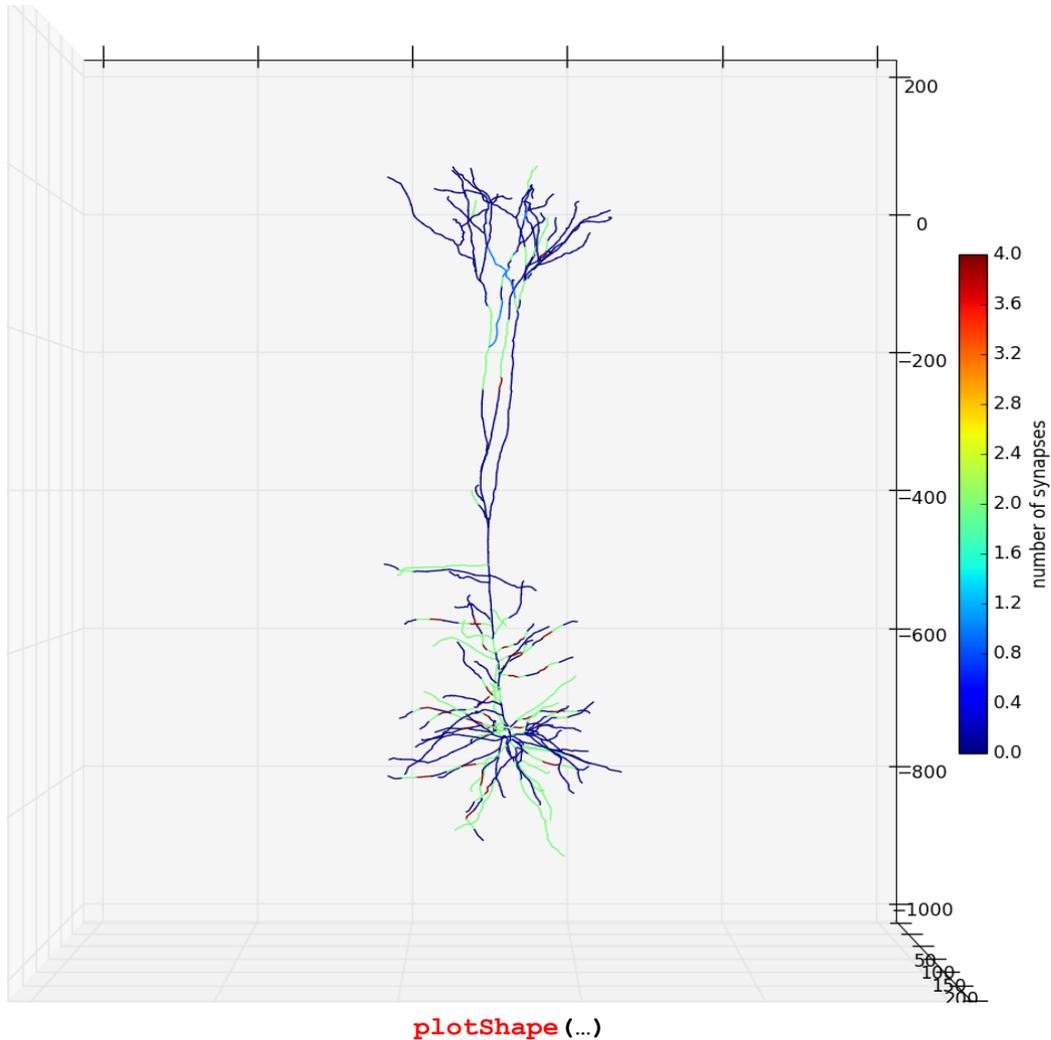
- **Connectivity plots** at cell or population level (weights, num connections, probability,...)



```
plotConn(include = ['allCells'], feature='strength',  
groupBy='pop', figsize=(9,9), showFig=True)
```

Analysis

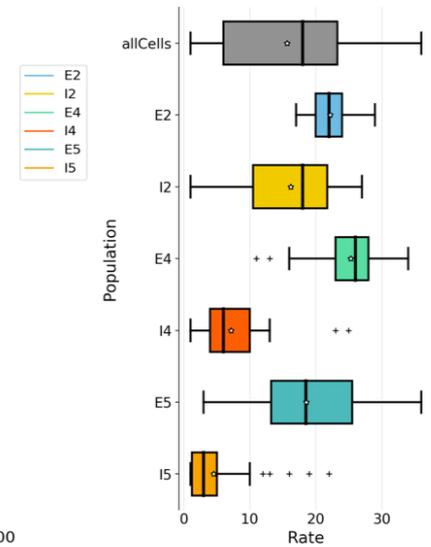
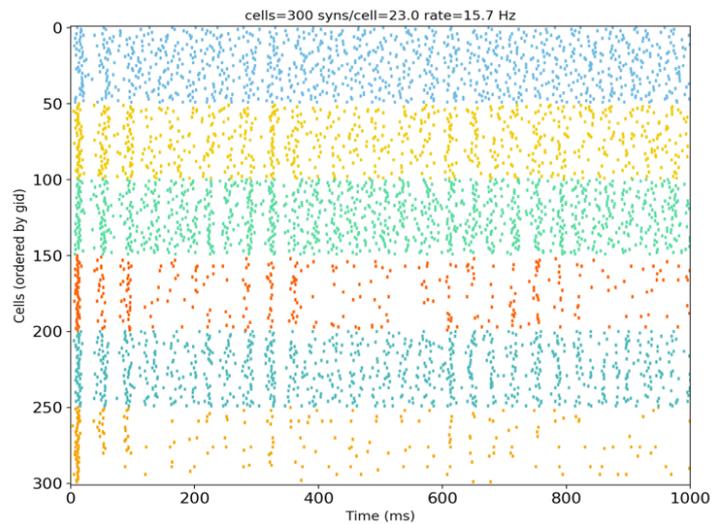
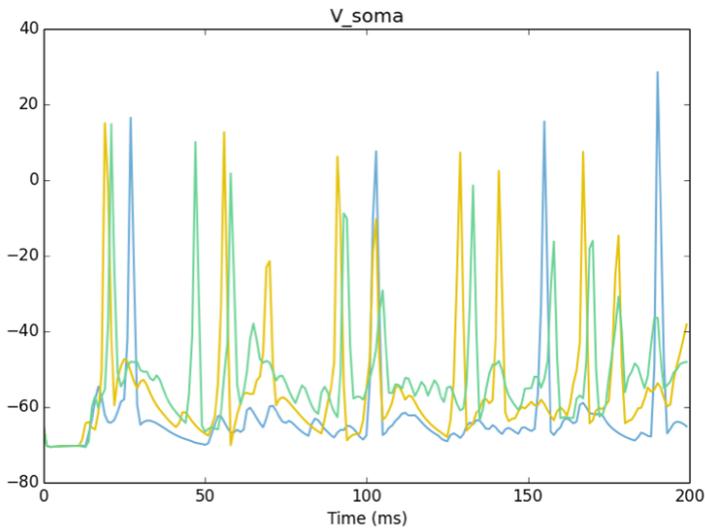
- 3D cell shape plot
- Option to include **color-coded variables** (eg, num of synapses)



Analysis

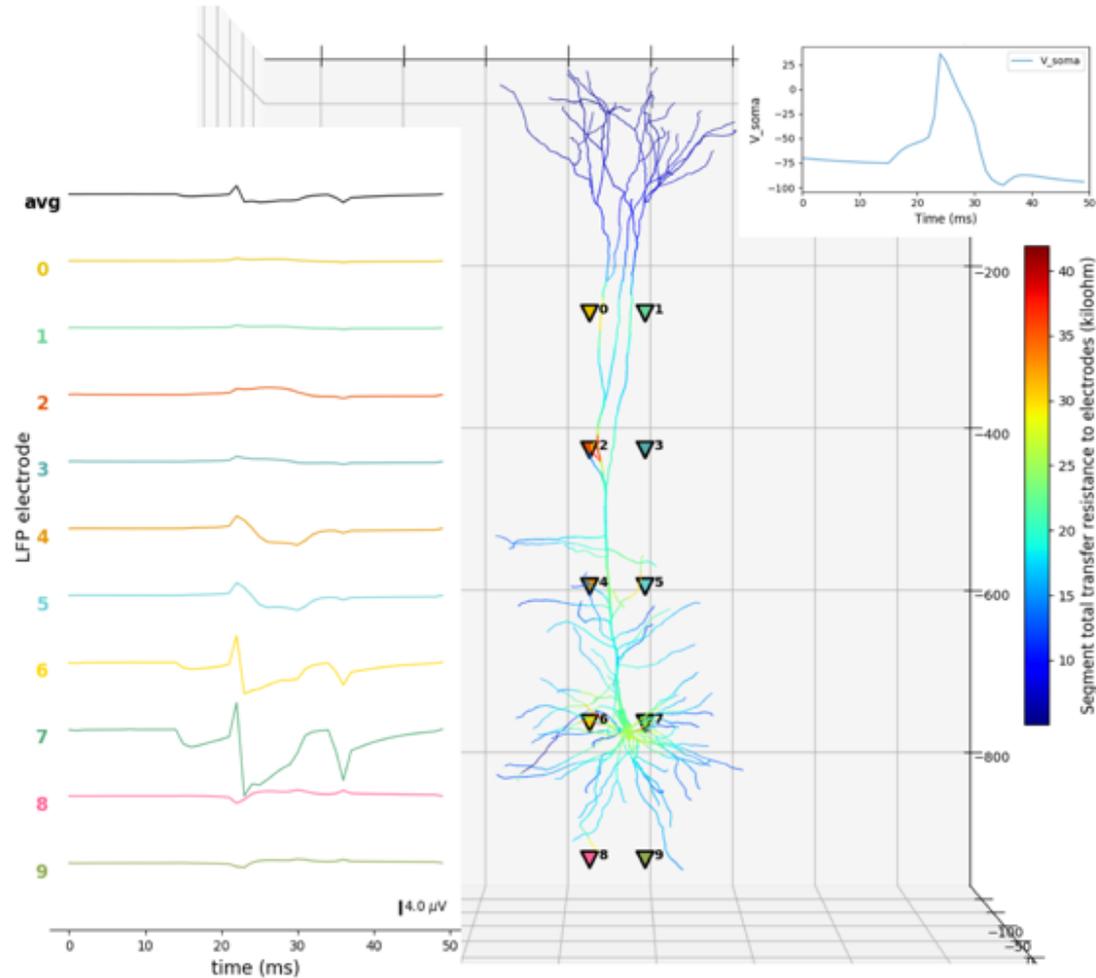
□ Simulation output

- Intrinsic cell variables (voltages, currents, conductance) **trace plots**
- **Raster plot** of any subset of cells
- **Spike histogram** of populations or subsets of cells
- Population **statistics**



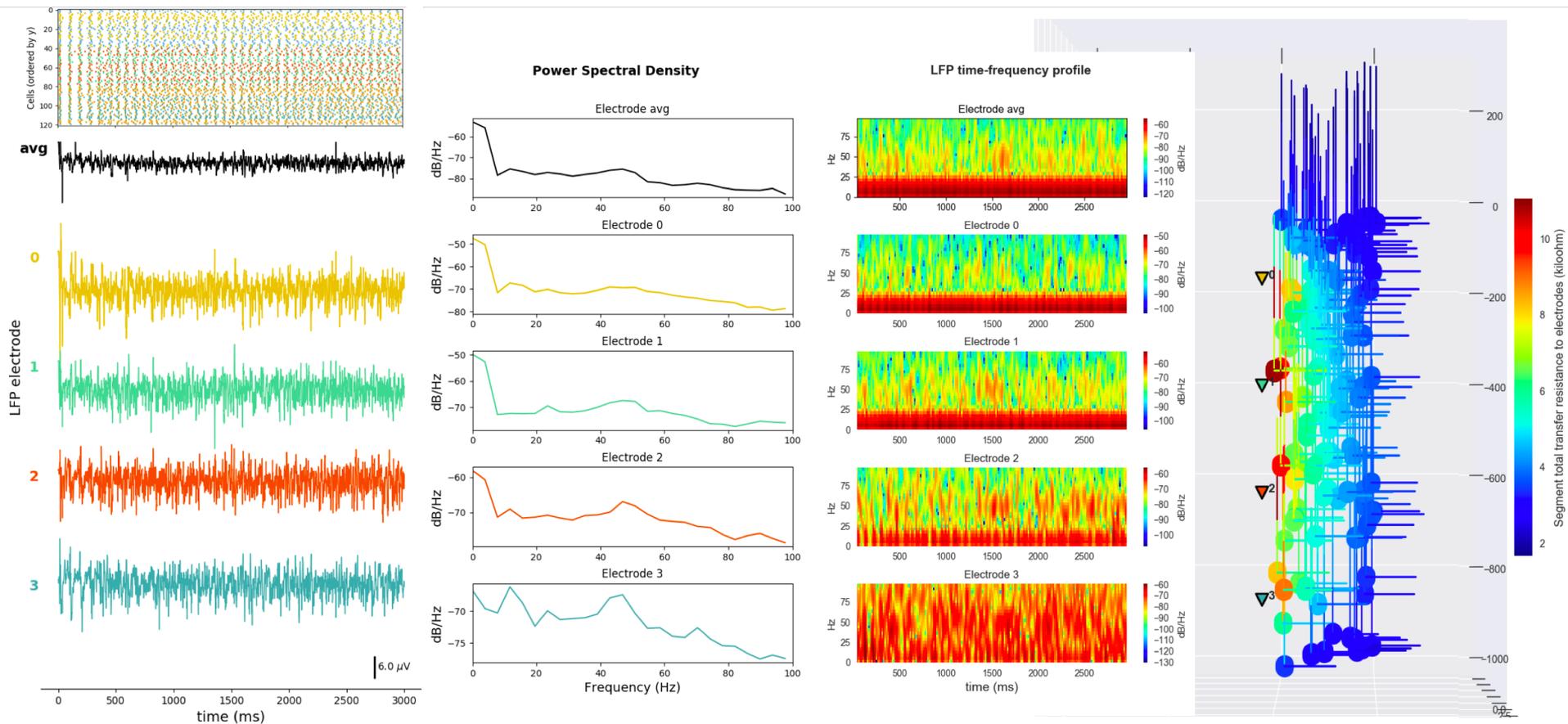
LFP

- LFP time-series, PSD, spectrogram and electrode locations (single cell)



Analysis

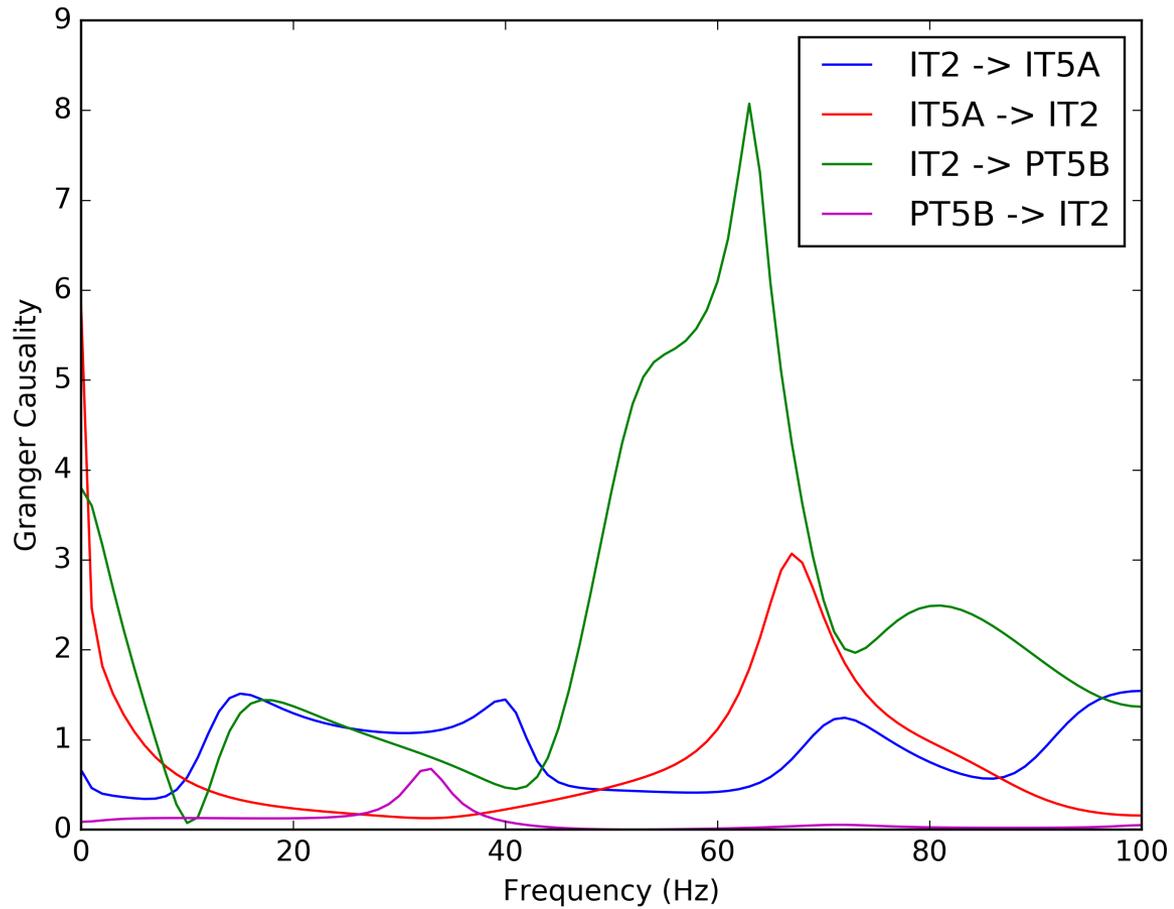
- LFP time-series, PSD, spectrogram and electrode locations (network)



plotLFP(...)

Analysis

- Spectral **Granger causality**
- Normalized **transfer entropy**



`plotGranger(...)`

Batch Parallel Simulations

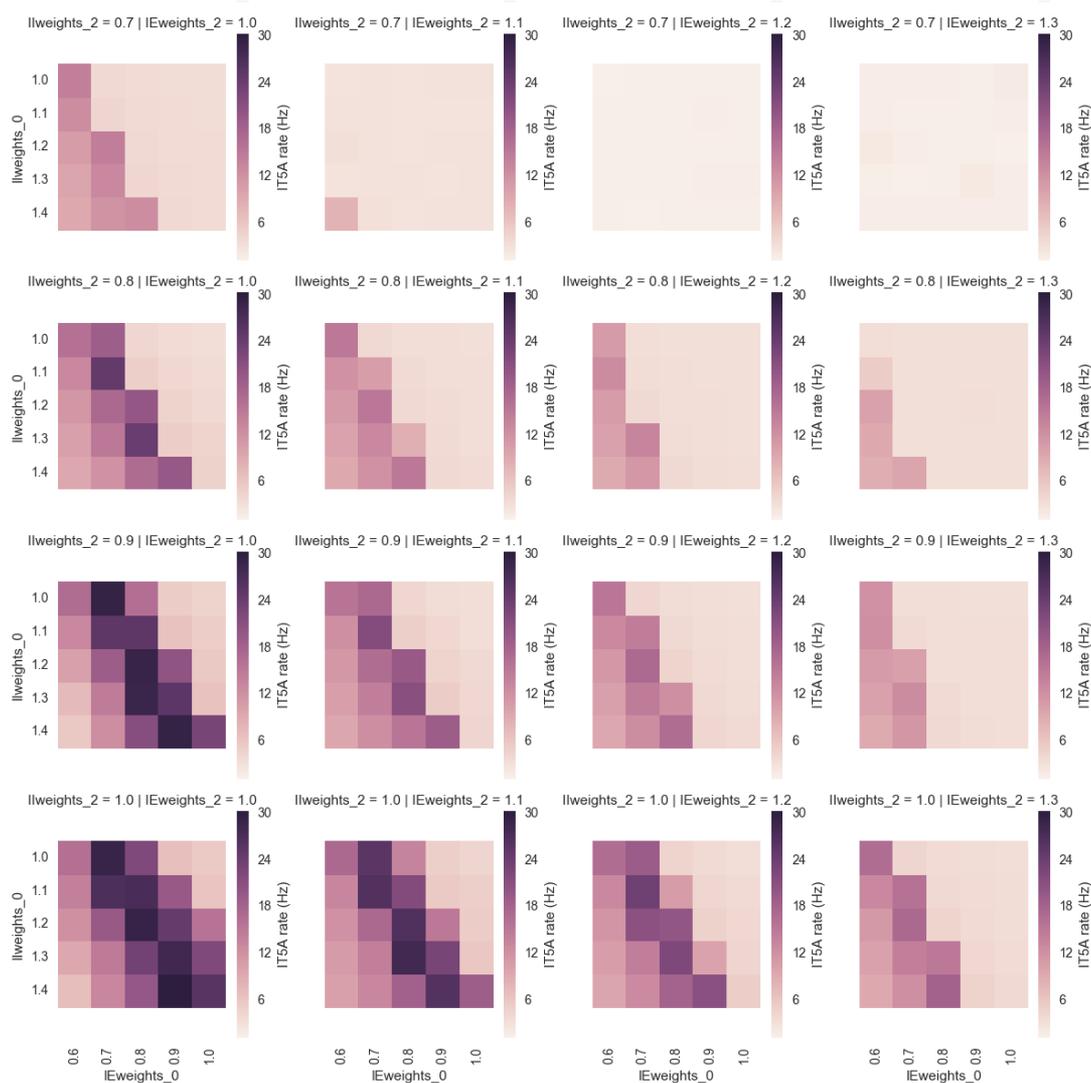
- ❑ **Easy specification** of parameters and range of values to explore in batch simulations.
- ❑ **Pre-defined, configurable** setups to automatically **submit jobs** in multicore machines (Bulletin board) or supercomputers (SLURM or PBS Torque)



SDSC SAN DIEGO
SUPERCOMPUTER CENTER

Batch Simulation Analysis

- Analysis and visualization of multidimensional batch simulation results.



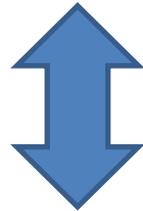
Data saving and exporting

- ❑ **Save and load** high-level specifications, network instance, simulation config and/or simulation results.
- ❑ **Multiple formats** supported: pickle, Matlab, JSON, CSV, HDF5
- ❑ **Export/import** network instance to/from **NeuroML**, the standard format for neural models.

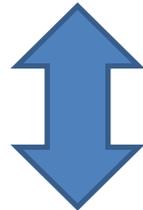
{JSON}



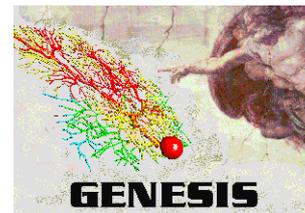
Data saving and exporting



Import/export to standard format



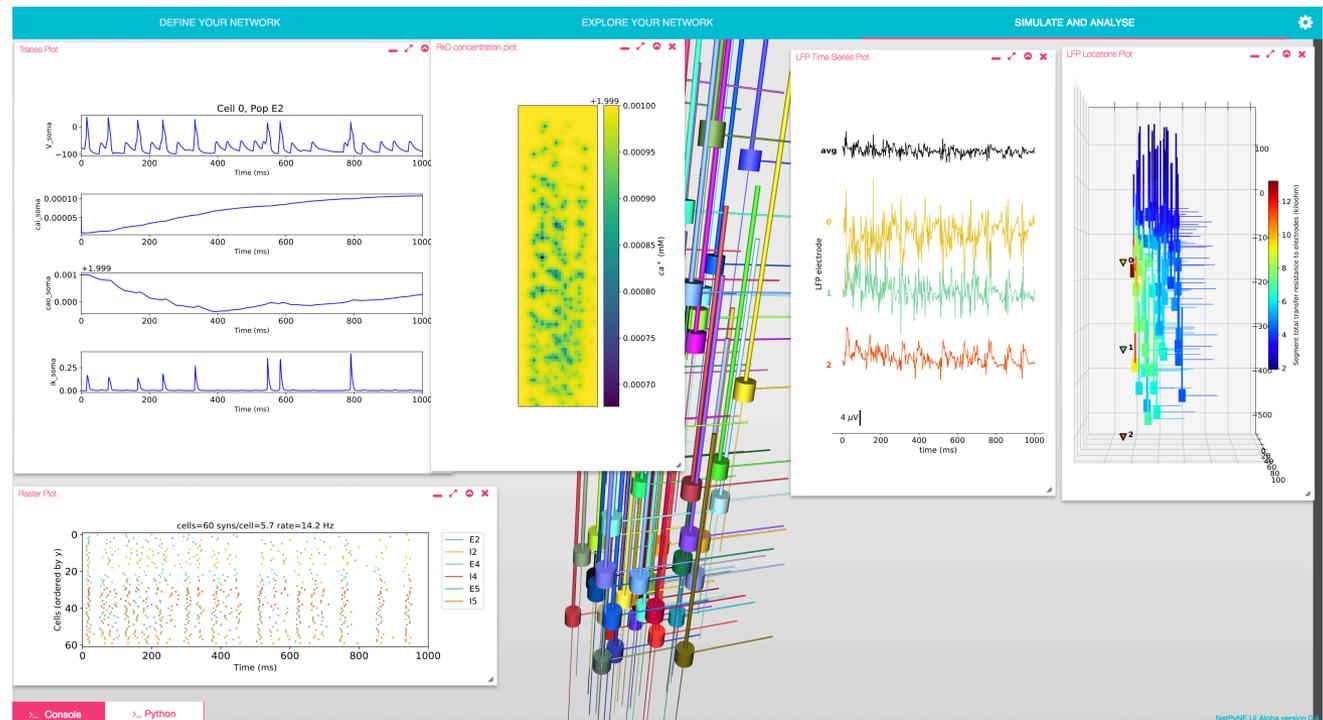
Import/export to other simulators



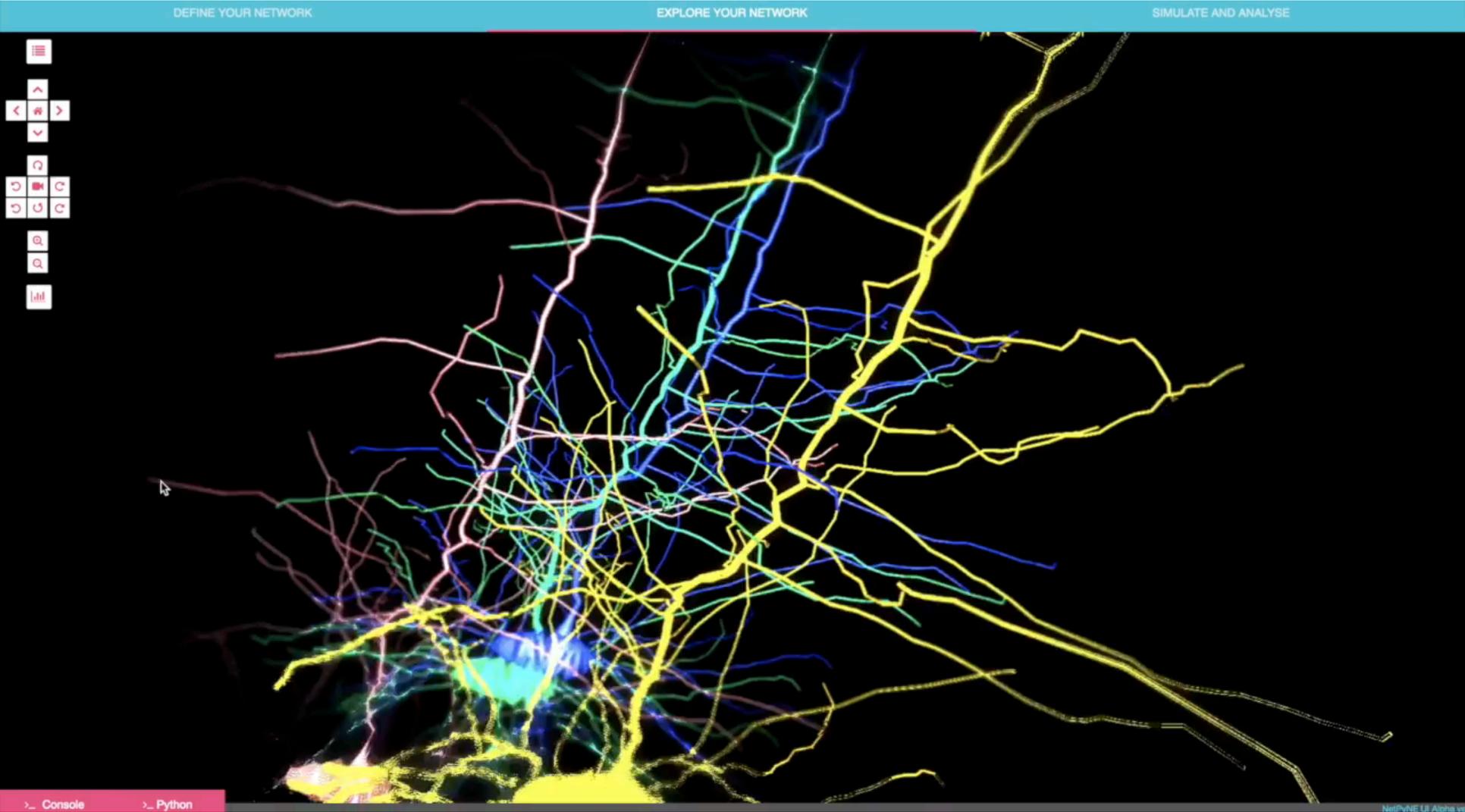
Development, simulation and analysis in GUI

□ Useful for:

- 1) Students/beginners
- 2) Prototyping model (can export to script)
- 3) Exploring/visualizing existing models



Development, simulation and analysis in GUI



NetPyNE: Documentation and Tutorials

www.netpyne.org

Welcome to NetPyNE's documentation!

NetPyNE is a python package to facilitate the development and parallel simulation of biological cell networks using the NEURON simulator.

Table of Contents

- Overview
 - What is NetPyNE?
 - What can I do with NetPyNE?
 - Main Features
- Installation
 - Requirements
 - Install via pip
- Tutorial
 - Very simple and quick example
 - Network parameters
 - Simulation configuration options
 - Network creation and simulation
 - Adding a compartment (dendrite) to cells
 - Using a simplified cell model (Izhikevich)
- Package Reference
 - Model components and structure
 - Network parameters
 - Simulation configuration
 - Structure of data and code
 - Network, Population and Cell classes
 - Package methods
 - Structure of saved data

NetPyNE: Q&A Forums

www.neuron.yale.edu
The NEURON Forum

Search...

Quick links FAQ Register Login

Board index < Tools of Interest to NEURON users < NetPyNE

NetPyNE
Moderator: tom_morse

New Topic Search this forum...

28 topics • Page 1 of 1

ANNOUNCEMENTS	REPLIES	VIEWS	LAST POST
VERSION RELEASES by salvadord » Fri Jun 09, 2017 10:41 pm	12	7554	by bremen Sat Apr 28, 2018 4:05 pm
Welcome to the NetPyNE Forum! by salvadord » Tue May 16, 2017 10:50 pm	0	7863	by salvadord Tue May 16, 2017 10:50 pm

TOPICS	REPLIES	VIEWS	LAST POST
Spike source and target sections by salvadord » Mon Nov 27, 2017 12:03 pm	17	4342	by bremen Sat May 12, 2018 12:07 pm
Import json format of morphology to NetPyNE by Javad » Fri May 04, 2018 3:02 pm	2	75	by ted Sun May 06, 2018 1:30 pm
Slow speed to save sim results by bremen » Sat Apr 21, 2018 10:32 am	2	51	by bremen Sat Apr 28, 2018 3:15 pm
Field names are restricted to 31 characters by bremen » Sat Mar 24, 2018 1:36 pm	2	55	by bremen Sun Mar 25, 2018 6:21 am
plotLFP by atknox » Fri Mar 02, 2018 6:44 pm	1	72	by salvadord Wed Mar 21, 2018 6:20 pm
Mat file not saved properly in batch functions by Vittorio » Thu Feb 15, 2018 10:58 am	1	91	by salvadord Thu Feb 15, 2018 11:30 am
Gap junction support - parallel simulation? by tmc » Wed Jan 24, 2018 10:18 pm	3	108	by salvadord Thu Feb 08, 2018 12:41 pm

<https://www.neuron.yale.edu/phpBB/viewforum.php?f=45&sid=99554ea5df10540d9b31e0c74929eaf0>

<https://groups.google.com/forum/#!forum/netpyne-forum>

Google Search for messages

Groups **NEW QUESTION** Mark all as read Actions Filters

My groups **NetPyNE Q&A forum** Shared publicly
3 of 3 topics

Home Starred Favorites Recently viewed HNNsSolver NetPyNE mailing list NetPyNE Q&A for... Recent searches Recently posted to NetPyNE Q&A for...

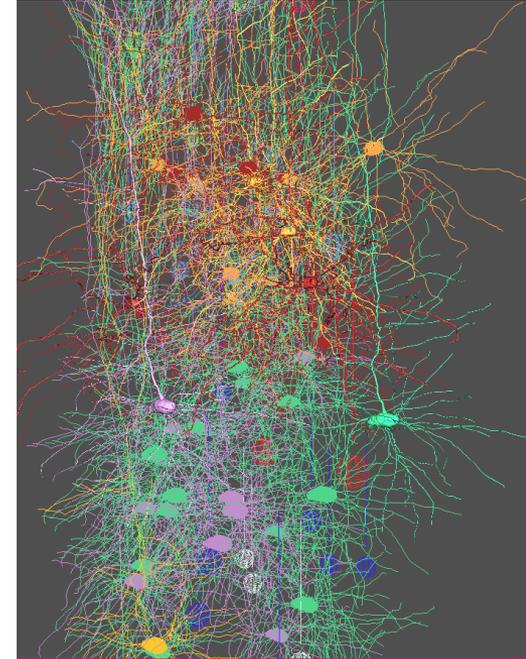
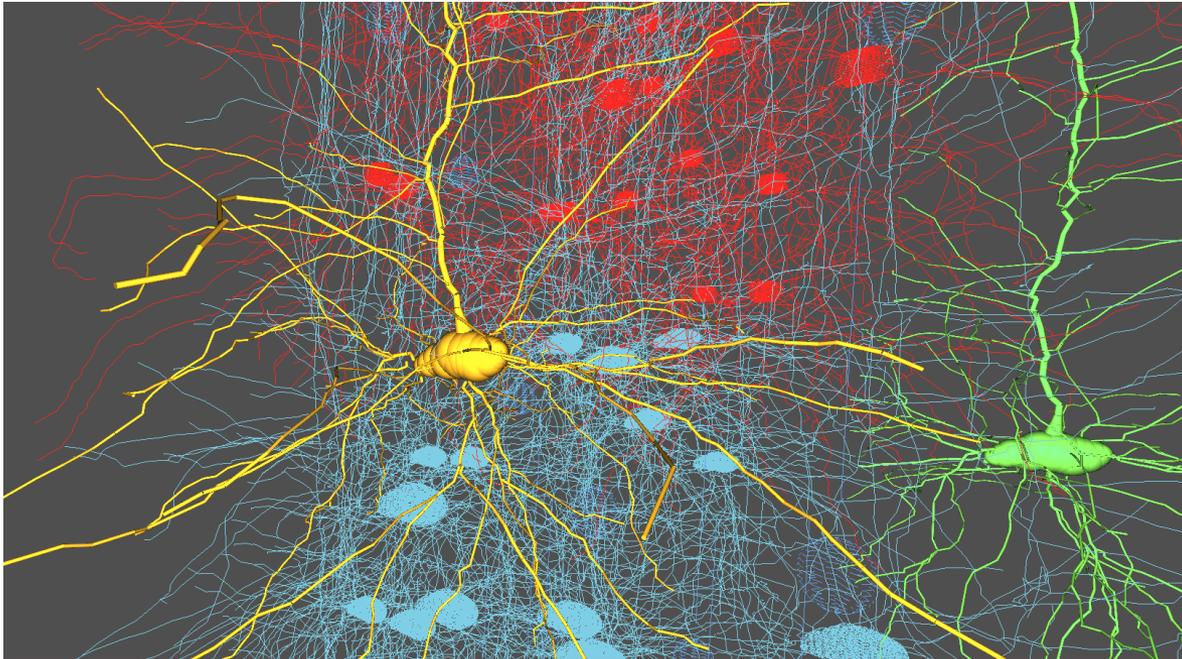
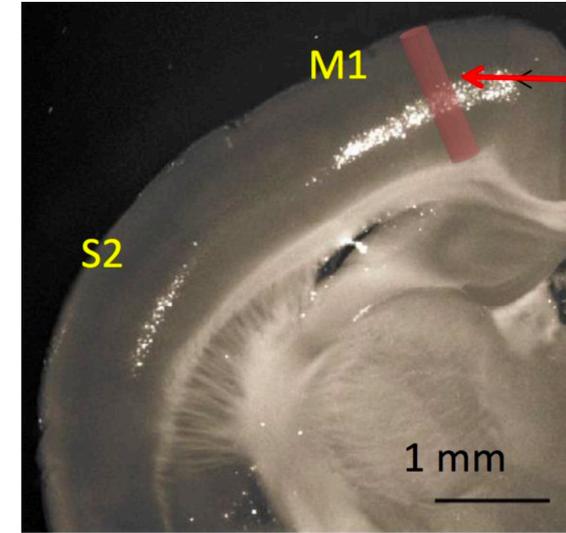
NetPyNE (www.netpyne.org) is a high-level python interface to NEURON that facilitates the development, parallel simulation and analysis of biological neuronal networks. This Q&A forum enables users and developers to post questions, answers and comments about the tool. Our previous Q&A forum with many posts can be found here: <https://www.neuron.yale.edu/phpBB/viewforum.php?f=45>

Edit welcome message Clear welcome message

<input type="checkbox"/>	Import json format of morphology to NetPyNE By Javad Paknahad - 2 posts - 3 views	May 5
<input type="checkbox"/>	GPUs or intel xeon phi coprocessor By atknox@gmail.com - 2 posts - 2 views	Apr 21
<input type="checkbox"/>	plotLFP By atknox@gmail.com - 2 posts - 3 views	Mar 23

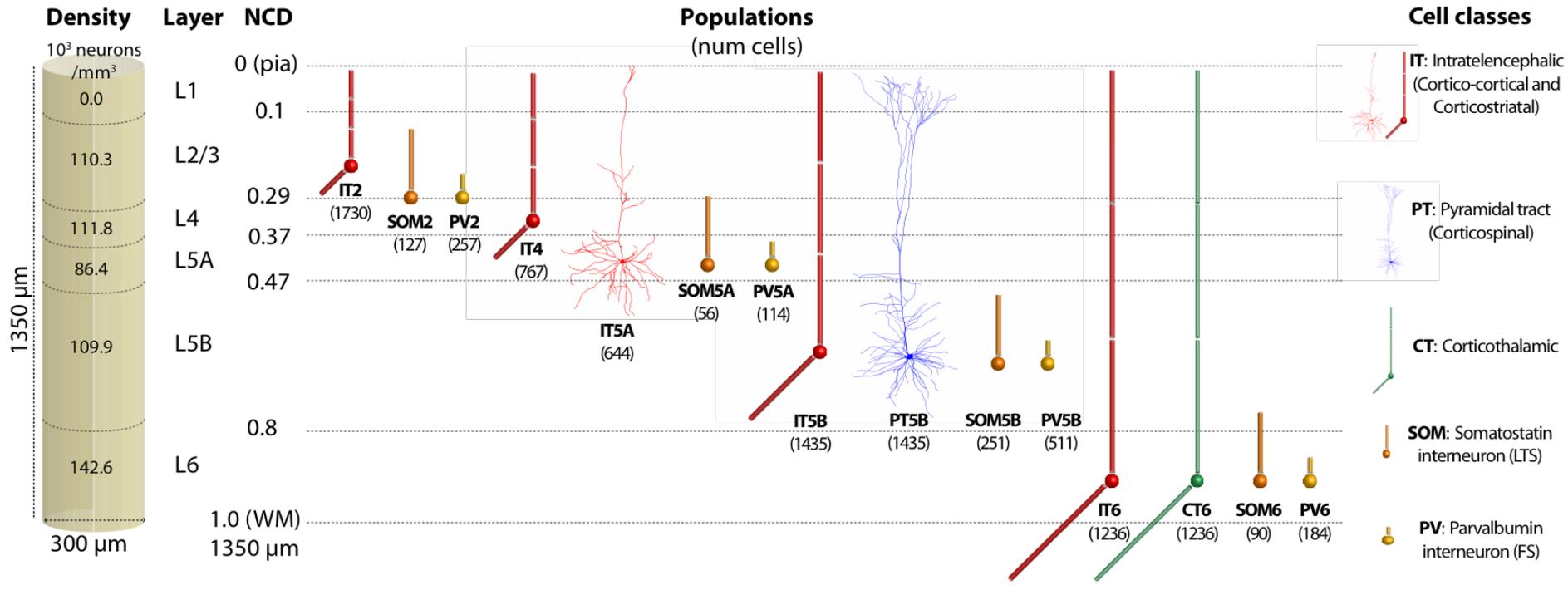
M1 microcircuits model

- ❑ Data-driven multiscale network model of **M1 microcircuits**
- ❑ Full scale cylindric volume of **300 μm** (diameter) x **1350 μm** (cortical depth)
- ❑ **~10,000 neurons** of 5 classes distributed in 15 populations
- ❑ **~30M synapses**

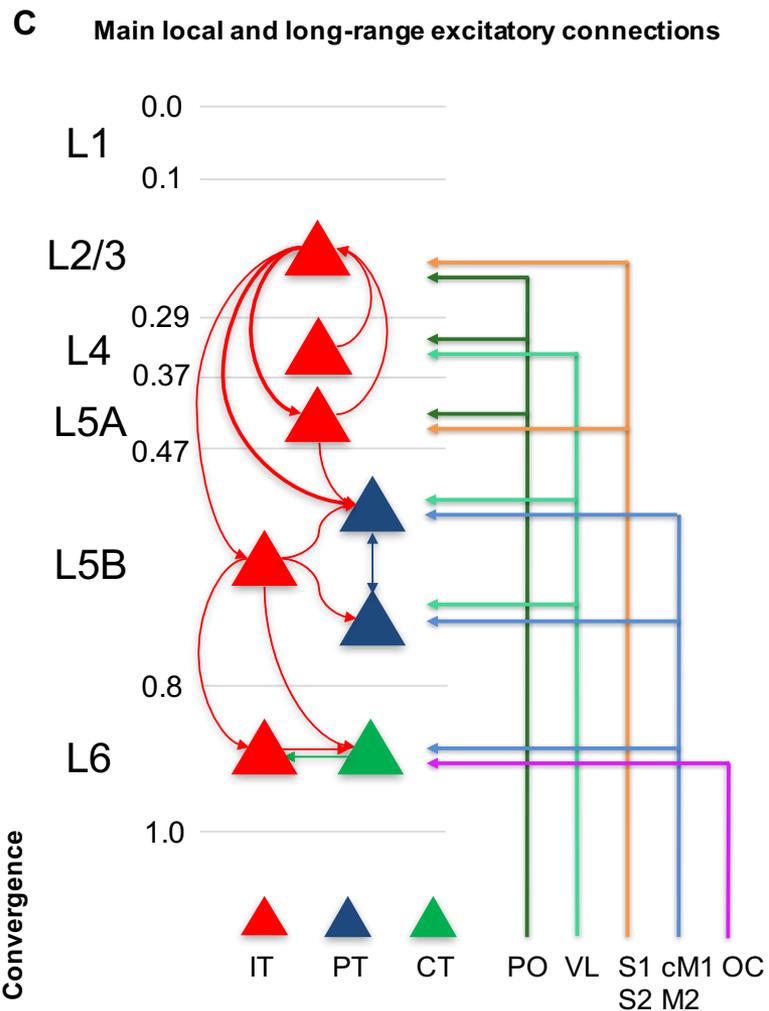
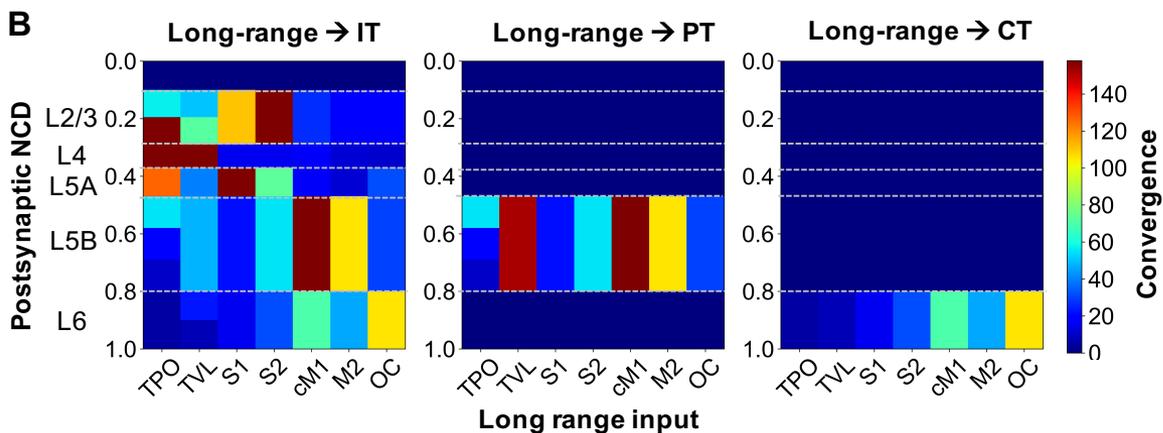
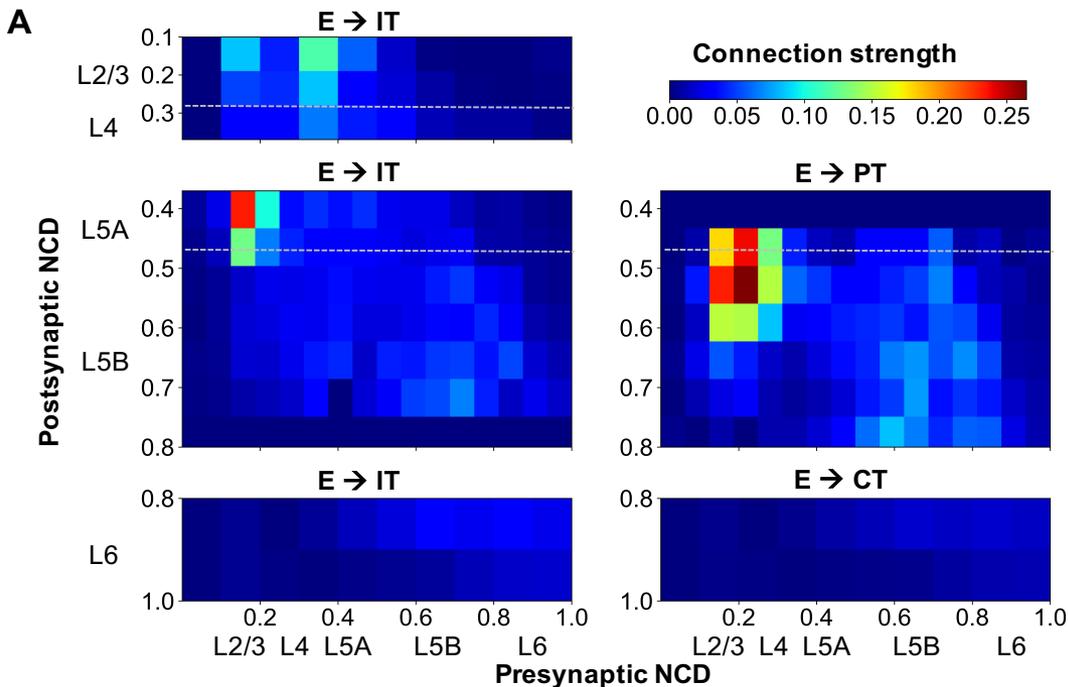


M1 microcircuits model

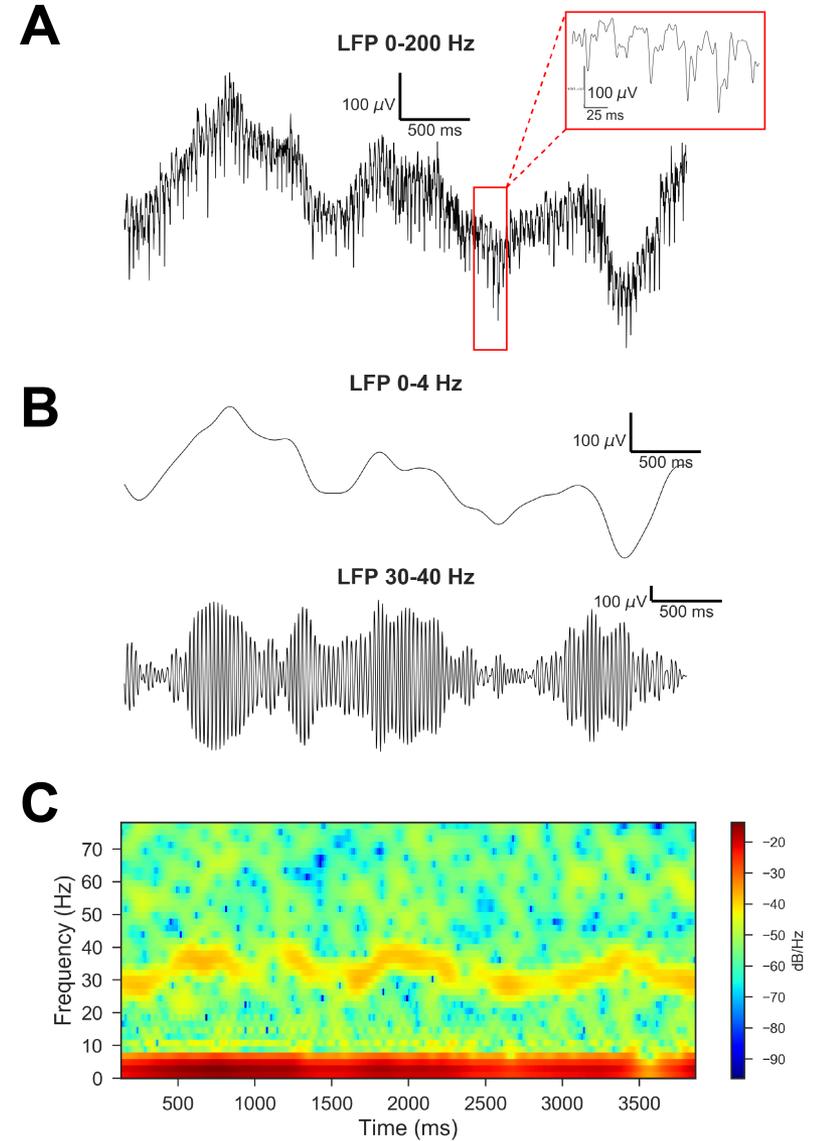
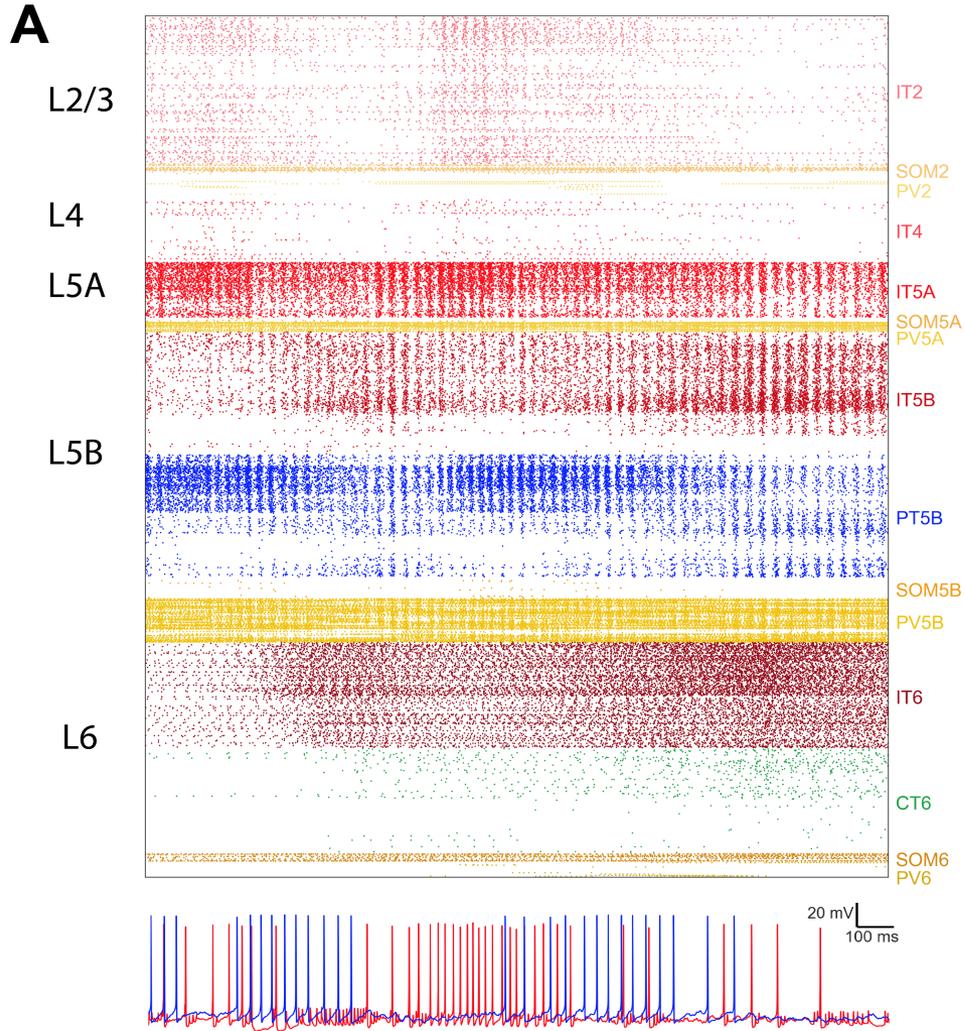
□ Data-driven multiscale network model of **M1 microcircuits**



M1 microcircuits model

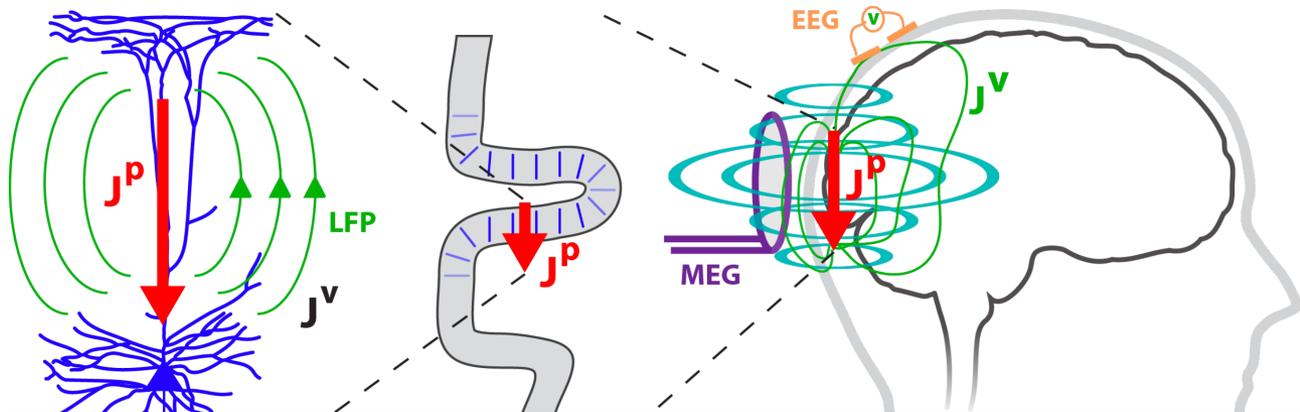


M1 microcircuits model



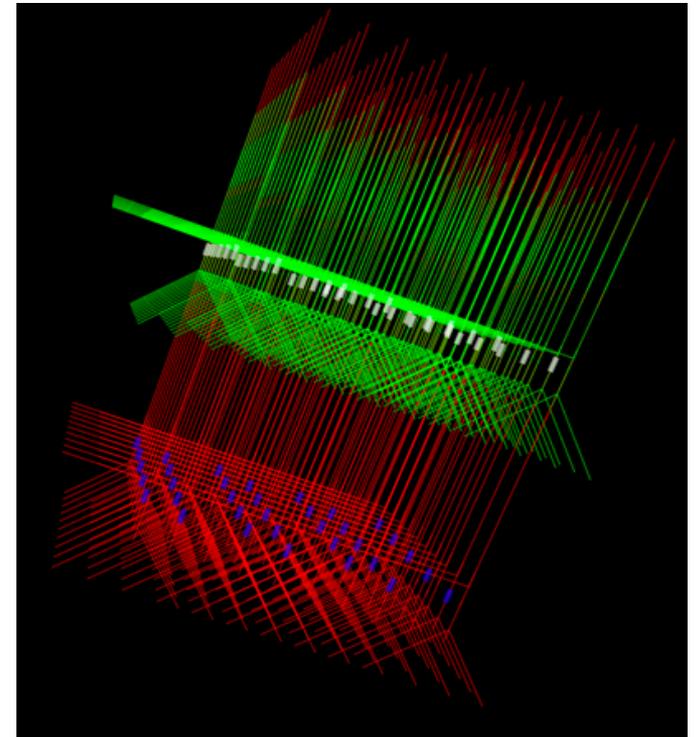
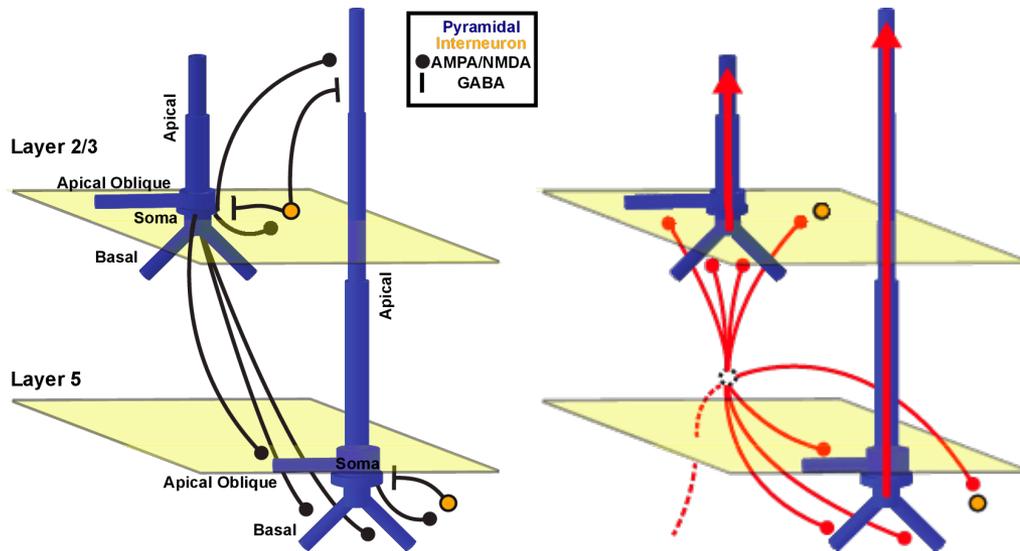
Human Neocortical Neurosolver

- Stephanie Jones (Brown University),
PI of NIH BRAIN R01
- Tool to reproduce/understand EEG/MEG
signals using biophysical circuit model



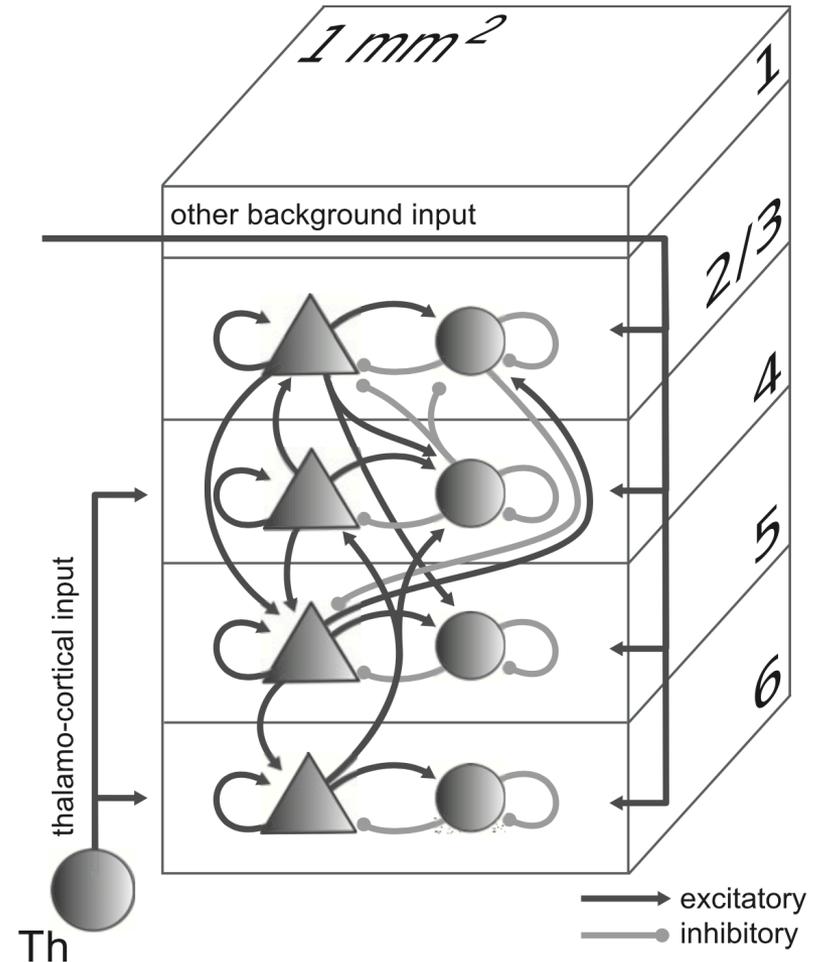
Human Neocortical Neurosolver

- ❑ Converted circuit model to NetPyNE
- ❑ Facilitate scaling, extension and customization



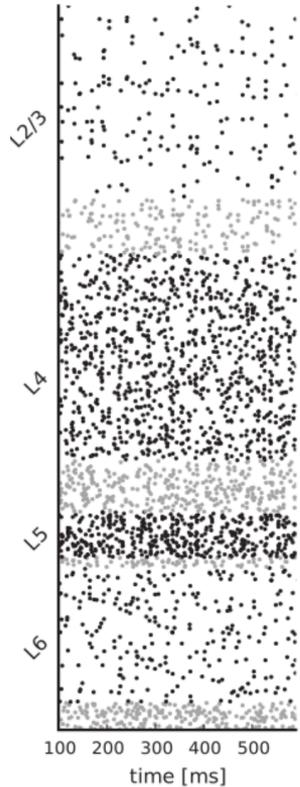
Potjan's & Diesmann model

- ❑ ~80k neurons (point model in NMODL)
- ❑ ~300M synapses
- ❑ Converted to NetPyNE
- ❑ Executed on Google Cloud

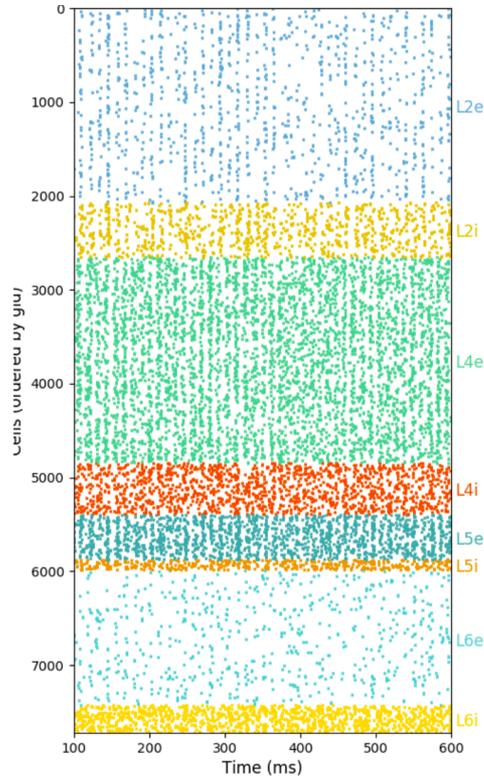


Potjan's & Diesmann model

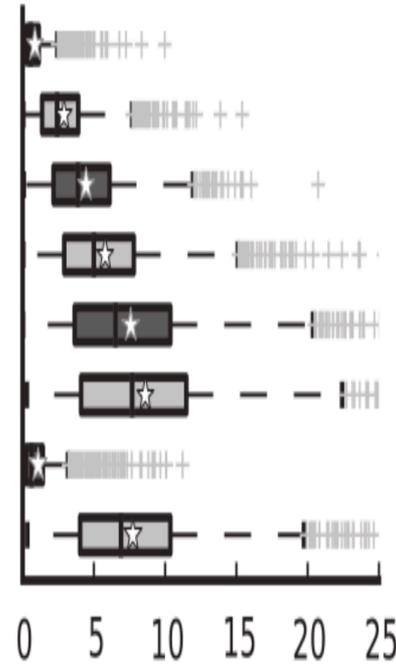
NEST



NetPyNE/NEURON



NEST



NetPyNE/NEURON

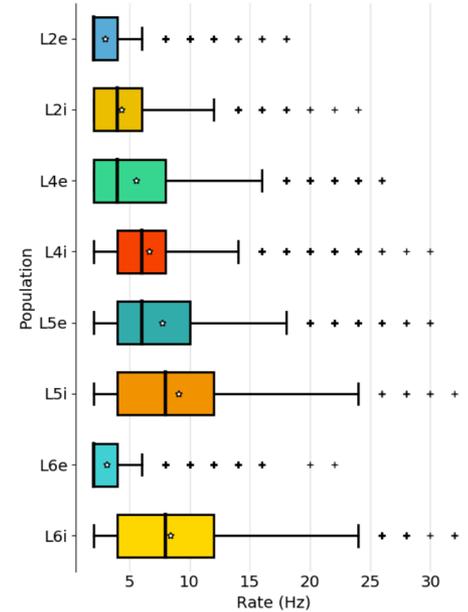


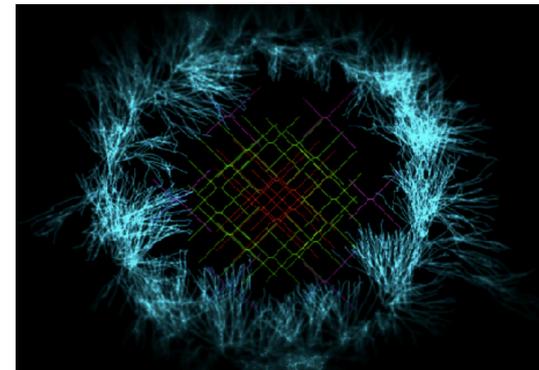
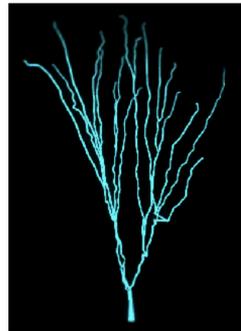
Figure 2: Raster plots network models scaled down to 100% of the original size (with around 80,000 neurons). NEST model on the left and NetPyNE model on the right.

Figure 3: Statistics of spiking activity of all 8 neural populations for rescaling of the PD model to 100% of its original size (around 80,000 neurons). NEST model on the left and NetPyNe model on the right.

NetPyNE: Existing models

□ Other models in progress:

- Traub **thalamocortical** network (P. Gleeson, UCL / S. Crook, Arizona)
- Hippocampus **CA3** (B. Tessler, SUNY DMC)
- **Spinal cord** circuits (V. Caggiano, IBM Watson)
- **Striatal** microcircuits (Hanbing/Christina Weaver, Franklin and Marshall College)
- **V1** network (Vinicius/Antonio Roque, Sao Paulo University)
- **Cerebellum** (Sergio Salines/Stefano Masoli, University of Pavia)
- **Dentate Gyrus** (F. Rodriguez, SUNY DMC)
- **Ischemia** in cortical network (Alex Seidenstein, SUNY DMC)
- **TMS/tDCS** network (Aman Aberra, Duke University)
- **LFP** oscillations (Christian Fink, Ohio Wesleyan)
- **Dendritic** computations (Birgit Kriener, Oslo)
- Thalamocortical **epilepsy** network (Andrew Knox, Cincinatti Hospital)
- Full list of 43 models: <https://drive.google.com/open?id=1bkWHakgZoEkYIkzrAS8sIKCvO5PSuUXLLRjNdN2pseY>



NetPyNE: Acknowledgments

❑ Contributors:

- Salvador Dura-Bernal (SUNY DMC)
- Ben Suter (Northwestern)
- Matteo Cantarelli (Metacell Ltd)
- Adrian Quintana (EyeSeeTea Ltd)
- Dario del Piano (Metacell Ltd)
- Facundo Rodriguez (SUNY DMC)
- Cliff Kerr (Sydney)
- Pdraig Gleeson (UCL)
- Robert McDougal (Yale)
- Michael Hines (Yale)
- Gordon MG Shepherd (Northwestern)
- William Lytton (SUNY DMC)

❑ Lab website: www.neurosimlab.org

❑ NetPyNE Website: www.netpyne.org

❑ NetPyNE-UI Website:
www.github.com/MetaCell/NetPyNE-UI

❑ Github: www.github.com/Neurosim-lab/netpyne
(open source development; contributions welcome)

❑ Funding:

- NIH Grant U01EB017695
- NIH Grant R01EB022903
- NIH Grant R01MH086638
- NYS Grant DOH01-C32250GG-3450000



SUNY
DOWNSTATE
Medical Center



PYTHON 3:

To install the the package run:

pip3 install netpyne_py3 (Linux or Mac OS) or
python -m pip install netpyne_py3 (Windows)

To upgrade to a new version run:

pip3 install netpyne_py3 -U (Linux or Mac OS) or
python -m pip install -U netpyne_py3 (Windows)

TO TEST:

In python interpreter type: ***from netpyne import sim***